Inside NETBIOS

3rd Edition

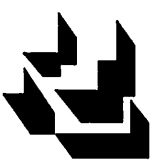
J. Scott Haugdahl

ARCHITECTURE
TECHNOLOGY
CORPORATION
SPECIALISTS IN COMPUTER ARCHITECTURE

Inside NETBIOS

3rd Edition

J. Scott Haugdahl



ARCHITECTURE
TECHNOLOGY
CORPORATION
SPECIALISTS IN COMPUTER ARCHITECTURE
P.O. BOX 24344 • MINNEAPOLIS, MINNESOTA 55424 • (612) 935-2035

First Printing April 1986
2nd Edition Printing March 1988
3rd Edition June1990

Published by:
Architecture Technology Corporation
P.O. Box24344
Minneapolis, MN 55424

Library of Congress Cataloging-in-Publication Data

Communication Networks.Includes Index.

1. Computer Networks 2. SNA

I. Haugdahl, J. Scott

ISBN 0-939405-00-8

DISCLAIMER

Architecture Technology Corporation makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability of fitness for any particular purpose.

Further, reasonable care has been taken to ensure the accuracy of this document, but errors and omissions could have occurred. Architecture Technology assumes no responsibility for any incidental or consequential damages caused thereby.

Further, Architecture Technology Corporation reserves the right to revise this document and to make changes from time to time in the content thereof without obligation to notify any person or organization of such revisions or changes.

This disclaimer applies to all parts of this document.

FOREWORD

Network Basic Input/Output System (NETBIOS) started out as a high-level programming interface for PC-DOS applications to IBM's PC Network broadband local area network (LAN). The NETBIOS interface was originally developed by Sytek (now Hughes LAN Systems) and operated over proprietary Sytek protocols. IBM reiterated the importance of NETBIOS by offering a NETBIOS emulator when the Token-Ring was announced, allowing applications originally developed for the PC Network to be run on the Token-Ring. NETBIOS has since proliferated to virtually every PC LAN and has several vendors and dozens of LAN applications written for it.

Written to be beneficial to both technical and non-technical readers, Inside NETBIOS includes general and overview information in addition to programming and design information.

The book begins with a history and conceptual view of NETBIOS and its relationship to the OSI Reference Model, continues with the command and packet structure of NETBIOS, details the server message block protocols, looks at various NETBIOS products from several vendors, and examines NETBIOS standards efforts.

This book is essential reading for network managers, implementors, and serious users of PC local area networks.

J. Scott Haugdahl
Minneapolis, Minnesota
June 1990

Table of Contents

List of Figures

To Nancy and Daniel

Chapter 1

Introduction

History

Network Basic Input/Output System (NETBIOS) started out as a high-level programming interface for PC-DOS applications to IBM's PC Network broadband local area network (LAN). The NETBIOS interface was originally developed by Sytek (now Hughes LAN Systems) and operated over proprietary Sytek protocols. IBM reiterated the importance of NETBIOS by offering a NETBIOS emulator when the Token-Ring was announced, allowing applications originally developed for the PC Network to be run on the Token-Ring. Another NETBIOS milestone occurred when IBM announced the PC LAN Support Program (which includes a NETBIOS driver) in conjunction with the Personal System/2. The PC LAN Support Program incorporates a NETBIOS emulator for operation in the newer PC Network broadband adapters, PC Network baseband adapters, and Token-Ring adapters and works with the Personal System/2 family.

It is important to keep in mind that NETBIOS is an interface, not a protocol per se. It defines a series of commands, but not how these commands are to be communicated between two stations on a LAN. This is one of the problem areas with NETBIOS. Vendors will implement NETBIOS on various protocols "stacks" (such as Novell's IPX, Xerox XNS, TCP/IP, etc.). Programs written for a NETBIOS interface will generally port to various networks with no major

problems. However, mixing protocol stacks on the same LAN may prevent NETBIOS programs from communicating with each other!

Another important aspect of NETBIOS is how it relates to the Open Systems Interconnection (OSI) reference model. For now, we note that NETBIOS is essentially a layer 5 or session level interface, with some support for lower level datagram services. The details of this will be discussed later.

By the time NETBIOS was included with IBM's OS/2 Extended Edition, it became a defacto industry standard. Thus the desires of many developers of PC LAN software and hardware have been met in that NETBIOS developed into a standard interface for PC LANs. Appendix C contains a listing of numerous applications using the NETBIOS interface available from several vendors.

Protocols

Protocols are simply a set of conventions that allow two or more end points to communicate. There are three major elements in protocols: syntax, semantics, and timing. The syntax of a protocol defines the fields; for example, there may be a 16-byte field for the addresses, a 32-byte field for check sums, and 512 bytes per packet. The semantics of the protocol attach meanings to those fields. For example, if an address field consists of all ones, it is a broadcast packet. Timing, the number of bits per second at which data is sent, is also important, not only at very low levels of protocols, but at high levels of protocols as well.

Figure 1-1 illustrates a diagram of a typical message format. At the beginning of the message, which is flowing one-way over the network, synchronization characters are assigned such that another node on the network can see that the message is coming and synchronize its receiver with the transmitter. The message header contains addressing information -- where the message is going and where it comes from. The message text itself is information which is going to be sent over the network. It has a header and in some instances a trailer, which is an indication of where the message ends. There may be control and synchronization characters at the end of the message too.

There are several considerations for end-to-end protocols. One issue is addressing structure: will it be flat so that there is one large addressing space for an interconnected system, or will it be hierarchical so that there is a tree-like structure of addresses such as network, station, and socket within a station? What is the addressing space of the system -- how many nodes, or PCs, can logically be addressed on the

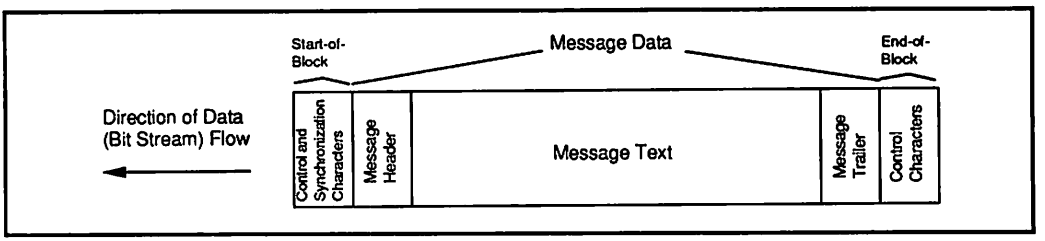


Figure 1-1: Typical Message Format

system? Despite what many vendors claim, the number of PCs that can attach to a system is usually far fewer than the addressing space. What should the data unit size be? There must be a compromise between large and small data unit sizes as large units may "hog" the system and small units may have more overhead. Does the system have some type of error control? If something goes wrong, can the protocol system show what the problem is, and does it provide for error recovery? How do packets synchronize themselves in the protocol layers? Suppose someone inadvertently affects someone else's data packet or writes into the wrong file server, is protection provided? Is there protocol monitoring for resource management and performance analysis of the system?

As we shall see in later chapters, NETBIOS addresses many, but not all, of these protocol considerations.

PC Network and Token-Ring

Applications (such as IBM's PC LAN Program, Asynchronous Communications Server, and the SNA 3270 Program) written to NETBIOS for PC Network will work "as is" with the Token-Ring emulator, but the two networks carry out the underlying protocols differently. The implementation of NETBIOS for PC Network and the Token-Ring is illustrated in Figure 1-2. The important difference is that NETBIOS in PC Network is totally self-contained on the PC Network Adapter. In the Token-Ring, by contrast, it is contained within the host PC itself.

Also note that in the Token-Ring implementation, there is no network nor transport layer of protocol (a discussion of network and transport protocol concepts follows). Although some debate exists on whether the Token-Ring implementation can perform as well without a dedicated NETBIOS co-processor, tests performed by IBM suggest at least a factor of 2 better performance with the Token-Ring NETBIOS emulator for reasons discussed later.

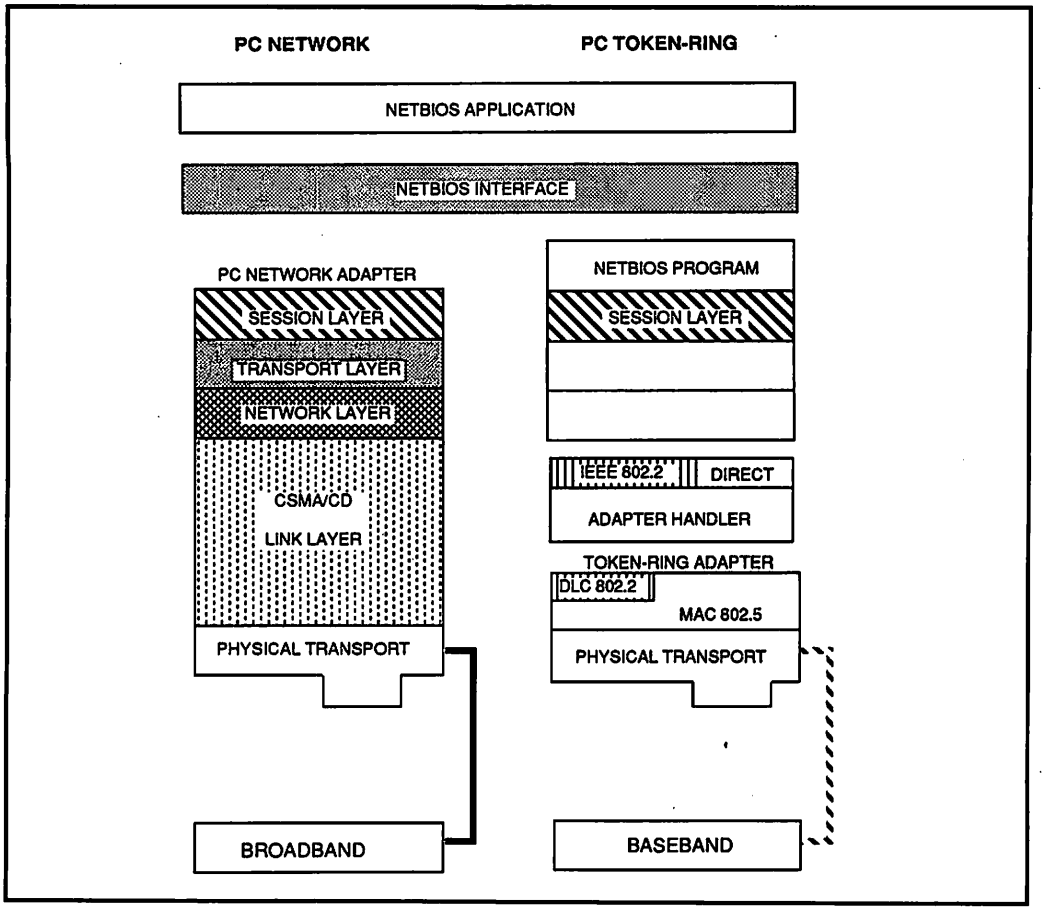


Figure 1-2: NETBIOS Implementation

IBM PC LAN Program

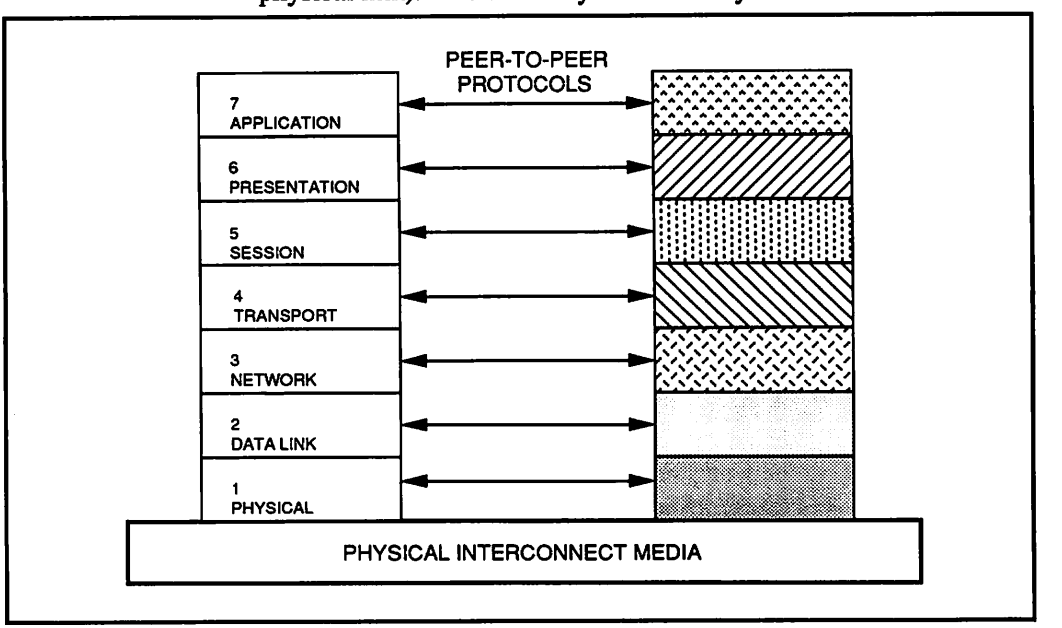
The IBM PC Local Area Network Program (formerly the IBM PC Network Program) is an example "application" that relies on NETBIOS for its operation. It implements the Server Message Block (SMB) protocol (discussed in Chapter 4). The PC LAN Program provides the user with workstation functions (redirector, receiver, and messenger) and non-dedicated server functions (workstation functions plus server). The redirector (which was written by Microsoft and included by IBM in the PC LAN Program) intercepts requests from DOS to determine if the request is for a remote resource. The receiver allows a workstation to receive text messages from other users. The messenger allows one to send messages to other users. The server is a non-dedicated file server which allows a user to share

the hard disks and printers associated with his or her PC with others in the network.

OSI Reference Model

When talking about NETBIOS and its services, it is useful to refer to the International Organization for Standardization (ISO) reference model for Open Systems Interconnection (OSI). The model is an architecture for the interconnection of computing devices in non-homogeneous environments. It encompasses not only LANS, but other networks as well, such as the wide-area X.25-type networks (e.g., ARPANET, Telnet) and short-haul mainframe networks. Most standards committees, including IEEE and ISO, have constructed the specific syntax and semantics of protocols for the implementation of the various OSI-defined layers.

The OSI model consists of seven layers of protocol as depicted in Figure 1-3. Each layer provides a service to the layer directly above it, and relies on the services of the layer directly below it. (The exception to this rule is, of course, the lowest layer of the model, the physical link). The seven layers are briefly defined below.



1-3: OSI Reference Model

Layer 1 is the physical link layer. It consists of the bit stream of data as sent and received from the network. Included is the raw transmission rate (2.5 Mbps on PC Network, 4 Mbps on Token-Ring) and the

encoding scheme (both networks use Manchester encoding, PC Network over a broadband channel, Token-Ring on baseband).

Layer 2 is the data link layer. This layer defines the meaning or structure of the bit stream. The packet format which Layer 2 describes is transmitted over the network using the services of Layer 1. The Token-Ring uses the IEEE 802.2 Logical Link Control (LLC) and IEEE 802.5 Data Link Control (DLC) protocols.

Layer 3 defines the network layer. The network layer is the lowermost layer of NETBIOS. It is responsible for the routing and switching of data throughout the LAN and interconnected LANs. It must recognize network addresses and be able to route information (packets) to the appropriate networks or pass it up to the transport layer for further interpretation.

Layer 3 is one of the weak spots in NETBIOS. NETBIOS was not designed with internetworking in mind, thus it lacks features needed to support this function efficiently. For example, internetworking between the PC Network and Token-Ring (as provided by the IBM Interconnect Program) is done by an application that resides in a PC as a gateway, and passes names between systems and packets between sessions (in a "store and forward" manner). It is interesting to note that the Interconnect Program can only internet two LANs.

Layer 4 defines the transport layer of the OSI model. The transport layer is responsible for the reliable transfer of information between stations on the network, and it implements features such as acknowledgments, sequence numbers, and time-outs. Again, in the original PC Network (broadband), NETBIOS uses protocols proprietary to Sytek for the implementation of the transport protocols.

Layer 5 is the session layer. The interface between NETBIOS and the host PC occurs at this level. The session layer supports naming and establishes sessions or logical links between two names on the network, or even two names within the PC. Like the network and transport layers, the session layer in NETBIOS is a proprietary implementation. The interface to NETBIOS is public domain information, however, and many vendors, such as Banyan, Novell, and 3Com, offer NETBIOS emulators for their networks. As long as the interface to the host remains identical, any protocols can be used for the actual peer-to-peer protocols within the layers.

Layer 6 is the presentation layer of the OSI model. It is not part of NETBIOS. The presentation layer is responsible for the negotiation of syntax that will be in use when passing information to and from

the application layer (Layer 7). Included are character formats such as EBCDIC vs. ASCII, and other formats for number representation or file formats. Layer 7 may have to do conversion if the application layer format is not compatible with the application layer format of another PC or service on the network.

The presentation layer in a PC Network or PC Token-Ring is virtually non-existent. To some extent, PC-DOS is part of the presentation layer since it is the format that is used to communicate with applications. PC-DOS cannot, however, recognize the format of any other files or characters besides its own.

Layer 7 is the application layer. It is responsible for presenting services to the end user. Example applications are the IBM PC Local Area Network Program and the IBM PC Interconnect Program, as discussed briefly above. The IBM PC LAN program relies on PC-DOS 3.1 (or higher) and NETBIOS for its operation and provides network file and print services to the end user.

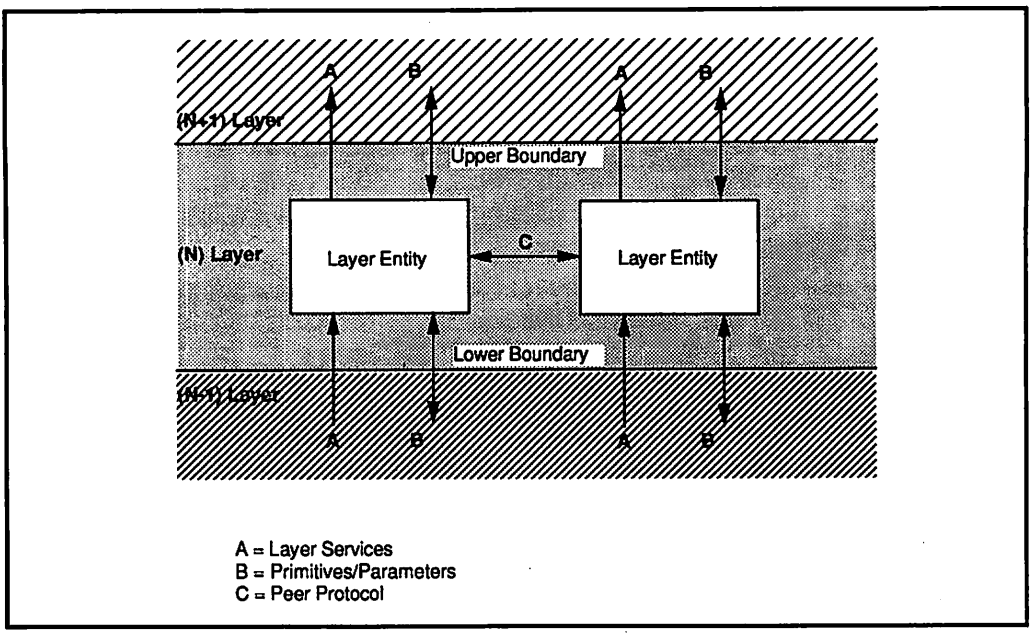
Layer Communication

In the OSI model, each layer communicates by passing its data to the layer above or beneath it; the message header is appended, and then the data packet is passed to the next layer. This process continues until it reaches the physical layer; at that point, the entire encapsulated packet is sent out through the network in the form of a bit stream. When the bit stream reaches the receiving node, it becomes de-encapsulated at the data link level. If the data link level recognizes the data packet, and if the address is correct, it will pass the packet up to the next level. This process continues until the data packet reaches the application level. Although it is possible to send thousands of bytes of data at a time through the physical and data link levels, the actual usable data that is sent is small due to encapsulation and appending of headers.

Layer Interaction

In Figure 1-4, layer "N" interacts with layers "N - 1" and "N + 1" through primitives or parameters that pass back and forth between the layers. Each layer provides a service for the layer above it. Protocols emphasize peer-to-peer level communication within a layer of a protocol: one layer entity communicates with another layer entity.

A level of protocol passes a packet to the level beneath it until it goes through the network and the receiving node. Layer "N" only knows what is happening in layers "N - 1" and "N + 1." It is not aware of what is happening in the other levels of protocol. This means that system implementors can easily change levels of protocol to adapt to



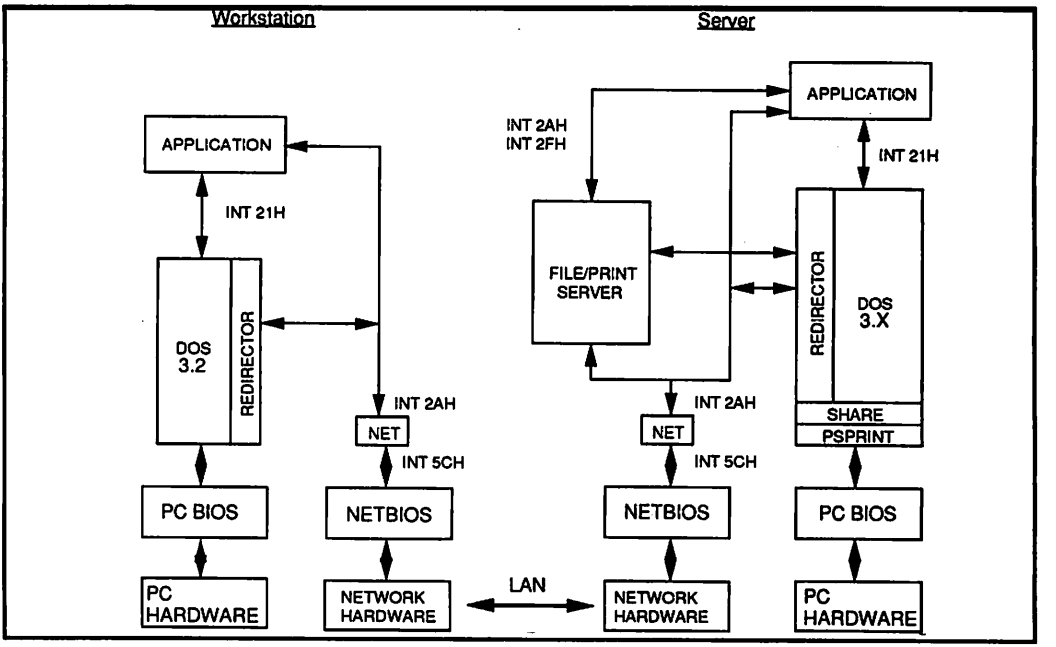
1-4: OSI Layer Interaction

new standards and new protocols as they evolve with a minimum impact on the system.

Relationship to PC-DOS and Applications

Sharing information in an IBM NETBIOS-based LAN requires three important pieces of systems software: 1) PC-DOS (3.1 or higher); 2) NETBIOS; and 3) the redirector. Figure 1-5 illustrates the manner in which PC-DOS, NETBIOS, and redirector fit into a system. NET, accessible from an application or the redirector via interrupt 2AH, is part of the IBM PC Local Area Network Program. A complete implementation of the PC Network Program is given on the right side of the figure; the file and print server is shown executing as an application in the background.

An application can do one of three operations that concerns the network: a user application (such as a word processor) will call DOS and have the redirector send I/O to/from a server via the NET program; a "multi-user" word processor will use the extended DOS calls to lock/unlock files; a specialized server application will call NETBIOS directly using the interrupt 5CH. A fourth option is for an application to directly call the file/print server, if it is implemented (via the PC LAN Program), using the interrupts 2AH or 2FH.



1-5: NETBIOS/DOS Service

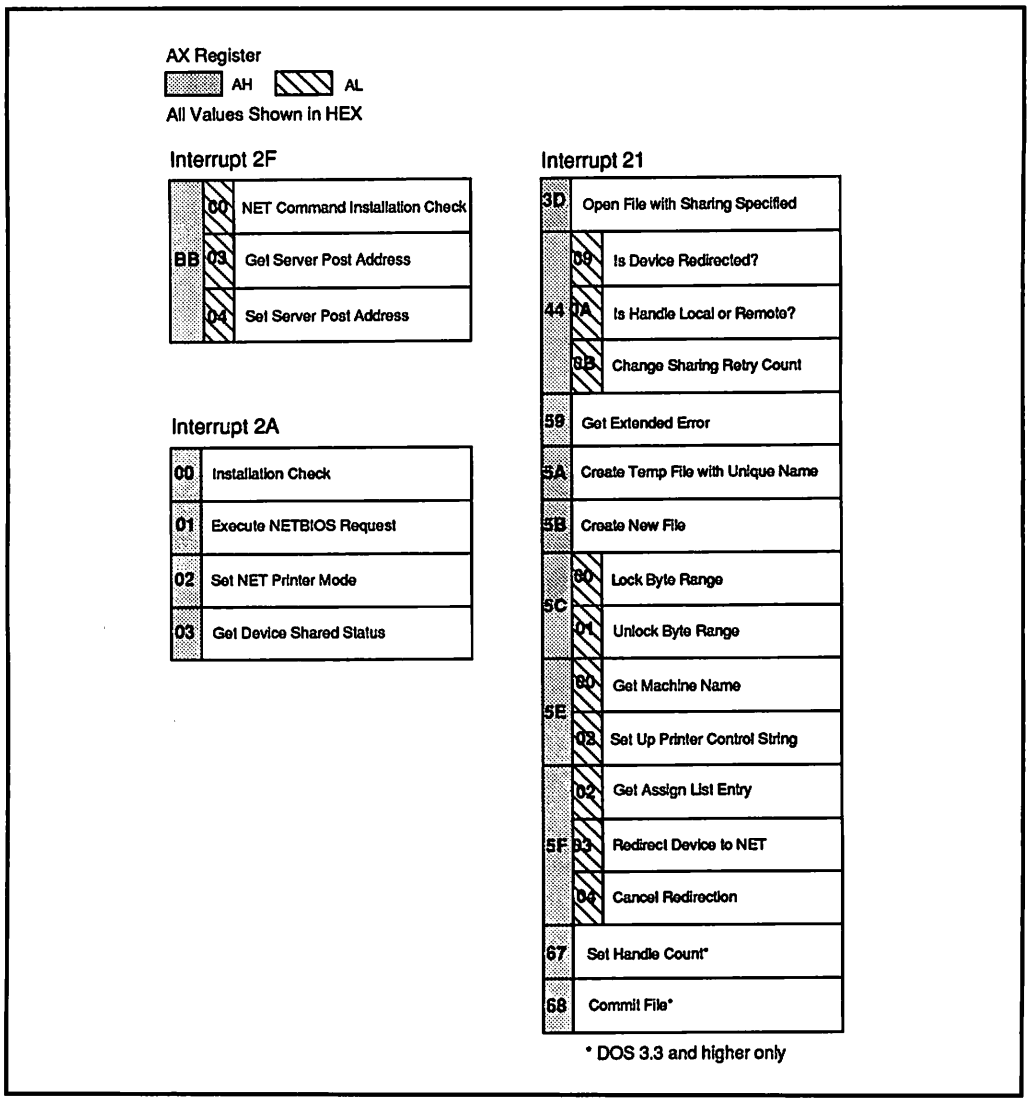
Other vendor implementations vary from IBM in the area of the redirector and file/print services. For example, Novell's shell is analogous to the redirector and also handles file/print requests from DOS.

A summary of the functions provided by interrupts 2FH, 21H, and 2AH are given in Figure 1-6. Later chapters discuss the 5CH NETBIOS interrupt in greater detail.

NETBIOS Implementations

It is important to stress that the NETBIOS interface option available for the Token-Ring is an emulation of the NETBIOS interface contained within the original PC Network Adapter card. Therefore, while the actual way in which the NETBIOS commands are carried out may differ between the Token-Ring (or other networks for that matter) and the PC Network, what the user or programmer sees is virtually identical interfaces and operation.

On the Token-Ring, the host processor handles NETBIOS, whereas on the IBM PC Network, an on-board 80188 processor does the NETBIOS processing. Interestingly, NETBIOS tests on both networks have shown the Token-Ring implementation to operate more than a factor of two better than PC Network in terms of raw data rate. This



1-6: Interrupt Functions 2FH, 21H, and 2AH

is attributable to overhead between the four microprocessors on the PC Network Adapter and the way in which the original NETBIOS interface is programmed plus the fact that the Token-Ring does not have the overhead of network and transport layers for NETBIOS to operate on top of.

The host PC communicates with NETBIOS via the Network Control Block (NCB -- also called the Message Control Block or MCB in the IBM Token-Ring Network PC Adapter technical reference manual -- Chapter 2 details the format and operation of the NCB.) Once the NCB is set up by the host application, it interrupts NETBIOS for service. NETBIOS then takes over and invokes the service requested by the host.

The data link layer provides the link access protocol (called LAP in PC Network terminology) to PC Network or Token-Ring. This is where a major discrepancy exists between the two networks in terms of NETBIOS implementation. The Token-Ring provides the IEEE standard 802.2 Data Link Control (DLC) and 802.5 Media Access Control (MAC) -- NETBIOS calls are directly translated into 802.2 and 802.5 frames bypassing any network or transport layer protocols.

IBM PC Network provides a proprietary DLC and Ethernet-like Media Access Control (MAC) protocol. The PC Network link Access Control Protocol (LAP) provides service for the packet transfer protocol (PTP).

PTP is a weakness of the original PC Network in that the "routing" function is a simplistic name-mapping scheme. PTP does not have the facilities to implement internetworking, making gateways between two networks difficult to design and limited in functionality. For example, the original PC Network to Token-Ring Interconnect Program can link only two networks together with a maximum of just 16 "services" between them. Using a newer version of the program with newer PC Network Broadband adapters, this number has improved somewhat.

The PC Network RSP resides at the transport layer. It provides error-free virtual connection services with other stations through end-to-end acknowledgments and retransmissions. RSP provides transport layer services to the session management protocol (SMP). PC Network Datagram transport protocol (DTP) also resides at this level. It provides unacknowledged datagram services between session layer entities, including the user datagram protocol (UDP) and the diagnostic and monitoring protocol (DMP).

The session layer provides host access to several protocols. The PC Network session management protocol (SMP) provides support for user sessions between nodes. SMP allows users to establish connection to a named process. It is responsible for interacting with the

name management protocol (NMP) within the local node to determine the address of the named process. Once the initiating SMP determines the destination node, it can communicate with the SMP within the destination node to provide session level services.

Along with naming, the PC Network user datagram protocol (UDP) provides support for datagrams between two names (nodes). The PC Network name management protocol (NMP) provides the binding of alias names and network addresses within the entire local network. NMP provides all name management services, including the translation of remote names to network addresses. This part of the protocol is one reason that it takes so long to become part of a NETBIOS network when starting up -- the node will broadcast its name several times to make sure all other stations on the name "receive" it. This also occurs when SMP has to establish a connection with another name.

One of the more interesting protocols provided by the PC Network is the diagnostic and monitoring protocol (DMP). DMP allows the collection of diagnostic and status information. DMP can query other adapter cards via the network to find out their state.

NETBIOS Versions

Versions of NETBIOS before the IBM LAN Support Program became available are denoted as Version numbers 1.X. The LAN Support Program 1.00 has NETBIOS version 2.0; 1.01 has NETBIOS version 2.1; and 1.02 has NETBIOS 2.2. This is important to note -- later we will see additional commands and features added to later releases.

NETBIOS or APPC/PC?

Like most higher-level protocols, NETBIOS is hardware-independent, which makes it useful across a variety of systems. The future of NETBIOS for heterogenous systems is uncertain, however. IBM introduced APPC/PC for the Token-Ring, which has a much broader appeal over the entire line of IBM computers and communications equipment. APPC is of great importance to developers of applications for IBM computers, since IBM offers it for every major computer line it sells and is actively promoting it as an "open" interface. It is also a strategic component of IBM's Systems Application Architecture (SAA); NETBIOS is not. The communication manager component of IBM's OS/2 Extended Edition for the Personal System/2, includes not only NETBIOS, but APPC/PC as well.

NETBIOS does have the advantage of having been established in hundreds of PC LANs, while APPC/PC is still "catching up". Another edge NETBIOS has over APPC is that APPC contains com-

plex SNA protocols which consume much of the system resources (RAM + CPU). One wants to minimize this overhead in a personal computer. This may become less of an issue however, as the more powerful PS/2 and 80386 PCs are in widespread use.

Developing applications to NETBIOS on the Token-Ring requires an interface with PC-style servers and gateways (see Chapter 5). However, it also means those applications probably won't port to other larger systems which IBM supports on the Token-Ring such as System/36 and the 9370. APPC is a safer bet for developing long term, distributed processing applications, while NETBIOS is certainly a safe bet of PC LAN only applications. This could change if IBM develops a NETBIOS emulator for non-PC computers, which appears unlikely.

According to IBM, the destiny of APPC is to become "one architecture for general purpose program-to-program communication,"and designed for "any-to-any" connectivity. Furthermore, the design of APPC is such that it has a low entry cost (ala APPC/PC) and to be highly functional with strong subset control. APPC is IBM's foundation for developing distributed applications.

The bottom line may be this: use NETBIOS to support traditional PC applications and APPC/PC for PC-to-mainframe and peer-to-peer applications; or NETBIOS for PC only LANs and APPC/PC for mixed environments that need to support IBM host applications such as those that fall under the IBM SAA umbrella.

Chapter 2

Programming

General Procedure

Applications that require direct sending and receiving of messages between stations are prime candidates for using NETBIOS. A example of such an application would be a communications server that provides LAN-wide access to its RS-232 ports.

To use NETBIOS, a station name is first added to the local name table. A station name is a unique name by which a given station is known on the network. Alternatively, one can simply use the permanent node address (a unique 48-bit address in ROM assigned to that adapter), in which case one does not have to enter a "name". However, assigning a phonetic name makes a station's name more meaningful.

After the station's name is set, an application may establish a session with another name on the network. The name may even exist in the name table of the station, in which case a "local" session is established. Usually, however, sessions will be established with "remote" names, such as a server session on another machine.

Once the session is established, messages can be sent and received over that logical link. The session provides reliable data transfer in which all messages are acknowledged; otherwise, one can opt for datagram transfer, which means NETBIOS will send the message directly to the data link layer and not be concerned with acknowledgments. Datagram service is useful for broadcasting messages.

Using NETBIOS

Before an application can use NETBIOS, it must be installed. In the original PC Network, NETBIOS is in ROM on the adapter card. In the Token-Ring (and most other LANs), NETBIOS is loaded at the PC-DOS command prompt by executing a program that remains memory resident (such as IBM's older NETBEUI -- NETBIOS Extended User Interface) or as a device driver such as IBM's PC LAN Support Program. In addition to Token-Ring, IBM's PC LAN Support Program also works with the newer version of PC Network broadband and PC Network baseband adapters. We will first examine NETBEUI followed by the PC LAN Support Program.

NETBEUI

The format to load IBM's NETBEUI is: NETBEUI work0, SAP0, stn0, work1, SAP1, stn1. The parameters have the following meaning:

work0/work1:

The amount of PC work storage (RAM) assigned to an adapter, from 1Kb to 18Kb. 18Kb is recommended for a PC that will also function as a server. Since NETBEUI supports two adapters, work0 and work1 are work areas for the respective adapters. A restriction is that work0 plus work1 must be fewer than 18Kb. If this parameter is missing, then 9Kb is used by default.

SAP0/SAP1:

Service access point (a reference point for connection between two nodes at the logical link -- or Layer 2 -- level for adapter 0/adapter 1. These are the additional SAPs requested on an implicit OPEN. Up to 9 additional SAPs may be in supporting non-NETBIOS applications. The default is 0.

stn0/stn1:

The number of additional link stations (up to 9) requested on an implicit OPEN. It equates stn to the number of additional non-NETBIOS remote SAPs that other applications might expect to communicate with at any one time.

NETBEUI occupies roughly 46Kb of RAM in the host PC's memory.

Device Driver

NETBIOS is configured into the IBM PC LAN Support Program as a device driver. Applications written to NETBIOS for PC Network will work "as is" with the Token-Ring emulator. (Refer to Figure 1-2 for the implementation of NETBIOS for PC Network and the

Token-Ring.) The device driver occupies roughly 23Kb of RAM in its default configuration.

With the PC LAN Support Program, NETBIOS has more parameters than with earlier releases. Support for earlier releases of NETBIOS parameters are provided such that all previous parameters may be used (as a migration aid).

One of the new options provides support for high-speed asynchronous adapters -- parameters called ENABLE that allocates less time to NETBIOS and more time for asynchronous adapter operation.

The table in Figure 2-1 is a summary of the available parameters. The parameters are supplied via DEVICE=DXMT0MOD.SYS in the CONFIG.SYS file and are not position dependent (like the old NETBEUI). An example would be DEVICE=DXMT0MOD.SYS ST=50 N=40 S=30 to support 50 stations, 40 names, and 30 sessions.

Keyword	ABBR	Valid Values	Minimum	Default
Stations	ST	0 - 254 *	1	6
Sessions	S	0 - 254	1	6
Commands	C	0 - 255	1	12
Names	N	0 - 254	2	17
Open.On.Load	O	Yes/No	-	Yes
Datagram.Max	DG	Yes/No	-	No
Close.On.Reset	CR	Yes/No	-	No
DHB.Size	DS	200 - 9999 *	200	**
DHB.Number	DN	0 - 9 *	-	**
Receive.Buffer.Size	R	0 - 9999 *	***	**
Transmit Timeout	TT	0 - 20	0	2
Transmit.Count	TC	0 - 1-	0	3
DLC.Maxout	MO	0 - 9	0	2
DLC.Maxin	MI	0 - 9	0	1
Ring.Access	RA	0 - 7	0	0
Extra.Saps	ES	0 - 99 *	0	0
Extra.Stations	EST	0 - 99 *	0	0

* Making this value too large will cause a failed adapter open.

** If default value is used, the NETBIOS Driver sets the values of these items based on adapter resources.

*** Minimum is set by adapter on open, not NETBIOS.

2-1: NETBIOS Device Driver Parameters

Programming NETBIOS

The basic procedure for using a NETBIOS service is as follows:

- Set up and fill in the Network Control Block (NCB)
- Allocate any buffers required by the NETBIOS service about to be invoked
- Set the ES:BX register pair to point to the NCB
- Issue the 5C interrupt for direct NETBIOS or issue interrupt 2A for DOS 3.1 or higher to call NETBIOS (see the DOS technical reference manual for more details)
- Examine the result in the AL register or return code field of the NCB

The Network Control Block

The following details the interpretation and operation of each field in the NCB (refer to Figure 2-2).

NCB_COMMAND

This field contains the actually NETBIOS command to be executed. When an application issues commands to NETBIOS, it may choose

Field Name	Length (in bytes) and Meaning	
NCB_COMMAND	1	- NCB COMMAND FIELD
NCB_RETCODE	1	- NCB RETURN CODE FIELD
NCB_LSN	1	- NCB LOCAL SESSION NUMBER FIELD
NCB_NUM	1	- NCB NUMBER OF YOUR NAME
NCB_BUFFER@	4	- NCB POINTER TO MESSAGE BUFFER ADDRESS (OFFSET:SEGMENT)
NCB_LENGTH	2	- NCB BUFFER LENGTH (IN BYTES)
NCB_CALLNAME	16	- NCB NAME ON LOCAL OR REMOTE ADAPTER - FOR CHAIN SEND THE FIRST 2 BYTES INDICATES LENGTH OF SECOND BUFFER THE NEXT 4 BYTES INDICATES THE BUFFER ADDRESS SECOND
NCB_NAME	16	- NCB NAME ON LOCAL ADAPTER
NCB_RTO	1	- NCB RECEIVE TIMEOUT VALUE
NCB_STO	1	- NCB SEND TIMEOUT VALUE
NCB_POST@	4	- NCB POINTER TO POST ROUTINE (OFFSET:SEGMENT)
NCB_LANA_NUM	1	- NCB ADAPTER NUMBER; 00H FOR FIRST ADAPTER, 01H FOR SECOND ADAPTER
NCB_CMD_CPLT	1	- NCB COMMAND STATUS FIELD
NCB_RESERVE	14	- NCB RESERVED AREA

2-2: Network Control Block

to wait for them to be completed or be interrupted upon completion.

The application may set the high-order bit of the command to 1 for no-wait, or 0 for wait. With the wait operation, control is not returned to the next instruction until NETBIOS completes the command. The calling routine must then check the AL register or NCB_RETCODE field for the status of the completed command. The preferable approach is the no-wait option, since NETBIOS executes as a background task, and multiple commands may be queued up. Control is returned to the next instruction of the application, with a return code in AL.

The possible return codes are: 00H - successful command completion; 03H - command invalid; 21H - interface is busy; 22H - too many commands queued; 23H - invalid NCB_LANA_NUM field; 24H - command completed while a cancel is occurring; 26H - command cannot be cancelled; 4XH - "unusual network condition"; 50-FEH - adapter has malfunctioned. Return code values of 40H - 4FH are unique to the Token-Ring implementation of NETBIOS.

The application can choose to be interrupted upon a 00H (OK) return code, or "poll" the NCB_CMD_CPLT field (initially set to FFH while the command is executing). If the interrupt option is chosen, then the NCB_POST@ field must be set (non-zero). If interrupted, the application can check AL or NCB_RETCODE for the final return code from NETBIOS.

NCB_RETCODE

NETBIOS mirrors the return code in the AL register. See above for possible return code values. If the return code is not 00H, then the application should take appropriate error recovery action.

NCB_LSN

After doing a CALL or LISTEN command, this field shows the assigned local session number. This field must be set when issuing a SEND or RECEIVE command for that session. NETBIOS assigns the number sequentially, wrapping at 254 back to 1 (255 or FFH and 0 are never used).

NCB_NUM

The number associated with a name. NETBIOS returns it after the application requests an ADD NAME or ADD GROUP NAME. As with NCB_LSN, NETBIOS assigns the number sequentially, wrap-

ping at 254 back to 1. This number must be in use when sending datagrams and for RECEIVE ANY.

NCB_BUFFER@

If required by the command (such as SEND), NCB_BUFFER@ is a 4- byte pointer that denotes the offset:segment address of the buffer that the command uses.

NCB_LENGTH

Length, in bytes of the buffer (pointed to by NCB_BUFFER). For the SEND commands, it is the actual number of bytes to be sent. For receive commands, NETBIOS sets it to the actual number of bytes received.

NCB_CALLNAME

The 16-byte name of the session with which one is communicating. When doing a chain send, the first 2 bytes specify the length, and the next 4 bytes specify the buffer address, in the same format as NCB_BUFFER@.

NCB_NAME

The station user's 16-byte name. If the permanent node address is in use, then the first 10 bytes are set to 0, followed by the 48-bit (6 byte) node address.

NCB_RTO

Specifies the receive time-out in 500 ms increments before a RECEIVE command times-out. A value of 0 is no time-out. NCB_RTO is set once a session is established.

NCB_STO

Like NCB_RTO, except for SEND commands.

NCB_POST@

The offset:segment address of the post-processing routine that executes after NETBIOS has completed a no-wait command. The standard interrupt return instruction, IRET, is used upon completion of the post-processing routine. If NCB_POST@ is 0, then NETBIOS does not invoke the routine, and the application has to poll the NCB_CMD_CPLT field for a non-FFH value.

NCB_LANA_NUM

Used to show which adapter the command is for. A 00H is for the first adapter, and a 01H is for the second adapter.

NCB_CMD_CPLT

A value of FFH shows that the command has not yet be executed. A value of 00H shows completion. As noted above, a non-zero value indicates an error.

NCB_RESERVE

A 14-byte reserved space, used partially by the Token-Ring implementation of NETBIOS.

NETBIOS Commands

The commands which NETBIOS carries out can be broken up into four groups:

- general
- name support
- session support
- datagram support

General commands are non-communicating commands that deal primarily with the adapter itself.

Name support commands allow applications to associate resources and services with logical names instead of discrete addresses.

Session commands allow an application to establish a reliable link between two names for communicating information. Again, keep in mind that a session link may be within an adapter or with a name on a remote adapter.

Datagram commands allow the broadcasting of short (fewer than 5 byte) messages and the sending of unacknowledged messages to another name.

The basic operation for exchanging information between two names on the network is as follows: 1) add names to the respective application's local name table at that PC; 2) establish a session between the two names using the CALL and LISTEN commands; 3) transfer data using the SEND and RECEIVE commands; 4) end the session using the HANG UP or RESET commands. Figure 2-3 illustrates a generic timing of packet exchanges.

Value (in Hex)	Meaning
00H	Good return; command complete.
01H	Invalid buffer length for SEND DATAGRAM, SEND BROADCAST, ADAPTER STATUS, OR SESSION STATUS.
03H	Illegal command code.
05H	Command time-out period has expired.
06H	Message received was partial since receive buffer was not large enough.
08H	Session number specified is not active.
09H	Not enough space available in adapter for session.
0AH	Session is closed.
0BH	Command was cancelled.
0DH	Duplicate name in local name table.
0EH	Local name table is full.
0FH	Name to be deleted is active in a session.
11H	Local session table is full.
12H	Session open was rejected since no LISTEN is outstanding on remote computer.
13H	Illegal name number.
14H	Cannot find name called or there is no answer.
15H	Name not found in local table.
16H	Name is in use elsewhere.
17H	Name deleted with no outstanding commands for that name.
18H	Session ended abnormally.
19H	NETBIOS has detected two or more identical names in use on network!
1AH	Incompatible packet protocol received.
21H	Interface is busy.
22H	Too many commands are outstanding.
23H	Bad number in NCB_LANA_NUM field.
24H	Command completed before cancel request or command never existed.
26H	Command is illegal to cancel.
4XH	Undeterminable network error.
50-FEH	Adapter has malfunctioned.
FFH	Command is still pending.

2-3: NETBIOS Error Return Codes

A summary of the possible commands follow. Note that besides the fields required by each command as outlined here, the NCB_COMMAND field must be set to the appropriate number as shown (in hex notation). The adapter number (0 or 1) must be selected by setting NCB_LANA_NUM, and the NETBIOS commands return a result in the NCB_RETCODE field. An application may request that a command be executed by NETBIOS in the background (indicated below

by the word return) or to wait until NETBIOS completes the option (indicated by the word wait). Some commands do not have this option and are considered to be a "wait until completion" command.

The format is list below is Command/NCB_COMMAND (in hex -- both no-wait and wait are listed if applicable) with an explanation of the command.

General Commands

RESET/32H. Resets local adapter status and clears name and session tables.

By resetting the adapter, an application can change the number of sessions and the number of NCB command blocks supported by NETBIOS. The default power-on for PC Network values are 6 and 12, respectively. These numbers must be carefully set or else performance may suffer, since more sessions and NCB command blocks could mean smaller packet sizes, depending on how much adapter memory there is.

The only NCB fields (other than NCB_COMMAND and NCB_LANA_NUM) required for RESET are NCB_LSN, NCB_NUM.

CANCEL/35H. Requests that the pending command whose NCB is found at NCB_BUFFER@ be cancelled.

One can cancel any pending NETBIOS command except ADD [GROUP] NAME, DELETE NAME, SEND DATAGRAM, SEND BROADCAST DATAGRAM, SESSION STATUS, CANCEL, AND RESET. Cancelling a SEND command will also drop the session. NCB_BUFFER@ is required (the buffer to be canceled).

STATUS/33H (wait) B3H (return). Gives status information for local or remote adapter.

This command performs diagnostics on local and remote adapters, even if the remote PC cannot communicate properly with its adapter, or is in a "hung" state. Fields required include NCB_BUFFER@, NCB_LENGTH, NCB_CALLNAME, and NCB_POST@ (for no-wait option only).

The information returned in the buffer for PC Network includes the 6-byte permanent node address, the 1-byte status of external jumpers on the network adapter card, the 1-byte results of the last self-test, 2 bytes containing the software revision number, 48 bytes of traffic and

error statistics, 26 bytes of adapter resource statistics, 2 bytes for the number of names in the local table, and 16 entries of 18 bytes each for the local name table.

The traffic and error statistics returned for PC Network include 2 bytes for the reporting period (in minutes), 2 bytes for the number of CRC errors, 2 bytes for the number of alignment errors, 2 bytes for the number of collisions, 2 bytes for the number of aborted transmissions, 4 bytes for the number of successfully transmitted packets, 4 bytes for the number of successfully received packets, 2 bytes for the number of retransmissions, and 2 bytes for the number of times the receiver has exhausted its resources.

The adapter resource statistics returned for PC Network include 8 bytes of reserved space, 2 bytes for the number of free command blocks, 2 bytes for the maximum configured NCBs, 2 bytes for the maximum free command blocks, 4 bytes reserved, 2 bytes for the number of pending sessions, 2 bytes for the maximum pending sessions, 2 bytes for the total maximum number of sessions, and 2 bytes for the maximum session data packet size.

TRACE/79H (wait) F9H (return). Token-Ring only. TRACE begins a trace of all MCB commands and some of the control block (CCB) commands issued by the NETBIOS program.

UNLINK/70H. Used with remote program load (RPL) to unlink session with IBMNETBOOT.

UNLINK is used only if a call to IBMNETBOOT was made at PC power up time; i.e. a remote boot was performed.

The session with IBMNETBOOT is dropped and the redirector (INT 13) is dropped.

Name Support Commands

Names allow an application and the PC upon which it runs to be known by other applications and PCs on the network. Names must be 16 bytes long and are entered into the local name table; the original PC Network broadband accommodates up to 16 names while the PC LAN Support Program accommodates 32 or more names. A name that is unique to a PC can also be part of a group (a group name). Recall that each station is always assigned a permanent node name (6 bytes of address followed by 10 zeros) by default. An application can retrieve this name by doing an ADAPTER STATUS command with an asterisk in the callname field. The first 6 bytes in the return

buffer show the address of the adapter. Both PC Network and Token-Ring use 6-byte node addresses.

ADD NAME/30H (wait) B0H (return). Adds a (unique) 16-character name to (return) name table.

NETBIOS does a broadcast to ensure that the name is unique. This command requires the NCB field NCB_POST@ if the no-wait option is used. Error codes are returned to indicate a full table, duplicate name, a name that is not unique, etc.

ADD GROUP NAME/36H (wait) B6H (return). Adds a group name to the name table.

NETBIOS does a broadcast to ensure that the name is not in use as a unique name on another PC. NCB fields and error conditions are the same as ADD NAME.

Since names can be up to 16 bytes in length, and the actual address size (at the data link layer) is only 6 bytes, NETBIOS will derive a group address for itself using one of two methods.

The first method requires applying the following function:

group_name = 0000 concatinated with (N1 xor N2 ... N5 xor N6) concatinated with FF

where N1 ... N5 are the first through fifth char fields of the name and N6 is the last char of the name.

The second method is to derive the group address from the permanent node name by applying the following function:

group_name = 0000 concat (ID3 ID2 ID1) concat FF

where ID3 ... ID1 are the three low-order bytes of the permanent node name.

These addresses derived by NETBIOS are not normally available to the application, but can be calculated using the formulas. The above formulas were chosen to minimize the chance that two different 16-byte names will "hash" down to the same 6-byte group address.

DELETE NAME/31H (wait) B1H (return). Deletes name from name table.

DELETE NAME removes a name entered by ADD NAME or ADD GROUP NAME from the local name table. DELETE NAME is typically done after ending a session with the HANG UP command (described below). If there are still active sessions, NETBIOS will

delay the delete until all active sessions are terminated. This command requires NCB_POST@ if the no-wait option is used.

Session Support Commands

The session commands form the heart of NETBIOS and are responsible for the actual passing of information (up to 65,535 bytes per request) over the network. The application uses sessions commands to establish a link between any two names on the network, or even within the PC itself. Note that names are used to initiate the process, but NETBIOS returns a number in the NCB_LSN field that is used from then on.

CALL/10H (wait) 90H (return). Opens a session with another name specified by the NCB_CALLNAME field.

CALL initiates a session with the name specified in the NCB_CALLNAME field, using the local name supplied by the NCB_NAME field. When CALLing another name, it must have already set up a LISTEN command. NETBIOS returns a session number in the NCB_LSN field. NCB fields required include NCB_RTO (receive time-out), NCB_STO (send time-out), and NCB_POST@ if the no-wait option is used.

LISTEN/11H (wait) 91H (return). Enables session establishment with the name specified in the NCB_CALLNAME field.

The complement of CALL, LISTEN allows session establishment with the name in the NCB_CALLNAME field and the name in the NCB_NAME field. NCB_CALLNAME can be set to an asterisk, in which case any name is accepted from a CALL. The name that initiated the CALL is then returned in the NCB_CALLNAME field. It is important to note that a LISTEN occupies a session entry. Fields required include NCB_NAME, NCB_RTO (receive time-out), NCB_STO (send time-out), and NCB_POST@ if the no-wait option is used.

HANG UP/12H (wait) 92H (return). Closes the session with another name.

HANG UP ends a session and all pending RECEIVE commands. This command requires NCB_POST@ for the no-wait option.

SEND/14H (wait) 94H (return). Sends data by the session number shown by the local session number (LSN).

SEND NO_ACK/71H (wait) F1H (return). Provides a SEND that does not require NO_ACK NETBIOS to transmit a data acknowledgement. Available only with NETBIOS version 2.2 and higher.

SEND transmits (reliably) up to a 65,535 byte buffer pointed to by NCB_BUFFER@ via the session shown by NCB_LSN. Multiple SEND commands can be queued. If the SEND cannot complete, the session terminates and must be re-established.

CHAIN SEND/17H (wait) 97H (return). Like SEND, except that data is taken from the buffers for the indicated number of bytes. Two buffers can be chained together.

CHAIN SEND NO_ACK/72H (wait) F2H (return). Provides a CHAIN SEND that does not require NETBIOS to transmit a data acknowledgement. Available only with NETBIOS version 2.2 and higher.

NETBIOS sends the buffers as one concatenated message with a size limit of 65,535 bytes. NCB_CALLNAME is used to specify the length (first 2 bytes) and address (next 4 bytes) of the second buffer. Fields required include NCB_BUFFER@, NCB_LENGTH, NCB_CALLNAME (0000H length format, 00000000H address format), and NCB_POST@ if the no-wait option is used.

RECEIVE/15H (wait) 95H (return). Receive data from a specified session. Time-out values can be specified.

RECEIVE sets up the adapter for receiving data from a specific session. If the received data exceeds the available buffer size, then a code of 06H is returned in the NCB_RETCODE field. Fields required include NCB_BUFFER@, NCB_LENGTH, and NCB_POST@ if the no- wait option is used.

RECEIVE ANY/16H (wait) 96H (return). Receive data from any station with which a session has been established. Like RECEIVE, except that it allows data to be received from any session. NCB_NUM (as returned from ADD NAME or ADD GROUP NAME) must be used instead of a name. Fields required are the same as RECEIVE.

SESSION STATUS/34H (wait) B4H (return). Obtain status of all active sessions for station name.

SESSION STATUS returns status information on all active sessions for a given local name (NCB_NAME) or all local names (if an asterisk is in the first byte of the NCB_NAME field). Fields required

include NCB_BUFFER@, NCB_LENGTH, AND NCB_POST@ if the no- wait option is used.

The format of the returned status information is as follows: 1 byte for the number of sessions being reported; 1 byte for the number of sessions with this name; 1 byte for the number of datagram commands pending; 1 byte for the number of RECEIVE ANY commands pending; 36 bytes for the information about a session, including 1 byte for the local session number, 1 byte for the state of the session (01H = LISTEN pending; 02H = CALL pending; 03H = session establish; 04H = HANG UP pending; 05H = HANG UP complete; 06H = session aborted); 16 bytes for the local name; 16 bytes for the remote name; 1 byte for the number of RECEIVE commands pending; and 1 byte for the number of SEND and CHAIN SEND commands pending.

Datagram Support Commands

The last group of NETBIOS commands is for datagrams. Datagrams allow the user to send unacknowledged messages of up to 512 bytes to a name or group name or broadcast to all names.

SEND DATAGRAM/20H (wait) A0H (return). Send a datagram to a unique name or group name at a local or remote node.

SEND DATAGRAM sends a datagram to a name or group name. That name must be set up for a RECEIVE DATAGRAM. Fields required include NCB_BUFFER@, NCB_LENGTH, NCB_NUM, and the optional NCB_POST@ for the no-wait.

SEND BROADCAST DATAGRAM/22H (wait) A2H (return). Send a message to everyone who has a RECEIVE BROADCAST DATAGRAM outstanding.

SEND BROADCAST DATAGRAM sends a broadcast message that any application picks up that has a RECEIVE BROADCAST DATAGRAM pending. Fields required are the same as SEND DATAGRAM.

RECEIVE DATAGRAM/21H (wait) A1H (return). Receive a datagram message from any name or anyone on the network.

RECEIVE DATAGRAM receives any datagram addressed to a local name or group name at that PC. The fields required are the same as SEND DATAGRAM. If NCB_NUM is set to FFH, then a datagram can be received from any name for any of the local names.

RECEIVE BROADCAST DATAGRAM/23H (wait) A3H (return). Receive a datagram from anyone who issues a SEND BROADCAST DATAGRAM command.

RECEIVE BROADCAST DATAGRAM receives any BROADCAST DATAGRAM. Fields required are the same as SEND DATAGRAM.

Figure 2-4 is a listing of the possible error codes returned by NETBIOS when the application uses the NCB and interrupt 5CH.

NETBIOS on Token-Ring

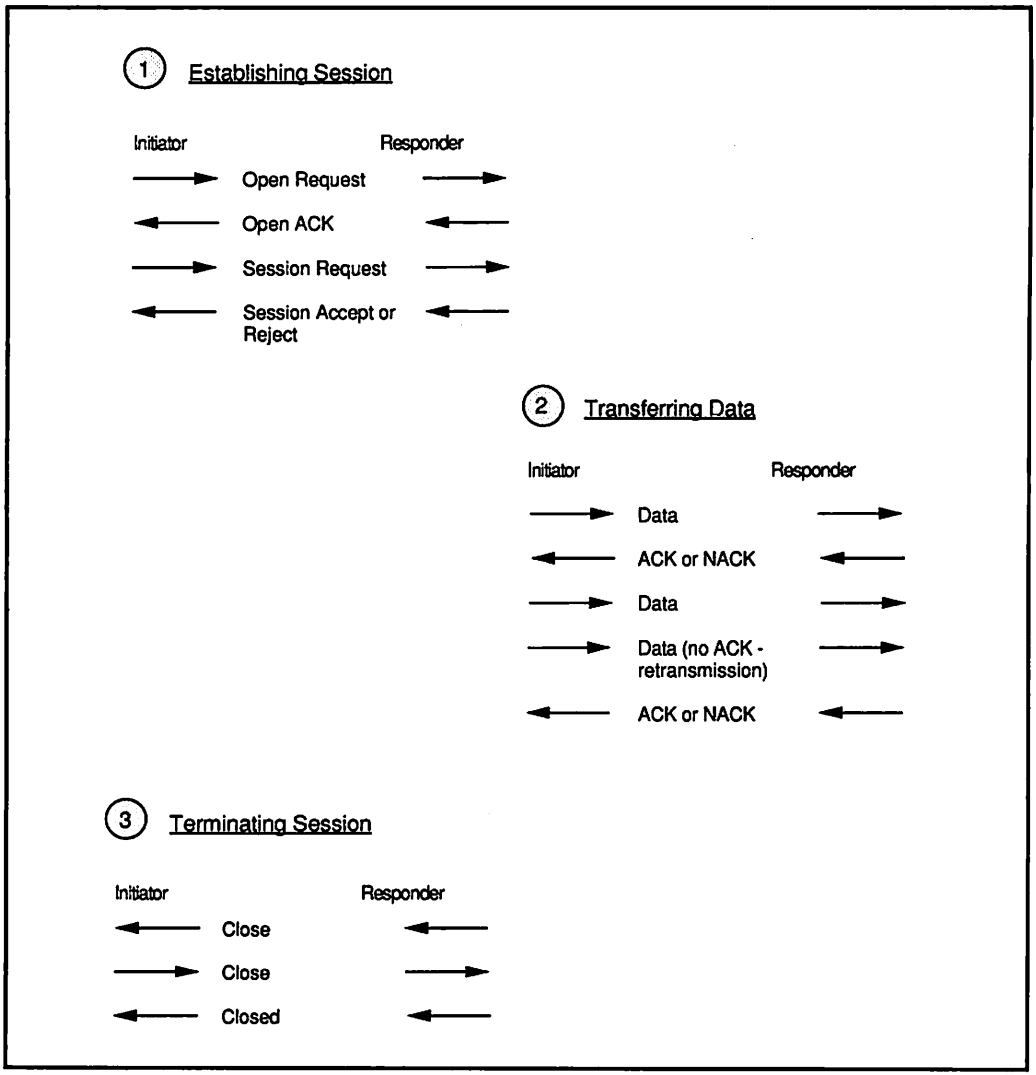
As noted in Chapter 1, NETBIOS on the original PC Network (broadband) adapter does not implement the standard 802.2 LLC or MAC. Therefore, on the Token-Ring, NETBIOS has been assigned an architected functional address of 00000080H to satisfy 802.2 requirements. With the NETBIOS program operational, all adapters with the functional address set will receive all frames destined for that address. The architected SAP value is F0H. Frames destined for DLC SAP F0H are routed to the NETBIOS program whether received through functional address or specific node address detection.

Token-Ring NETBIOS makes use of the broadcast capability of the ring. In all cases but one, frames are sent as "Limited Broadcast" -- that is, a bridge delivers only one frame to each ring in a multi-bridged ring network. The broadcast bit and the limited broadcast bit in the routing control field are both set to 1.

In the one case, the frame is sent as a "General Broadcast" -- that is, all bridges will forward the frame. The broadcast bit in the routing control field is set to 1 and the limited broadcast bit is set to 0.

The following are considerations of the NETBIOS Token-Ring. Initializing the Adapter handler can be done explicitly by an application in which an established shared RAM address is used and the error appendages (routines) are defined either by the application or implicitly by the NETBIOS program when a RESET or the first NCB is encountered. Here, shared RAM addresses D8000H/D4000H for adapters 00/01 will be used and NETBIOS defines error appendages.

OPEN CCB is an optional NETBIOS call used to define a set of NETBIOS program specific parameters. OPEN CCB can be done explicitly by an application, and must be issued before the first NCB and after NETBIOS is loaded. OPEN CCB can be done explicitly by a RESET or first NCB.



2-4: Generic Session Packet Timing

A typical initialize sequence would be as follows: the NETBIOS device driver issues to the Token-Ring handler DIR.INITIALIZE, DIR.OPEN.ADAPTER, DIR.STATUS, DLC.OPEN.SAP (with SAP set to F0H), DIR.SET.FUNCTIONAL.ADDRESS, DLC. MODIFY, and SET.TIMER.

The following is the sequence of NETBIOS events occurring when an application issues a NETBIOS command (via the NCB) to estab-

lish a session: the NETBIOS device driver issues to the Token-Ring handler DIR.SET.TIMER (for name recognized response), issues DIR.TRANSMIT.UI (broadcast NAME.QUERY), returns immediate return code if no-wait NCB, RECEIVE data response (name recognized), issues DIR.CANCEL.TIMER, issues DIR.FREE. BUFFER, issues DLC.OPEN.STATION (establish link station), issues DLC.CONNECT_STATION (connects the nodes), issues DIR.TRANSMIT.FRAME (send session initialize), RECEIVE data response (session confirmed), and returns final NCB return code.

Implementation Differences

There are some subtle differences in the original PC Network (broadband) NETBIOS vs. the PC LAN Support Program emulation. The Support Program implementation contains a few enhancements which should not be used if compatibility with the PC Network is desired. The following details the key differences.

NETBIOS	Original PC Network (broadband) NETBIOS
254 links per Adapter	16
254 sessions per Adapter	32
254 names	17
255 outstanding commands	32

The Token-Ring NETBIOS emulator adds additional NETBIOS Return Codes. When the following return codes are set in the NCB, the emulator will also return related status information in the NCB RESERVE field.

Extended Return Code	Meaning
4EH	Ring status not temporary
4FH	Permanent ring status error
F7H	Error on implicit INITIALIZE
F8H	Error on implicit OPEN
F9H	TOKREUI internal error
FAH	Adapter machine check
FBH	NETBIOS emulator not present
FCH	OPEN adapter or OPEN_SAP failed
FDH	Unexpected close of adapter

Both a PC Network adapter (either broadband or baseband) and Token-Ring Adapter may coexist in the same PC. If a PC Network adapter is present and operational, all NCB commands issued for that Adapter number are routed to it. If a PC Network Adapter is not present, then all NCB commands issued for that Adapter number are routed to the Token- Ring NETBIOS program. The user can jumper select which adapter is the primary (00) adapter and which adapter is the secondary (01).

Protocol Driver

The IBM PC Network Protocol Driver is another product introduced at the same time as the Personal System/2. The protocol driver provides NETBIOS emulation for the newer PC Network (broadband) adapters (both Micro Channel and PC), that were also introduced at the time of the PS/2, to maintain compatibility with the original PC Network adapter. However, the protocol driver will not communicate with the newer PC Network adapters operating the PC LAN Support Program. Another difference is the number of names and sessions supported -- up to 62 and 64 respectively.

A major drawback to the protocol driver is that unlike the PC LAN Support Program, it does not provide an IEEE 802.2 Logical Link Control (LLC) (OSI layer 2) interface. This means that certain non-NETBIOS software such as IBM's APPC/PC will not operate across a network with the protocol driver.

As we can see, the protocol driver is somewhat limited. It is only provided as a mechanism to preserve backward compatibility with NETBIOS in original PC Network Broadband installations.

Chapter 3

PC Network and Token-Ring Frame Formats

This chapter intends to give the reader a feel for what happens in an implementation of NETBIOS in terms of the protocols and packet formats used to carry the NETBIOS information between stations. Two networks are used as examples: the original IBM PC Network broadband and the IBM PC Token-Ring.

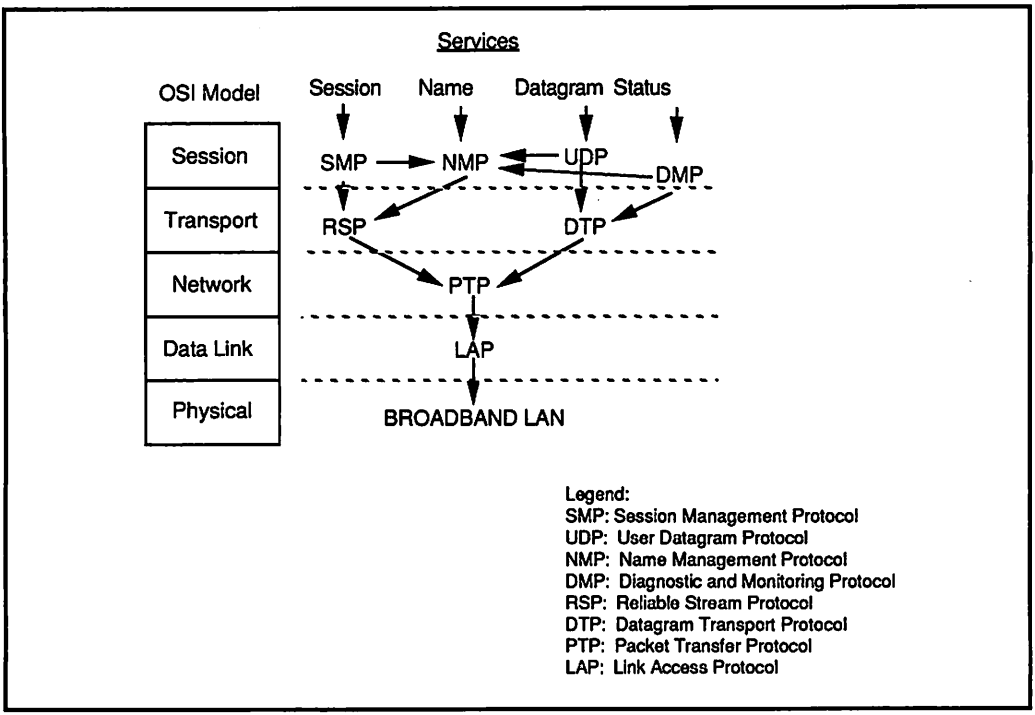
PC Network

Figure 3-1 illustrates the dependencies between the various protocols as implemented in the original IBM PC Network broadband adapter.

Session Layer Commands/Protocol Actions

The following section examines the actions taken by NETBIOS for various NCB commands. The commands and protocols are associated with the IBM PC Network Session Management Protocol (SMP). What is given here is a high-level description of the protocols used by SMP. The packet formats are also given when they are first referenced. The formats are the ones used by the original IBM PC Network broadband NETBIOS; they may be different for other implementations.

All packets received by NETBIOS have already gone through CRC checking and address recognition at the data link level. In the IBM PC Network broadband, this is done by the Intel 82586 CSMA con-



3-1: PC Network Protocol Relationships

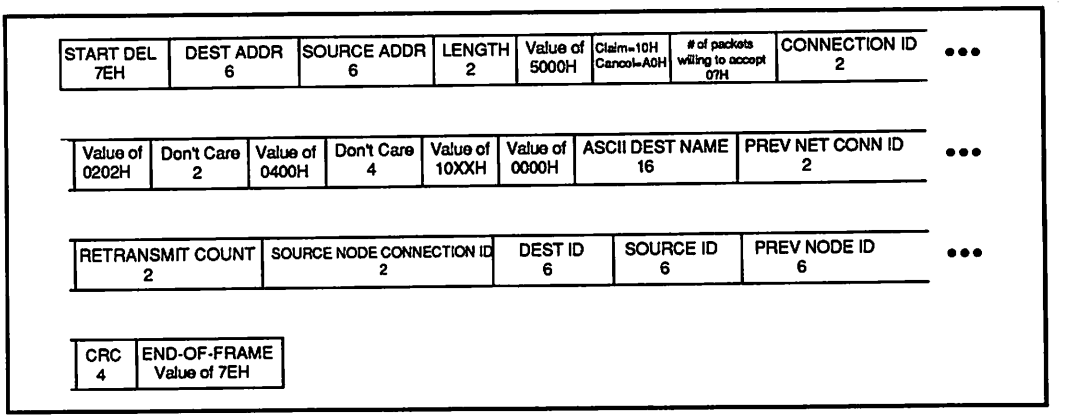
troller. With the Token-Ring, it is done by proprietary IBM protocol handlers.

ADD NAME

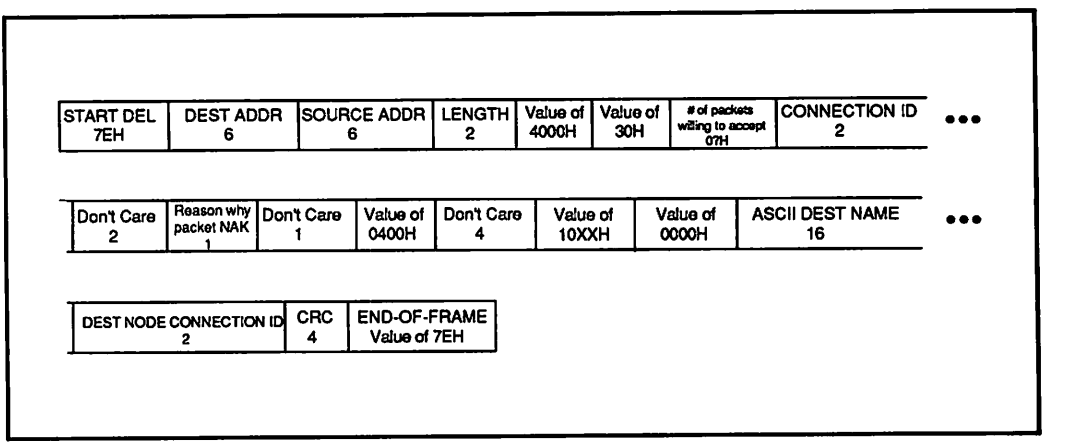
NETBIOS checks the name to make sure it is valid and continues if it is okay. If it doesn't find the name in the local name table, it broadcasts a "name claim" packet (Figure 3-2) several times to ensure that all stations see the request. If a response is received, the packet is in the form of Figure 3-3. If NETBIOS does not receive a response, then the name is added to the local name table.

DELETE NAME

As with ADD NAME, NETBIOS checks for a valid name and continues if it is okay. If it finds a non-active session associated with that name, it is terminated. Otherwise NETBIOS queues the delete request until the "session count" (active sessions) is zero, in which case the name is then removed from the name table.



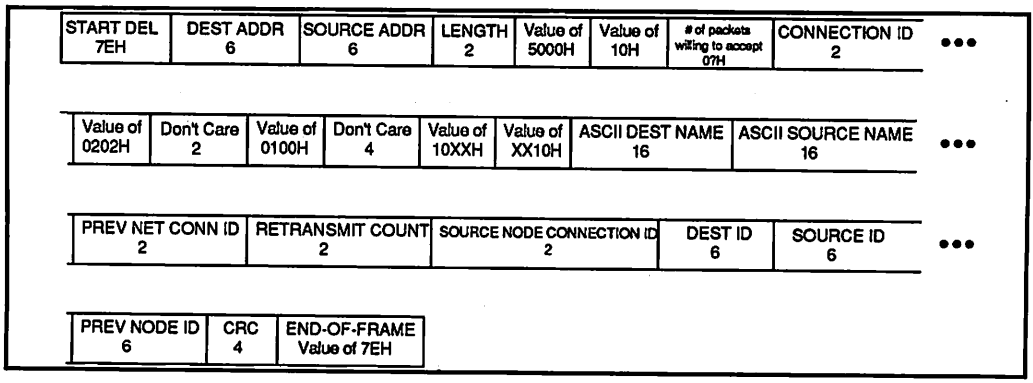
3-2: Name Claim/Name Cancel Packet



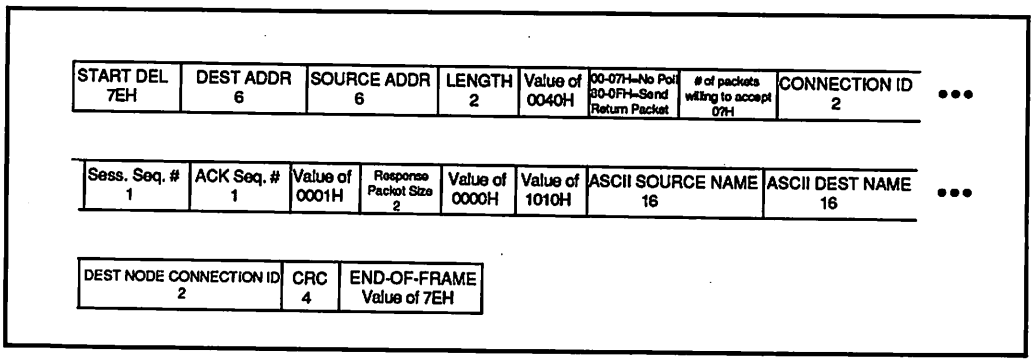
3-3: Name Claim Response Packet

CALL

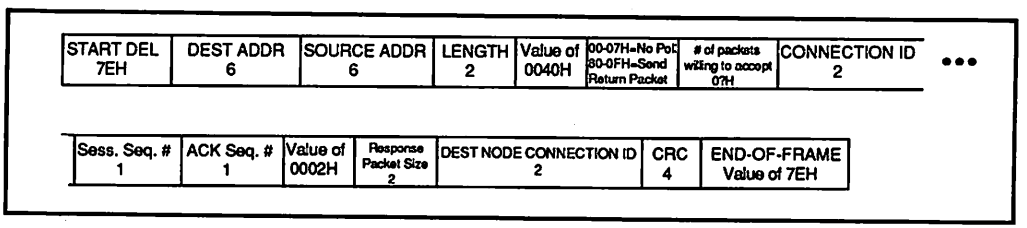
NETBIOS first checks to make sure that it finds the local name in the name table. If it is found, NETBIOS checks for the remote name in the name table, and if it is not found, broadcasts a "name query" packet (Figure 3-4) to the NETBIOS first checks to make sure that it finds the local name in the name broadcasts, then a "session request" packet (Figure 3-5) is sent to the destination and NETBIOS executes a LISTEN command to wait for the reply. If a "session accept" packet (Figure 3-6) is received before LISTEN times-out, then NETBIOS sets a session established flag in the session table, returns the local session number (LSN) to the application, and returns the command completed status (CMD_CPLT) to the application.



3-4: Name Query Packet



3-5: Session Request Packet



3-6: Session Accept Packet

LISTEN/LISTEN ANY

NETBIOS first checks to make sure that it finds the local name in the name table. If so, then NETBIOS makes sure that room is available in the session table and waits for a "session request" packet. If the source of the "session request" packet is the same as the remote name specified by the application, then LISTEN is completed. A "session accept" packet is returned to the source, the session_established flag

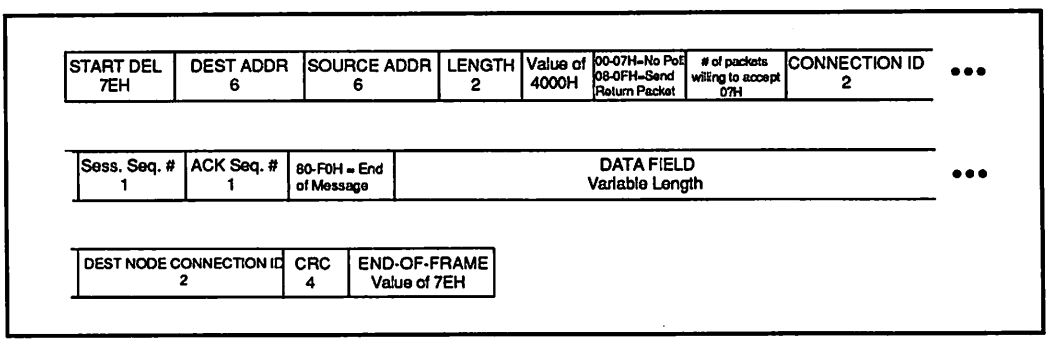
is set in the session table, the local session number (LSN) is set, and the appropriate status is returned to the application. If it is a LISTEN ANY request, then any "session request" packet will satisfy the request.

HANG UP

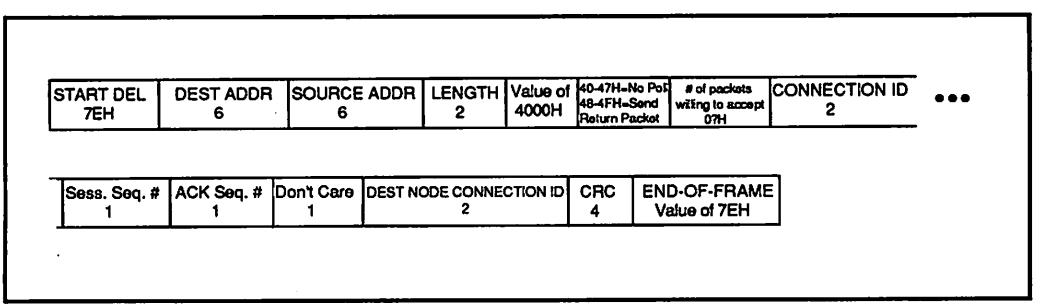
If the requested session number is legal and the session is "open," NETBIOS ends any RECEIVE commands and then ends the session. If a SEND command is pending, then NETBIOS waits until the SEND command has completed or timed-out.

SEND/CHAIN SEND

If the session number is legal and the session is "open," then NETBIOS sends the session data packet (Figure 3-7), as pointed to by NCB_BUFFER@, to the destination node and waits for an acknowledgment packet (Figure 3-8), or times-out and returns the appropriate status to the application.



3-7: Session Accept Packet



3-8: Acknowledgement Packet

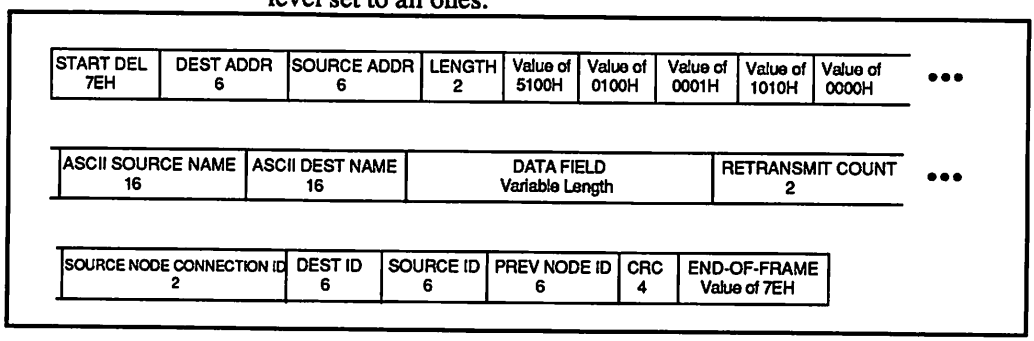
RECEIVE/RECEIVE ANY

If the session number is legal and the session is "open," then NETBIOS will wait for a specific (name) session message for the duration of the receive packet time-out set by the application. If NETBIOS receives a session packet within the time-out interval, then an acknowledgment is sent back to the source and the data is transferred to the buffer pointed at by NCB_BUFFER@. NETBIOS also checks to ensure that the length of the received message is not greater than the buffer length (set by NCB_LENGTH). The operation of RECEIVE ANY is similar except that reception can be from any name.

SEND DATAGRAM/SEND BROADCAST DATAGRAM BROADCAST DATAGRAM

For SEND DATAGRAM, NETBIOS checks the requested name number against a match in the name table.

If found, NETBIOS sends the datagram packet (Figure 3-9) to the destination node. For a BROADCAST, the name number is checked, and if valid, the datagram is broadcast to all nodes on the network. The datagram is sent only once, with the address field at the data link level set to all ones.



3-9: Datagram Packet

RECEIVE DATAGRAM/RECEIVE BROADCAST DATAGRAM

If NETBIOS finds the name number in the local name table, then it waits for the arrival of a datagram (other than broadcast). The application can designate a specific name from which to receive a datagram or else "any." If NETBIOS receives a datagram but it is not from the name the application had requested, then NETBIOS continues to wait. If the application wants any datagram, then NETBIOS returns to the application. In both cases, the received message

BIOS returns to the application. In both cases, the received message is copied to the NCB_BUFFER@, and the local name number of the received DATAGRAM is a specific case that watches for a destination address of all ones.

Transport Layer

The session layer calls on the transport layer to establish a reliable connection between two names (source and destination PCs). NETBIOS tries up to a maximum number of times to make the connection. If an acknowledgement (Figure 3-9) is received, then the connection is successful. From then on NETBIOS uses the reliable stream protocol (RSP). This protocol is proprietary to the original PC Network. Other NETBIOS implementations use some other protocol such as Xerox Network Systems (XNS), the Transmission Control Protocol (TCP), or the ISO/NBS transport level protocols.

Network Layer

The network layer on the IBM PC Network uses the packet transfer protocol (PTP). PTP consists of four major procedures: send PTP packet, send link access protocol (LAP) frame, receive LAP frame, and received frame.

The send PTP procedure requests that a buffer be sent to a specific network connection ID. If a connection exists, then PTP formats the buffer and sends it via a LAP frame. Note that even though this protocol/procedure allows internetworking to be implemented, the IBM PC Network NETBIOS implementation allows a processing of a packet to only one other network (adapter).

The send LAP frame procedure sends a buffer to a specified destination node or broadcast address by passing a properly formatted request to the data link layer.

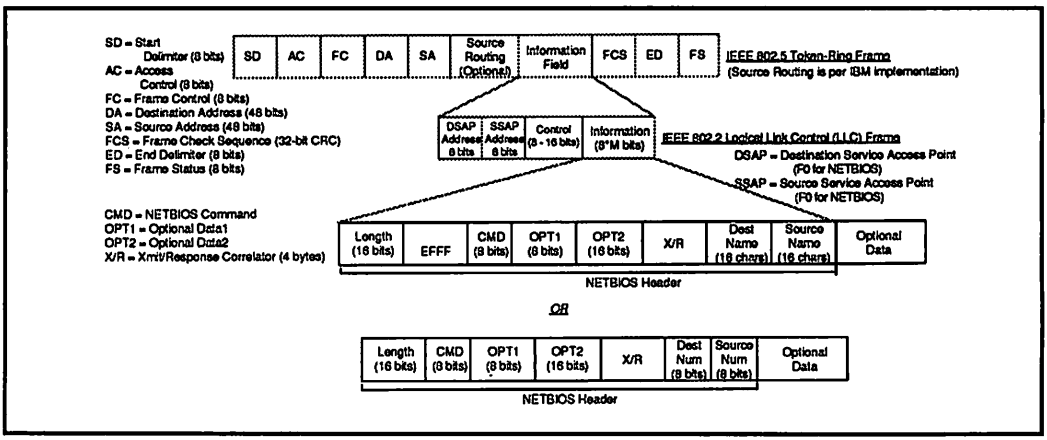
The receive LAP frame procedure receives valid frames directly from the data link layer. This procedure is also responsible for allocating and deallocating buffers as frames are received.

If a frame is received from the previous procedure, then the received frame procedure is invoked. This procedure checks the packet type for one of the following: connection data; route completion; discovery; datagram; route establishment; or a duplicate (which is ignored). Datagrams are passed to the transport layer for further processing. Datagrams are the lowest form of packet in the network, and all higher level protocols including the NETBIOS SEND commands, eventually use them indirectly in a chain of events (session to transport to network to data link).

Token-Ring

In the IBM Token-Ring network, a NETBIOS frame is directly embedded in an IEEE 802.2 Logical Link Control (LLC) frame at Layer 2 of the OSI model. Layer 2 frames are in turn encapsulated in the Token-Ring control frame at layer 1, the IEEE 802.5 Media Access Control (MAC) layer. The following is a discussion of the basic format.

Figure 3-10 shows the basic NETBIOS frame format. This frame is embedded in the information field of the 802.2 frame which in turn is embedded in the information field of the 802.5 frame. The 802.5 frame itself is transmitted around the ring to the destination station, where it is broken down until the NETBIOS emulator can process the actual NETBIOS frame itself.



3-10: Token-Ring NETBIOS Frame Format

The first 16 bits of the NETBIOS frame contain the length of the NETBIOS header, including the length field itself. The next 16 bits, or two bytes, contain the hex value EFFF which is a delimiter which indicates that subsequent data is destined for the NETBIOS emulator. The next byte is the actual NETBIOS function of the fame. In the Token-Ring, functions with hex values 00 through 13 are non-session frames that are sent using 802.2 unnumbered (U) information frames, while the functions with hex values 14 through 1F are session frames that are sent using 802.2 information (I) frames. U frames are analogous to datagrams (self-contained frames with no sequence numbers) while I frames contain sequence numbers and are reliably transmitted and received on the Token-Ring.

The next 8 bits or 1 byte is an optional data byte per a specific command. Likewise, the next 2 bytes are optional data bytes per a specific command.

The next 4 bytes in the frame are a correlator -- one or two numbers in the range hex 0001 to FFFF. It is used to associate received responses with transmitted requests. The transmit correlator is returned in a response to a given query (the value was received as the response correlator). The response correlator is the value expected (in the transmit correlator field) when the response to that message is received.

Non-session frames contain the NETBIOS 16-character destination name followed by the 16-character source name. Session frames contain a 1-byte destination session number followed by a 1-byte source session number.

As you can see, non-session frames have a NETBIOS header with a total of 43 bytes while session frames have fewer overhead bytes with a total of 13 in the header.

A summary of the frames used in the Token-Ring NETBIOS emulator is shown in table form in Figures 3-11 through 3-14. The figure lists the frames in table form by frame type.

The following frames are used to provide name management functions.

Command Name	Code	Function
ADD_GROUP_NAME_QUERY	X'00'	Check for duplicate group name on network
ADD_NAME_QUERY	X'01'	Check for duplicate name on network
ADD_NAME_RESPONSE	X'0D'	Negative response: add name is duplicate
NAME_IN_CONFLICT	X'02'	Duplicate names detected

3-11: Token-Ring NETBIOS Management Frames

The following frames are used to establish, maintain, and terminate sessions.

Command Name	Code	Function
NAME_QUERY	X'0A'	Request to locate a name on the network
NAME_RECOGNIZED	X'OE'	Name Recognized: NAME_QUERY response
SESSION_ALIVE	X'1F'	Verify session is still alive
SESSION_CONFIRM	X'17'	SESSION_INITIALIZE acknowledgement
SESSION_END	X'18'	Session Termination
SESSION_INITIALIZE	X"19'	A session has been set-up

3-12: Token-Ring NETBIOS Session Frames

The following frames are used to transfer both session and non-session data.

Command Name	Code	Function
DATA_ACK	X'14'	DATA_ONLY_LAST acknowledgement
DATA_FIRST_MIDDLE	X'15'	Session data message - first or middle frame
DATAGRAM	X'08'	Application-generated datagram
DATAGRAM_BROADCAST	X'09'	Application-generated broadcast datagram
DATA_ONLY_LAST	X'16'	Session data message - only or last frame
NO_RECIEVE	X'1A'	No receive command to hold received data
RECEIVE_CONTINUE	X'1C'	Indicates receive outstanding
RECEIVE_OUTSTANDING	X'1B'	Re-transmit last data - receive command up

3-13: Token-Ring NETBIOS Data Transfer Frames

Command Name	Code	Function
STATUS_QUERY	X'03'	Request remote node status
STATUS_RESPONSE	X'0F'	Remote node status information
TERMINATE_TRACE	X'07'	Terminate traces at remote nodes
TERMINATE_TRACE	X'13'	Terminate traces at local and remote nodes

3-14: Token-Ring NETBIOS Additional Frames

Chapter 4

Server Message Block Protocol

Overview

The redirector and server functions of the IBM PC LAN Program implement the Server Message Block (SMB) protocol developed by Microsoft, Intel, and IBM. SMB operates at the application level, and the IBM version requires NETBIOS for proper operation. SMB is designed to be machine- and operating system-independent, although the IBM implementation is closely tied to PC-DOS.

While SMB is an "open" protocol as published by IBM, few vendors have chosen to implement it in their PC LANs. One third-party vendor, 3Com Corporation, has decided to use it in its 3+ implementation. One reason for this choice is that 3Com and AT&T have an OEM agreement whereby AT&T bundles the 3Com software with its StarLAN 1 Mbps baseband LAN for small AT&T (6300 and 7300) computers. Another is 3Com's close relationship to Microsoft -- both vendors worked to develop the LAN Manager for OS/2 (see Chapter 6). Other vendors, such as Novell, have implemented their own "redirector" (the shell) and server protocols for reasons of efficiency and added functionality.

The IBM PC LAN Program can be broken up into four major functions: redirector, receiver, messenger, and server. The redirector intercepts DOS 21H function calls and determines whether the request is for a local device or a remote device. If the device is local, the redirector simply passes the request to the local operating system (DOS 3.1 or greater for the IBM PC Network, DOS 3.2 or greater for

the Token-Ring). If the request is for a remote device, then the redirector must translate it into SMB protocols.

The receiver listens for SMB protocols passed from another PC on the network, then removes a message portion that is human-readable and passes it to a local device such as the screen, a file, or a printer.

The messenger is a superset of the receiver; besides handling messages, it can also send messages the other way. In other words, it can translate messages from the user into SMB to be sent off over the network to another PC.

The server is the most complex function of the PC LAN Program in that it implements the full set of SMB protocols and manages local devices for shared use by other PCs on the network. Types of requests that it must handle from the other PCs include requests to access files and print spooling.

The first three functions -- redirector, receiver, and messenger -- can also be viewed as a subset of the server. Information exchanges are always initiated by an action from a requester PC. The requester is usually the redirector, and the request is sent to one of the four configurations as described above.

Naming

Naming is key to supporting communication between two points on the network. SMB supports two classes of names: network names and network paths.

Network names are 16-byte character strings that form the machine, server, redirector, main user, and additional user names added to a local name table of each PC. The names themselves can be up to 15 bytes long; if necessary, they are padded with blanks. A one-byte suffix identifies the type of name.

A machine name is an up-to-15 character name given to the computer by the configuring software, under the control of the end user. A server name is a 16-byte name consisting of the machine name with a 20H in the sixteenth byte. This name is used in a workstation PC to communicate with a server PC. To communicate with the redirector, a 16-byte name consisting of the machine name with a 00H in the sixteenth byte is used. The additional user name consists of the machine name with a 03H in the sixteenth byte. Its main purpose is to send and receive messages. A forwarded name is a user, main, or additional name whose sixteenth byte changes to 05H.

Network path names associate with resources to be shared. For each resource, such as a subdirectory on a server's hard disk, a network path name is created for the subdirectory (path) by prefixing the machine name to the resource (path) name.

A network path name has the following format: \\nnnnnnnnnnnnnnn\ ddddddddd...ddd where nnn...n is a 1 to 15 character machine name and ddd...d is a device name or directory path. The maximum network path length is 146 bytes.

Establishing a PC-to-Server Connection

When a user attempts to "connect" to the resources of a server (such as via the PC LAN Program NET USE command), the redirector attempts to establish a session with the server. If there is room in the local adapter's address table, then a session starts in which the redirector and server agree upon a protocol and begin communicating.

On the server side, the redirector requests that a connection be set up to the shared resource such as a subdirectory. The server will make sure that the requested resource exists, and if so, check the optional password for validity. The server then responds with a maximum server transmission block size and a connection handle called the network path ID (similar to the handle returned by PC-DOS when opening a file) for all future requests to the resource. When the connection terminates, the redirector tells the server to end the connection and free the handle.

A user may forward messages to another computer. If so, then the sender (such as a server) requests that the target computer add the user name to the network as a forwarded name. From then on, the messages for that user will go to the target computer.

SMB Protocols

The SMB protocol set consists of four types of server message blocks: session (connection) control, file, printer, and message. Session control fulfills two major functions: dialect determination and connection control.

Given that the PC LAN Program is in operation (and thus the SMB protocols are in effect) and after session establishment between a redirector PC and a server PC, the redirector sends the command VERIFY DIALECT with a list of supported dialects back to the server. The server then determines if it can support one of the dialects. If so, it sends back to the redirector an indication of which dialect to be used.

If the server cannot support any of the dialects, it sends an error response back to the redirector PC and the session is terminated; otherwise, communication may commence.

Connection control consists of commands which start and end a redirector connection to a shared resource at the server. The START CONNECTION command establishes the connection between a redirector PC and a shared resource at a server PC. All further commands and responses use this session. The END CONNECTION command terminates the connection between the redirector and the shared resource.

The redirector may use the file access commands to access files at the server once the START CONNECTION command establishes a connection. These commands are similar to the local PC-DOS function calls which allow access to files and directory. Additional commands have been added to determine the configuration and status of the remote shared resources. The supported file access commands include: check, create, and remove directories; create file, create temporary file, create new file; delete or rename file; get or set file attributes, search multiple files and get disk attributes; open and close files; read and write a byte block; commit process and end process; and finally, lock and unlock a byte block.

The print server commands allow a redirector to send files to a print queue at a server and obtain status information about the print queue. Commands include create spool file, spool byte block, close spool file, and return print queue.

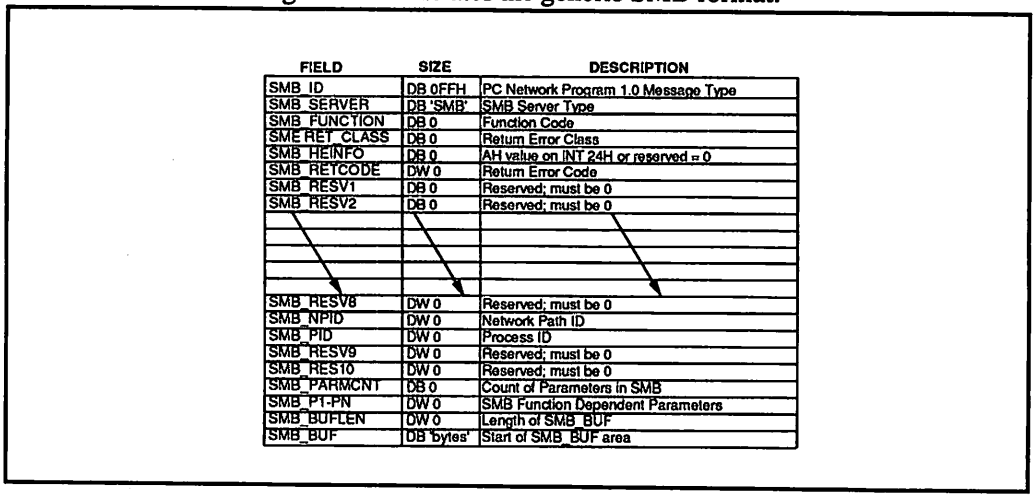
An application may use the message commands to send and receive messages. They include commands to send and receive short messages (one transmission) or long messages (multiple transmissions), to forward (or cancel) messages, and to send a broadcast message (short message only). While the message protocols allow multiple user names at one PC to send or receive messages, the IBM PC LAN Program implementation allows messages to be sent only from a single name. The commands that support messages include send single block message, send broadcast message, send start of multiple block message, send text of multiple block message, send end of multiple block message, forward user name, cancel forward, and get machine name.

SMB Format

This section describes the generic SMB fields and structure (format). Note that the terms "device name," "dirname," and "file name" refer to their PC-DOS equivalents (e.g., a device name such as PRN

(printer)). A dialect name is a string of characters with the same restrictions as a file name (8 characters plus an optional 3-character extension). The beginning of this chapter discusses the structure of the network name. An origin and destination name is a 1-to-15-character machine name (also discussed earlier). A password is a 1-to-8-character name with the same restrictions as a PC-DOS file name.

Figure 4-1 illustrates the generic SMB format.



FIELD	SIZE	DESCRIPTION
SMB_ID	DB 0FFH	PC Network Program 1.0 Message Type
SMB_SERVER	DB 'SMB'	SMB Server Type
SMB_FUNCTION	DB 0	Function Code
SME RET_CLASS	DB 0	Return Error Class
SMB_HEINFO	DB 0	AH value on INT 24H or reserved = 0
SMB_RETCODE	DW 0	Return Error Code
SMB_RESV1	DB 0	Reserved; must be 0
SMB_RESV2	DB 0	Reserved; must be 0
SMB_RESV8	DW 0	Reserved; must be 0
SMB_NPID	DW 0	Network Path ID
SMB_PID	DW 0	Process ID
SMB_RESV9	DW 0	Reserved; must be 0
SMB_RES10	DW 0	Reserved; must be 0
SMB_PARMCNT	DB 0	Count of Parameters in SMB
SMB_P1-PN	DW 0	SMB Function Dependent Parameters
SMB_BUFLEN	DW 0	Length of SMB_BUF
SMB_BUF	DB 'bytes'	Start of SMB_BUF area

4-1: SMB Generic Format

The SMB_FUNCTION field can take on the follow values:

Value	Meaning
00H	Create directory
01H	Delete directory
02H	Open file
03H	Create file
04H	Close file
05H	Commit all files
06H	Delete file
07H	Rename file
08H	Get file attribute
09H	Set file attribute
0AH	Read byte block

0BH	Write byte block
0CH	Lock byte block
0DH	Unlock byte block
0EH	Create unique file
0FH	Create new file
10H	Check directory
11H	End of process
12H	LSEEK
70H	Start connection
71H	End connection
72H	Verify dialect
80H	Get disk attributes
81H	Search multiple files
C0H	Create spool file
C1H	Spool byte block
C2H	Close spool file
C3H	Return print queue
D0H	Send message
D1H	Send broadcast
D2H	Forward user name
D3H	Cancel forward
D4H	Get machine name
D5H	Start multi-block message
D6H	End multi-block message
D7H	Multi-block message text

The SMB_RETCODE field can take on the following values (with SMB_RETCLASS = 00H):

ValueMeaning

0054H Message has been buffered

0055H Message has been logged

0056H User message displayed

The SMB_RETCODE field can take on the following values (with SMB_RETCLASS = 02H):

Value	Meaning
0000H	Reserved
0001H	Unknown error
0002H	Bad password
0003H	Device type mismatch on assign
0004H	Netname access level violated
0005H	Invalid network path ID
0006H	Network path not found
0007H	Invalid device
0031H	Print queue full (number of files)
0032H	Print queue out of space
0033H	End of file on print queue
0034H	Invalid print file ID
0051H	Server is paused
0052H	Not receiving messages
0053H	No room to buffer message
0057H	Too many remote user names
0058H	Duplicate name on network
FFFFH	Function not supported

Session Control Commands

VERIFY DIALECT -- This command is sent by the redirector to the server to establish the dialect.

START CONNECTION -- This command establishes a connection between the redirector and the server shared resource. The server contains a table that maps the shared resource from the network path name to the local name, identifies the type of resource, and contains an optional password. The server returns a network path ID to use for subsequent requests for that resource. An application can also use START CONNECTION for PC-to-PC communication in which the maximum transmission size is 512 bytes.

File Commands

CREATE DIRECTORY -- Sent from redirector to server to do the PC-DOS MKDIR (make directory) function.

REMOVE DIRECTORY -- Sent from redirector to server to do the PC-DOS RMDIR (remove directory) function.

CHECK DIRECTORY -- Sent by the redirector to determine if a directory at the server exists when a user does a DOS CHDIR (change directory) command.

OPEN FILE -- Sent by the redirector to a server to open a file and return the file handle (just like the local PC-DOS operation). Starting with PC-DOS 3.1, several additional file open modes add support for multi-user environments. The following table describes the various modes:

Compatibility -- Provides compatibility with applications which used previous versions of PC-DOS. A file may be opened any number of times.

Deny Read/Write -- Used to gain exclusive access to the file. The request is rejected if the file is already open in any other mode.

Deny Write -- Allows the file to be opened as many times as requested for reading. The request is rejected if the file has been opened with a write access or in compatibility mode.

Deny Read-- Allows the file to be opened for writing. The request is rejected if the file has already been opened for reading in compatibility mode.

Deny None-- Allows the file to be opened as many times as requested for read/write. The request is rejected if the file is already open in any other mode.

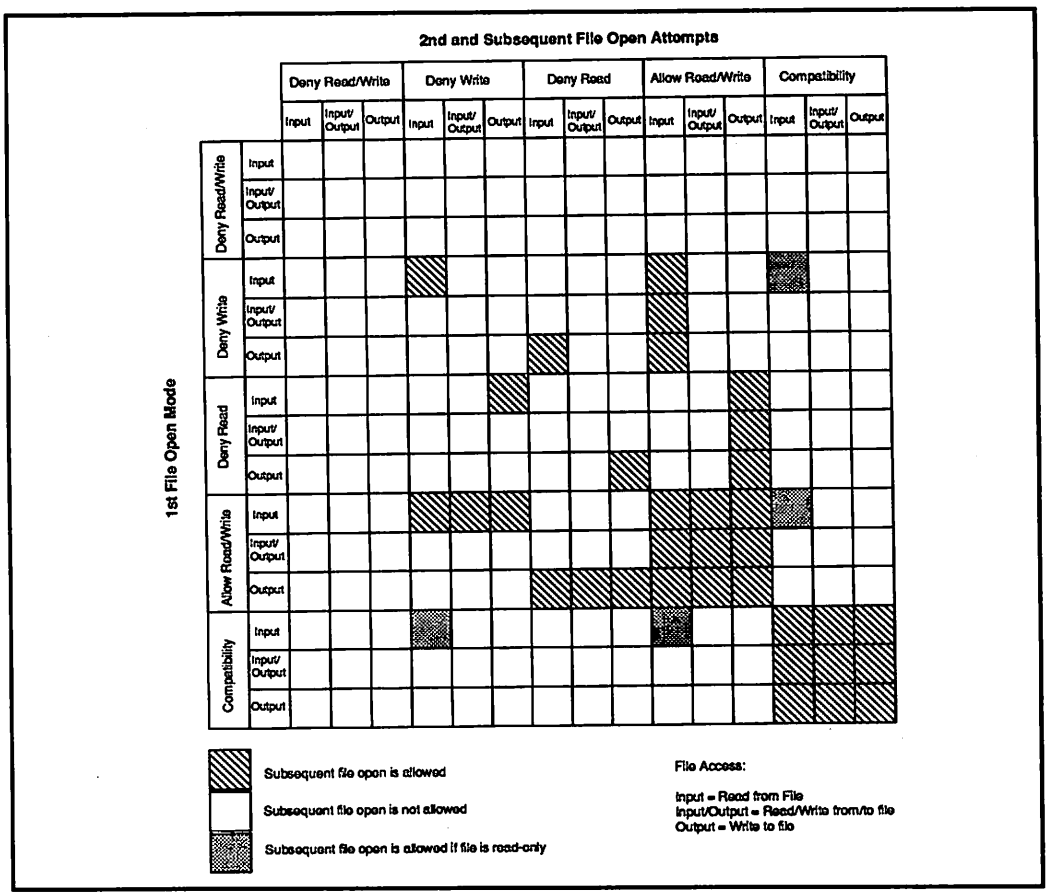
If an application opens a file using the older File Control Block (FCB) method via DOS INT 21H, function 0FH, then the sharing modes are not supported.

Figure 4-2 summarizes the various file modes and accesses.

CREATE FILE -- Sent by the redirector to a server to create a new file and return a handle. It is also used to delete an old file and create a new one with the same name. The request is rejected if the file is open or the file attribute is set to read only.

CLOSE FILE -- Sent by the redirector to a server to close a file. The redirector sends the file handle.

COMMIT FILE -- Sent by the redirector to a server to request that all buffers for a file be written to the server's hard disk. The redirector specifies a file handle, and the server responds when the operation is complete. Multiple files may be "committed" if the redirector



4-2: File Modes and Accesses

specifies that all files opened in the connection represented by the network path ID in the SMB_NPID field be committed.

DELETE FILE -- Sent by the redirector to a server to delete a file. The redirector specifies the file handle. The request is rejected if a file is open or marked as read only.

RENAME FILE -- Sent by the redirector to a server to rename a file.

GET FILE ATTRIBUTES -- Sent by the redirector to the server to obtain the file's attributes, time of last access, and size.

SET FILE ATTRIBUTES -- Sent by the redirector to the server to set file attributes.

READ BYTE BLOCK -- Sent by the redirector to the server to read a block of data from a file.

WRITE BYTE BLOCK -- The complement of READ BYTE BLOCK; sent by the redirector to the server to write a block of data from a file.

LSEEK -- (Long Seek) sent to the server to move the file pointer. The PC LAN Program uses this function to determine the size of a file. The file must have been previously opened in a mode that supports shared read.

LOCK BYTE BLOCK -- Supports the extended file byte-range locking function of PC-DOS 3.1 and higher. Sent by the redirector to the server to lock a region of bytes within a file. The region can be from one byte to the entire file. The request is rejected if any byte overlaps a region already locked. Note that this function does not adequately support transaction processing insofar as atomic locks (multiple lock requests in a single request to carry out a transaction) are not supported. Some servers, such as Novell's NetWare, support atomic locks.

UNLOCK BYTE BLOCK -- Complement of LOCK BYTE BLOCK; sent by the redirector to the server to unlock a region of bytes within a file.

CREATE UNIQUE FILE -- Sent by the redirector to the server to generate a unique file name by the server (actually PC-DOS running at that server). The server then returns this unique name to the redirector. A use for unique names is by applications that require temporary work files.

CREATE NEW FILE -- Similar to Create Unique File, except that the file name must be unique to files already in that directory.

END OF PROCESS -- Sent by the redirector to the server to terminate work within a connection. It is sent for each network path a redirector has active.

GET DISK ATTRIBUTES -- Sent by the redirector to the server to get information about hard disk storage size and layout.

SEARCH MULTIPLE FILES -- Sent by the redirector to the server to do the PC-DOS FCB and ASCIIZ search functions. The path and file name is passed to the server. The search can also check for hidden and system files.

Print Commands

CREATE SPOOL FILE -- Sent by the redirector to the server to set up a data stream in which to spool print data.

SPOOL BYTE BLOCK -- Sent by the redirector to the server to spool a block of data from a file. The first block sent contains set-up information for the printer.

CLOSE SPOOL FILE -- Sent by the redirector to the server to close (mark to end of) a print spool. The server then queues the file for printing.

RETURN PRINT QUEUE -- Sent by the redirector to the server to return the contents of the server print queue.

Message Commands

SEND SINGLE BLOCK MESSAGE -- Sent from one redirector to another, sends a short (single block) message consisting of a maximum of 128 characters.

SEND BROADCAST MESSAGE -- Sends a short message to all receivers on the network. A broadcast message is sent as a datagram; thus there are no responses from the receivers.

SEND START OF MULTI-BLOCK MESSAGE -- Sent by a sender to start a message of multiple blocks. The receiver returns a message group ID which subsequent message blocks use.

SEND TEXT OF MULTI-BLOCK MESSAGE -- Sent by a sender to send a message of up to 1,600 characters in 128-character blocks.

SEND END OF MULTI-BLOCK MESSAGE -- Sent by a sender to indicate the end of a multiple block message.

FORWARD USER NAME -- Sent by a sender to a server (or subset) requesting that the server receive messages for an additional user name. There must be room in the server's table to add the name.

CANCEL FORWARD -- Complement of FORWARD USER NAME. The server (or subset will delete the name from its table.

GET MACHINE NAME -- Sent to get the machine name of a user name. This command is usually used with CANCEL FORWARD in order to get the name to which to send the CANCEL FORWARD command.

Chapter 5

Other Vendor Implementations

Non-IBM Implementations of NETBIOS

This chapter describes the NETBIOS implementations of several vendors. Of the first four vendors, one implementation is from a PC board manufacturer, another is from an Ethernet front-end board manufacturer, the third is from a PC LAN file server software vendor, and the fourth is from a multi-user software vendor.

Common to all vendors is a NETBIOS-compatible session level interface that conforms to the NETBIOS 5CH interrupt function calls and values returned from the function calls. As discussed in chapter one, what differs from vendor to vendor is the underlying protocols that are used to implement the high-level protocols below the session layer (OSI Layer 5), that is the transport and network layers (refer to Chapter one for a discussion of the OSI model). Often, the network layer interfaces to a particular type of data link layer; thus, the particular NETBIOS implementation only runs on that vendor's network implementation. It should be noted, however, that all vendors are open to porting their intermediate protocols to other LAN technology; thus users could implement the NETBIOS emulator of their liking on the underlying LAN technology of their choice.

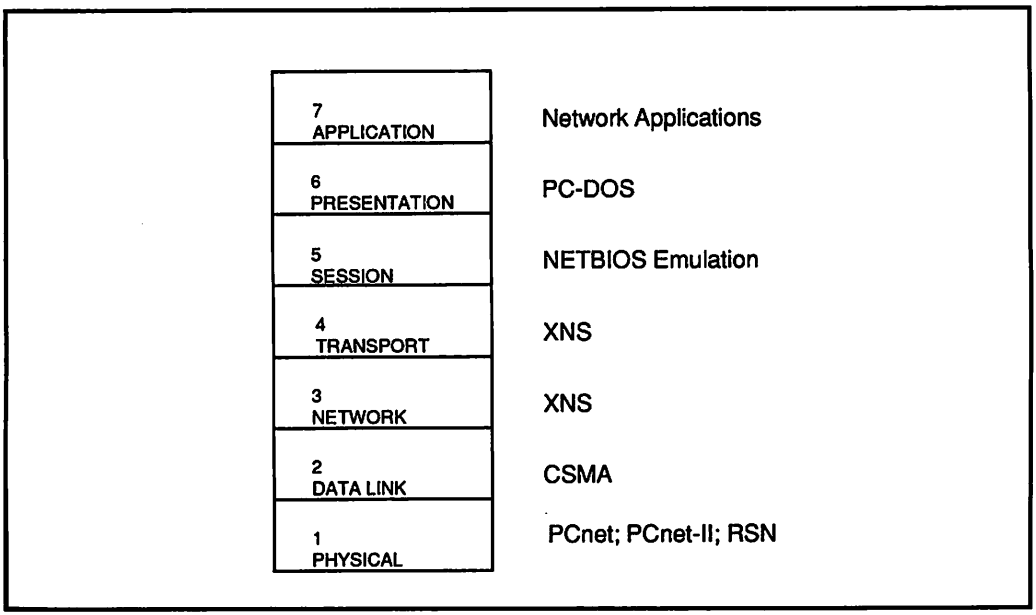
AST Research

The AST Research (Irvine, CA) NETBIOS option fully implements all NETBIOS functions, allowing the IBM PC LAN Program or other NETBIOS-compatible applications to operate properly. Large portions of it were written in the C language, making it somewhat portable. It is available for AST's PCnet, PCnet-II, and RSN. AST is

also willing to sell it to OEMs for customization. The underlying protocols used are based on XNS.

The AST implementation allows the user more control over the operation of NETBIOS than other implementations, including IBM's. When configuring the software, the user can accept the defaults or optionally specify the CALL, NAME, SESSION, and ACK time-out value, the NAME, SEND, SESSION, and DATAGRAM retry count, the number of open sessions, the number of outstanding commands, and adjustable data packet sizes.

Figure 5-1 illustrates the general implementation of AST's NETBIOS.

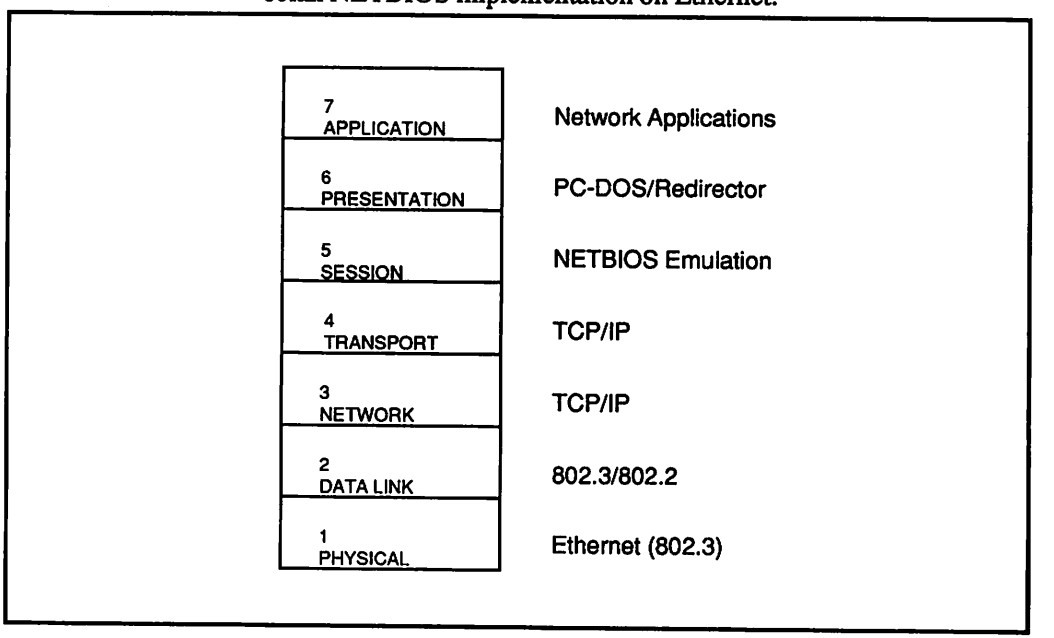


5-1: AST Research NETBIOS Implementation

Excelan

Excelan, Inc. (San Jose, CA; now a division of Novell) has a NETBIOS emulator for end users that covers all the IBM NETBIOS interrupt 5CH commands. This implementation runs on top of Excelan's $ITCP/IPTCP/IP protocols that in turn communicate over an Ethernet LAN. Excelan's OEM offering of TCP/IP protocols is designed to operate with Excelan equipment (Ethernet front-end processors). Excelan offers OEMs a version of NETBIOS/TCP/IP that can be customized. One should note that TCP/IP implementations from multiple vendors are more compatible with each other and

likely to communicate than implementations based on protocols such as Xerox Network Systems (XNS). Figure 5-2 summarizes the Excelan NETBIOS implementation on Ethernet.



5-2: Excelan NETBIOS Implementation

Novell, Inc., and Excelan Inc. have developed TCP/IP Workstation for NetWare. (The hardware/software product is an alternative to the use of a communication gateway and allows a personal computer workstation on a LAN to gain direct access to and transfer files concurrently with a NetWare server and TCP/IP host computer.)

The Excelan TCP/IP Workstation provides users with NetWare-based services and applications, and Excelan's TCP/IP-based applications, including FTP, Telnet, VT100 and VT220 terminal emulation, and Utilities. Thus users have direct access to TCP/IP host mini- or superminicomputers and to NetWare servers. Excelan's design allows Novell's IPX protocols and Excelan's TCP/IP in the workstation to run concurrently.

Novell

Novell, Inc. (Orem, UT) added NETBIOS emulation beginning with Version 2.0a of Advanced NetWare. Full IBM NETBIOS interrupt 5CH commands are supported in the workstation, and are recognized by the server. In fact, both Novell and IBM servers (via the IBM PC Local Area Network Program) can operate simultaneously, and a

workstation operating either the Novell shell or the IBM PC LAN Program can access a Novell server. Of course, a workstation running the Novell shell can access only the Novell server and not a PC configured as a server under the IBM PC LAN Program.

The Novell implementation operates on top of Novell's IPX (Internet Packet eXchange) protocols. The implementation thus gives users NETBIOS compatibility with any of the many PC LAN adapters supported by Advanced NetWare (including PC Network and token-ring), as well as improved performance. It is interesting to note that tests conducted by Novell have shown a factor of 2 better performance in end-to-end workstation throughput using the emulator with the orginal PC Network broadband adapter vs. using the built-in NETBIOS of the same adapter. The user can specify which NETBIOS to use when installing the workstation shell.

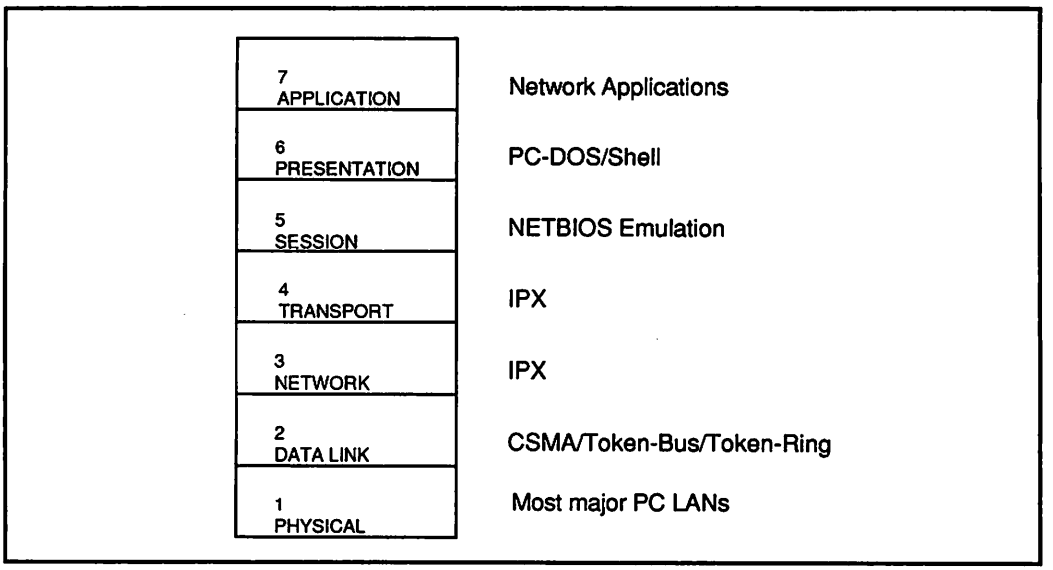
Another technique implemented by Novell to improve performance reduces the overhead associated with passing packets between layers. In theory, only packets are passed between layers, not information about buffer and work areas used by another layer. This promotes interoperability and compatibility with other vendors' implementations of a particular protocol.

In Novell's implementation, packets are not sent from NETBIOS (at the session layer) to the transport layer. Instead, a pointer is passed from one layer to the next, thus reducing processor and bus overhead from redundant copying.

Figure 5-3 illustrates the Novell NETBIOS implementation for NetWare-supported PC LANs. Also note that Novell's shell is functionally equivalent to Microsoft's redirector.

Novell and OS/2

The NetWare Requestor together with OS/2 Standard Edition supports external server-based OS/2 applications. Under this configuration, the NetWare Requestor is loaded on both the workstation, or client, and the application server. The client component of the application makes requests for application services through the NetWare Requestor; the network peer-to-peer transport mechanism then sends the request to the application server for processing. Replies are sent back to the workstation via the same path. If the application is to use the NetWare file system on a physically separate file server, the NetWare Requestor on the application server handles disk input/output (I/O) requests to the NetWare file server for the application.



5-3: Novell NETBIOS Implementation

Novell supports the standard application programming interfaces (APIs), as defined by IBM, that enable cooperative processing: these APIs include NETBIOS, APPC, and the API IBM has defined for OS/2. Novell will not support Named Pipes as they are not supported by IBM. Named Pipes is part of the Microsoft LAN Manager which is supported by 3Com.

The Software Link

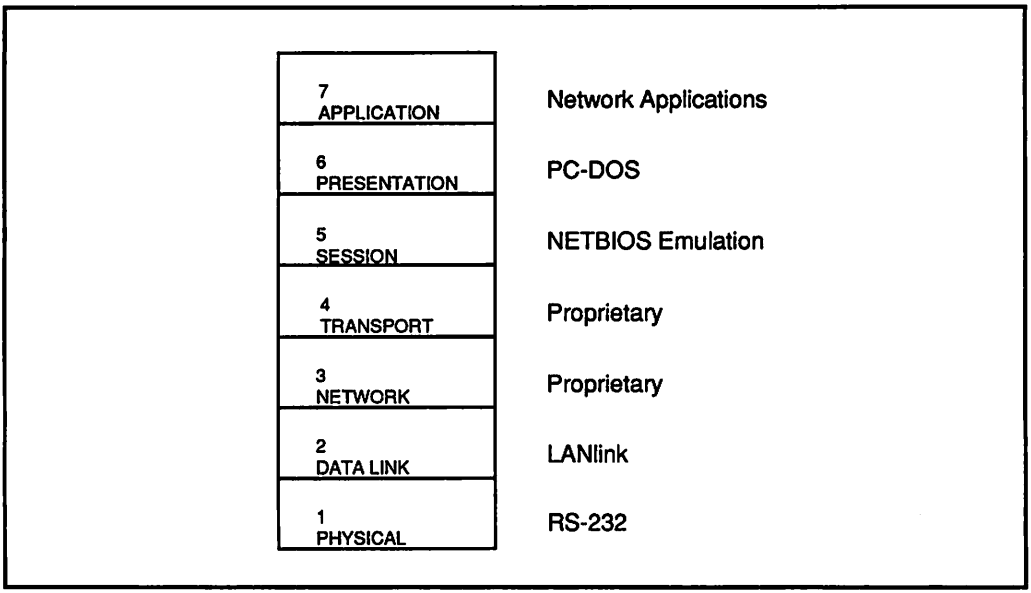
Unlike the three vendors discussed previously, The Software Link (Atlanta, GA) has developed a NETBIOS emulator for non-LAN products. The company has developed two products to allow users of IBM PCs to share resources (printers and hard disks).

The first product turns a PC or AT into a multi-user, multi-tasking machine. Multilink Advanced partitions RAM to concurrently run up to nine applications. As many as eight ASCII terminals (including The Software Link's Shadow, a terminal with IBM keyboard and screen emulation) connects to the host PC via RS-232 ports.

The second product, LANlink, allows one to configure a low-level network consisting of interconnected PCs in a star configuration. Stars can connect to other stars as well as to remote PCs, allowing a fairly arbitrary topology. Unlike Multilink Advanced, PCs attach to the host, not terminals. As with Multilink Advanced, RS-232 ports are used; however, the ports are driven at 56 Kbps with a proprietary data-compression technique that gives an effective throughput of 115

Kbps. The "satellite" PCs connect to a "server" PC and can use the server's hard disk as a local virtual drive (as in a PC LAN environment).

Both products can run all applications that are PC-DOS 3.1 and/or NETBIOS compatible. Figure 5-4 illustrates the implementation of NETBIOS by The Software Link for LANlink.



5-4: The Software Link NETBIOS Implementation

Other Vendors

Many other vendors offer a NETBIOS option for their networks (for a more extensive list, consult Appendix). They also take the approaches discussed above; in other words, they provide a 5CH session layer emulator and use the transport and network protocols of their choice. There are minor differences between the vendors in terms of enhancements (at the cost of downward compatibility to IBM's NETBIOS), and native services (e.g. Novell's shell) with which NETBIOS can co-exist in the vendor's LAN. To enhance performance, vendors such as 3Com and Codenoll are implementing NETBIOS protocols in the network adapters themselves. The following highlights addition implementations of interest.

CSI

Communications Solutions, Inc. (CSI) (San Jose, CA; now a division of 3Com) provides OEMs with SNA gateway software, including the

LAN-based versions of its Access/SNA 3270, Access/SNA APPC, and Access/DIA software.

CSI's NETBIOS-based LAN software enables users to concurrently access multiple SNA sessions. The software has been designed to minimize the memory requirements needed by workstations on the LAN. The code is split such that the presentation services/transaction layer is physically located on each individual workstation. The nucleus of the software resides in a single workstation serving as a gateway.

NCR

NCR Corporation (Dayton, OH) offers the NCR PC Token-Ring System. which consists of a PC adapter board, an adapter cable, an enhanced version of the Microsoft Networks network operating system, NETBIOS, menu- and command-driven user interfaces, and diagnostic software.

The NCR PC LAN and NETBIOS programs are also interoperable with other token-ring systems. The NCR PC LAN Program, a version of Microsoft Networks, manages the network's messaging, printer, disk, and directory functions. The NCR NETBIOS Program provides a PC-compatible transport facility for Logical Link Control and Media Access Control and runs applications written to use any industry-standard token-ring NETBIOS.

Network Research Corporation

Network Research Corporation (NRC) (Oxnard, CA) has added NETBIOS capability to its FUSION Network Software allowing the NETBIOS interface to run on multi-vendor networks.

The FUSION option allows users to run any NETBIOS application program over a TCP/IP network. In addition, NETBIOS users can transfer files between their systems and systems supported by FUSION Network Software, such as Digital VAX/VMS computers and UNIX workstations. Users can also use a variety of link layers such as Ethernet and token-ring from different vendors.

The FUSION NETBIOS option conforms to RFC 1001 (Protocol Standard for a NETBIOS Service on a TCP/UDP Transport: Concepts and Methods -- see Chapter 7) and RFC 1002 (Protocol Standard for a NETBIOS Service on a TCP/UDP Transport: Detailed Specifications) making it compatible for existing applications. The option is available for the MS-DOS/PC version of FUSION.

Pathway Design

Pathway Design, Inc. (Natick, MA) offers netPATH SNA-3770/NETBIOS, a local area network gateway to transfer data between any NETBIOS-compatible LAN and IBM mainframe at speeds up to 56 Kbps. The netPATH SNA-3270/NETBIOS gateway builds on the core technology embodied in the company's established netPATH product.

The netPATH/NETBIOS gateway is a hardware and software product that provides communication between PC users attached to a LAN and a host mainframe operating within an SNA network. The netPATH/NETBIOS product appears to the gateway as an IBM 3274 remote cluster controller with attached 3270 workstations. Up to 32 IBM PCs, ATs, or XTs can establish multiple, concurrent LAN sessions with the host and individual DOS sessions at each LAN workstation. The netPATH/NETBIOS gateway also includes Pathway Design's netPATH 3270 PC file transfer software.

The netPATH/NETBIOS gateway includes several network management tools, including a Network Supervisor, which provides statistics regarding all communications activities, including on-line session status and gateway session control. The Network Logging Facility provides comprehensive audit trails of all session activity, and an On-line Trace Facility allows both an SDLC and session-level capture and analysis of message traffic.

Pathway Design also offers COAXGATE, a LAN gateway that provides up to 40 SNA-DFT communications sessions on a single coaxial cable. Connecting directly to an IBM 3174 or 3274 Control Unit or to an IBM 9370 processor, COAXGATE gives PC users on a NETBIOS LAN access to remote or local host systems. The coaxial adapter board features 20-MHz microprocessor and private memory and -- when installed on the workstation selected as the LAN gateway -- enables each PC on the network to run up to five concurrent sessions with the mainframe for a maximum of 40 sessions/gateway.

A typical COAXGATE configuration consists of a LAN linked via coaxial cable or twisted-pair wiring into a 3299 multiplexer port on an IBM control unit.

Hughes LAN Systems

Hughes LAN Systems (formerly Sytek, the original developers of NETBIOS) offers to OEMs a baseband token-ring adapter card which conforms to the IEEE 802.2 logic link control and the IEEE 802.5

token-ring standards. The card connects into the IBM Token-Ring Network using NETBIOS as the programming interface, the IBM cabling system for physical interconnection, and the Token-Ring access protocol for the network traffic control and operation.

To complement the adapter card, Hughes LAN Systems also offers its version of the Token-Ring NETBIOS program. The card supports both the IBM PC LAN Program and Novell NetWare operating systems for the Token-Ring.

3Com

3Com Corporation (Mountain View, CA) incorporates in its 3+ network operating system, internetworking capability available to NETBIOS-based software. IBM PCs share data transparently on local Ethernet and Token-Ring networks and through remote modems or internetwork bridges.

3+ supports such NETBIOS-based applications as MDBS III version 3.09, IBI PC Focus, Hayes SmartCom II (for PC Network), DCA Crosstalk XVI (Network Version) and IrmaLan, PCOX 3270 gateway, Brightwork Software's Net Remote.

3Com achieved internetworking with NETBIOS by integrating the NETBIOS interface into its Xerox Networking Standard (XNS) communications drivers: 3+ translates NETBIOS requests into XNS requests to be sent out across the network; when responses are received from either remote or local networks, they are then translated back into the NETBIOS call format. As far as the NETBIOS application is concerned, all NETBIOS requests and responses (but not necessarily response time!) appear to be local.

3Com and OS/2

3+ open is the newest 3Com server offering that supports OS/2 workstations and DOS workstations. NETBIOS and data link control (DLC) interfaces are supplied. APPC is supported by DLC and uses the DLC data link interface to the physical network connection. The DLC interface also supports the IBM NETBIOS and Token-Ring protocols used by the IBM OS/2 LAN Server (see Chapter 6).

3+ open also supplies Named Pipes, a facility of the Microsoft LAN Manager, whichs provides an interface that extends the interprocess communications of OS/2 transparently across the network. Named Pipes offers an alternative to APPC, NETBIOS, or even the network transport layer, making it easier to create an application which uses

remote procedure calls across the network. Named Pipes are a complement to the interprocess pipes used throughout OS/2.

Named Pipes allow an application developer to access a computing resource (which may be located elsewhere on the network) as if it were local. Named Pipes simplify this effort of building network-intrinsic applications. The availability of Named Pipes does not exclude an application developer from developing applications which directly use the APPC, NETBIOS, or transport-level APIs.

Protocol Analyzers

As local area networks proliferate and become more complex, the need for specialized tools has arisen. These tools can aid in diagnostics, debugging, maintenance, management, and administration of a network.

The first available LAN tools were those that gathered statistics about a network's performance (e.g. network loading, packets per second throughput) and health (e.g. CRC errors, collisions, lost tokens). These tools monitor primarily the LAN's physical and data link layers.

The second category of tools to become available sat on top of the LAN systems software to allow a manager to control the operation of the network. With these tools, the manager could control user access to network nodes and the ports attached to these nodes and also define various parameters under which the nodes would operate (e.g. permanent virtual circuits, terminal characteristics, flow control).

The third category of tools becoming available are those that are able to monitor the intermediate protocols -- Layers 3 through 6 (network through presentation) of the OSI Reference Model. With these tools, a manager can monitor the performance of the protocols and identify bottlenecks or trouble spots. The tools also allow developers to implement and debug protocols and ensure that they are compatible with the way a certain set of protocols (such as NETBIOS) is defined.

The Sniffer

The first commercially available protocol analyzer for NETBIOS, is The Sniffer from Network General Corporation (Sunnyvale, CA).

The Sniffer is a combination of hardware and software which serves as a frame analyzer for the IBM Token-Ring, Ethernet, StarLAN, or ARCNET. It can capture all frames transmitted on the network for analysis. It monitors the network and analyzes data in much the same way that a logic analyzer does for digital signals. The software includes real-time performance monitoring and filtering of desired packet types. For more complex event sequences, the user can cus-

tom program for certain trap conditions. All frames are timestamped and can be stored to disk for later analysis.

Another useful function of the Sniffer is its ability to inject frames into the network. This allows a manager to determine how the network reacts to an increase in load. This "load analysis" can determine the affects of adding more active devices to the network. "Extra" or erroneous NETBIOS packets can also be injected into the network in order to see how well a workstation or server handles them.

The available protocol suites include: several TCP/IP varieties (including ARP, TCP, UDP, ICMP, DNS, Telnet); Sun Microsystems NFS protocol (including RPC, YP, and PMAP); ISO 8473 IP (internet); ISO 8073 TP class 4 (transport); XNS protocols used by Xerox, 3Com 3Plus, and Ungermann-Bass (including PEP, SPP, RIP, and Courier); the Server Message Block (SMB) protocol developed by Microsoft for MS-Networks and also used in the IBM PC LAN Program; NETBIOS; NetWare Core Protocol (used for dialog between the workstation and a file server); the IEEE 802.2 logical link control (LLC) Type 1 and Type 2 operation protocols used by virtually all token-ring and Ethernet LANs; and the media access control (MAC) protocol (which is a superset of the IEEE 802.5 protocol) used in the IBM Token-Ring.

The Sniffer is built around portable 80286 or 80386 personal computer technology, making it a useful tool both in the field and in the lab.

Other products similar to the Sniffer which have NETBIOS decode capabilities include the LANalyzer from Novell/Excelan and Spider-Analyzer from Spider Systems.

Chapter 6

Microsoft and IBM

When the IBM PC Network was first announced in 1984, there was much confusion in the industry about what networking software IBM had implemented. Shortly after the IBM announcement, many LAN vendors responded by announcing support for Microsoft Networks, thinking that that was what was being used in PC Network. There was some precedent for this belief: after all, Microsoft's MS-DOS did become PC-DOS to IBM. (Microsoft Networks is strictly an OEM product. It is not implemented on any particular LAN -- that is up to the OEM.)

History repeated itself when IBM announced OS/2 in April of 1987 and at the same time, Microsoft announced the LAN Manager. IBM did not reveal its server strategy for OS/2 until later that year, when announcing the IBM OS/2 LAN Server. Once again, confusion prevailed about the relationship of the IBM LAN Server to the Microsoft LAN Manager. There are several differences between the two -- IBM essentially implements a subset of the Application Programming Interfaces (APIs) of Microsoft. Eventually, IBM will incorporation all of the APIs into its LAN Server, probably by sometime in 1991.

This chapter traces the evolution of Microsoft Networks and IBM PC LAN Program from the original to the OS/2 implementations.

Microsoft

Microsoft Networks

It wasn't until after PC Network broadband was first released that users and vendors began to realize that the networking software in PC Network (which later became known as NETBIOS) was incompatible with Microsoft Networks and not written by Microsoft. As a result, a flood of Microsoft Networks- compatible products announced from many LAN vendors never materialized.

Microsoft did change the Microsoft Networks' user command structure after PC Network was released in order to closely correspond to the commands used by IBM. (For example, the CONNECT command became NET USE.) In fact, the Microsoft Networks commands are essentially a subset of those used in the PC LAN Program.

A component of IBM's NETBIOS implementation was written by Microsoft: the now infamous redirector. The redirector has close ties to PC/MS-DOS in the sense that the SMB protocol is used (see Chapter 5), even though the rest of Microsoft Networks and NETBIOS is designed to be both operating system- and LAN-independent. Microsoft Networks is available for both XENIX (Microsoft's implementation of UNIX, System V) and MS-DOS. The redirector is responsible for ensuring that requests for services (such as opening or printing files), which are normally handled locally, are, if necessary, intercepted, transformed into a network request, and sent to a server for execution.

Like NETBIOS, Microsoft Networks is designed to work with MS-DOS 3.1 (both the workstation and server rely on it) or higher. File-sharing modes and file locking are identical. The extended naming scheme is also the same (\\server_name\directory\file).

Microsoft Networks vs. NETBIOS

The key difference between early Microsoft Networks and NETBIOS is that Microsoft Networks provided a transport-level interface, while the NETBIOS interface is at the session level. Microsoft Networks also includes dedicated server and workstation software, while the IBM PC LAN Program, which provides these and other functions including a non- dedicated server.

Microsoft Networks' transport layer is used to send messages reliably via virtual circuits. Up to 64K bytes can be sent per request. Communication with the transport layer is via interrupt 21H, function 5BH (recalled that NETBIOS uses interrupt 21H, function 5CH).

Communication with the transport layer is accomplished by setting up the transport control block (TCB), then doing a 21H interrupt. The transport control block is analogous to setting up a NCB or MCB in

NETBIOS. In fact, many fields are common to both the TCB and NCB/MCB implementation. Figure 6-1 shows the structure of the transport control block.

Field Name	Length (in bytes) and Meaning	
COMMAND	1	- COMMAND FIELD
CID	1	- COMMAND ID
VCID	1	- VIRCUIT CIRCUIT ID-NUMBER
LENGTH	2	- DATA BUFFER SIZE
BADDR	4	- POINTER TO MESSAGE BUFFER ADDRESS (OFFSET:SEGMENT)
RES1	2	- RESERVED
LADDR	16	- LOCAL ADDRESS
RADDR	16	- REMOTE ADDRESS
ASYNC	4	- POINTER TO ADDRESS NOTIFICATION ROUTINE (OFFSET:SEGMENT)
LNET	4	- LOCAL LAN NUMBER
RNET	4	- REMOTE LAN NUMBER
RTO	1	- RECIEVE TIME-OUT (500 MS INCREMENT)
STO	1	- SEND TIME-OUT (500 MS INCREMENT)
RES2	8	- RESERVED

6-1: Transport Control Block (TCB)

As with the original NETBIOS on PC Network broadband, the network layer in Microsoft Networks is only minimally implemented. Included is support for a hierarchical address consisting of a 4-byte network address and a 16-byte station address. Also included is low-level support for datagram service allowing unreliable (unacknowledged) packets of up to 512 bytes to be sent/received. The OEM must determine how to map the station addresses into a network address and how to implement a routing algorithm if a gateway is to be implemented.

Unfortunately, there were many differences between Microsoft Networks and NETBIOS which make compatibility somewhat difficult. For example, besides the differences pointed out above, the two naming schemes are incompatible. NETBIOS allows multiple names, dynamic reassignment, and forwarding of names; whereas MS Networks requires that an administrator assign one logical name to each physical address.

While the two implementations differ, they suffer from one common fault: both rely on MS/PC-DOS to carry out services in the file server. In other words, they are both tied into the deficiencies of this single-user, single-tasking operating system. A previous version of this report stated that "It is not even clear that a multi-tasking version of DOS would resolve the problem, since in order to perform well, it more than likely would not be compatible with previous DOS versions, leaving five million PC owners up in arms." As it turns out, OS/2 only maintained some of the DOS commands and its underlying file structure, leaving in place a weakness for any LAN server than runs under OS/2, to deal with. (The DOS method of using a file allocation table or FAT requires excessive table searching when opening, closing, and maintaining a file).

Enter NETBIOS

Some of the LAN vendors that announced support for Microsoft Networks have shipped production versions for their networks. Many of them are offering NETBIOS as well since Microsoft Networks now includes a "sample" NETBIOS emulator that OEMs can integrate with their own offerings. It appears that the original Microsoft Networks' usefulness in pure MS-DOS environments was somewhat limited. Microsoft Networks was originally geared towards LANs with a combination of Xenix and DOS operating systems.

LAN Manager

With the introduction of the Microsoft LAN Manager (at the same time as PS/2 and OS/2), the redirector became known as the Requestor, and NETBIOS became an application programming interface (API) programming interface of OS/2. The LAN Manager itself operates with OS/2 and thus runs on a PC AT or PS/2 Models 50 and higher with the 80286 or 80386 processor. The workstation PCs can operate with either MS-DOS and the PC LAN Program (which requires NETBIOS) or with OS/2 and the new NETBIOS API.

The MS OS/2 LAN Manager (developed with 3Com) allows users to link personal computer systems running either MS OS/2 or MS-DOS together on a single network. Systems running Microsoft XENIX and XENIX Net may also connect to the same network.

An MS OS/2-based system may act simultaneously as a workstation and a server within such a network; MS-DOS systems running Microsoft Networks can act either as a server or a workstation. XENIX-based systems may act as both server and workstation simultaneously.

The MS OS/2 LAN Manager provides networking capabilities including transparent file and print sharing, user security features, and

network administration tools. Because the MS OS/2 Lan Manager is tightly integrated with the MS OS/2 operating system, MS OS/2 programming interfaces work transparently across the network. The interprocess communication (IPC) capability facilitates applications that can communicate directly with each other, even though they are resident on different machines on the network. Application developers will be able to write a single version of a software product which will be capable of running either on a single machine, or distributed between machines connected by the MS OS/2 LAN Manager. Vendors such as Novell and 3Com have offered their own versions of IPC before OS/2.

The MS OS/2 LAN Manager is fully compatible with the existing Microsoft Networks products for both the MS-DOS and XENIX operating systems. This allows new systems running MS OS/2 to integrate with existing networks supporting MS Networks with minimal disruption to the users of the network.

Most OEMs that offer MS Networks (including Hewlett-Packard, Tandem, DEC, 3Com/Bridge, AT&T, Northern Telecom, AST Research, Apricot NEC Japan, Xerox, SMT Group Research Machines, and Digital Microsystems) will also be offering the MS LAN Manager. IBM is the notable exception. Even though MS Networks and the PC LAN Program/NETBIOS have a lot in common, IBM did not include in its original announcement support for the LAN Manager opting later to offer the OS/2 LAN Server software. Another difference between Microsoft and IBM is the IBM OS/2 Extended Edition -- this appears to be a "Value-added" version of OS/2 by IBM that Microsoft will not offer. It will be up to the OEM's of MS OS/2 to decide whether to add support for IBM's Extended Edition. Many vendors will offer add-ons to Standard Edition, such as an APPC/PC package.

LAN Manager NETBIOS API Interface

The Applications Programming Interfaces (API) Interfaces for Microsoft Operating System/2 give access to network functions through a well-defined interface. High-level languages use the API interface. API provides two categories of functions:

1) Applications-oriented calls. These functions are for network aware applications that do sophisticated functions such as using network resources, interrogating or controlling remotely spooled printer, sending or receiving network messages, etc.

2) Administration-oriented calls. These functions allow complete access to or control over remote server functions. Examples are in-

stall/pause/continue server programs, share/unshare server resources, interrogate/control server sessions, etc.

The LAN Manager API follows the conventions of the standard DOS API: all calls are "far pascal" and all pointers are far. The API uses a remote procedure call programming model to allow remote servers and server resources to be controlled like local resources. Such calls take a server name parameter that indicates the target of the operation. To prevent unauthorized use, certain functions that are being invoked remotely execute only if the requesting user has administrative privilege at the target server.

The NETBIOS driver is a resource category defined by the LAN Manger API. Other categories the LAN manager defines, but not within the scope of this report for discussion include:

- WORKSTATION - workstation configuration information
- REDIRECT USE - workstation remote device connection
- MESSAGING - user-to-user messaging
- ALERT - monitors and raises network alerts
- NAMED PIPES - two-way interprocess communication (an alternative to NETBIOS for OS/2 LANs)
- MAILSLOTS - one-way interprocess communication
- PROFILE - saves/restores active uses/shares
- CONFIG - sets entries from the LANMAN.INI file
- SERVER - server configuration and statistics
- SHARES - server file and device shares
- SESSIONS - server machine sessions
- CONNECTIONS - server user connections
- FILES - server open files
- AUDITING - audit trail logging
- ERROR LOGGING - system error logging
- CHAR DEVICES - remote character device connections
- PRINT QUEUES - spooler job queues
- PRINT JOBS - spooler printer jobs
- PRINT DESTINATIONS - virtual devices in the spooler
- ACCESS PERMISSIONS - file and other resource access permissions

- USERS - user file permissions and group membership
- STATISTICS - workstation and server operating statistics
- REMOTE - remote functions such as remote program execution, file copying, and file moving
- MISCELLANEOUS - network time of day, etc.

The NETBIOS driver is accessed through an Enum call. Enum calls enumerate the resource type in question, whether it is user names, jobs in a queue, queues in a server, etc. If the user buffer is not large enough, some of the variable length data may not be returned, possibly only for the last entry but not necessarily. The return code is "more data" whenever the user-supplied buffer is not large enough to hold all entries, including all variable-length data. The "entries read" parameter shows the number of fixed length portions that were returned. Some or all these may have only a part of the associated variable data returned with them.

All Enum calls place the fixed structures in front of the buffer so you can iterate through them. Variable data is packed in from the other end.

The NETBIOS Enum call provides two levels of information:

Level 0

```
struct netbios_info_0 {char  nb_net_name[16] };
/* nb_net_name is 16 character NETBIOS name */
```

Level 1

```
struct netbios_info_1 {char          nb_net_name[16];
char                 nb_driver_name[9];      /* OS/2 device driver name */
unsigned byte        nb_lana_num;            /* lan adapter number */
unsigned short       nb_driver_type;         /* 1 = ncb, 2 = mcb */
unsigned short       nb_net_status;          /* bit 0 on = net started */
    /* bits 14/15 opcodes as follows: */
     /* 0 = net not opened */
    /* 1 = open in regular mode */
    /* 2 = open in privileged mode */
    /* 3 = open in exclusive mode */
unsigned long        nb_net_bandwidth;       /* network bandwidth bits/s */
unsigned short       nb_max_sess;            /* max number of sessions */
unsigned short       nb_max_ncbs;            /* max number of outstanding ncbs */
unsigned short       nb_max_names;           /* max number of names */
};
```

Procedure Calls

NETBIOSENUM

Purpose: enumerates NETBIOS drivers.

Calling Convention

```
init far pascal NetBiosEnum(servername,level,buf,buflen,entriesread,totalentries)
char far *          servername;          /* name of tart PC (null if local) */
char far *          buf;                 /* pointer to info buffer */
unsigned short      buflen;              /* byte length of info buffer */
unsigned short far * entriesread;        /* # of entries supplied on return */
unsigned short far * totalentries;       /* total # of entries available */
```

Return Value

The buffer contents on return (format for a single entry) can be one of the following:

Level 0 contains a "struct_netbios_info_0" Level 1 contains a "struct_netbios_info-1"

Error Return

The function returns 0 if successful. Possible error returns are as follows:

net not started
device not found
server not found
remote server communication failure

NETBIOSGETINFO

Purpose - Gets information about a given NETBIOS driver.

Calling Convention

```
int far pascal netbiosgetinfo(servername, netdevname,level,buf,buflen)
int far pascal netbisogetinfo(servername,netbiosname,level,buf,buflen)
char far *          servername;          /* name of target pc (null if local) */
char far *          netbiosname;         /* netbios network name */
short               level;               /* level of info requested */
char far *          buf;                 /* pointer to info buffer */
unsigned short      buflen;              /* byte length of info buffer */
```

level 1 contains a "struct netbios_info_1".

Error Return

The function returns 0 if successful. Possible error returns are as follows:

- buffer too small for fixed fields

- device not found
- server not found
- remote server communication failure

NETBIOSOPEN

Purpose - Gets a handle to a NETBIOS driver.

Description

This call creates a handle for sending ($INCBNCBs to a NETBIOS driver. A program can determine what these names are by calling NETBIOSENUM. The null string can be used as device name to refer implicitly to the first installed NETBIOS driver.

NETBIOSOPT specifies open options which include:

Access mode:	1 Regular
(mask 0x3)	2 Privileged
	3 Exclusive

The access mode specifies how the opener is willing to share access to the NETBIOS driver with other processes. Any number of processes can open a driver in regular mode. Besides those processes, exactly one other process can open the driver in privileged mode. One and only one process can open a driver in exclusive mode. NCB operations are restricted depending on the access mode.

Mode	Description
Regular	Does not allow reset, receive broadcast datagram, receive any-to-any NCBs, or use of permanent names in any NCB.
Privileged	Does not allow reset or receive any-to-any
Exclusive	Allows any NCB operation

Calling Convention

```
int far pascal netbiospen(netbiosname, netreserved, netopenopt, nethandle)
char far *          netbiosname;        /* Name of NETBIOS network */
char far *          netreserved;        /* reserved pointer; must be 0 */
unsigned short      netopenopt;         /* open options */
int far *           nethandle;          /* word for returned handle */
```

Error Return

The function returns 0 if successful. Possible error returns are as follows:

- NETBIOS driver does not exist
- invalid options
- open-mode conflict with existing open
- system resources unavailable

NETBIOS handles are process-to-driver associations. Only the process that created that handle can use it.

NETBIOSCLOSE

Purpose - Closes a NETBIOS driver handle.

Description

This call terminates access to a NETBIOS driver, invalidates the handle, and cancels any NCBs issued by the process that owned the handle.

Calling Convention

```
int far pascal netbiosclose(nethandle, netreserved)
int             nethandle;  /* handle to close */
short           netreserved; /* reserved, must be zero */
```

Error Return

The function returns 0 if successful. Possible error returns are as follows:

- invalid handle

NETBIOSSUBMIT

Purpose

Passes an NCB to a NETBIOS driver. A handle of 0 refers to the first installed NETBIOS driver. This driver will be automatically NETBIOSOPEN'ed if necessary (in regular access mode) the first time a NETBIOS call refers to it using the 0 handle.

NETNCB points to the NCB to be executed (unchained NCB) or to the link word preceding the NCB (chained NCB).

NETNCBOPT specifies NCB processing options which include:

Chaining:	0 Single NCB is being passed
(mask 0x3)	1 Single NCB with error retry
	2 NCB chain with proceed-on-error
	3 NCB chain with stop-on-error

The chaining option specifies whether a single NCB or an NCB chain is being passed. A single NCB can be executed with optional error retry in which case the net kernel reissues the NCB a set number of times in response to the following errors:

09H - no resource available

12H - session open rejected

21H - interface busy

Chained NCBs must be in the same segment and are linked by a 16-bit offset pointer that precedes the NCB. An offset of 0xFFFF specifies the end of chaining.

Although any sequence of NCB commands may be chained, not all possibilities are useful. For example, you can't open a session and send a data packet on it by chaining the Send with Call. The NCB_LSN field returned in the Call NCB must be copied into the Send NCB -- the net kernel support doesn't do this automatically.

NCBs in a proceed-on-error chain are executed independently of one another, regardless of errors in the chain; such a chain simple provides a quick way to pass a set of NCBs to the driver. Proceed-on-error chains make the most sense when no-wait NCBs are being submitted.

A stop-on-error chain is terminated by the net kernel when an NCB in the chain causes an error. NCBs that were not processed because of an error earlier in the chain will have their NCB-CMD-CPLT field set to 0xB (Command Cancelled). This type of chain should normally have wait-mode NCBs only. No-wait NCBs are accepted, but in this case it is the immediate return and no the final return that determines whether the chain stops or proceeds.

Calling Convention

```
int far pascal netbiossubmit(nethandle,netncbopt,netncb)
int                 nethandle;          /* handle to issue ncb against */
unsigned short      netncbopt;          /* option flags */
struct ncb far *    netncb;             /* Address of NCB */
```

The function returns 0 if successful. Possible error returns are as follows:

- invalid handle
- invalid options
- access denied
- driver resources unavailable

- netbios-defined immediate return codes (no wait NCB)
- netbios-defined final return codes (wait mode NCB)

Operation

The Microsoft Networks kernel used by the LAN Manager supports multiple NETBIOS drivers by treating each one as a named, installable device driver that can be opened and closed. Applications using the same NETBIOS driver can share access to names and sessions they have created simply by passing the name or session number to another process. This allows other processes to send and receive data on behalf of the owing process, or to free the shared names and sessions.

Protected mode NETBIOS NCBs are processed the same as those under MS-DOS 3.X via interrupt 5C and 2A except as follows:

1. The NCB_POST@ field now contains a system semaphore handle, not the address of an asynchronous notification routine. The net kernel will clear this semaphore when a no-wait NCB completes. A semaphore handle of 0 is considered null and will not be cleared.

2. The NCB_POST@ and NCB_BUFFER@ fields are temporarily modified by the net kernel during processing of an NCB. Applications should not depend on the values of these fields until an NCB completes.

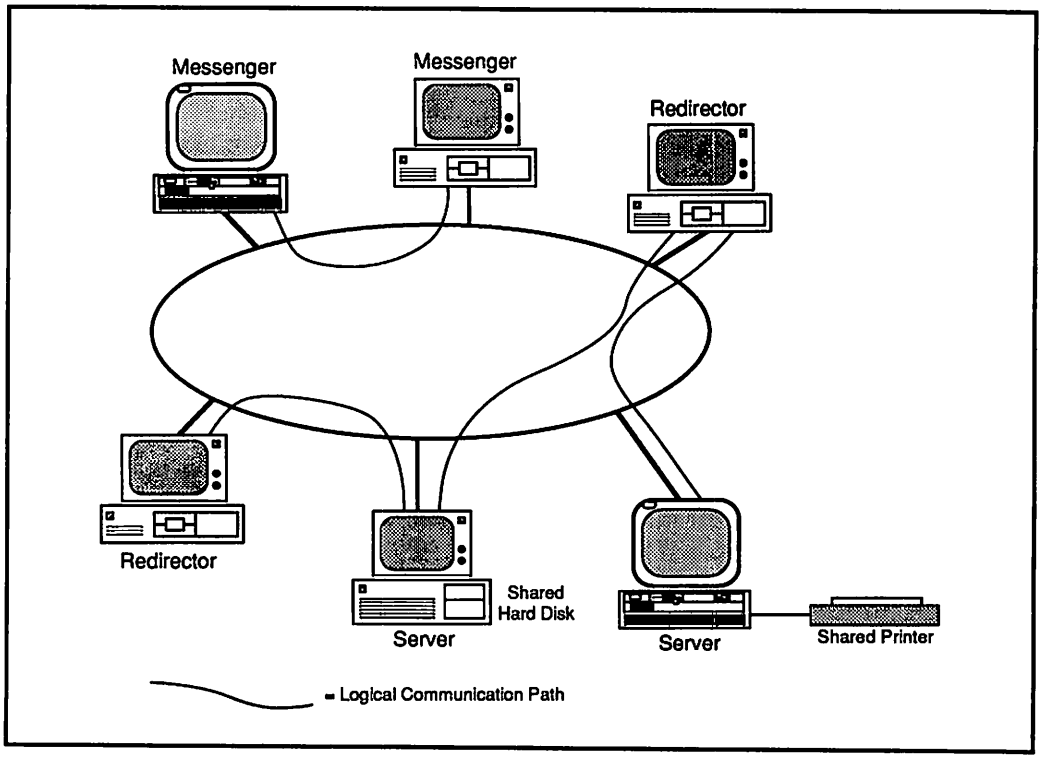
3. Certain NCB functions are disallowed, depending on the access mode specified on the NETBIOSOPEN call.

4. Processes may share NETBIOS resources they own simply by passing the desired names and session numbers to other processes (as discussed above). The recipient processes simple use the shared names or session numbers in NCBs that they create. Such processes must, or course, issue the NCBs against a handle that they themselves have had NETBIOSOPEN'ed to the same driver.

IBM PC LAN Program

The IBM PC Local Area Network Program (formerly called the IBM PC Network Program) was originally designed to operate with DOS 3.1 and the IBM PC Network (broadband). The PC LAN Program requires NETBIOS support available via the PC LAN Support Program and DOS 3.2 or higher, to operate on the Token-Ring.

The PC LAN Program consists of a single large executable file which can be brought up one of four ways: the user executes a NET START command and then specifies a redirector, receiver, messenger, or server. The first three (redirector, receiver, and messenger) are for workstations, and the last one (server) is a non-dedicated file/print server implementation which runs as a background task in a workstation. Figure 6-2 shows an example implementation with one PC configured as a file server and another as a print server.



6-2: PC LAN Program

Using the redirector is the most basic way to get onto the local network. It intercepts the workstation's printer and disk I/O to send to a server; users can also send messages to other machines. The receiver, messenger and server perform the same as the redirector

with the addition of: the receiver receives and logs messages to any device or file; the messenger allows a user to transfer files; and the server allows sharing of hard disks and printers.

The installation procedure is menu-driven. Once installed, the PC Network Program can be operated by typing commands at the DOS prompt or by menus.

The original version of the PC LAN Program has suffered heavy criticism -- primarily its poor throughput and lack of administrative features. The performance problem relate to the fact that unlike Novell's NetWare, PC LAN Program relies on DOS for its file and printer operation. As DOS continues to improve and the migration is made to Operating System/2, the operating system problems related to PC LAN Program will become less of an issue. DOS 3.3, for example, has added additional file allocation table (FAT) caching (the FASTOPEN command) and PC LAN Program Version 1.2 added disk caching.

PC LAN Program Version 1.3 has added the following features to address the administration problem: password controlled access (login) to a server; central resource definition and control including setting the time/date on a master server to synchronize dates and times in all workstations and servers, printer management, define users and modify privileges, and to manage application selector menus for individual users; administrator access to resources from any workstation; support for remote program and operating system load (i.e., support for diskless workstations); the ability to view logged-in users; and finally, remote workstation printer selection, queuing, and status. Version 1.3 has also minor improvements in performance.

LAN Server

The IBM OS/2 LAN Server implements NETBIOS to communicate across a LAN. The Server provides, to both DOS (via the PC LAN Program) and OS/2 applications, resource sharing for disks, printers, and serially attached devices, plus facilities for defining, controlling, and managing access to LAN resources, along with enhanced security and print management (for up to eight printers). These security and administration features are similar to those provided by PC LAN Program Version 1.3 except that file security is down to the file level. The LAN Server requires OS/2 Extended Edition and utilizes the OS/2 multi-session and caching functions. LAN Server also provides facilities for remote execution of programs.

Portions of the LAN Server have been licensed from Microsoft, most notably the redirector. As previously noted, IBM has not imple-

mented many of the Microsoft Application Program Interfaces (API) in LAN Manager, most notably interprocess communication facilities that are inconsistent with Systems Application Architecture (SAA). Instead, IBM hopes that one uses APPC/PC included with OS/2 Extended Edition to develop distributed applications -- especially in mixed computer Token- Rings. For strictly PC LANs, the issue is not as critical -- one is probably safe with LAN Manager or server-based products from other LAN vendors. Major third-party LAN vendors such as 3Com, Banyan, and Novell, are offering at least some level of compatibility (such as APPC) with OS/2 Extended Edition and OS/2 LAN Server and will increase this level of compatibility over time.

It appears that IBM is positioning the LAN Server product as a means to migrate from DOS to OS/2. This is supported by the fact that IBM has offered a trade-in allowance to upgrade to OS/2 from previous PC-DOS licenses and that the LAN Server operates under OS/2 and services both the DOS-based PC LAN Program and OS/2 NETBIOS emulation.

Chapter 7

NETBIOS Standards Efforts

Efforts are underway to implement NETBIOS in other network environments such as those based on the Department of Defense Transport Control Protocol/Internet Protocol (TCP/IP) or those based on the International Organization for Standardization (ISO) Transport and Network protocols. TCP/IP was defined by the U.S. Department of Defense for Arpanet and is now in widespread use in many commercial systems. ISO network and transport protocols are somewhat newer, and many vendors are migrating to them as a truly international standard.

TCP/IP

The following material is, in essence, the excerption of a Request For Comments (RFC) memo outlining the concept and methodology of the NETBIOS Working Group of the Internet Activities Board -- a critical working group whose existence and efforts are of special importance to NETBIOS devotees. Vendors that provide NETBIOS on top of TCP/IP include Excelan, Ungermann-Bass, 3Com/Bridge, Syntax, and Communication Machinery Corporation.

Memo Status

This Request For Comments specifies a proposed standard for the Internet community. Since this topic is new to the Internet community, discussions and suggestions are specifically requested. Written comments are to be sent to: Karl Auerbach, Epilogue Technology Corporation, P.O. Box 5432, Redwood City, CA 94063.

Acknowledgements

This Request For Comments has been developed under the auspices of the Internet Activities Board, especially the End-to-End Services Task Force.

The system proposed by this RFC does not reflect any existing NETBIOS-over-TCP implementation; however, the design incorporates considerable knowledge obtained from prior implementations. Special thanks goes to the following organizations which have provided this invaluable information: CMC/Syros; Excelan; Sytek; and Ungermann-Bass.

Introduction

This RFC describes the ideas and general methods used to provide NETBIOS on a TCP and UDP (User Datagram Protocol) foundation. A companion RFC, "Protocol Standard For a NETBIOS Service on a TCP/UDP Transport: Detailed Specifications" contains detailed descriptions of packet formats, protocols, and defined constants and variables.

The NETBIOS service has become the dominant mechanism for personal computer networking. NETBIOS provides a vendor- independent interface for the IBM PC and compatible systems.

NETBIOS defines a software interface, not a protocol: there is no "official" NETBIOS service standard. In practice, however, the IBM PC Network version is used as a reference. That version is described in the IBM document 6322916 "Technical Reference PC Network."

Protocols supporting NETBIOS services have been constructed on diverse protocol and hardware foundations. Even when the same foundation is used, different implementations may not be able to interoperate unless they use a common protocol. To allow NETBIOS interoperation in the Internet, this Request For Comments defines a standard protocol to support NETBIOS services using TCP and UDP.

NETBIOS has generally been confined to personal computers to date. However, since larger computers are often well-suited to run certain NETBIOS applications, such as file servers, this specification has been designed to allow an implementation to be built on virtually any type of system where the TCP/IP protocol suite is available.

This standard defines a set of protocols to support NETBIOS services.

These protocols are more than a simple communications service involving two entities: rather, this note describes a distributed system

in which many entities play a part even if they are not involved as an end-point of a particular NETBIOS connection.

This standard neither constrains nor determines how those services are presented to application programs.

Nevertheless, it is expected that on computers operating under the PC-DOS and MS-DOS operating systems, the existing NETBIOS interface will be preserved by implementers.

Design Principles

In order to develop the specification, the following design principles were adopted to guide the effort. Most are typical to any protocol standardization effort; however, some have been assigned priorities that may be considered unusual.

Preserve NETBIOS Services - In the absence of an "official" standard for NETBIOS services, the version found in the IBM PC Network Technical Reference is used.

NETBIOS is the foundation of a large body of existing applications. It is desirable to operate these applications on TCP networks and to extend them beyond personal computers into larger hosts. To support these applications, NETBIOS on TCP must closely conform to the services offered by existing NETBIOS systems.

IBM PC-Network NETBIOS contains some implementation-specific characteristics. This standard does not attempt to completely preserve these. It is certain that some existing software requires these characteristics and will fail to operate correctly on a NETBIOS service based on this Request For Comments.

Use Existing Standards - Protocol development, especially with standardization, is a demanding process. The development of new protocols must be minimized.

It is considered essential that an existing standard which provides the necessary functionality with reasonable performance always be chosen in preference to developing a new protocol.

When a standard protocol is used, it must be unmodified.

Minimize Options - The standard for NETBIOS on TCP should contain few, if any, options. Where options are included, the options should be designed so that devices with different option selections should interoperate.

Tolerate Errors And Disruptions - NETBIOS networks typically operate in an uncontrolled environment. Computers come on-line at

arbitrary times. Computers usually go off-line without any notice to their peers. The software is often operated by users who are unfamiliar with networks and who may randomly perturb configuration settings.

Despite this chaos, NETBIOS networks work. NETBIOS on TCP must also be able to operate well in this environment.

Robust operation does not necessarily mean that the network is proof against all disruptions. A typical NETBIOS network may be disrupted by certain types of behavior, whether inadvertent or malicious.

Do Not Require Central Management - NETBIOS on TCP should be able to operate, if desired, without centralized management beyond that typically required by a TCP-based network.

Allow Internet Operation - The proposed standard recognizes the need for NETBIOS operation across a set of networks interconnected by network (IP) level relays (gateways). However, the standard assumes that this form of operation will be less frequent than on the local MAC bridged-LAN.

Minimize Broadcast Activity - The standard presupposes that the only broadcast services are those supported by UDP. Multicast capabilities are not assumed to be available in any form.

Despite the availability of broadcast capabilities, the standard recognizes that some administrations may wish to avoid heavy broadcast activity. For example, an administration may wish to avoid isolated non-participating hosts from the burden of receiving and discarding NETBIOS broadcasts.

Permit Implementation On Existing Systems - The NETBIOS on TCP protocol should be implementable on common operating systems, such as UNIX and VAX/VMS, without massive effort.

The NETBIOS protocols should not require services typically unavailable on presently existing TCP/UDP/IP implementations.

Require Only The Minimum Necessary To Operate - The protocol definition should specify only the minimal set of protocols required for interoperation. However, additional protocol elements may be defined to enhance efficiency. These latter elements may be generated at the option of the sender, although they must be accepted by all receivers.

Maximize Efficiency - To be useful, a protocol must conduct its business quickly.

Minimize New Inventions - When an existing protocol is not quite able to support a necessary function, but with a small amount of change it could, that protocol should be used. This is felt to be easier to achieve than development of new protocols; further, it is likely to have more general utility for the Internet.

Facilities Supported

The protocol specified by this standard permits an implementer to provide all of the NETBIOS services as described in the IBM "Technical Reference PC Network".

The following NETBIOS facilities are outside the scope of this specification. These are local implementation matters and do not impact interoperability:

- RESET
- SESSION STATUS
- UNLINK
- RPL (Remote Program Load)

Required Interfaces and Definitions

The protocols described in this RFC require service interfaces to the following:

- TCP
- UDP

Byte ordering, addressing conventions (including addresses to be used for broadcasts and multicasts) are defined by the most recent version of:

- Assigned Numbers

Additional definitions and constraints are in:

- IP
- Internet Subnets

Related Protocols and Services

The design of the protocols described in this RFC allow for the future incorporation of the following protocols and services. However, before this may occur, certain extensions may be required to the protocols defined in this RFC or to those listed below.

- Domain Name Service
- Internet Group Multicast

NETBIOS Scope

A "NETBIOS Scope" is the population of computers across which a registered NETBIOS name is known. NETBIOS broadcast and multicast datagram operations must reach the entire extent of the NETBIOS scope.

An internet may support multiple, non-intersecting NETBIOS Scopes.

Each NETBIOS scope has a "scope identifier". This identifier is a character string meeting the requirements of the domain name system for domain names.

NOTE: Each implementation of NETBIOS-over-TCP must provide mechanisms to manage the scope identifier(s) to be used.

Control of scope identifiers implies a requirement for additional NETBIOS interface capabilities. These may be provided through extensions of the user service interface or other means (such as node configuration parameters.) The nature of these extensions is not part of this specification.

NETBIOS End-nodes

End-nodes support NETBIOS service interfaces and contain applications.

Three types of end-nodes are part of this standard:

- Broadcast ("B") nodes
- Point-to-point ("P") nodes
- Mixed mode ("M") nodes

An IP address may be associated with only one instance of one of the above types.

Without having preloaded name-to-address tables, NETBIOS participants are faced with the task of dynamically resolving references to one another. This can be accomplished with broadcast or mediated point- to-point communications.

B nodes use local network broadcasting to effect a rendezvous with one or more recipients. P and M nodes use the NETBIOS Name Server name server (NBNS) and the NETBIOS Datagram Distribution Server same purpose.

End-nodes may be combined in various topologies. No matter how combined, the operation of the B, P, and M nodes is not altered.

NOTE: It is recommended that the administration of a NETBIOS scope avoid using both M and B nodes within the same scope. A NETBIOS scope should contain only B nodes or only P and M nodes.

Broadcast Nodes

Broadcast (or "B") nodes communicate using a mix of UDP datagrams (both broadcast and directed) and TCP connections. B nodes may freely interoperate with one another within a broadcast area. A broadcast area is a single MAC-bridged "B-LAN".

Point-to-Point Nodes

Point-to-point (or "P") nodes communicate using only directed UDP datagrams and TCP sessions. P nodes neither generate nor listen for broadcast UDP packets. P nodes do, however, offer NETBIOS level broadcast and multicast services using capabilities provided by the NBNS and NBDD.

P nodes rely on NETBIOS name and datagram distribution servers. These servers may be local or remote; P nodes operate the same in either case.

Mixed Mode Nodes

Mixed mode nodes (or "M") nodes are P nodes which have been given certain B node characteristics. M nodes use both broadcast and unicast. Broadcast is used to improve response time using the assumption that most resources reside on the local broadcast medium rather than somewhere in an internet.

M nodes rely upon NBNS and NBDD servers. However, M nodes may continue limited operation should these servers be temporarily unavailable.

Support Servers

Two types of support servers are part of this standard:

- NETBIOS name server ("NBNS") nodes
- Netbios datagram distribution ("NBDD") nodes

NBNS and NBDD nodes are invisible to NETBIOS applications and are part of the underlying NETBIOS mechanism.

NETBIOS name and datagram distribution servers are the focus of name and datagram activity for P and M nodes.

Both the name (NBNS) and datagram distribution (NBDD) servers are permitted to shift part of their operation to the P or M end-node which is requesting a service.

Since the assignment of responsibility is dynamic, and since P and M nodes must be prepared to operate should the NETBIOS server delegate control to the maximum extent, the system naturally accommodates improvements in NETBIOS server function. For example, as Internet Group Multicasting becomes more widespread, new NBDD implementations may elect to assume full responsibility for NETBIOS datagram distribution.

Interoperability between different implementations is assured by imposing requirements on end-node implementations that they be able to accept the full range of legal responses from the NBNS or NBDD.

Name Server Nodes

The NBNS is designed to allow considerable flexibility with its degree of responsibility for the accuracy and management of NETBIOS names. On one hand, the NBNS may elect not to accept full responsibility, leaving the NBNS essentially a "bulletin board" on which name/address information is freely posted (and removed) by P and M nodes without validation by the NBNS. Alternatively, the NBNS may elect to completely manage and validate names. The degree of responsibility that the NBNS assumes is asserted by the NBNS each time a name is claimed through a simple mechanism. Should the NBNS not assert full control, the NBNS returns enough information to the requesting node so that the node may challenge any putative holder of the name.

This ability to shift responsibility for NETBIOS name management between the NBNS and the P and M nodes allows a network administrator (or vendor) to make a tradeoff between NBNS simplicity, security, and delay characteristics.

A single NBNS may be implemented as a distributed entity, such as the Domain Name Service. However, this RFC does not attempt to define the internal communications which would be used.

Topologies

B, P, M, NBNS, and NBDD nodes may be combined in various ways to form useful NETBIOS environments.

There are three classes of operation:

- Class 0: B nodes only.
- Class 1: P nodes only.
- Class 2: P and M nodes together.

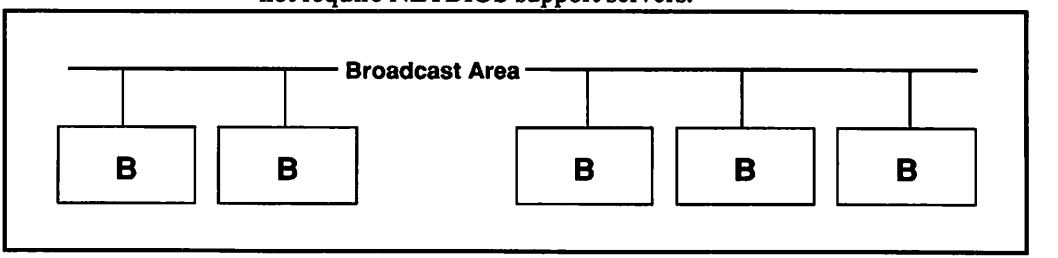
In the figures which follow, any P node may be replaced by an M node. The effects of such replacement will be mentioned in conjunction with each example below.

Local

A NETBIOS scope is operating locally when all entities are within the same broadcast area.

B Nodes

Local operation with only B nodes is the most basic mode of operation. Name registration and discovery procedures use broadcast mechanisms. The NETBIOS scope is limited by the extent of the broadcast area. This configuration as illustrated in Figure 7-1, does not require NETBIOS support servers.

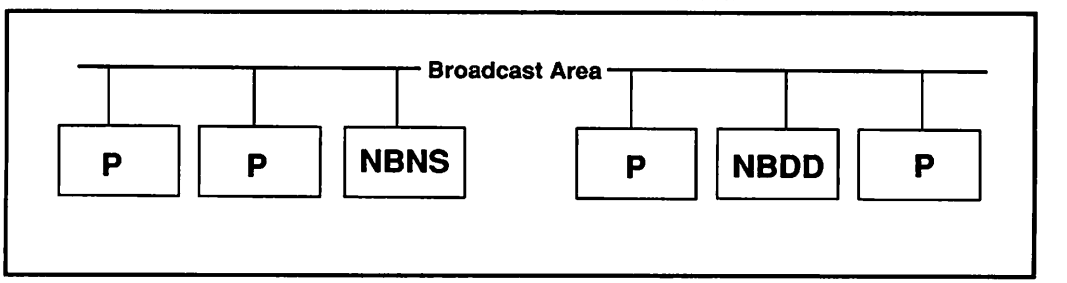


7-1: B Nodes

P Nodes

This configuration would typically be used when the network administrator desires to eliminate NETBIOS as a source of broadcast activity.

This configuration as illustrated in Figure 7-2, operates the same as if it were in an internet and is cited here only due to its convenience as a means to reduce the use of broadcast.



7-2: P Nodes

Replacement of one or more of the P nodes with M nodes will not affect the operation of the other P and M nodes. P and M nodes will be able to interact with one another. Because M nodes use broadcast, overall broadcast activity will increase.

Mixed B and P Nodes

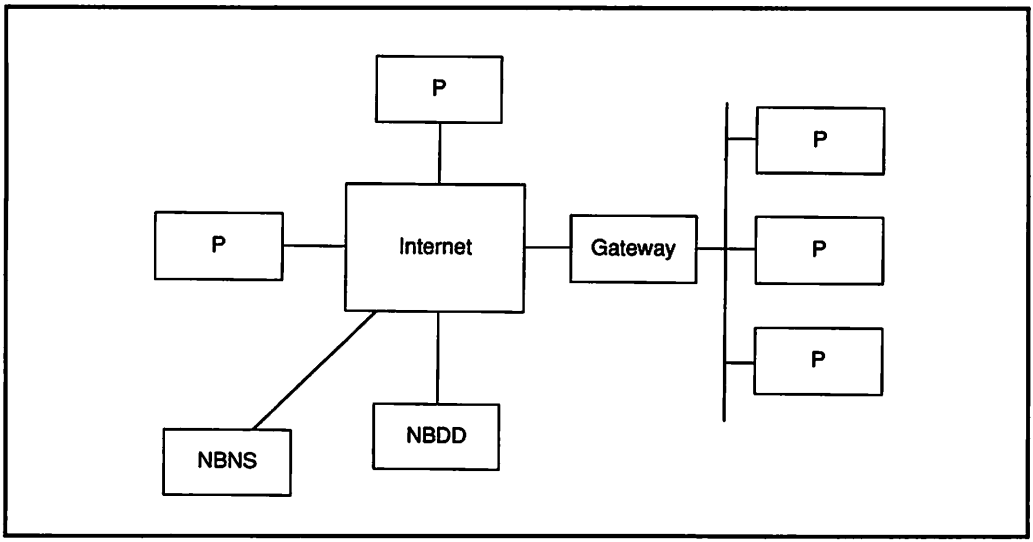
B and P nodes do not interact with one another. Replacement of P nodes with M nodes will allow B's and M's to interact.

B nodes and M nodes may be intermixed only on a local broadcast area. B and M nodes should not be intermixed in an internet environment.

Internet

P Nodes

P nodes may be scattered at various locations in an internetwork as shown in Figure 7-3. They require both an NBNS and an NBDD for NETBIOS name and datagram support, respectively.



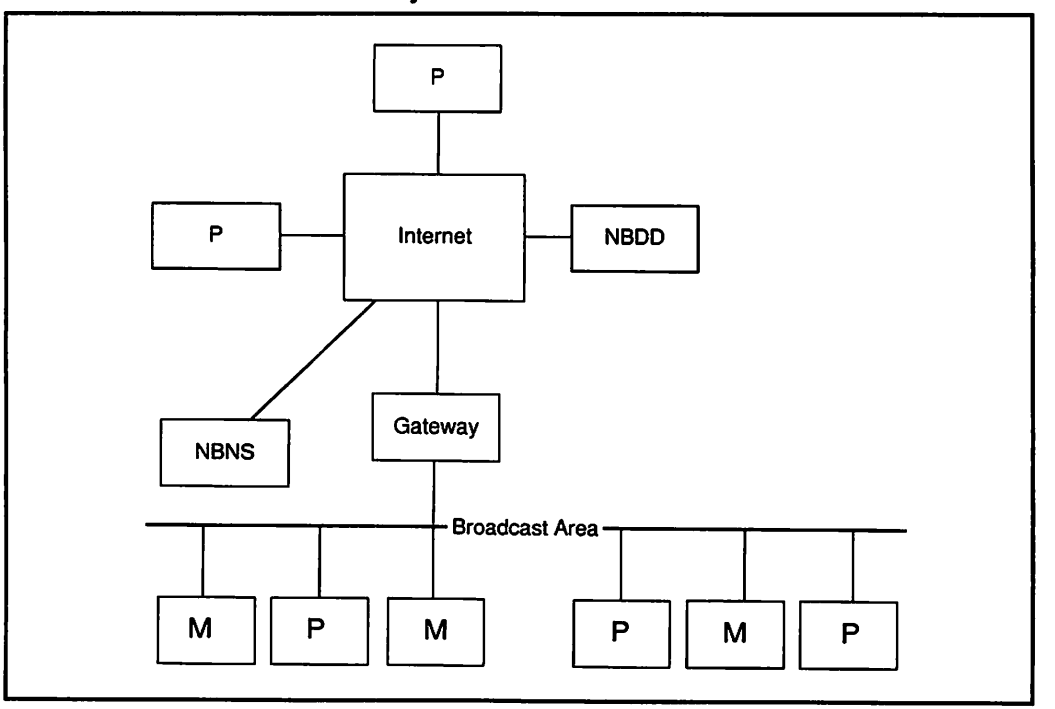
7-3: Internet P Nodes

The NETBIOS scope is determined by the NETBIOS scope identifier (domain name) used by the various P (and M) nodes. An internet may contain numerous NETBIOS scopes.

Any P node may be replaced by an M node with no loss of function to any node. However, broadcast activity will be increased in the broadcast area to which the M node is attached.

Mixed M and P Nodes

As illustrated in Figure 7-4, M and P nodes may be mixed. When locating NETBIOS names, M nodes will tend to find names held by other M nodes on the same common broadcast area in preference to names held by P nodes or M nodes elsewhere in the network.



7-4: Internet P & M Nodes

NOTE: B and M nodes should not be intermixed in an internet environment. Doing so would allow undetected NETBIOS name conflicts to arise and cause unpredictable behavior.

General Methods

Overlying the specific protocols, described later, are a few general methods of interaction between entities.

Request/Response Interaction

Most interactions between entities consist of a request flowing in one direction and a subsequent response flowing in the opposite direction.

In those situations where interactions occur on unreliable transports (i.e. UDP) or when a request is broadcast, there may not be a strict interlocking or one-to-one relationship between requests and responses.

In no case, however, is more than one response generated for a received request. While a response is pending the responding entity may send one or more wait acknowledgements.

Retransmission of Requests

UDP is an unreliable delivery mechanism where packets can be lost, received out of transmit sequence, duplicated and delivery can be significantly delayed. Since the NETBIOS protocols make heavy use of UDP, they have compensated for its unreliability with extra mechanisms.

Each NETBIOS packet contains all the necessary information to process it. None of the protocols use multiple UDP packets to convey a single request or response. If more information is required than will fit in a single UDP packet, for example, when a P-type node wants all the owners of a group name from a NETBIOS server, a TCP connection is used. Consequently, the NETBIOS protocols will not fail because of out of sequence delivery of UDP packets.

To overcome the loss of a request or response packet, each request operation will retransmit the request if a response is not received within a specified time limit.

Protocol operations sensitive to successive response packets, such as name conflict detection, are protected from duplicated packets because they ignore successive packets with the same NETBIOS information. Since no state on the responder's node is associated with a request, the responder just sends the appropriate response whenever a request packet arrives. Consequently, duplicate or delayed request packets have no impact.

For all requests, if a response packet is delayed too long another request packet will be transmitted. A second response packet being sent in response to the second request packet is equivalent to a duplicate packet. Therefore, the protocols will ignore the second packet

received. If the delivery of a response is delayed until after the request operation has been completed, successfully or not, the response packet is ignored.

Requests Without Responses

Some request types do not have matching responses. These requests are known as "demands". In general a "demand" is an imperative request; the receiving node is expected to obey. However, because demands are unconfirmed, they are used only in situations where, at most, limited damage would occur if the demand packet should be lost.

Demand packets are not retransmitted.

Transactions

Interactions between a pair of entities are grouped into "transactions". These transactions comprise one or more request/response pairs.

Transaction ID

Since multiple simultaneous transactions may be in progress between a pair of entities a "transaction ID" is used.

The originator of a transaction selects an ID unique to the originator. The transaction ID is reflected back and forth in each interaction within the transaction. The transaction partners must match responses and requests by comparison of the transaction ID and the IP address of the transaction partner. If no matching request can be found the response must be discarded.

A new transaction ID should be used for each transaction. A simple 16 bit transaction counter ought to be an adequate ID generator. It is probably not necessary to search the space of outstanding transaction ID to filter duplicates: it is extremely unlikely that any transaction will have a lifetime that is more than a small fraction of the typical counter cycle period. Use of the IP addresses in conjunction with the transaction ID further reduces the possibility of damage should transaction IDs be prematurely re-used.

TCP and UDP Foundations

This version of the NETBIOS-over-TCP protocols uses UDP for many interactions. In the future this RFC may be extended to permit such interactions to occur over TCP connections (perhaps to increase efficiency when multiple interactions occur within a short time or when NETBIOS datagram traffic reveals that an application is using NETBIOS datagrams to support connection- oriented service.)

NETBIOS Session Service

The NETBIOS session service begins after one or more IP addresses have been found for the target name. These addresses may have been acquired using the NETBIOS name query transactions or by other means, such as a local name table or cache.

NETBIOS session service transactions, packets, and protocols are identical for all end-node types. They involve only directed (point-to-point) communications.

Session service has three phases:

Session establishment - it is during this phase that the IP address and TCP port of the called name is determined, and a TCP connection is established with the remote party.

Steady state - it is during this phase that NETBIOS data messages are exchanged over the session. Keep-alive packets may also be exchanged if the participating nodes are so configured.

Session close - a session is closed whenever either a party (in the session) closes the session or it is determined that one of the parties has gone down.

NETBIOS Datagram Service

Every NETBIOS datagram has a named destination and source. To transmit a NETBIOS datagram, the datagram service must perform a name query operation to learn the IP address and the attributes of the destination NETBIOS name. (This information may be cached to avoid the overhead of name query on subsequent NETBIOS datagrams.)

NETBIOS datagrams are carried within UDP packets. If a NETBIOS datagram is larger than a single UDP packet, it may be fragmented into several UDP packets.

End-nodes may receive NETBIOS datagrams addressed to names not held by the receiving node. Such datagrams should be discarded. If the name is unique then a DATAGRAM ERROR packet is sent to the source of that NETBIOS datagram.

Datagram Fragmentation

When the header and data of a NETBIOS datagram exceeds the maximum amount of data allowed in a UDP packet, the NETBIOS datagram must be fragmented before transmission and reassembled upon receipt.

A NETBIOS Datagram is composed of the following protocol elements:

- IP header of 20 bytes (minimum)
- UDP header of 8 bytes
- NETBIOS Datagram Header of 14 bytes
- The NETBIOS Datagram data.

The NETBIOS Datagram data section is composed of 3 parts:

- Source NETBIOS name (255 bytes maximum)
- Destination NETBIOS name (255 bytes maximum)
- The NETBIOS user's data (maximum of 512 bytes)
- The two name fields are in second level encoded format.

A maximum size NETBIOS datagram is 1064 bytes. The minimal maximum IP datagram size is 576 bytes. Consequently, a NETBIOS Datagram may not fit into a single IP datagram. This makes it necessary to permit the fragmentation of NETBIOS Datagrams.

On networks meeting or exceeding the minimum IP datagram length requirement of 576 octets, at most two NETBIOS datagram fragments will be generated. The protocols and packet formats accommodate fragmentation into three or more parts.

Node Configuration Parameters

B NODES:

- Node's permanent unique name
- Whether IGMP is in use
- Broadcast IP address to use
- Whether NETBIOS session keep-alives are needed
- Usable UDP data field length (to control fragmentation)

P NODES:

- Node's permanent unique name
- IP address of NBNS
- IP address of NBDD
- Whether NETBIOS session keep-alives are needed
- Usable UDP data field length (to control fragmentation)

M NODES:

- Node's permanent unique name
- Whether IGMP is in useBroadcast IP address to use
- IP address of NBNS
- IP address of NBDD
- Whether NETBIOS session keep-alives are needed
- Usable UDP data field length (to control fragmentation)

Minimal Conformance

To ensure multi-vendor interoperability, a minimally conforming implementation based on this specification must observe the following rules:

a) A node designed to work only in a broadcast area must conform to the B node specification.

b) A node designed to work only in an internet must conform to the P node specification.

ISO

The following information regarding NETBIOS for ISO Networks is written by Stephen Thomas of Communication Machinery Corporation, Tempe, Arizona. It originally appeared in the July/August 1987 issue of Computer Communication Review.

Introduction

The protocol suite specified by the International Organization for Standarization (ISO) promises universal interoperability. Network products from many different vendors will communicate with each other, and users will be able to mix and match equipment to more perfectly satisfy their needs. Unfortunately, the ISO protocol suite has not yet matured, and users cannot yet acquire all the ISO application programs that they need today.

In the world of IBM Personal Computers, most network application programs use IBM's NETBIOS interface, and until those applications are replaced by applications that conform to ISO standards, users will continue to require NETBIOS-compatible networks. Fortunately, a NETBIOS interface can be added to ISO network products. When an ISO product includes a NETBIOS- compatible interface, users can run both ISO applications and standard PC applications over the same network.

Communication Machinery Corporation (CMC) includes a NETBIOS- compatible interface with its Technical Office Protocol (TOP) products for IBM personal computers, and, because of the populari-

ty of the NETBIOS interface, several other vendors will eventually offer NETBIOS interfaces to their ISO products.

The simplest way to add NETBIOS-compatibility to a protocol suite is to define a "NETBIOS protocol" at the top-most level of that suite. This approach has been taken by the group of vendors standardizing NETBIOS over TCP/IP networks. In effect, they have defined a new protocol that resides above TCP on their network products.

This method suffers from a major disadvantage, however, because the new protocol must be used by both endpoints of the connection. A PC with such a NETBIOS protocol can communicate with a UNIX system, for example, only if the UNIX system runs a special program or driver that supports the NETBIOS protocol, and using that special driver may preclude running normal UNIX network applications.

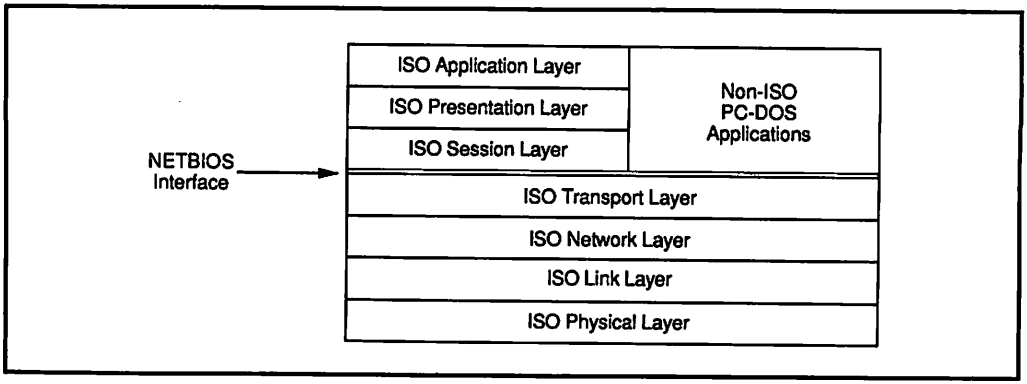
CMC has eliminated this problem in its ISO products by defining a NETBIOS interface for ISO that serves only as an interface. CMC achieves NETBIOS compatibility without adding another protocol, and CMC's NETBIOS becomes a transparent interface to normal ISO protocols. Even standard ISO applications communicate with CMC's network hardware through the NETBIOS interface. NETBIOS-equipped PCs communicate with UNIX, VMS, and other systems that do not even realize that they are talking to a PC.

By opening publishing details of CMC's implementation of its NETBIOS-ISO interface, CMC hopes to encourage other vendors to adopt the same approach in adding NETBIOS-compatibility to their ISO network products. Such a standardization would allow NETBIOS-ISO products from many vendors to communicate with each other, and it would benefit not only the vendors themselves but, more importantly, network users.

NETBIOS as a Transport Level Interface

If NETBIOS serves as a transparent interface to the ISO protocol suite, it must, of course, interface to one or more ISO protocols. Because the NETBIOS specification provides reliable data delivery on logical connections, it most naturally interfaces to the ISO transport layer. Figure 7-5 illustrates the place in the OSI model best suited for a NETBIOS interface.

NETBIOS requires four different types of network services: general services, name services, "session" services, and datagram services. The first two of these, general services and name services, do not really fit any ISO protocol, so only the last two services affect the choice of a protocol layer. Since NETBIOS "sessions" must be reliable, NETBIOS must interface at the ISO transport layer or higher.



7-5: NETBIOS Interface and the OSI Model

But NETBIOS does not provide an interface to higher layer services such as synchronization, activity management, and token management. If NETBIOS were selected as an interface to a protocol layer above transport, it would only offer an interface to a small subset of the layer's services. The most natural protocol layer for a NETBIOS interface, therefore, is the transport layer.

Choosing a transport layer interface contradicts the terminology of IBM's PC Network Technical Reference, which refers to NETBIOS as a session layer interface. The discrepancy arises because the session layer protocol to which IBM's NETBIOS interfaces is not the ISO session protocol. In fact, IBM's session layer does not meet the general criteria for a session layer established by the OSI reference model.

Within the transport layer, Class 4 of the connection-oriented transport protocol (TP4) best supports NETBIOS "sessions." Because NETBIOS "sessions" must be reliable and because NETBIOS cannot, as an interface, perform error detection and recovery, NETBIOS requires the full error detection and recovery services that TP4 provides.

NETBIOS Names

NETBIOS names identify the specific computer on which they reside, and since a single computer may have several names, they also define a "NETBIOS service access point" at that computer. If NETBIOS interfaces to the ISO transport layer then, by definition, NETBIOS names must correspond to ISO addresses at the transport level. Each NETBIOS name, therefore corresponds to a unique Network Service Access Point (NSAP) and Transport Service Access Point (TSAP) pair. The NETBIOS interface translates between names and NSAP-

TSAP pairs before sending NETBIOS commands to the ISO transport protocol.

CMC's local directory service protocol, the Dynamic Naming Protocol (DNP), actually performs the translation. That protocol, discussed in "A Dynamic Naming Protocol for ISO Networks," provides name services for all of CMC's ISO protocols, not just the NETBIOS interface.

When a NETBIOS user issues an ADD NAME or ADD GROUP NAME command, the NETBIOS interface formulates a TSAP for that name and asks DNP to register the specified name with that TSAP. Because NETBIOS interfaces to the transport level, a NETBIOS name has no session or presentation layer service access point (SSAP or PSAP).

When a NETBIOS user issues a DELETE NAME command, the NETBIOS interface checks to see if the specified name has any active "sessions." If the name has none, the NETBIOS interface deletes the name immediately and asks DNP to de-register it. If the name does have active "sessions," however, the NETBIOS interface simply marks the name appropriately. Only after all the "sessions" have closed does the NETBIOS interface ask DNP to de- register the name.

NETBIOS users may optionally use permanent node names instead of network names. A permanent node name consists of ten bytes of zeroes followed by six bytes which designate the node name of the computer. CMC's ISO NETBIOS takes the six non-zero bytes to be the sub-network point of attachment (SNPA) for the computer, and it builds a network service access point (NSAP) from this SNPA.

For the rest of the permanent node name's NSAP, the interface simply copies from its own NSAP. All permanent node names for a given local network will therefore have identical authority and format identifiers (AFI), network identifiers (NID), primary subnetwork identifiers (PSI), LSAP suffixes (LSS), and NSAP suffixes (NSS).

The NETBIOS interface also uses a specific, designated TSAP for permanent node names. Currently that TSAP is two bytes, each with the hexadecimal value FE. Figure 7-6 shows an example of the translation between permanent node name and NSAP-TSAP address. (Note that the NETBIOS interface recognizes and translates permanent node names; CMC's Dynamic Naming Protocol has no conception of NETBIOS permanent node names.)

If local NSAP is:
49 01 00 00 00 00 01 y1 y2 y3 y4 y5 y6 FE 00

permanent node name:
00 00 00 00 00 00 00 00 00 00 x1 x2 x3 x4 x5 x6

converts to:
NSAP - 49 01 00 00 00 00 01 x1 x22 x3 x4 x5 x6 FE 00
TSAP - FE FE

7-6: Node Name to NSAP Conversion

NETBIOS "Session" Services

The NETBIOS interface maps "sessions" transparently onto TP4 connections. Each "session" has its own connection, and the interface transfers data for a "session" directly to its connection. The NETBIOS interface never generates data to send on a connection.

When a NETBIOS user asks the NETBIOS interface to establish a "session," the user usually refers to the endpoints of that "session" with NETBIOS names. The interface must, of course, translate those names into network addresses (NSAPs and TSAPs) before it can establish a transport connection. CMC's Dynamic Naming Protocol (see "A Dynamic Naming Protocol for ISO Networks") performs this translation.

Once the interface establishes a transport connection, applications may transfer data on that connection by issuing SEND, CHAIN SEND, RECEIVE, and RECEIVE ANY commands. In general, these commands simply send and receive data on the TP4 connection; however, minor differences between NETBIOS data transfer and TP4 data transfer exist.

The most obvious difference is that NETBIOS does not support the TP4 concept of expedited data. Consequently, application programs restricted to the standard NETBIOS interface cannot send TP4 expedited data. For applications that require expedited data, CMC provides extensions to NETBIOS to allow its use. A subsequent section discusses those extensions.

The second difference between TP4 and NETBIOS results in a minor deviation from the NETBIOS interface specified in the PC Network Technical Reference. Data delivery as provided by the ISO transport layer is an unconfirmed service. When a TP4 user sends data, it can

count on the data being sent, but TP4 does not indicate when the received data is actually delivered to the remote user. In effect, the sending user has to trust TP4 to deliver the data.

The NETBIOS interface, however, specified that a SEND (or CHAIN SEND) command does not complete until the remote user has actually received the data. Because CMC's NETBIOS interface only uses services that are normally available from TP4, it returns completed SEND commands when TP4 indicates that they have completed. TP4 may indicate a completion before the remote user actually receives the data, so a SEND command may complete before the remote user's RECEIVE command completes. This behavior differs slightly from the NETBIOS specification, but, except under unusual circumstances, applications cannot detect the difference.

NETBIOS Datagram Services

The NETBIOS services most difficult to satisfactorily implement over an ISO network are, perhaps, the NETBIOS datagram services. NETBIOS datagrams, like datagram services for other protocol suites, are intended as a simple, quick, and efficient method of data transfer. NETBIOS datagrams, however, use network names, and name services are rarely simple, quick, or efficient.

Because of this basic incompatibility, all methods of implementing NETBIOS datagrams within ISO result in compromises. CMC has chosen a method that it feels to be the best compromise, and this section details that implementation. This section also outlines other possible implementations, and it describes CMC's reasons for rejecting those approaches.

A straightforward implementation would have NETBIOS datagrams use the ISO connectionless transport protocol as NETBIOS "sessions" use TP4. When a user sends a datagram, the NETBIOS interface requests the name services convert the destination name into an NSAP and a TSAP; the interface then sends the datagram to that NSAP-TSAP. When the destination node receives the datagram, it asks the name services to convert the source NSAP and TSAP into the sender's network name. The interface then delivers the datagram to the remote user.

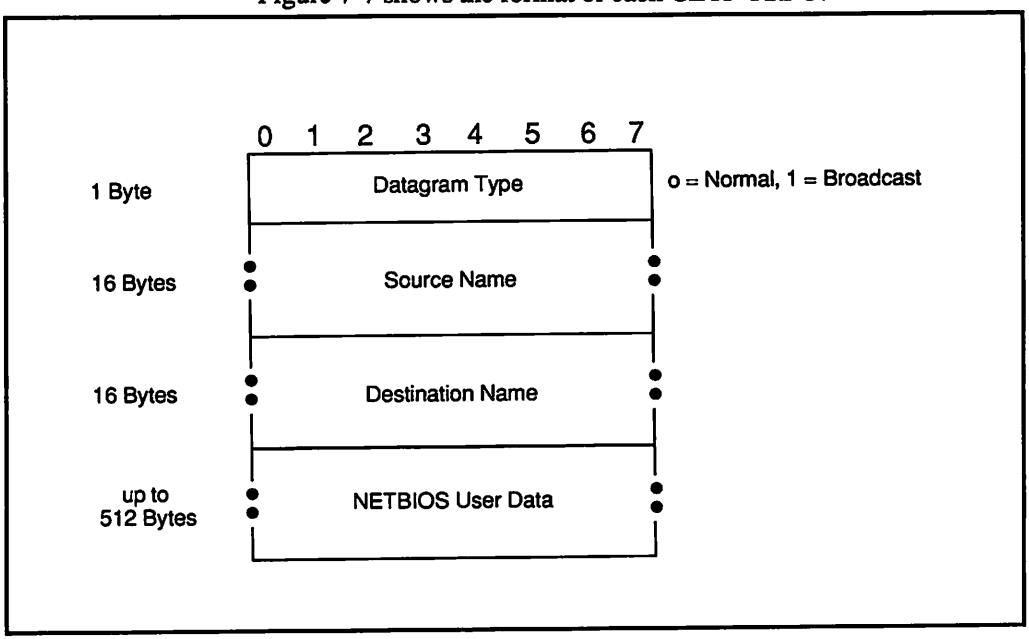
This method has two major flaws, however, and consequently CMC's NETBIOS interface cannot use it. The most obvious flaw is the method's inefficiency: it requires five network transactions in order to send a single datagram. (The five transactions are the name discovery request, the name discovery response, the datagram, the address resolution request, and the address resolution response.)

The method also treats group names very poorly, its second major flaw. When an application directs a datagram to a group name, this method requires that every node with the group name be identified and that the datagram be copied and sent to each of these nodes.

As an alternative, the NETBIOS interface may broadcast every datagram and include in the datagram the source and destination names. This method significantly improves upon the efficiency of the first; no name services are required and therefore each datagram results in only one network transaction.

At first glance, it may appear that having all nodes on the network receive the datagram offsets this advantage. Closer examination, however, shows this assumption to be incorrect. The first method also requires that all nodes on the network process one network transaction per datagram. Instead of the datagram itself, each node must process the name discovery request.

CMS uses this second approach in its NETBIOS interface. The interface places the application datagram, along with the source and destination names and the datagram type, in service data units (SDUs) of the ISO connectionless transport protocol (CLTP), DIS 8602. Figure 7-7 shows the format of each CLTP TSDU.



7-7: NETBIOS Datagram CLTP TSDU

To broadcast these datagrams, CMC uses a destination NSAP which contains a MAC-layer broadcast address as its sub-network point of attachment (SNPA). All other fields of the NSAP (AFI, NID, PSI, LSS, and NSS) are copied from the adapter's local NSAP. A sample NSAP that the NETBIOS datagram services could use is (in hexadecimal) 49 01 00 00 00 00 01 FF FF FF FF FF FF FE 00. All NETBIOS datagrams also use a single, well-known TSAP for both source and destination. That TSAP consists of two bytes; the first byte has a value of zero and the second byte has a value of (hexadecimal) 81.

This approach suffers from one significant disadvantage, a lack of transparency. By including the source and destination names within connectionless transport's service data unit, CMC requires the destination node support the NETBIOS interface. Fortunately, nodes that do not support NETBIOS do not receive NETBIOS datagrams. The NETBIOS interface directs all datagrams to a specific TSAP, so only those protocols which listen for datagrams on that TSAP will receive NETBIOS datagrams.

ISO Extensions to NETBIOS

Because the NETBIOS interface was not specifically designed for ISO protocols, it does not present a perfect interface to those protocols. CMC has minimized this mismatch by choosing to interface to the ISO Transport layer; nonetheless, there are a few features of the ISO protocols that cannot be supported strictly within the NETBIOS standard.

Specifically, applications have no direct access to NSAPs and TSAPs for addressing, nor can they send expedited data or explicitly indicate the end-of-text within a message. In order to provide these features, CMC has extended the NETBIOS interface used by its TOP-NETBIOS products.

To access these ISO extensions, application programs place a specific "signature" in the NETBIOS NCB. That signature is placed in the first four bytes of the NCB's CALLNAME field, and it consists of a byte with the binary value of zero, followed by three bytes with the values, in ASCII code, for uppercase "I", uppercase "S", and uppercase "O". Since standard NETBIOS does not allow names to begin with a binary value of zero, non-ISO applications will not use a name beginning with the ISO signature, and a non-ISO application cannot inadvertently request an ISO extension.

Furthermore, if an ISO application tries to use an ISO extension over a non-ISO NETBIOS, the network adapter will, in most cases, simp-

ly reject the NCB because it has an invalid name. This method ensures that only ISO applications can use an ISO extension, and it ensures that no harm results when an ISO application mistakenly tries to use a non-ISO network.

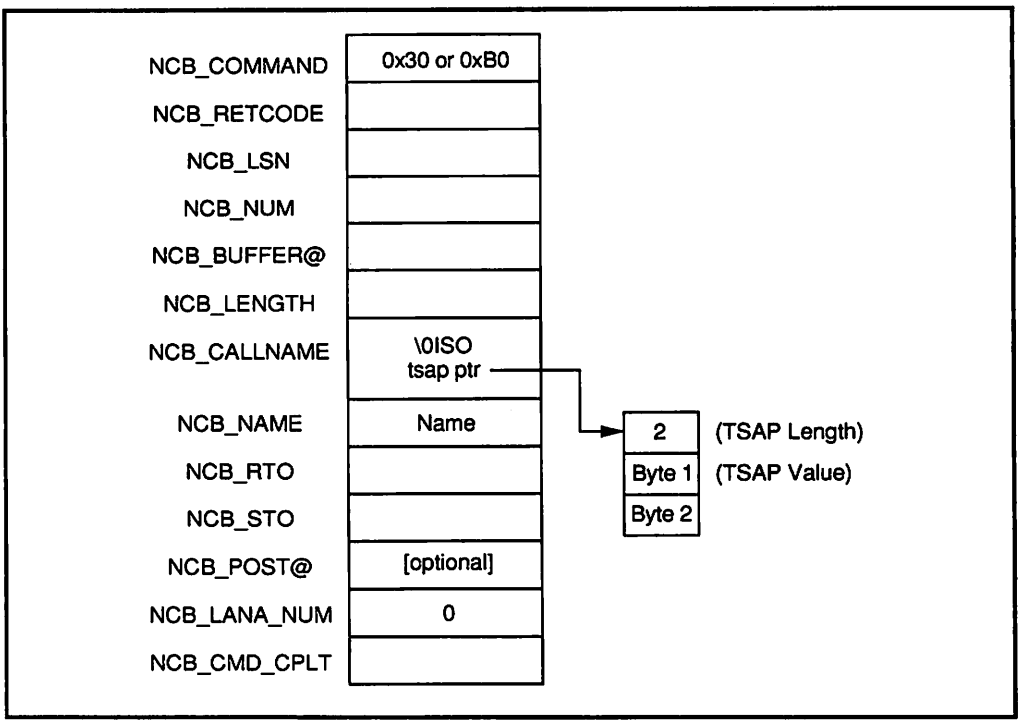
The first ISO extension allows applications to specify a TSAP when they register a name. Normally the NETBIOS interface chooses a TSAP for them. To specify a TSAP, the application places the ISO signature in bytes zero through three of the NCB_CALLNAME field of its ADD NAME NCB. It also places a pointer to the specified TSAP in bytes four through seven of the NCB_CALLNAME field.

That pointer, like all NETBIOS consists of an offset and a segment. Byte four contains the least significant byte of the offset, byte five the most significant; byte six contains the least significant byte of the segment, and byte seven contains its most significant byte. The pointer points to the TSAP structure, the first byte of which holds the length of the TSAP. The bytes that actually comprise the TSAP follow the length indicator. In the ISO tradition, the length indicator does not include itself in its count. The rest of the NCB, including the name itself, duplicates the standard NETBIOS ADD NAME command. Figure 7-8 shows a sample ISO-extended ADD NAME command.

The second ISO extension allows applications to call each other by directly specifying the remote NSAP and TSAP. Normally, of course, NETBIOS expects applications to specify a remote name; it then uses the Dynamic Naming Protocol to discover that name's address. To use NSAPs and TSAPs directly, the application again places the ISO signature in the NCB callname field. Following that signature, it places a pointer to the NSAP it is calling and then a pointer to the TSAP it is calling.

Both pointers are again specified with offsets and segments. The NSAP pointer occupies bytes four through seven of the NCB_CALLNAME field, and the TSAP pointer occupies bytes eight through eleven. Each pointer points to a length indicator, and the bytes of the NSAP or TSAP immediately follow that length indicator. Figure 7-9 shows a sample ISO-extended CALL command.

A third ISO extension provides expedited data services to the application. It also allows an application to explicitly specify whether or not the end-of-transmission flag is set. To use these features, the application places the ISO signature in the SEND NCB. (Neither feature may be used in a CHAIN SEND command.)

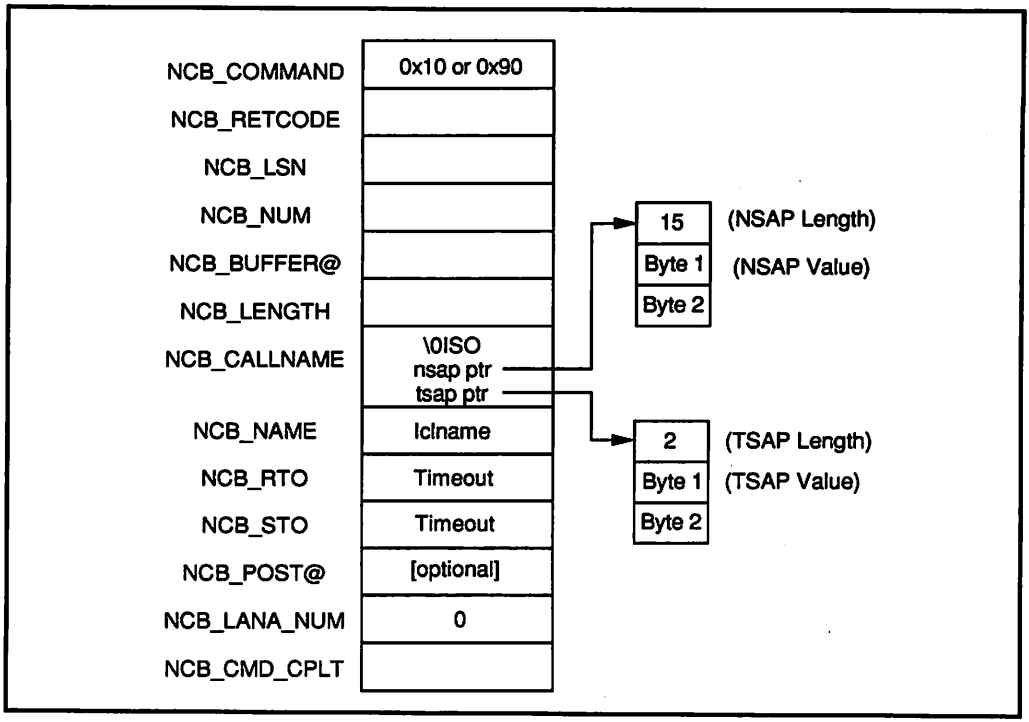


7-8: ISO-extended ADD NAME Command

In byte four of the NCB_CALLNAME field, the application sets specific bits to indicate whether or not to use expedited data and whether or not to set end-of-transmission. By setting the fourth least significant bit to one (xxxx1xxx), the application selects end-of-transmission; a zero (xxxx0xxx) indicates additional data. Similarly, the fifth least significant bit selects expedited (xxx1xxxx) or normal (xxx0xxxx) data delivery.

To receive expedited data, the application must place the ISO signature in its RECEIVE and RECEIVE ANY commands. When NETBIOS returns the NCB, it will fill in byte four of the NCB_CALLNAME field with the bits described above. The fourth least significant bit indicates end-of-transmission (xxxx1xxx or xxxx0xxx), and the fifth least significant bit indicates expedited data (xxx1xxxx or xxx0xxxx).

The ISO signature also tells the NETBIOS interface that the application can accept the extended return code of "Message interrupted" (hexadecimal value 0x07). The NETBIOS interface uses this non-standard return code when it has begun reassembly of a data message



7-9: ISO-extended CALL Command

and receives expedited data before that message completes. NETBIOS obviously cannot reassemble the expedited data into the message in progress, so it returns the NCB being used for reassembly with the "Message interrupted" return code. The next RECEIVE or RECEIVE ANY that completes will then contain the expedited data, and, after it, the following NCB will contain the remainder of the first message.

If an application does not place the ISO signature in its RECEIVE and RECEIVE ANY commands and NETBIOS receives expedited data, NETBIOS will still try to deliver that data. As long as NETBIOS has not begun a reassembly process for that NCB, it can package the received data normally and return the completed NCB. The application, of course, will be unaware that the data is expedited. If a reassembly process has begun on the NCB, however, the NETBIOS interface cannot use the extended return code and has no choice but to abort the "session."

This chapter has presented Communication Machinery Corporation's NETBIOS-to-ISO protocol interface. Because that interface is

transparent, both ISO applications and standard IBM PC applications can use NETBIOS as an interface to the network. Standard PC applications may communicate with each other, and ISO applications on the PC communicate with each other and with normal ISO applications on other, non-PC systems. CMC welcomes questions and comments on this NETBIOS-ISO interface.

Appendix A: Acronyms

API	Application Programming Interface
APPC	Advanced Program-to-Program Communication
ASCII	American Standard Code for Information Interchange
ASYNC	Asynchronous
BIOS	Basic Input Output System
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
DMA	Direct Memory Access
DOS	Disk Operating System
DSAP	Destination Service Access Point
EBCDIC	Extended Binary-Coded Decimal Interchange
EPROM	Erasable Programmable Read-Only Memory
FAT	File Allocation Table
FCS	Frame Check Sequence
HDLC	High Level Data Link Control
ID	IDentification
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
Kbps	thousand bits per second
LAN	Local Area Network
LLC	Logical Link Control
LPDU	Logical Link Protocol Data Unit
LSAP	Link Service Access Point

MAC	Media Access Control
Mbps	Million bits per second
MCB	Message Control Block
MMIO	Memory Mapped I/O
MS-DOS	Microsoft Disk Operating System
NCB	Network Control Block
NETBIOS	Network Basic Input Output System
OS/2	Operating System/2
OSI	Open Systems Interconnection
PC	Personal Computer
PC-DOS	Personal Computer Disk Operating System
PDU	Protocol Data Unit
PS/2	Personal System/2
RAM	Random Access Memory
ROM	Read-Only Memory
SAP	Service Access Point
SMB	Server Message Block
SNA	Systems Network Architecture
SSAP	Source Service Access Point
TCP/IP	Transport Control Protocol/Internet Protocols
VLSI	Very Large Scale Integration
XNS	Xerox Network Systems

Appendix B: References

Architecture Technology Corporation publications may be obtained by calling (612) 935-2035.

IBM publications should be ordered through your local representative.

IEEE publications can be ordered from The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017.

1. Architecture Technology Corporation, Token-Ring Technology Report, November 1987.

2. Haugdahl, J. Scott, Analyzing Network Traffic, PC Tech Journal, Volume 5 Number 10, October 1987, pp. 48-62.

3. Haugdahl, J. Scott, Inside the Token-Ring 3rd Edition, Architecture Technology Corporation, May 1990.

4. IBM Corporation, IBM NETBIOS Application Development Guide 68X2270.

5. IBM Corporation, IBM PC Network Technical Reference, 632205.

6. IBM Corporation, IBM PC Network Local Area Network Program, 75 1129.

7. IBM Corporation, IBM Personal Computer Seminar Proceedings: IBM PC Network SMB Protocol.

8. IBM Corporation, Token-Ring Network Architecture Reference, 6165877.

9. IBM Corporation, Token-Ring Network PC Adapter Technical Reference, SC30-3383.

10. IBM Corporation, IBM Token-Ring Network NETBIOS Program: Formats and Protocols.

11. IEEE, An American National Standard--IEEE Standards for Local Area Networks: Token-Ring Access Method and Physical Layer Specifications (ANSI/IEEE STD 802.5-1985 or ISO Draft Proposal 8802/5).

12. IEEE, An American National Standard--IEEE Standards for Local Area Networks: Logical Link Control (ANSI/IEEE Std 802.2-1985, or ISO Draft International Standard (DIS) 8802/2).

13. Microsoft, Inc., Microsoft Operating System/2: LAN Manager API Interface.

14. Novell, Inc., LAN Operating System Report 1986, October 1986.

Appendix C: NETBIOS Vendor/Product Listing

Products listed are for PC-DOS/MS-DOS except where noted.

NETBIOS Workstation Interfaces	Alloy Computer	(386/MultiWare)
	Atlantix	
	Applied Knowledge Group	
	Artisoft	
	AST Research	
	AT&T	
	Banyan	
	CMC	
	Corvus Computer Pathways	
	Corvus	
	Datapoint	
	D-Link	
	DCA	
	DEC	
	DSC	
	Harris	
	Hewlett Packard	
	IBM	
	IBM	(OS/2 EE)
	Madge	
	NCR	
	Novell	(DOS and OS/2)
	Performance Technology	(Unix)
	Racal Interlan	
	Sage	
	Simpact	(MicroVAX Token-Ring)
	Software Link	(PC MOS - DOS compatible)
	3Com	
	TOPS	
	Torus	
	Ungermann-Bass	
	Western Digital	
	Wollongong	

NETBIOS Workstation Interfaces over TCP/IP	CMC	(TCP-IP for DOS)
	Connections Laboratories	(qBNet - NETBIOS over TCP on VMS)
	Network Research Corporation	(Fusion)
	Novell/Excelan	(LAN Workplace for DOS)
	3Com	(PCS-TCP; PCS/1)
	Western Digital	(NETBIOS for TCP/IP)
	Wollongong	(Win/TCP)
NETBIOS Workstation Interfaces over ISO	AT&T	
	3Com	
	Microsoft	
	RCE	
	Ungermann-Bass	
File/Print Servers Supporting NETBIOS	Alloy Computer	(386/MultiWare)
	Applied Knowledge Group	(ZeroNet Plus)
	Artisoft	(LANTASTIC)
	Atlantix	(CocoNet - DOS and Unix)
	Computer Pathways	(Grapevine)
	Datapoint	(dataLAN)
	DCA	(10Net)
	D-Link Systems	(Lansmart/NV)
	DEC	(PCLAN/Server (LAN Manager))
	IBM	(LAN Server - OS/2 and DOS)
	Hewlett Packard	(LAN Manager - OS/2 and DOS)
	Madge	(PC LAN Program Clone)
	Microsoft	(LAN Manager - OS/2 and DOS)
	Racal Interlan	(LAN Manager - OS/2 and DOS)
	Sage	(MainLAN)
	Syntax Systems	(SMB Server - DOS-to-minicomputers)
	TOPS	(TOPS DOS)
	Torus	(Tapestry II)
	Ungermann-Bass	(Net/One LAN Manager - OS/2 and DOS)
	Wollongong	(Pathway - PC-to-VAX server)
CD ROM Servers Supporting NETBIOS	CBIS	(CD Connection)
	Meridian Data	(CD Net)
	Online Products Corporation	(Opti-Net)
Databases Supporting NETBIOS	Anju Technology	(Second Wind for SQL)
	Data Access Corporation	(DataFlex)
	Fox Software	(FoxBASE Plus/LAN)
	Gupta Technologies	(SQLBase)
	Oracle	(Oracle Server)

	Progress Software Corporation	(RDBMS)
	Raima	(Db-Vista III)
	Richmond Systems	(Fault Tolerant Transaction Manager)
	Small Computer Company	(filePro)
	Sybase/Ashton-Tate	(SQL Server)
	XDB Systems	(XDB-Server for SQL)
	Turbo Power Software	(B-Tree Filer)
	Via Information Systems	(DBMS server)
	WordTech	(DBXL/LAN)
NETBIOS-to-NETBIOS Bridges	IBM	(via Token-Ring and/or PC Network)
	IBM	(8209 LAN Bridge for Ethernet/Token-Ring)
	EICON Technology	(via X.25)
RS-232/Modem Servers Supporting NETBIOS	Concept Development	(Sparkle)
	IBM	(Asynchronous Communication Server)
	IBM	(Asynchronous Connection Server)
	Gateway Communications	(G/Async II)
	J&L Information Systems	(NCS)
	Network Products Corporation	(ACS2)
	Omnitel	(Asynchronous Communication Server)
Gateways Supporting NETBIOS	Attachmate	(Extra! - 3174 emulation)
	Datapoint	(PCNS - gateway to RMS)
	DCA	(IRMALAN 802.2 Gateway - 3174 emulation)
	Golden Gate Communications	(V:Access - PC-to-VAX)
	IBM	(PC/3270 Emulation Program Version 3.0)
	ICOT	(NetPath/A SNA 3270 Gateway)
	NSA	(AdaptSNA - 3174)
	Rabbit Software	(RabbitGate - 3174)
	3Com	(MAXESS SNA Gateway)
	Tandem	(MULTILAN Communications Server)
	Techland	(BlueLynx 3270 Gateway)
Terminal Emulation Supporting NETBIOS	Crosstalk Communications	(Crosstalk MK/4 - terminal emulation)
	Crystal Point	(PCTerm - terminal emulation)
	IBM	(PC/3270 Emulation Program Version 3.0)

	ICOT	(NetPath/A 3270 Terminal Emulation)
	SoftKlone	(Mirror III - terminal emulation)
	Techland	(Bluelynx 3270)
	Wall Data	(Rumba - 3270 emulation)
Email Supporting NETBIOS	Conetic Systems	(Higgins)
	Consumers Software	(Network Courier)
	Cross Information Company	(Cross+Point LAN)
	Daystrom Data	(Gallery)
	Data Access Corporation	(OfficeWorks)
	DaVinci Systems	(DaVinci eMail)
	Transcend	(PC COMplete)
Groupware Supporting NETBIOS	DEC	(All-in-1)
	Information Research	(Syzygy - project schedular)
	Informix Software	(SmartWare II)
	Lotus	(Notes)
	Nokia Data	(Alfaskop Workgroup System)
	R+R Associates	(Shoebox 3)
	Siemens AG	(Comfoware - office communications)
	Tandy Computer	(Workgroup)
Station Monitoring Using NETBIOS	Artisoft	(Network Eye)
	MicroNet	(LANshare)
	Norton-Lambert	(Close-Up/LAN)
Miscellaneous Applications with NETBIOS Support	Brightwork	(NetRemote+ - remote NETBIOS access)
	C&C Technology	(Interface for ARCNET adapter for VMS)
	Commtex	(NETBIOS interface for ISDN)
	Exycon, Ltd	(Exycomm/2 or OS/2)
	IBM	(CICS OS/2)
	IBM	(Remote NETBIOS Access) Facility Program
	Indelec	(CIM)
	Lambda Group	(Pipes Platform)
	LAN Systems	(LANSpace - load NETBIOS to extended memory)
	Lattice	(Lattice C Compiler NETBIOS library)
	Netwise	(Remote Procedure Call (RPC) Tool)
	Nynex	(LANPath Fax Server)
	Pacific Bell	(Starlink - private network)
	Pernetix	(DOS-to-UNIX via NETBIOS)

	Softbridge Microsystems	(Bridge/386 - application integrator)
	Sterling Software	(DMS/IB - disk backup)
	Teleclone Datasoft	(Datatalk - remote NETBIOS access)
	Traveling Software	(Viewlink - remote program execution)
	Vadis	(NETBIOS interface for ISDN)
	Yourdon International	(Analyst/Designer - CASE tool)
Protocol Analzyers with NETBIOS Decode	Novell/Excelan	(LANalyzer)
	Network General	(Sniffer)
	Spider Systems	(SpiderAnalyzer)
	Vance System	(ATS 1000)

Appendix D: NETBIOS Vendor/Address Listing

3Com, 3165 Kifer Rd., Santa Clara, CA 95052-8145; (408) 562-6486

Alloy Computer, 165 Forest St, Marlborough, MA 01752; (508) 481-8500

Artisoft, Artisoft Plaza, 575 East River Rd, Tucson, AZ 85704; (602) 293-6363

AST Research, 2121 Alton Ave, Irvine, CA 92714; (714) 756-4984

Atlantix, 5401 NW Broken Sound Boulevard, Suite 100, Boca Raton, FL 33431; (407) 241- 8101

Attachmate, 13231 S.E. 36th St. Bellevue, WA 98006

Banyan, 115 Flanders Rd, Wesboro, MA 01581; (508) 898-1000

Brightwork, 766 Shrewbury Ave., Jerral Center West, Tinton Falls, NJ 07724; (800) 552-9876

C&C Technology, Bldg. 9, Unit 60, 245 W. Roosevelt Rd., W. Chicago, IL 60185; (312) 231-0015

CBIS, 5875 Peachtree Industrial Blvd., Bldg. 100/Ste. 170, Norcross, GA 30092; (404) 446-1332

CMC, 125 Cremona Dr, Santa Barbara, CA 93117; (805) 968-4262

Commtex, 1655 Crofton Blvd., Crofton, MD 21114-1305; (301) 721-3666

Computer Pathways, 19102 N. Creek Parkway, Bothell, WA 98011; (206) 487-1000

Concept Development, 2775 Main St. Suite E, Kennesaw, GA 30144; (404) 424-6240

Conetic Systems, 1470 Doolittle Dr., San Leandro, CA 94577; (415) 430-8875

Connections Laboratories, 1301 Highway 36, Hazlett, NJ; (201) 739-0968

Consumers Software, 73 Water St., 7th Floor, Vancouver, BC V6B 1A1, Canada; (604) 688-4548

Corvus, 160 Great Oaks Blvd., San Jose, CA 95119-4199; (513) 433-2338

Cross Information Company, 1881 9th St., Suite 34, Boulder, CO, 80302; (303) 444-7799

Crystal Point, 12707 120th Ave. NE # 202 Kirkland, WA 98034; (206) 487-3656

Data Access Corporation, 14000 S.W. 119th Ave., Miami, FL 33186; (305) 238-0012

Datapoint, 9725 Datapoint Drive, San Antonio, TX 78229-8500; (512) 699-7900

D-Link Systems, 5 Musick St., Irvine, CA 92718;(714) 455-1688

DaVinci Systems, P.O. Box 17449, Raleigh, NC 27619-7449; (919) 781-5923

Daystrom Data, 15 Sunrise Hill Rd, Fishkill, NY 12524; (914) 896-7378

DCA, 1000 Alderman Drive, Alpharetta, GA 30201-4199; (513) 433-2338

DSC, P.O. Box 110940, Campbell, CA 95011; (408) 441-1300

EICON Technology, 2196 32nd Ave, Montreal, Quebec, Canada H8T 3H7; (514) 631-2592

Exycon, Ltd Luton, Bedfordshire (0582) 402599

Fox Software, Intech House, 34-35 Cam Centre, Wilbury Way, Hitchin, Herts SG4 OAP (0469)

Gateway Communications, 2941 Alton, Irvine, CA 92714; (800) 367-6555

Golden Gate Communications, 2140 Shattuck Ave, Suite 2290, Berkely, CA 94704; (415) 524-6166

Gupta Technologies,1040 Marsh Rd., Menlo Park, CA 94024; (415) 321-9500

Harris, 2101 W. Cypress Crk Rd., Ft. Lauderdale, FL 33309-1892; (305) 973-5125

Hewlett-Packard, 3000 Hanover St., Palo Alto, CA 94304; (415) 857-1501

ICOT, 3801 Zanker Rd., P.O. Box 5143, San Jose, CA 95150-5143 (408) 433-3300

Information Research, 2421 Ivy Rd., Charlottesville, VA 22901; (800) 368-3542

Informix Software, 4100 Bohannon Dr., Menlo Park, CA 94025; (415) 926-6300

J&L Information Systems, 9238 Deering Ave., Chatsworth, CA 91311 (818) 709-1778

Lambda Group, 555 DeHaro St. San Francisco, CA (415) 626-4545

LAN Systems, 300 Park Ave. So., New York City, NY 10010; (212) 473-6800

Lattice, 2500 S. Highland Ave., Lombard, IL 60148; (312) 916-1600

Lotus, 55 Cambridge Rky., Cambridge, MA 02142; (617) 577-8500

Madge, 1508 Oakland Rd., C-206, San Jose, CA 95131; (408) 441-1300, or, 100 Lodge Lane, Chalfont St.,Giles, Bucks HP8 4AH, England; telephone, 011-44-2404-5651

Meridian Data, 5615 Scotts Valley Dr., Scotts Valley, CA 95066; (408) 438-3100

MicroNet, 20 Mason, Irvine, CA 92718; (714) 837-6033

Microsoft, 16011 NE 36th Way, Box 97017, Redmond, WA 98073-9717; (206) 822-8080

NCR, 1635 Aeroplaza Dr. Colorado Springs, CO 80916; (800) 525-2252

Netwise, 2477 55th St., Boulder, CO 80301; (303) 442-8280

Network General, 1945 A Charleston Rd., Mountain View, CA 94043; (415) 965-1800

Network Research Corporation, 2380 N. Rose Ave, Oxnard, CA 93030; (805) 485-2700

Nokia Data, 710 Lakeway, Suite 290, Sunnyvale, CA 94086; (408) 720-0895

Norton-Lambert, P.O. Box 4085, Santa Barbara, CA 93140; (805) 964-6767

Novell, 122 East 1700 South, Provo, UT 84606; (801) 379-5900

NSA, 39 Argonaut, Laguna Hills, CA; (714) 768-4013

Nynex, 4 West Red Oak Ln., White Plains, NY 10604; (914) 644-7844

Omnitel, 3500 W. Warren Ave., Fremont, CA 94538; (415) 490-2202

Oracle, 20 Davis Dr., Belmont, CA 94002; (415) 598-8000

Performance Technology, 800 Lincoln Center, San Antonio, TX 78230

Pernetix, 13633 Gamma Rd., Dallas TX 75244: (214) 385-2376

Progress Software Corporation, 5 Oak Park, Bedford, MA 01730 (617) 275-4500

R+R Associates,39 Carwell Ave., Mt. Vernon, NY; (914) 668-4057

Rabbit Software, Great Valley Corp. Center, Seven Great Valley Pky. E., Malvern, PA 19355;

(215) 647-0440

Racal Interlan, 155 Swanson Road, Boxborough, MA 01719; (508) 263-9929

Raima, 3245 146th Place, S.E., Ste. 230, Bellevue, WA 98007; (206) 747-5570

Richmond Systems, 3 Church Terrace, Richmond, Surrey TW10 6SE, England

Siemens AG, 2191 Laurelwood Rd., Santa Clara, CA 95054; (408) 980-4500

Simpact, 9210 Sky Park Court, San Diego, CA 92123; (619) 565-1865

Softbridge Group, 123 Cambridge Park Dr., Cambridge, MA 02140; (617) 926-5030

SoftKlone, 327 Office Plaza Dr., Suite 100, Tallahassee, FL 32301; (904) 878-8564

Software Link, 3577 Parkway Lane, Norcross, GA 30092; (404) 448-5465

Spider Systems, 12 New England Executive Park, Burlington, MA 01803; (617) 270-3510

Sterling Software, 21050 Vanowen St., P.O. Box 9152, Canoga Pk., CA 91304; (818) 716-1616

Sybase/Ashton-Tate, 6475 Christie Ave., Emeryville, CA 94608; (415) 596-3500

Syntax Systems, 1501 W. Valley Hwy, N./Suite 104, Auburn, WA 98001; (206) 833-2525

Techland, 401 Winston St., Cumberland, MD 21502; (800) 237-5888

TOPS, 950 Marina Village Pky., Alamada, CA 94501; (415) 769-9669

Torus, Beauford Court, Waterside, Marsh Wall, London E149XL, England

Transcend, 884 Portola Rd., Portola valley, CA 94028; (415) 851-3402

Traveling Software, 18702 North Creek Parkway, Bothell, WA 98011; (206) 483-8088

Turbo Power Software, Scotts Valley, CA (408) 438-8608

Ungermann-Bass, 3900 Freedom Circle, Santa Clara, CA 95054; (408) 562-7958

US Sage, 2005 Tree Fork Ln., Suite 125, Longwood, FL 32750; (407) 331-4400

Vadis, 1201 Richardson Dr., Suite 200, Richardson, TX 75080; (214) 996-0100

Vance System, 3901-V Centerview Dr., Chantilly, VA 22021-3200; (703) 471-9402

Wall Data, 17769 N.E. 78th Place, Redmond, WA 98052-4992; (206) 883-4777

Western Digital, 2445 McCabe Way, Irvine, CA 92714; (714) 863-0102

Wollongong, 1129 San Antonio Rd., Palo Alto, CA 94303; (415) 962-7100

WordTech, 21 Altwiati Rd., Orinda, CA 94563; (415) 254-0900

XDB Systems, 7309 Baltimore Ave., Suite 220, College Park, MD; (301) 779-5486

Yourdon International, 1501 Broadway, New York City, NY 10036; (212) 391-2828

Index

V

X

About the Author...

J. Scott Haugdahl is a Senior Technical Consultant at Architecture Technology Corporation. His work has included simulation and performance analysis of multiprocessor computer systems, design and implementation of specialized server software and protocols for personal computer local networks, design and testing of advanced hardware for servers of PC local networks, and analysis of the latest developments and product offerings in the local-area network marketplace. He has researched, written, and presented numerous seminars on LANs in both the U.S. and Europe. He has also presented papers for IEEE and CW Communications conferences and is the author of the books *Inside the Token-Ring, Inside NETBIOS,* and *Inside SAA*. Mr. Haugdahl received his B.S. in Computer Science from the University of Minnesota, Institute of Technology.

Other Books of Interest from
Architecture Technology Corporation

Inside APPC
APPC (Advanced Program-to-Program Communications) is an architecture destined for implementation on an extremely wide range of IBM and non-IBM datacom products. APPC is the architecture most central to IBM's distributed processing strategy. Written by Michael Hurwicz, this book begins with an conceptual and historical overview of APPC, continues with coverage on: LU6.2 basics --states, functions, services, and relationship to PU2.1; the LU6.2 architecture and formats; SNA Service Transaction Programs (TPs) -- SNADS, DIA, and DDM; and concludes with detailed examples of how APPC verbs are used.

Inside SAA
SAA (Systems Application Architecture), is IBM's set of architectures for implementing distributed computing and data systems across its strategic family of computers; the Personal System/2, Systems 3X, and the System/370. Written by J. Scott Haugdahl, this book begins with a historical and conceptual overview of SAA, including its relationship to IBM's SNA (Systems Network Architecture) and the benefits it offers to vendors and users. Subsequent chapters discuss the key elements of SAA in more detail -- including Low Entry Networking (LEN), Advanced Program-to-Program Communication (APPC), Distributed Data Management, Enhanced Connectivity Facilities (ECF), Integrated Office Systems, Network Management with discussion of NetView, and the SAA elements in Operating System/2 Extended edition.

Inside The Token-Ring
The Token-Ring has achieved status in the local area network marketplace that took Ethernet over ten years to achieve (Ethernet is now in its sixteenth year). Needless to say, the critics are somewhat embarrassed by the success of the IBM Token-Ring -- while Ethernet continues to currently dominate, we can not ignore this "new" technology. Written by J. Scott Haugdahl, Inside the Token-Ring covers all aspects of IBM's Token-Ring local area network, from history to IEEE Standards to interfaces and products. The reader of this book will gain a solid understanding of the IBM cabling system, token ring operation, and management features unique to the token ring.

ISBN 0-939405-01-6