Understanding Performance Tuning Theory for IBM OS/2 LAN Server

Document Number GG24-4430-00

November 1994

International Technical Support Organization
Austin Center

Take Note!

Before using this information and the product it supports, be sure to read the general information under "Special Notices" on page xiii.

First Edition (November 1994)

This edition applies to both IBM OS/2 LAN Server Version 3.0 and Version 4.0.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

An ITSO Technical Bulletin Evaluation Form for reader′s feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. 948S, Building 821 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

Abstract

This document provides a comprehensive overview of tuning LAN Server products (LAN Server Version 3.0 / 4.0) to obtain optimum performance in various system environments. It details how to set up a system to provide a good balance between hardware and software for the most common file server applications. It also describes, in detail, the performance issues related to each of the server subsystems and the characteristics concerning different server environments.

This document is intended for IBM customers, dealers, system engineers, and consultants who need to understand the issues related to LAN Server performance tuning. Basic knowledge of LAN Server and LAN concepts is assumed.

(221 pages)

Contents

Figures

Tables

Special Notices

This document is intended to help readers understand the performance issues related to the IBM OS/2 LAN Server Version 3.0 / 4.0 product. It contains information discussing the approach to identify and solve any performance problems within LAN Server. The information in this document is not intended as the specification of any programming interfaces that are provided by OS/2 LAN Server. See the PUBLICATIONS section of the IBM Programming Announcement for OS/2 LAN Server Version 3.0/4.0 for more information about which publications are considered to be product documentation.

References in this document to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (VENDOR) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms, which are denoted by an asterisk (*) in this publication, are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AT	IBM
DatagLANce	DB2/2
EtherStreamer	LAN Distance
LANStreamer	Micro Channel
NetDoor	OS/2
Personal System/2	Presentation Manager
PS/2	Ultimedia

The following terms, which are denoted by a double asterisk (**) in this publication, are trademarks of other companies:

80386SX, i386, i486, Pentium, Intel	Intel Corporation
AppleTalk	Apple Computer, Inc.
Banyan, VINES	Banyan Systems, Inc.
DEC, DECnet	Digital Equipment Corporation
Freelance, Lotus Notes, Lotus 1-2-3, Lotus Freelance Graphics	Lotus Development Inc.
Microsoft, Microsoft Windows, Excel, Microsoft Word	Microsoft Corporation
Novell, NetWare	Novell, Inc.
NFS, Network File System	Sun Microsystems, Inc.

Preface

This document is intended to give insight into issues related to IBM OS/2 LAN Server Version 3.0 / 4.0 performance and provide the necessary guidance to achieve optimum server performance.

It is intended to assist IBM customers, dealers, system engineers and consultants who need to know more about LAN Server in order to get the best performance. Throughout this document we assume the reader has access to the reference books *IBM OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning* for Version 3.0 and Version 4.0.

How This Document is Organized

This document is organized as follows:

- Chapter 1, "Introduction" gives you a brief overview of the methodology behind general server tuning and provides a basic understanding of server performance.
- Chapter 2, "Server Hardware Review" describes the hardware components in relation to the server performance. LAN Server performance is limited by the hardware configuration.
- Chapter 3, "LAN Server Architecture and Tuning" is one of the main chapters which describes the LAN Server architecture and its performance considerations.
- Chapter 4, "LAN Server Network Tuning" discusses the network interface components such as NetBEUI, TCPBEUI and adapter interface.
- Chapter 5, "LAN Requester Performance Tuning" explains the performance tuning for the requesters such as DOS LAN Requester, DOS LAN Services and OS/2 LAN Requester.
- Chapter 6, "Further Analysis of Tuning" expands the theory to the analysis of LAN trace data in order to understand the details of SMB and NetBEUI protocol exchanges.

- Appendix A, “Net Errors” provides several error messages related to performance or capacity tuning. These are quick tips for your performance problems.

The following two diagrams guide you through the topics in this document and help to identify the sections which might be of most interest to the readers.

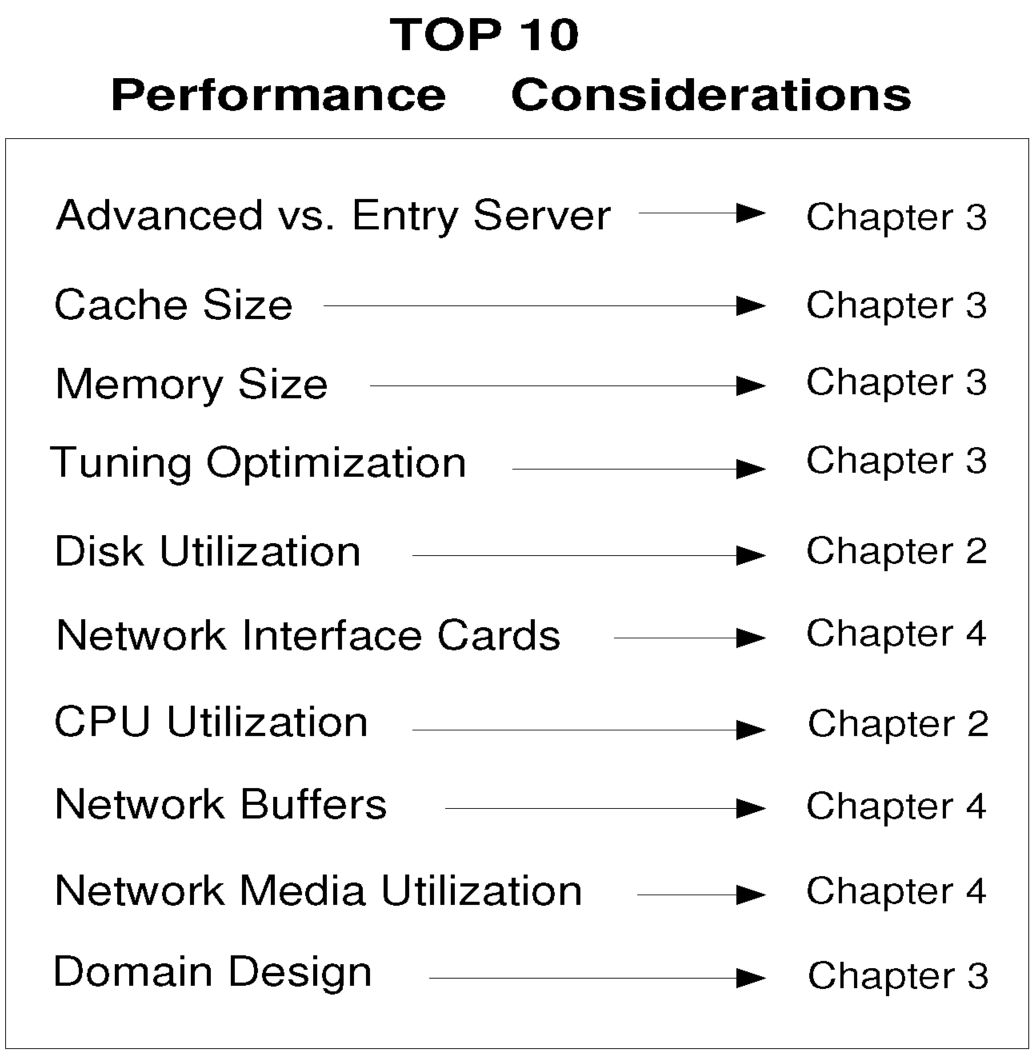

Figure 1. Top 10 Performance Considerations

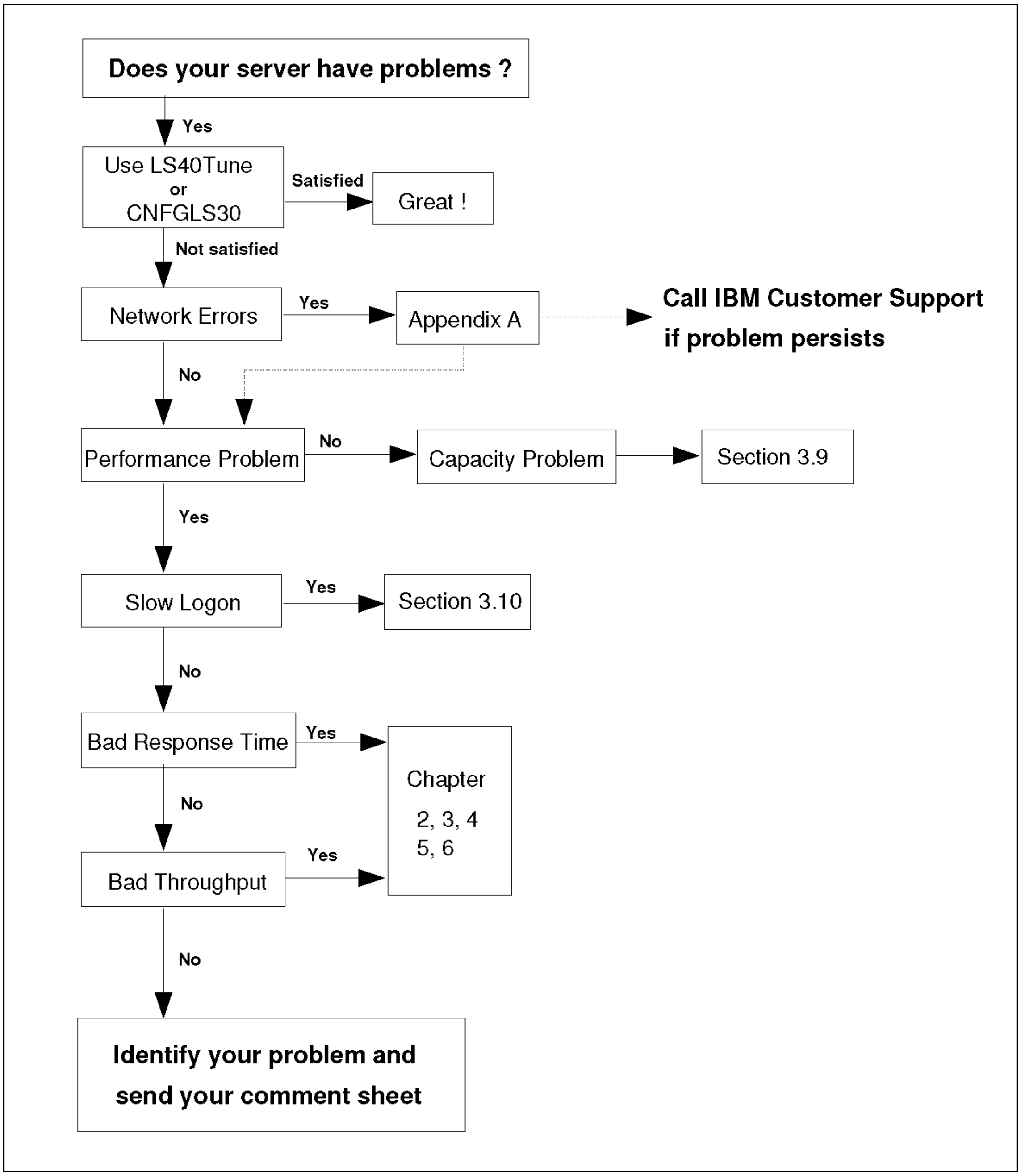

Figure 2. Problem Solving Flow

Related Publications

The publications listed in this section are considered particularly suitable for further readings:

- *IBM OS/2 LAN Server Version 4.0 Network Administrator Reference Volume 1: Planning and Installation*, S10H-9680
- *IBM OS/2 LAN Server Version 4.0 Network Administrator Reference Volume 2: Performance Tuning*, S10H-9681
- *IBM OS/2 LAN Server Version 4.0 Network Administrator Reference Volume 2: Administrator Tasks*, S10H-9682
- *Multi-Protocol Transport Services - AnyNet for OS/2: Configuration Guide*, S10H-9693
- *IBM Network Transport Services/2 LAN Adapter and Protocol Support Configuration Guide*, S96F-8489
- *LAN Technical Reference*, SC30-3587
- *Protocols for X/OPEN PC Interworking: SMB Version 2*

International Technical Support Organization Publications

- *Experiences with the IBM OS/2 LAN Server Version 3.0 New Functions*, GG24-3959-01
- *Understanding IBM OS/2 LAN Server Ultimedia Version 1.0*, GG24-4224-00
- *Advanced PS/2 Servers Planning and Selection Guide*, GG24-3927-01

A complete list of International Technical Support Organization publications, with a brief description of each, may be found in:

> *International Technical Support Organization Bibliography of Redbooks,* GG24-3070.

Acknowledgments

This publication is the result of the project conducted at the International Technical Support Organization, Austin Center.

The advisor for this project is:

Toshi Shimizu ITSO Austin Center

Authors of this document are:

Peter Gehrig IBM Switzerland
Emi Takamura IBM Japan
Damian Tambirasa IBM United Kingdom

Thanks to the following people in IBM Austin for the invaluable advice and guidance provided in the production of this document:

Brice Bartek
Len Brissette
Steve Dobbelstein
Daw Feng
Steven French
Bob Hanner
Tom Herrick
Carol McKinney
Ken Whitfield
Russ Young

Chapter 1. Introduction

This chapter introduces some of the basic concepts of performance tuning and provides a brief overview of the principle factors that affect server performance. You will see where the LAN Server performance tuning is positioned in the whole server performance task.

1.1 Overview of PC Server Performance Tuning

Performance tuning is not straight-forward and should be looked upon as a complex task. It requires a great deal of discipline and exactness. Unless appropriate tests are used to identify specific bottlenecks within the system it is difficult to interpret any results. Occasionally performance tuning can be very frustrating and tedious at times especially when after a great deal of analysis the results are still inconclusive. Nevertheless, performance tuning can be very rewarding and provide long term benefits.

A PC server is subjected to various LAN loads from the workstations on the LAN. The load can vary widely depending on the number of workstations used and the type of applications being run. For example, 50 LAN users running an e-mail application will generate a different load to the server than 50 users running an SQL database application. Obviously the number of workstations and the type of applications being run will vary widely over the period of the server′s working life. Consequently, changes have to be made to the server′s hardware and software setup to accommodate these changing conditions.

Engineers often refer to any degradation of service as a *bottleneck* in the server system, but users who are less aware might simply consider the LAN to be *running slow* or that something is wrong. Bottlenecks need to be understood and compensated for, if the network engineers are to keep the users satisfied with LAN performance.

There are a range of parameters that can be changed to affect server performance. However, before making any changes, the current server configuration should be carefully assessed, and any current or potential bottlenecks must be identified. Once bottlenecks are identified, both hardware and software tuning can be applied using this document as a guide to the most appropriate actions.

In this document we describe some basics that will help improve server and requester performance in general. In fact the required actions may be hardware related, software tuning, or a combination of both. The purpose of this document is to provide LAN Server tuning but related PC hardware basics and tuning tasks are also described, because software tuning cannot improve the system performance more than the hardware capabilities. This means that you cannot get the best performance out of the server by tuning only the software. Tuning will help you get the best out of your existing server hardware, but to get the best total system performance it is essential that any hardware bottlenecks are eliminated before you perform your final software tuning.

When we talk about LAN Server performance, it is necessary to understand the statistics and the utilizations of CPU, disk I/O, LAN I/O and the memory component in the server machine. Without knowing these resources utilization, simple changes of LAN Server/Requester parameters will not solve your problem.

To assist with analyzing system activities on your server, IBM System Performance Monitor/2 is recommended.

Note

Use one of the performance tuning tools such as LAN Server 4.0 Tuning Assistant or CNFGLS30 tool for LAN Server 3.0. LAN Server Tuning Assistant is a part of the LAN Server 4.0 package. They can both help you attain the appropriate parameters for CONFIG.SYS, IBMLAN.INI and PROTOCOL.INI.

Under normal circumstances these tools will be adequate to provide good server performance. However, you will not learn the internal logic of the LAN Server from these tools. When you want to know the logistics, these tools are not the answer. If you are still not satisfied with the performance of your server, or you need more in-depth information in understanding LAN Server performance tuning, then this document should provide an assistance.

1.2 Understanding Server Performance in General

Within a server there are limited resources that can affect the performance of a given system. Each of these resources work together hand-in-hand and they each are capable of influencing the behavior of one another. If performance modifications are not carefully administered then the overall effect could be a deterioration of server performance.

These resources comprise the fundamental subsystems found within a server:

- Central Processing Unit (CPU) and memory
- Disk I/O
- LAN I/O
- Controlling software

Obviously the controlling software in this case is IBM* OS/2* Version 2.X and LAN Server Version 3.0 and/or LAN Server Version 4.0, including transport layer software. Efficiency of the controlling software will maximize the hardware performance.

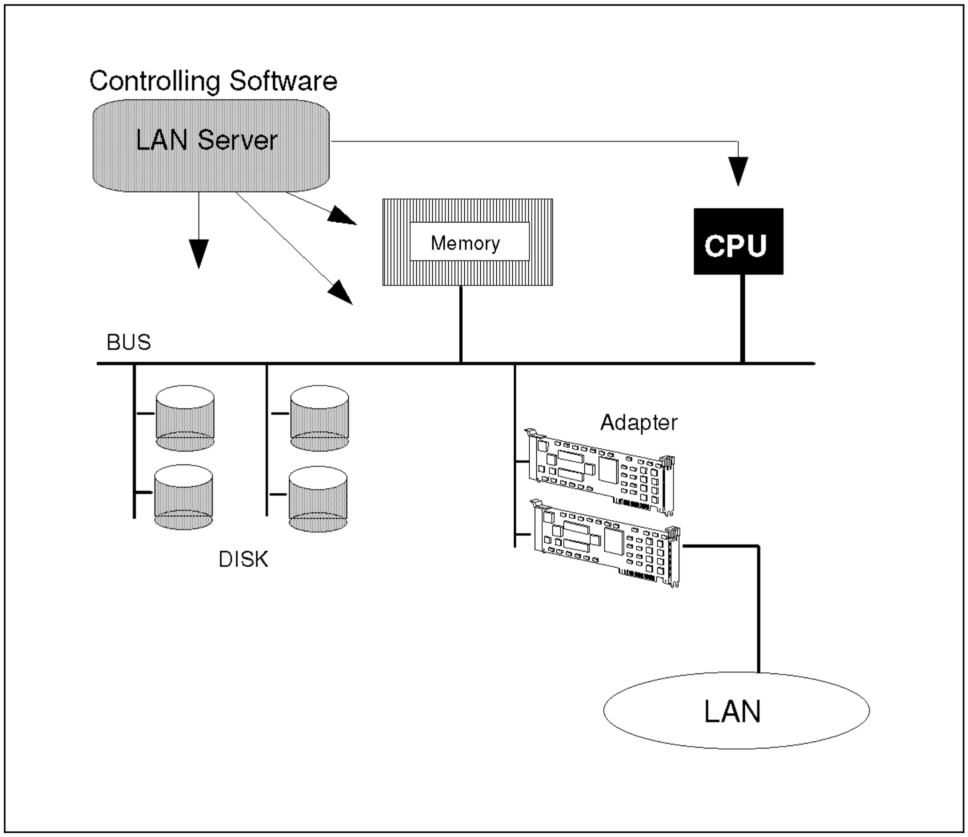

Figure 3. LAN Server Controlled Subsystems

As server performance is distributed throughout each server component it is essential to identify the most important factor(s), for example bottlenecks, that will affect the performance for a particular activity.

Detecting the bottleneck within a server system depends on a range of factors such as:

1. Server hardware configuration
2. Application software workload
3. Operating system configuration parameters
4. LAN Server software parameters

File servers need fast LAN adapters and fast disk subsystems. In contrast, database server environments typically produce high CPU and disk utilization requiring fast processors or multiple processors and fast disk subsystems. Both file and database servers require large amounts of memory for operating system caching.

If the bottleneck is the processor then a faster processor could be installed. An alternative to processor upgrade is to off-load processing requirements by using busmaster LAN and disk adapters. Busmaster adapters have the capability of off-loading the server processor by carrying out unassisted data transfers. In contrast, non-busmaster adapters must use the server processor for all data transfer operations.

If the bottleneck is memory then additional memory could be installed. Memory bottlenecks often cause excessive disk I/O by the operating system. Memory bottlenecks can also cause high disk utilization, if there is insufficient memory available or disk caching algorithms are ineffective for a particular application workload.

If the bottleneck is the disk subsystem then either additional disks and/or disk adapters can be installed, or a specialized high performance disk subsystem such as the IBM Array Controller with IBM Fast SCSI-2 drives could be used. This allows an overlap of disk I/O requests.

If the bottleneck is the LAN adapter then a faster LAN interface such as the IBM LANStreamer* MC 32 adapter with 40MB data streaming support could be installed. Another optimization technique that can be employed is to utilize multiple LAN adapters in the server increasing throughput onto one or multiple segments.

Finally LAN Server software could be a bottleneck in controlling or using the system resources. For example, inefficient use of buffers will waste the memory resources, so every buffer must be allocated and used properly.

General Performance Characteristics

The disk subsystem is responsible for filling or emptying the contents of operating system disk cache memory in order to satisfy network requests for data retrieval or storage. Performance efficiency depends on whether the operating system disk cache is sufficiently large and intelligent to contain the correct data required to satisfy most network request from cache memory. In addition, a fast disk subsystem must provide read data for the cache and empty the cache when data must be written to disk. Thus, the disk subsystem must offer performance that can keep the operating system cache filled with useful data. Failure to do so results in a lower disk cache hit rate, slower user access times and decreased performance.

The LAN adapter must move data between the network and system memory as fast as possible. For applications that issue large read and write requests, the data width of the adapter bus interface is an important factor. The larger the amount of data transferred at one time by an application, the greater the requirement for a wide (32-bit) data path and 40MB/second data streaming technology.

When, on the other hand, an application transfers small data frames, *latency* becomes a critical factor. That is, when a server must respond to application requests for small data transfers, performance can be greatly affected by the network adapter′s ability to send or receive each small frame, its latency. Since small frames contain little data, the width of the data interface of the adapter has very little influence on performance, since data transfers are short.

Systematic measurements of server performance employing adequate resolution typically produce a graph shaped like the one shown in Figure 4. The characteristics that shape this graph are important for understanding potential bottlenecks in a file server.

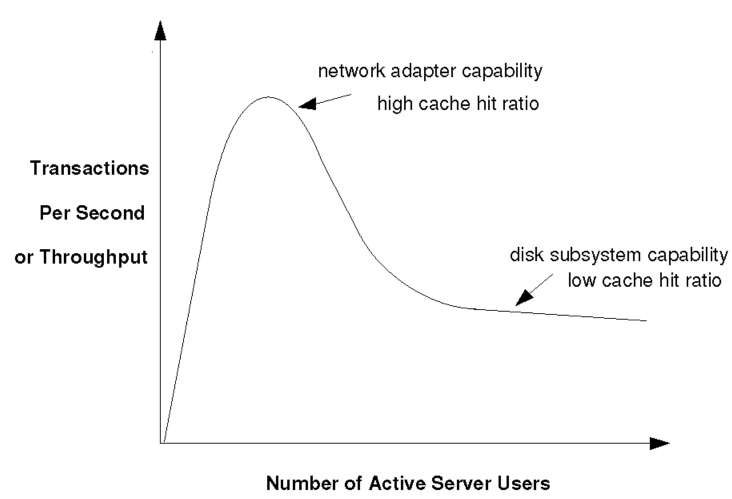

Figure 4. File Server Performance - General Characteristics

The horizontal axis shows the number of simultaneous active users. The vertical axis indicates server throughput, or number of transactions per second. This is the total number of transactions per second at the server, rather than the individual transaction rate for each workstation.

Throughput indicates how many units of work or transactions can be accomplished per second. A transaction can be an arbitrary unit of work. However, it is more meaningful to consider a transaction to be a typical user task, such as loading and saving of a spreadsheet or word-processing document.

A transaction can be a networked client loading a spreadsheet, processing it at the workstation and then saving it back to the server. This process can be executed by a large number of workstations attached to a single file server to determine the maximum number of load-and-save transactions that can be accomplished per second. It is important to note that the number of transactions that can be accomplished per second is greatly dependent upon the transaction definition. Obviously, a one page document load-and-save transaction can complete much sooner than a 100-page document load-and-save transaction. Thus, arbitrary file server transaction rates are meaningless unless the specific type of transaction is clearly defined.

Figure 4 on page 6 shows a graph of file server performance where throughput initially increases at a constant rate as users are added to the server. As the total number of users is increased, the server operating system is able to maintain a sufficiently high disk cache hit rate. Most user I/O requests are being serviced directly from the simultaneous requests from multiple users. The curve continues to climb until it reaches a peak, which represents the maximum server transactions per second or throughput rate.

Following the peak throughput obtained by the server, the curve begins to slope downward. As the number of users is increased, the *caching engine* of the operating system begins to break down. The reduction in the disk cache hit rate is caused by the increasing amount of data that each additional user requests the server to access. The ratio of requested data to the size of the network operating system disk cache increases, to a point where the network operating system disk caching is no longer effective. Furthermore, the disk cache is often storing data that was requested to be written to disk by application users. This write caching, when employed, causes write-data to occupy disk cache memory space until it is finally written to the disk surface. Thus, slow disk write operations can compound the performance degradation of servers that support write caching because the amount of useable disk cache memory is reduced by write-data waiting for service by the slower disk system.

The curve begins to break down until it reaches the transaction rate sustainable by the disk subsystem. In this state, throughput of the server has degraded because fewer I/O requests can be serviced directly from disk cache. When the addition of users generates an increase in requests for data, I/O operations are delayed even further, since each I/O request then requires a direct disk access. The server then must service almost every I/O request directly from disk and reaches saturation.

The initial slope of the curve is dependent on how quickly transactions can be processed by the server, which, in turn, depends primarily on how quickly the LAN adapter is able to transfer data.

The peak of the curve is the maximum server transaction rate that the particular server configuration is capable of sustaining for a specific transaction type. The maximum transaction rate is primarily dependent upon performance of the network adapter and disk subsystem combination. When the graph flattens out, performance of the disk subsystem has the greatest influence on overall server performance.

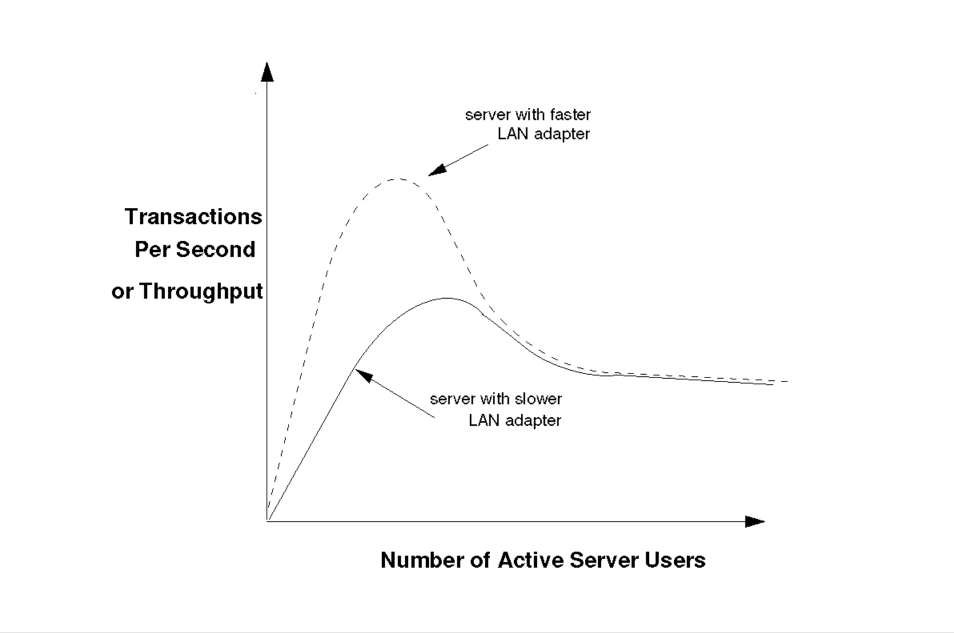

Figure 5. Differences in LAN Adapters

Changing the LAN adapter or the disk subsystem can alter the height or width of the graph. In most cases the disk subsystem becomes the bottleneck when a large number of users becomes active. Since most disk subsystems are significantly slower than a cache-hit operation, the throughput curve begins to decline. We can begin to obtain a better understanding of how to tune server hardware by studying the curve dynamics as we modify various hardware components.

An exception to this is for high performance disk subsystems. One such subsystem is the IBM Array Controller with Fast SCSI-2 drives. This combination offers such a high level of performance that for many applications it allows the peak transaction rate to be sustained indefinitely. The general shape of the curve will be different than the one shown in Figure 6. Where the peak performance in transactions per second occurs, the line will continue horizontally rather than dipping as it would when the bottleneck is the disk subsystem. That is, for this workload the disk subsystem is no longer the system bottleneck and the peak transaction rate is sustained.

For example, in Figure 5 on page 8, adding a faster network adapter will increase the initial slope of the graph and also provide additional throughput. Therefore, the total number of transactions that can be processed per second will be improved. Furthermore, response time seen by any individual application user will be reduced. However, the reduction in response time is a limited benefit. As more users are added, the effects of the improved network adapter will be offset by increased disk I/O and increased write-data (which reduces the disk cache hit rate and overall server performance).

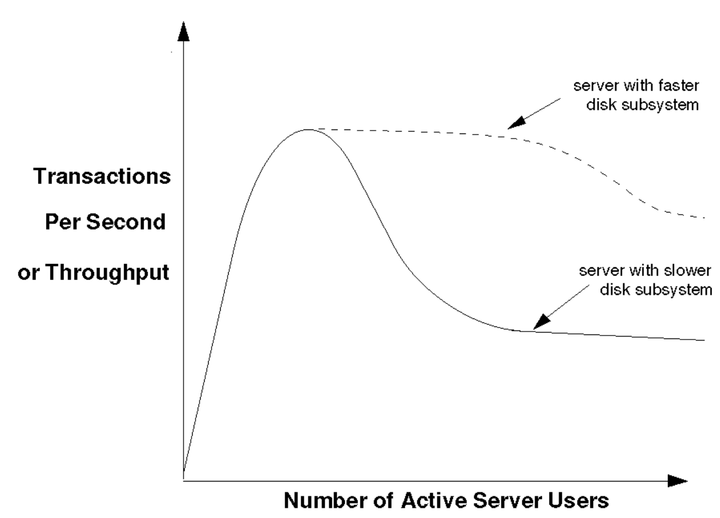

Figure 6. Differences in Disk Subsystems

Improving performance of the disk subsystem will usually prolong the maximum transactions per second rate shown in Figure 6. The effect of a faster disk can be to improve the disk cache hit rate. This is because write-data can be flushed from the cache quicker, so that the amount of memory available for caching of new data is increased.

1.3 General Performance Tuning Approach

Before any tuning is actually performed it is worth understanding the framework within which performance testing is done. A set of simple guidelines need only be followed to assist in any type of performance analysis.

There are many trade-offs related to performance tuning that have to be considered. In order to chose the best set of options it is vital to ensure that there is a balance between them. The trade-offs are:

Cost vs Performance

There are situations where the only way performance can be improved is by using more or faster hardware.

Conflicting Performance Requirements

When more than one OS/2 application is used simultaneously, there may be conflicting performance requirements. For example, DB2/2* needs a large memory block for database caching and the HPFS386 component of LAN Server needs a large cache memory to serve non-database I/Os. They cannot share the same cache area. In such a case, it will be necessary to review and decide on a compromise between these two requirements until an acceptable level of performance is achieved for all affected functions.

Speed vs Functionality

Here, for example, resources may be increased to improve a particular section, but serve as an overall detriment to the system. For example in DOS LAN Requester, increasing the number of transmit and receive buffers may help response time, but may also take RAM away so that certain DOS applications no longer can run.

Using a methodical approach you can obtain improved server performance, such as by:

- Understanding the factors which can affect server performance, for the specific server functional requirements and for the characteristics of the particular system
- Measuring the current performance of the server
- Identifying a performance bottleneck
- Upgrading the component which is causing the bottleneck
- Measuring the new performance of the server to check for improvement

An example of a methodical approach for performance tuning can be illustrated by the following flowchart:

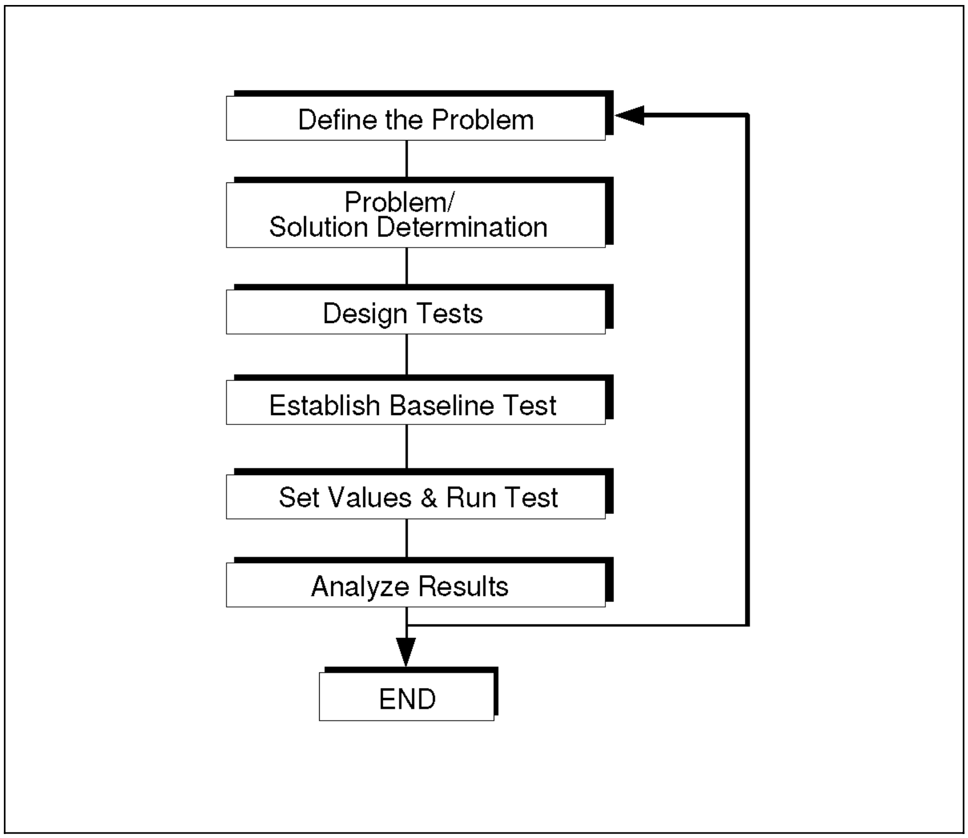

Figure 7. Problem Determination Flow Chart

1. **Defining the Problem**

   This can be achieved by understanding that there is a problem and knowing what factors that would affect performance. This is normally done by using some kind of performance tool. Once the problem has been identified it is worth defining a realistic goal that can be achieved if the necessary modifications are done.

2. **Problem/Solution Determination**

   Once the performance problem has been experienced it is necessary to identify the possible bottlenecks within the server system that could be the source of the problem. Bottlenecks usually occur at points in the system where requests arrive faster than they can be handled. An example would be a server supporting too many requests with insufficient buffering to hold the data. Once a bottleneck is identified and resources have been applied to relieve it, another bottleneck may be generated. At this point it would be possible to brainstorm some possible solutions that may eliminate the bottleneck by, for example, changing the LAN topology or redistributing the RAM.

3. **Design Tests**

   Now as the problem is identified it is possible to design appropriate tests that can be used for understanding the problem and providing a solution. These tests should be prioritized so that the ideas that are most likely to have an impact on performance with minimal effort are executed first. However, it is vital that the tests are repeatable to ensure that multiple tests can be run to determine different trends and behaviors.

4. **Establish Baseline Test**

   Before any testing can be done a baseline test should be established and adapted to the particular environment. It should be the starting point of the *benchmarking* process.

   Benchmarking essentially consists of taking performance readings, making appropriate changes and measuring the resulting performance again. It is a continuous process and gives an insight into the behavior of the system once modifications have been done. To choose an appropriate benchmark, it is advisable to consider the many factors that will influence the establishment of the baseline test, such as:

   - What is your scenario?
   - What tool or tools are required to measure performance of your particular environment?
   - Always start at the same state. Make sure that you are only measuring the relevant activities and avoid interference from outside.
   - Document everything including server configuration, PROTOCOL.INI, CONFIG.SYS and steps followed during test.

5. **Set Values & Run Test**

This involves running the tests that have been designed earlier to measure the progress of system. In order to achieve this a representative set of programs and data are required to evaluate the performance of computer hardware and software in a given configuration. It is strongly advisable that you only work on one problem at a time and only change one variable at a time so that an absolute result can be ascertained from the test.

6. **Analyze Results**

   This involves analyzing the results of the performance tests and being able to interpret and explain the resulting data. This normally refers to observing the response time and throughput patterns for the particular scenario.

Following through this methodology once is not sufficient to obtain any definite results. This process has to be continually performed so that conclusive results can be obtained from any trends that are experienced.

It is worth mentioning that accurate management of network user activity and application usage is an important part of tracking the changes in LAN load put on the server. Adequate records must be kept to note the changes made to the LAN, so as to relate the changes to any degradation in file server performance. This is an important factor in network management.

1.4 Server Functional Requirements

LAN Server has the capabilities to provide users with many unique functional services depending on the applications that reside on it. This allows a LAN Server to provide either a dedicated service or incorporate a number of server functions. For example, it can provide a range of file and print serving functions within one system, or be used in non-dedicated mode to provide application or communication serving capabilities (for example, LAN Distance* Connection Server, 3270 Communications gateway, Lotus Notes** server, and others).

The actual allocation of server function to a LAN Server will depend on the number of users, the size of the workload, the capabilities of the server itself and the design criteria established for the LAN server environment. However, care should be taken when mixing the primary role of a mission critical or heavily used server system or performance may degrade considerably.

The following table gives a brief overview into the various server functions available to LAN Server. They are divided into three categories:

1. File Servers
2. Client-Servers
3. Communications Servers

Table 1. Server Functional Requirements - File Server Category

File Server Category	**Description**
File Server	Provides access to files that may be used by individual users or shared by several users. Shares fixed disk or equivalent storage media. Facilities to control multiple access must be included.
Package Server	Where specific application programs are required by a number of users, especially when the computing resources needed are significant, the application may be provided by a server, and accessed through the LAN by individual user workstations.
Multimedia Server	An example of specialized device support, which would be used to provide access to devices not normally supported by the workstation itself. The image conversion or other processing is carried out on the server, and the results transmitted across the LAN to the workstation. Other specialized device servers may be similarly defined, depending on the system requirements.
Remote IPL Server	Supports medialess workstations by providing initial program load images for initializing the workstations on the LAN. This is a specialized form of file servers.

Table 2 (Page 1 of 2). Server Functional Requirements -Client-Server Category

Client-Server Category	**Description**
Application Server	May be used to provide processing capability purely to relieve the processor in the workstation, using the remote execution capability of the LAN.

Table 2 (Page 2 of 2). Server Functional Requirements -Client-Server Category	
Client-Server Category	**Description**
Database Server	Like the file server, this provides access to data but it additionally incorporates database management system components. The database may reside on the server, or there may be links to databases on other LANs or on central systems. These links are managed independently of the user, who need not be aware of the specific location or means of access used.
Mail Server	Manages electronic mail communications, provides local mail box facilities, and manages mail user definitions and access, or acts as an intermediary to a centrally provided electronic mail service.
Security Server	A security system may be provided by a server that could hold access control databases, specialized encryption adapters or tables for validity checking. This could be used for example to keep up-to-date lists of stolen credit cards, or for the management of centrally distributed passwords and access codes.

Table 3 (Page 1 of 2). Server Functional Requirements -Communications Server Category	
Communications Server Category	**Description**
Communications Server	Handles the communications devices, either directly through communications adapters and modems, or indirectly by routing communications requests to a LAN-attached communications controller. The server in the latter case is often referred to as a gateway.
Transaction Server	An alternative means of distributing the processing requirement across the LAN is to design the application in the form of transactions. This is similar to the transaction processing traditionally provided by centralized systems, but the server either carries out the processing locally, or acts as an intermediary to check and forward transactions. This could include a mechanism for reducing the perceived number of sessions that a central system may need to handle.

Table 3 (Page 2 of 2). Server Functional Requirements -Communications Server Category	
Communications Server Category	**Description**
Print Server	Allows access to shared printer facilities, and may include provision of spool space for interim storage of print tasks. Print job management facilities will be needed, and appropriate capabilities for form type, print orientation, font facilities and so on. The printer itself may be directly attached to the LAN, using a hardware LAN attachment device, in which case the functions of the server are related to the routing of data and the management of print tasks.

Application Performance Considerations

Here are some recommendations that could help the performance of particular server functions.

Print Server: The print spooler function can severely degrade server performance if it is running on a file server. Most spool files are accessed only once and derive no benefit from caching. The data sould be written and read directly off the server hard disk. It is advisable to move the print server function onto a separate machine. Low speed PCs can often be used as print servers, as processor speed is not normally an issue. It is even possible to install a low speed DOS-based machine as a print server using IBM PC LAN Program and accessing the DOS print server as an external resource from the OS/2 LAN Server domain.

Client-Server Application: OS/2 provides an excellent platform for servers of all types. The multitasking operating environment with memory protection enables you to run multiple server functions within the same machine with a high level of confidence. However, if your file server is used under heavy load, it is not wise to run a mission critical client-server application within the same system, unless LAN server is a prerequisite.

Applications running on the server will, by default, be of a lower processing priority than the LAN server functions. This can be changed for the Entry Server by modifying bit 6 of *srvheuristics* (see Chapter 3 of *LAN Server Network Administrators Reference Volume 2 : Performance Tuning*). This will degrade the performance of the server while the client-server applications are running. This may be an adequate solution if the file server services on that system are not heavily used. To get optimal performance out of both servers it may be necessary to install each one on physically separate systems.

Database Application: With the industry trend towards downsizing from mainframe to smaller database server which provides the support for mission critical or line of business applications, it has become popular to develop SQL database applications which use a more efficient design in the way that data is queried on the server. These database applications follow the client-server design model which utilizes a client front end application which will pass SQL statements to the server back end database engine. One of the advantages from a performance aspect is the reduction of LAN traffic to execute a given database query.

Performance characteristics of an SQL database application utilizing a client-server design model will vary widely. Some basic performance characteristics and consideration are:

1. An SQL database application will generally produce less network traffic than a flat file database application for the same database query or instruction.
2. A back-end SQL database residing on a server will utilize the CPU, memory and disk subsystem more intensively than the flat file server.
3. Upgrading the CPU can have a direct benefit to SQL database performance, which is not always true for other file based applications.

It is recommended that you install DB2/2 on a separate machine from OS/2 LAN Server. This is because DB2/2 has its own cache management. Any memory you dedicate to DB2/2 is taken away from DISKCACHE or CACHE386 on LAN Server. If you run both server types within the same system you may need a very large amount of memory to maintain a good performance.

Multimedia Application: To understand the issues involved with running multimedia applications over a network, we must first understand how multimedia data is different from standard data. Multimedia data has different requirements in terms of data transfer method and band width.

Data Transfer Method - Unlike standard office automation applications such as word processing and spreadsheets, multimedia demands that large amounts of sequential data be delivered in a continuous manner. In other words, for the multimedia data to be usable, it must be kept together,even if the network experiences a slowdown in the data transmission rate. Multimedia is more concerned with all the necessary data arriving in a complete stream, as opposed to transactional data that is delivered in short bursts of small packets.

Bandwidth Requirements - For many network applications, bandwidth limitations can create bottlenecks. Although LAN Server Advanced is equipped to transfer data very quickly, low-level network protocols and hardware topologies can only handle so many bytes per second. These bandwidth limitations are considerations for multimedia as well. Some performance considerations when running multimedia applications is that they require:

1. More disk space than most current network applications
2. High speed LAN adapters
3. Faster video displays (client only)

IBM supplies an add-on software to LAN Server specifically for use on multimedia data servers. LAN Server Ultimedia* provides guaranteed delivery of multimedia data on token-ring networks for up to 40 DOS, DOS Windows or OS/2 clients. Most common multimedia data formats are supported. Refer to *Understanding IBM OS/2 LAN Server Ultimedia Version 1.0* for more detail.

Chapter 2. Server Hardware Review

Before we discuss the detail of LAN Server architecture and performance tuning related topics, it is probably a good idea to review the considerations related to the server hardware.

2.1 Processor Hardware Considerations

Processor is the first server subsystem we will consider in this document. We will discuss how the system processor and bus architecture affects the performance of a particular server.

Potentially the system processor can be a major source of performance degradation in servers. To eliminate any performance degradation caused by the processor there are two main approaches that can be adopted:

- The processor could be upgraded to a faster speed so that the processor can cope with the workload demand.
- The workload could be off-loaded from the processor and redistributed more evenly.

While a faster processor is necessary in some cases, such as in application and database servers, replacing a processor with a faster one may just disguise an underlying problem within the server system that will reoccur at a later date. It is therefore vital that you understand the processes and load on the server system.

The following information in this chapter describes the various processor characteristics that can increase the performance of a server. Thereafter suggestions on how to redistribute the workload within the system is discussed.

The choice of a suitable processor for a server is one of the major considerations for the performance of a server. Ideally the PC with the better specification is the most appropriate; however, this may not be economically viable if the server workload is not excessive. In some cases the processor may not even be the bottleneck and therefore upgrading it would be a fruitless exercise. To determine the relative performance benefits within the processor the following considerations have to be taken into account.

Processor Complex

In the first PS/2* models, most components were integrated into the planar of the system. This severely limited upgrade options and upgrade flexibility. While one component was upgraded, for example the processor, the other components such as the I/O controller and the memory controller were not. This created combinations of fast and slow components, which created unbalanced systems. Unbalanced systems are not as efficient as balanced systems where every components′ performance is matched against other components′ performances.

With this in mind, the server key components have been grouped together on a separate card known as a processor complex. Now the processor is contained on a removable processor complex board, which also holds the processor/memory bus, the memory controller, DMA controller, and Micro Channel* bus interface. Placing the processor on a complex together with key components means that when a system is upgraded, balanced systems performance can be maintained.

IBM has provided an upgrade path for existing and future file servers that allows network design engineers to replace the system processor complex with a faster and more efficient system processor complex at a later date. This policy of upgrading allows the server to accommodate increased server CPU utilization without the need to buy a complete new machine.

Within the processor complex there are many features that are capable of providing more efficient data transfer. They consist of:

- Cache
- Dual Path to Memory
- Two-Way Interleaved Memory Banks
- 32-bit DMA Controller
- 40MBps Data Streaming

Processor Cache

Processor cache is a memory that can be accessed 5 to 10 times faster than standard memory. Cache memory uses Static Random Access Memory (SRAM) which is much faster than the Dynamic Random Access Memory

(DRAM) used for system memory. SRAM is more expensive and requires more power, which is why it is not used for all memory.

There are two levels of cache. The cache incorporated into the main system processor is known as Level 1 (L1) cache. The 486** incorporates a single 8KB cache. Pentium** features two 8KB caches, one for instructions and one for data. These caches act as temporary storage places for instructions and data obtained from slower, main memory. When a system uses data, it will be likely to use it again, and getting it from an on-chip cache is much faster than getting it from main memory.

The second level of cache, called second-level cache or Level 2 cache, provides additional high speed memory to the Level 1 cache. This additional cache memory works together with the cache memory native to the main processor (L1). If the processor cannot find what it needs in the processor cache (a first-level *cache miss*), it then looks in the additional cache memory. If it finds the code or data there (a second-level *cache hit*), the processor will use it, and continue. If the data is in neither of the caches, an access to planar memory must occur.

Within Level 2 cache there also two possible types of caching:

1. Write-Through Cache

   Read operations are issued from the cache but write operations are sent directly to the standard memory. Performance improvements are obtained only for read operations.

2. Write-Back Cache

   Write operations are also done to the cache. Transfer to standard memory is done if:

   - Memory is needed in the cache for another operation
   - Modified data in the cache is needed for another application

It is believed that OS/2 uses much wider memory area than DOS or Microsoft Windows**, so write-back cache is sometimes worse than write-through cache mode.

Dual Path to Memory

When bus masters were implemented on Micro Channel servers, it was found that there was often contention for memory access between the processor and the bus masters, and that the processor was being delayed waiting for bus masters to release the path into memory.

The new design of the processor complexes addresses these issues by providing a *dual-path* into memory, effectively providing two paths to system memory, one from the processor and one from the Micro Channel.

These two separate paths to system memory allow overlapping of processor and bus master cycles. Three different kinds of overlapped cycles can occur:

- Processor reads to Level 2 cache simultaneously with bus master I/O
- Processor reads to Level 2 cache simultaneously with bus master memory access
- Processor reads to memory simultaneously with bus master I/O

In addition, both processor and Micro Channel cycles are buffered into 16 bytes blocks, further alleviating the contention for memory by reducing the frequency of the accesses. Implementing dual-path access to memory and the buffering of cycles can give a system throughput of up to three times that of a server without it.

Two-Way Interleaved Memory Banks

Another performance advantage is gained when the processor is accessing memory in burst mode. Memory is split into two banks, and data or code is stored sequentially across these banks; for example addresses 0 and 2 are held in bank 1, and addresses 1 and 3 are held in bank 2. The reason for this arrangement is that when a 486 burst mode request is made, the accesses to memory will be sequential. When the memory controller detects such a burst request from, for example, bank 0, it also pre-fetches the next 32 bits of data from bank 1. This way, the processor is not kept waiting while the information is being retrieved from memory.

32-bit DMA Controller

A Direct Memory Access (DMA) controller is a dedicated unit with the ability to move data between system memory and a device on the Micro Channel. It is used by simple adapters, and also by the parallel and serial ports.

Earlier versions of the PS/2 Model 95 implemented a 24-bit DMA, limiting DMA memory transfers to below 16MB (whereas the 486 processor was able to address up to 4GB of memory). On 32-bit systems with more than 16MB of memory, this could cause problems if a DMA access was for memory above 16MB. The operating system could work around the problem by ensuring that

DMA buffers were always below 16MB when a DMA transfer was done, but this imposes a performance penalty.

40MBps Data Streaming

The 40MBps data streaming transfer offers considerably improved I/O performance.

As in many cases, blocks transferred to and from memory are stored in sequential addresses, so repeatedly sending the address for each four bytes is unnecessary. With data streaming transfer the initial address is sent, then the blocks of data are sent and it is then assumed that the data requests are sequential.

IBM SynchroStream

At the heart of the computer, data is moving continually between processor, cache, main memory and the Micro Channel. Typically there is a single path to memory, so fast devices like processors have to wait for much slower I/O devices, slowing down the performance of the entire system to the speed of the slowest device. The IBM SynchroStream controller was designed to overcome this problem. It synchronizes the operation of fast and slow devices and streams data to these devices to ensure all devices work at their data at their optimum levels of performance.

Synchrostream is an intelligent device in that it predicts what data the devices will need and loads it from memory before it is requested. When the device wants the data, it is presented to it from the IBM SynchroStream controller and the device can continue working immediately, as it does not have to wait for the data to be collected from memory. When devices are moving data into memory, the IBM SynchroStream controller holds the data, and writes it to memory when it is most efficient to do so.

Since devices are not moving data to and from memory directly, but to the SynchroStream controller, each device has its own logical path to memory. Devices do not have to wait for other slower devices.

The SynchroStream engine operates by using a spinning valve that continuously forms different connections between pipes. Once a connection is made, data is streamed to the Micro Channel or processor at the highest possible rates. Parallel paths allow data to stream to multiple sources at the same time. The pipes even continue to stream after the connection is changed. Data is always streaming to the Micro Channel and processor, allowing them to operate at full bandwidth.

IBM used the latest in chip design technology to integrate all SynchroStream functions on a single chip, improving performance dramatically by not having to move data between chips. The IBM SynchroStream controller uses a single RISC-like chip architecture to move data fast and efficiently between memory and requesting devices, as shown in Figure 8.

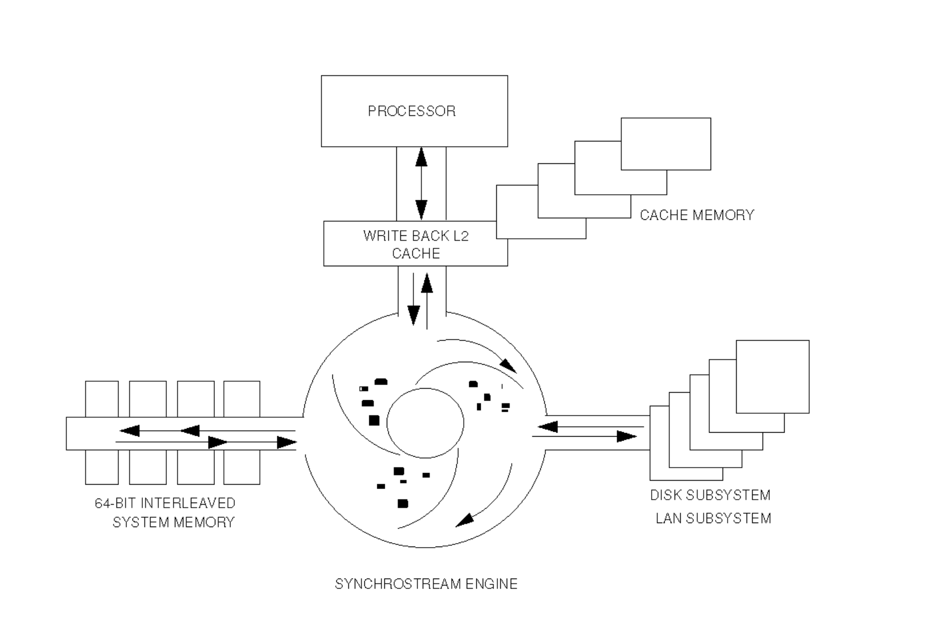

Figure 8. SynchroStream Technology

The IBM SynchroStream controller is located on the i486 DX2 66MHz and Pentium processor complexes, featured in the Server 95 and Server 95 Array systems. The implementation on the processor complex means that current PS/2 Server 95 and PS/2 Model 90 users can easily upgrade their machines to have IBM SynchroStream controller functions.

The key advantages of the SynchroStream technology are as follows:

- Fast single chip implementation

Competitive designs are multi-chip and have the performance overhead of moving information between chips. SynchroStream technology provides a *Zero Wait State* Pentium implementation.

- Intelligence

  IBM SynchroStream is intelligent in that it predictively loads data from memory so that requesting devices are not kept waiting. In addition, writes to memory are stored within the IBM SynchroStream controller and written to memory to optimize memory utilization.

- RISC-like architecture

  Pipelines are used to move data in a fast, efficient manner between memory and the requesting device.

- Stream data to Micro Channel devices

  SynchroStream can stream data to Micro Channel devices at 40MBps.

- Upgradable system implementation

  Competitive system designs do not have the unique Upgradable Processor Complex design so you cannot upgrade to SynchroStream-like functions from earlier models.

2.2 Bus Architecture

The system bus provides a path for data transfer to be transferred between the CPU, memory and peripherals. It provides a method for resolving the many conflicts that can arise during a data transfer. From a performance point of view the most important characteristic of the bus is its speed. The bus must be fast enough to transfer data from high speed peripherals to the host CPU with no noticeable degradation in system performance.

Throughout the evolution of the microprocessor the system bus has progressively enhanced to maintain the level of performance serviced by the CPU and memory. Presently there are many types of buses to choose from each having their own unique characteristics. First there is the AT* expansion bus which is a 16-bit data bus designed mainly for the Intel** 80286 microprocessor. It is universally acknowledged as the industry standard and is commonly referred to as the Industry Standard Architecture (ISA) bus. It is a relatively slow bus and specifically designed to run slow enough to keep the more sluggish peripherals from holding up the faster CPU.

However, now as the peripheral performance and microprocessors have improved everything is running faster except the ISA bus and data transfer especially between the processor and memory is very slow. To cater for the newer technology two other buses have been designed with improved characteristics. The Micro Channel Architecture (MCA) bus was developed by IBM and the Enhanced Industry Standard Architecture (EISA) was developed by a consortium of developers. Both these architectures sought to overcome some of the limitations in the ISA design. They both provide an extended 32-bit data bus and more tightly specified bus transfer protocols which generated greater transfer rates.

In addition, both buses have the capability to support multiple bus masters. That is to say, the system allows expansion boards to take complete control of the bus for certain operations. Bus mastering is essential especially for servers as they normally support intelligent drive controllers and network interface cards. Data can now be transferred from a peripheral card without affecting the CPU control program. This essentially off-loads some of the host processor′s tasks and introduces concurrent processing.

The primary limitation of a bus is the response times of the cards in the bus and the nature of the motherboard logic, which uses wait states to slow down the bus to synchronize with its own internal response times. There is no need to use an advanced EISA or MCA bus when the primary limitation is the processor or is network-related.

Local Bus Technology

As RAM is the most speed critical resource that a CPU needs, to improve system speed the standard bus limitations can be bypassed. Local bus technology essentially closely couples the peripherals to the CPU and memory. Local bus provides peripherals with direct access to the CPU along a 32-bit wide data path, and they can be along side an ISA, EISA or MCA standard bus. The result is faster data transfer between the CPU and the peripherals. The type of standard bus used does not affect the performance of the local bus. Modern drives can produce data faster than the ISA bus can transfer it. Local bus technology promises to close this gap by accelerating data transfer rate from 2 to 10 MBps. The three main types of local bus are:

.

- Proprietary local bus is located on the motherboard itself and limited to a specific vendor
- VL-Bus (VESA local bus) is created for the 386/486 architectures
- PCI (Peripheral Component Interconnect) local bus is processor independent, which means it operates removed from the processor bus

Both the VL-Bus and proprietary local buses operate at the CPU′s external speed and PCI runs at 33MHz. For any peripheral to take advantage of this added performance it would have to be fast enough to operate at high speeds, if not the peripheral components will limit the speed your system can achieve.

Hard disks with buffers and RAM cache on the disk controller connected to the local bus can speed up data transfers. In this case track-to-track access would not generate a bottleneck within the system and data can be moved from cache to the CPU via the local bus. This provides significant performance improvement over the same controllers on either EISA or MCA.

PCI Local Bus

PCI is technically a more elegant solution to the problem of displaying more information faster. Compared to the VL-Bus and other local buses, PCI offers higher performance, automatic configuration of peripheral cards and superior compatibility.

It is not exactly a true bus, but one step removed from the system′s processor bus. It occupies an intermediate level between the processor and standard expansion buses such as ISA, EISA and MCA, with electronic bridges spanning the gap. These bridges are essentially the PCI controller and expansion bus interface, as in the Figure 9 on page 28, and they both translate signals from one bus to another.

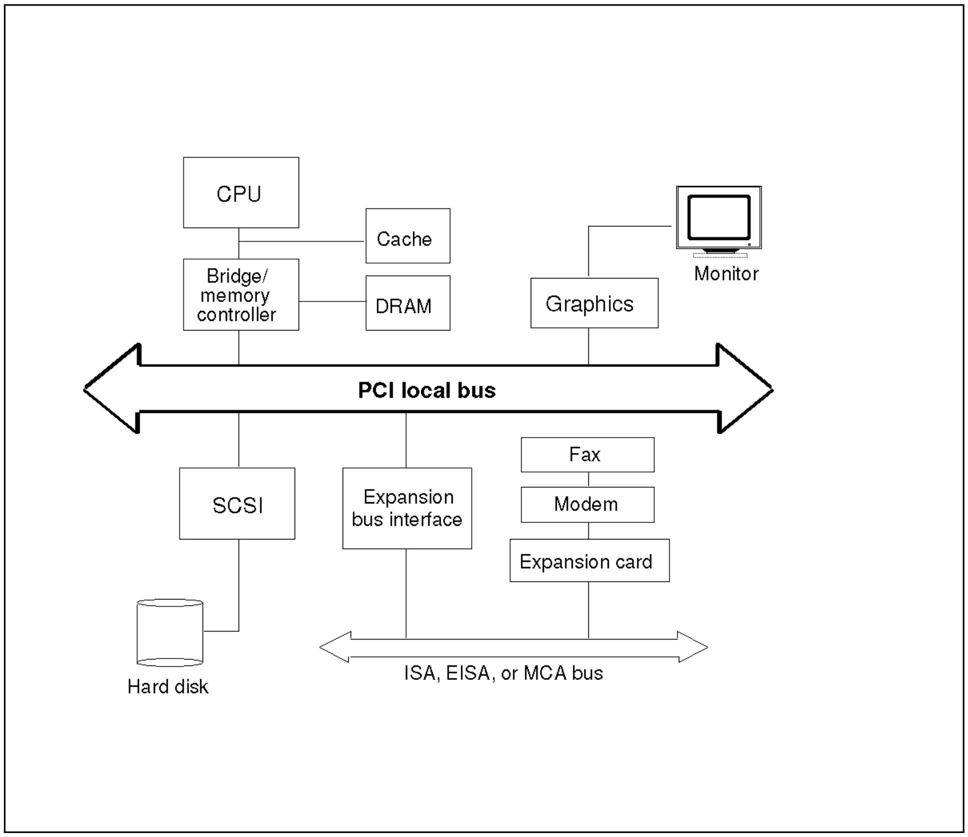

Figure 9. PCI Local Bus Architecture

By isolating the PCI bus from the CPU′s local bus the PCI local bus has two main advantages. One, it can now support more devices than any of the other local buses, because PCI devices won′t electrically load down the CPU bus, and the other is that it is capable of supporting faster processor speeds. In theory it would be possible to connect a PCI bus to a 100MHz Pentium where a true local bus would be out of the question.

Generally speaking the overall performance of the PCI and VL-Bus standards on a 486 is very much the same for both read and write operations. However, it is only within Pentium systems where the technical capabilities will supersede the VL-Bus. Within true multitasking operating systems such as OS/2, PCI can provide better data throughput as it is capable to work concurrently with the CPU. It creates improved performance without burdening the processor. On current VL-Bus designs it is not possible for the CPU to operate independently of the local bus.

As a result, PCI can be beneficial for:

.

- Servers that require top-notch hard disk or LAN adapter throughput, and have the peripherals that take advantage of the bus speeds.
- Multimedia servers or servers that use many applications within a graphical user environment. This is important especially when using OS/2 as the base operating system and for the new GUI interface for LAN Server 4.0.

2.3 Faster Processor and Multiprocessor

With the recent advanced technology the faster processor chip and the system unit are available but appropriate software support is required to fully support these advanced technologies.

Pentium Processor

IBM has used the Intel i386**/i486 processors as the basis of its system processor complexes for many years. More recently, the choice has broadened to include the new Pentium processor. The Pentium CPU is designed to process more than one instruction at a time by a technique called *superscalar execution*. This not only provides superior performance when handling data transfers, but also maintains compatibility with applications written for the Intel i386/i486 family of processors.

However, performance enhancements can be made if an application written for the Intel i386 processor is redesigned to use this superscalar feature. The simple C-programs might get some performance improvement by using some of the available compilers that support the Pentium optimizations. However, optimizing a system code such as the LAN Server Advanced running in Ring 0 is not so easy, because 386 HPFS is already written with assembler language and achieved a high level of hardware optimization. So, the performance available from a Pentium based system can vary considerably depending on the software.

LAN Server Version 3.01 and Version 4.0 are aware that one of Pentium′s features which is active with 32MB or more memory size. However, the performance gain is not noticeable.

Symmetric Multiprocessing

Within a symmetric multiprocessor environment the server workload can be distributed over multiple processors to optimize the performance of the processor activity. With LAN Server 4.0, symmetric processing is now supported and is able to obtain the same performance as other architectures under the same environment. However symmetric processing is only beneficial under certain conditions.

If the server experiences heavy CPU workloads and is utilized to its maximum capabilities causing a deterioration in server performance, then symmetric multiprocessing becomes useful. It enables other Ring3 applications workload to be distributed over the other processor(s), thereby off-loading the workload on the main processor. A good example of this is if the LAN Server has Lotus Notes or DB2/2 running on the same machine. As these two applications are processor intensive applications they will experience better response times if a multiprocessing environment is adopted.

2.4 Processor Hardware Summary

Upgrading the server processor is not always the right answer. There are a number of alternatives within LAN Server that should be explored before taking such drastic action.

It is unusual for simple file/print servers to be processor-bound as the majority of the server activity is involved in disk and LAN I/O functions. Both disk and LAN I/O problems may manifest as processor over-load symptoms if not addressed correctly. However, systems that are used as application or database servers may have a real processor over-load problem due to the activities performed by the processes running within the server system itself.

If the CPU utilization level reaches 80% frequently and remains there for more than a few seconds, performance is going to be impacted by the CPU′s ability to satisfy its workload demands. Depending on the specific server setup and workload one or more of the following adjustments can offer significant improvements to overall performance:

1. Use the appropriate version of LAN Server package for the function of your server. The CPU efficiency of the Advanced server is several times

that of the Entry server, particularly if the primary purpose is data sharing. The Advanced server offers a tightly coupled SMB server and 32-bit 386 HFPS file system which provide significant performance advantages on servers with high levels of redirected disk I/O. The section on disk I/ O addresses 386 HPFS tuning in detail.

2. Large memory size such as 32MB will enable a very large cache, so it will reduce the CPU utilization, in some cases the disk I/O uses the cycle.

3. High levels of I/O on the network interface card (NIC) can cause excessive levels of CPU activity. This is frequently demonstrated in server systems with single, or multiple shared RAM type LAN adapters installed. Each time an I/O request is made the processor must perform work. By replacing shared RAM type network interface cards (NICs)with busmaster type, the I/O processing is off-loaded onto the adapters own processor, freeing the CPU for more productive work. In cases of excessive LAN I/O, multiple busmaster NICs, for example, LANStreamer MC32, can offer significant performance improvements. Both LAN Server 3.0 and 4.0 support up to four NICs per server. Automatic load balancing between NICs is also provided to the same or multiple LAN segments during session initialization.

4. Redistribute server load between a number of servers to take advantage of excess capacity within the network as a whole. For example backup domain controllers can be used to relieve bottlenecks at the domain controller at logon time. Applications can be distributed between multiple servers (using products such as IBM NetDoor* to assist with administration and load-balancing.) Domain controllers, print servers and serial device servers do not require powerful processors and will not benefit from fast disks or the OS/2 LAN Server Advanced.

5. If the above suggestions do not provide sufficient improvement in processor utilization, a faster processor should be considered. If a Pentium processor is installed, make sure you are using LAN Server 3.01 or LAN Server 4.0 to take full advantage of the processor.

2.5 Disk Subsystem Considerations

The disk subsystem is a major component of server performance and is frequently the location of the server bottleneck. With the interdependencies of the CPU and disk subsystems and their close working relationship it is essential that there is a swift transfer of data between them.

In the event of the disk being over-utilized, which can normally be detected by the fixed disk activity indicator flashing on more than it is off for long periods of time, many considerations can be put into practice to improve disk I/O performance. It is important to understand how the hard disk drive subsystem operates and the relationship between the hardware and the operating system software. As a result, issues related to both hardware and software are discussed in this chapter giving an insight into the factors that can affect disk I/O performance.

Disk Storage Subsystem

The hard disk storage subsystem contains the files which have to be accessed by the LAN workstations. The hard disk performance is directly related to the file server performance. To understand how the disk controller services requests from the LAN workstation to read or write a file to disk and how this can affect performance it is necessary to understand the process by which LAN Server handles the disk read/write requests. One of the standard indicators of hard drive performance is the average access time. This is the amount of time required for the drive to deliver data after the computer sends a data request. Any application that needs frequent access to different parts of the hard disk, for example a database server, will benefit from a drive with a faster access time. There are many factors that control the duration of the disk access times. To achieve a higher data throughput and faster response time for a short record, the disk subsystem should have a better performance in the following specs:

.

- Average access time
- Maximum Transfer rate
- Shorter latency time
- On-board cache size

With the same physical size, for example 3.5 inch disk, the larger capacity disk will have a better access time or transfer rate.

Hard Disk Interfaces

The disk interface is the electronic liaison between a hard disk and a computer. The interaction established by the physical and electrical connection is handled by the drive controller. It is essential that the interface protocols and signals understood by the controller match or are compatible with those on the hard drive. There are four main types of hard disk interfaces each possessing different levels of drive performance:

1. **ST506** - This interface has been the microprocessor industry standard and has the capabilities to transfer data to and from the hard disk at 5 MBps. :
2. **Enhanced Small Device Interface**, ESDI - This is an enhanced version of the ST506 interface. It provides a greater data transfer rate, 10MBbps and in some implementations 15Mbps, and higher storage capacities.
3. **Intelligent Drive Interface**, IDE - This interface mainly use RLL data encoding, which results in denser storage and faster data transfer. IDE is similar to SCSI but its standard is built from an aspect of low cost and for PC use. The latest enhancements to this interface include caching at adapter level, CD-ROM interface, and extending the maximum disk storage beyond 500MB. However, IDE still limits maximum two hard disks per interface, this limitation makes IDE more favorable for personal desktop.
4. **Small Computer System Interface**, SCSI - This is a high speed parallel interface that transfers eight bits at a time rather than one bit at a time for the ST506 and ESDI serial interfaces. It is similar to IDE in that the controller is present on the drive itself instead of on a separate controller board. However, SCSI is suitable for server configuration because of its expandability and well established standard.

To provide improved performance SCSI adapters are normally used on the server to provide greater data throughput between external devices and the server. Its high data transfer rates allow enhanced performance over other, non-SCSI systems. A SCSI drive and controller connected to an EISA, MCA or PCI local bus is already capable of 10MBps transfers. There will be more and more PCI based SCSI adapters available in the market, along with PCI based Pentium systems.

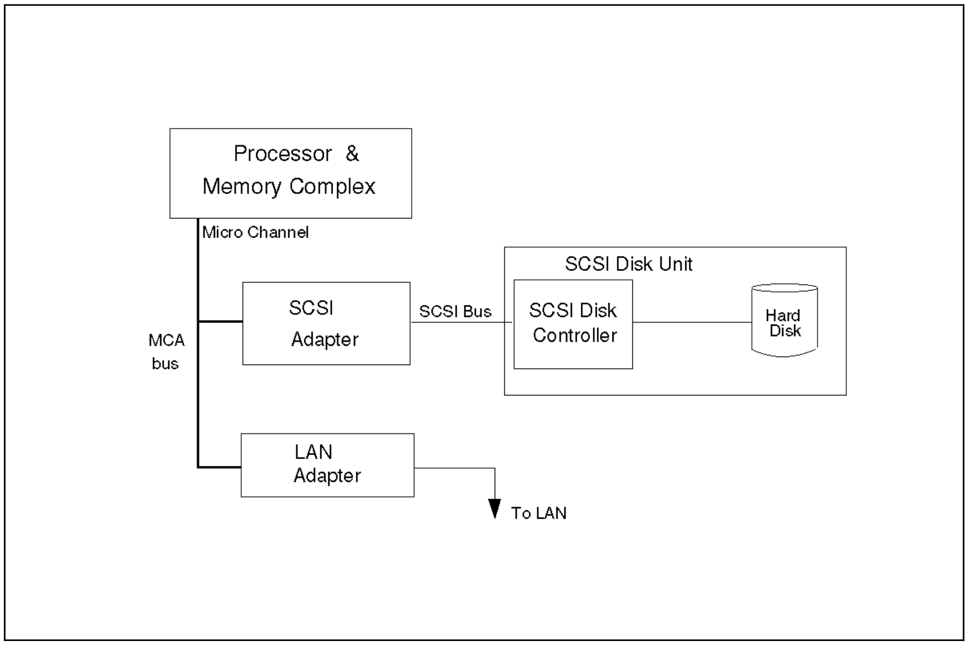

Figure 10. SCSI Disk Interface

SCSI Technology

The most widely used disk subsystem technology in advanced servers is SCSI. It is a standard interface bus through which computers may communicate with attached intelligent peripheral devices such as fixed disks, CD-ROMs, printers, plotters, and scanners. With SCSI, a large number of devices of different types can be connected to the system unit via a single SCSI bus cable and a SCSI attachment feature. This SCSI attachment feature may be in the form of an adapter or an integrated unit on the planar board. The SCSI interface is also device independent, allowing you to attach intelligent devices of any form that adhere to the SCSI standard.

SCSI has the ability to provide features such as arbitration and disconnect / reconnect, allowing several devices to be operate concurrently and share the SCSI bus efficiently. This is an obvious benefit in a multitasking environment.

In essence the SCSI structure defines the following:

.

- Bus structure
- Device addressing

- Data transfer width and rate
- Common command set

Write Caching: For a high performance, the drive should be able to provide write caching features. Normally during a computer write cycle the system copies the data from memory to the memory on the drive. It has the ability to continue to work but cannot until the drive signals the system that it has completed the actual write operation. To eliminate this valuable idle time, write caching is adopted. Now the drive signals the completion of the write immediately after it receives the data and before the data is actually written to disk. The system then continues to process data while the hard disk is actually writing the data. Performance is significantly better because subsequent write operations can overlap getting data from the system to actually storing the cached information on disk. Matching the performance of your drive and SCSI adapter is vital for getting the best performance from the available resources. Slower controllers can seriously degrade performance and become the bottleneck in a disk I/O transfer; therefore, faster SCSI drives are required that can support SCSI-2 fast speeds. Of course, you want to have an UPS (uninterruptible power supply) in order to prevent the data loss.

SCSI Adapters: In a server each component has to be optimized. Together with the disk subsystem the SCSI Adapter forms an integral part of the servers performance. IBM has several offerings in this arena. They are:

.

- IBM Personal System/2* Micro Channel
- SCSI Adapter with Cache
- IBM SCSI-2 Fast/Wide Adapter/A
- IBM SCSI-2 Fast/Wide Streaming-RAID Adapter/A

Note

.

- *FAST* refers to the doubling of the data transfer rate from the SCSI-1 5Mbps to 10Mbps.
- *WIDE* generically means wider than the original 8-bit path defined in SCSI-1. Its use is currently limited to mean the 16-bit implementation as 32-bit is not currently in use.

- ***IBM Personal System/2 Micro Channel SCSI Adapter***

  The PS/2 Micro Channel SCSI Adapter is a 16-bit Micro Channel bus master adapter that provides the ability to connect multiple SCSI devices to all PS/2 Micro Channel systems. This Micro Channel SCSI adapter adheres to the industry standard SCSI interface. It features an 8.3MB per second burst data transfer rate, 16-bit data path with 32-bit address capabilities, and can be installed in either a 16- or 32-bit system card slot. A SCSI physical device could be a fixed disk, scanner, plotter, printer, CD-ROM, or controller.

The bus master capability of this SCSI adapter optimizes data flow from each SCSI device configured to the system. This capability can provide performance benefits in applications where multitasking or high-speed data flow is essential. It allows the processor to be off-loaded from many of the input/output activities common to DASD transfers. This SCSI Adapter also conforms to the Subsystem Control Block (SCB) architecture for Micro Channel bus master.

- ***IBM Personal System/2 Micro Channel SCSI Adapter with Cache***

  The redesigned PS/2 Micro Channel SCSI Adapter Cache provides system configuration flexibility to support multiple internal and/or external SCSI devices such as high performance fixed disk drives, CD-ROM drives, tape drives, scanners, and printers. It has a burst transfer rate of 16.6MBps, adheres to the industry standard ANSI SCSI interface and is supported in either a 16- or 32-bit Micro Channel slot.

  The IBM PS/2 Micro Channel SCSI Adapter with Cache is a 32-bit busmaster SCSI adapter containing a 512KB cache buffer that allows system memory to be totally dedicated to running the application rather than a portion being reserved for software caching. This SCSI adapter is recommended where improved data transfer rates and multiple SCSI devices are required and system memory is constrained. However, 512KB cache is too small for the LAN Server application.

- ***IBM SCSI-2 Fast/Wide Adapter/A***

  The IBM SCSI-2 Fast/Wide Adapter/A is a SCSI-2 adapter that IBM has announced as an option for it′s range of PS/2s. Fast refers to a data transfer method. With *fast*, data is moved to fixed disks at 10MBps--twice the speed of SCSI 1. Wide refers to the bus width which is increased from 8 to 16 bits, enabling transfers of up to 20MBps.

  The IBM SCSI-2 Fast/Wide Streaming Adapter/A is a high performance SCSI-2, 32-bit Micro Channel 40MBps Data Streaming bus master adapter. It has dual SCSI-2 16-bit, fast and wide channels (one 20MBps internal, one 20MBps external). The dual bus of the adapter prevents access to internal DASD from the external port and also allows the maximum cable length to be calculated individually for each bus. This allows for additional capability externally.

This adapter has a dedicated 80C186 local processor on-board. It supports SCSI Tagged Command Queuing (TCQ) which increases performance in DASD-intensive server environments. With SCSI-1 systems, only two commands could be sent to a fixed disk - the disk would store one while operating on the other. With TCQ it is possible to send multiple commands to the fixed disk, and the disk will store the commands and execute each command in the sequence which will give optimal performance.

The supporting cables and terminators allow attachment of SCSI devices internally or externally for PS/2 Models 8590/95 and 9585/90/95. Up to seven SCSI physical devices may be attached. The IBM SCSI-2 Fast/Wide Adapter/A also supports standard 8-bit SCSI devices using either asynchronous, synchronous, or fast synchronous (10MBps) SCSI data transfer rates per ANSI Small Computer System Interface 2 (X3T9.2/375R REV10K) for SCSI-2 features.

- ***IBM SCSI-2 Fast/Wide Streaming-RAID Adapter/A***

  The IBM SCSI-2 Fast/Wide Streaming-RAID Adapter/A is a high performance SCSI-2 RAID fast and wide (16-bit SCSI bus) adapter capable of 20MBps SCSI bus transfer rates. The combination of SCSI-2 Fast/Wide offers substantial performance increases over current SCSI solutions. RAID offers the additional data protection security inherent in RAID configurations. If you need to attach IBM SCSI options to a PS/2 Server (9585 or 9595) the SCSI-2 Fast/Wide Streaming-RAID Adapter/A is a cost-effective solution. This adapter offers functional advantages by supporting a wide range of IBM options, including the hot-swappable fast and wide hard drives.

  The SCSI-2 Fast/Wide Streaming-RAID Adapter/A supports a 40MBps Micro Channel streaming data rate. Microcode is optimized for database and video server environments. Two independent SCSI buses are available for internal and external array configurations, further enhancing performance and fault tolerant configurations. The adapter also supports 8-bit SCSI devices using either asynchronous, synchronous, or fast synchronous (10MBps) SCSI data transfer rates per ANSI Small Computer System Interface 2 (X3T9.2/375R Rev10K) for SCSI-2 features. The dual bus of the adapter allows for maximum connection of up to 14 drives, seven on each individual bus; for example, one bus cannot support internal and external devices simultaneously.

Table 4. SCSI Adapters Summary				
Attribute	**SCSI Adapter with no Cache**	**Enhanced SCSI Adapter with Cache**	**SCSI-2 Fast/Wide Adapter/A**	**SCSI-2 Fast/Wide Streaming RAID Adapter/A**
SCSI Bus Width	16-bit	32-bit	32-bit	32-bit
Micro Channel Data Transfer Rate	8.3 MBps	16.6 MBps	20 MBps	40 MBps
Parity	Optional	Optional	Yes	Yes
Tagged Command Queueing (TCQ)	N/A	N/A	Yes	Yes
Note: Fast and Wide are data transfer methods as defined in SCSI-II N/A = not available				

Recommendation: Consider PCI based fast/wide SCSI adapters if your system has a PCI bus. You could use IDE interface as a boot drive if motherboard has an integrated IDE interface. For EISA or MCA machines, use fast/wide SCSI adapters which will work in busmastering mode.

It is also worth distributing the disk intensive workload from a single physical disk drive to multiple disk drives, enabling concurrent disk seeks and read/writes. If this is done, SCSI disk controllers should be used to allow over-lapped read-writes. Multiple SCSI disk controllers can provide further performance enhancements to the point where the rate that the disk head can write to disk itself becomes the bottleneck.

LAN Server Advanced and network device drivers that support scatter/gather read/write for busmaster adapters, which allows devices to transfer data to and from the scattered cache area independently of the CPU, increase CPU overlap.

2.6 Disk Subsystem Summary

The following is the summary of this section:

1. The hard disk subsystem plays an important role in providing resource sharing. Since the read and write requests from the LAN workstation are

handled by the disk controller, the efficiency of the input/output (I/O) of this subsystem can greatly affect the overall performance of the file server.

2. The choice of disk controller and the number of controllers per disk can have a marked effect on disk I/O performance. Using the latest disk controller subsystems which have integrated busmastering technology can help performance. IBM RAID configuration can provide performance advantages in addition to data protection.

3. Use some of the available tools to view the disk drive status, number of files and directories, available cache buffers and system memory. Generally, noting the changes in the file server as part of a planned network management strategy, is key to identifying bottlenecks in the server subsystems.

4. Redistribute users, files or applications to other servers to take advantage of excess capacity elsewhere on the network.

Chapter 3. LAN Server Architecture and Tuning

Before going into the performance issues of LAN Server it is first worth understanding the basic architecture of the product. Both the Entry and Advanced servers are explained here illustrating the differences between the two architectures. This manual is intended to provide extra information to the base publication *OS/2 LAN Server Version 4.0, Network Administrator Reference Volume 2: Performance Tuning*. If you need a complete description of IBMLAN.INI parameters such as setting up **wrkheuristics** or **srvheuristics**, you should refer to the base publication.

3.1 Design Overview of LAN Server - Entry

The Entry server runs at application privilege level (Ring 3). The Entry server uses OS/2 services to satisfy network file I/O requests, session setup, and resource sharing. Network file I/O requests and responses are sent by way of SMB frames. The Entry server processes SMBs using internal network buffers called *request buffers*. The IBMLAN.INI file parameters that define the size and number of network buffers on the server are **sizreqbuf** and **numreqbuf** and corresponding buffers on the requester are **sizworkbuf** and **numworkbuf**.

An SMB received from the network is copied into the adapter receive buffers by the network adapter. The NetBIOS device driver called NetBEUI copies the data from the adapter's receive buffer into the request buffer, using a global descriptor table (GDT) selector. NetBEUI will send the acknowledgement to the message or piggyback the acknowledgement on a subsequent frame. The PROTOCOL.INI file contains the configuration information for NetBEUI.

The SMB is passed through the redirector to the server. The redirector is a requester component that directs file system request traffic among the server, the file system and the network. There are three types of SMB protocols that are used for transferring data between a requester and a server. The detail of SMB frame type is discussed in the 3.7, "Logon Performance Tuning" on page 83.

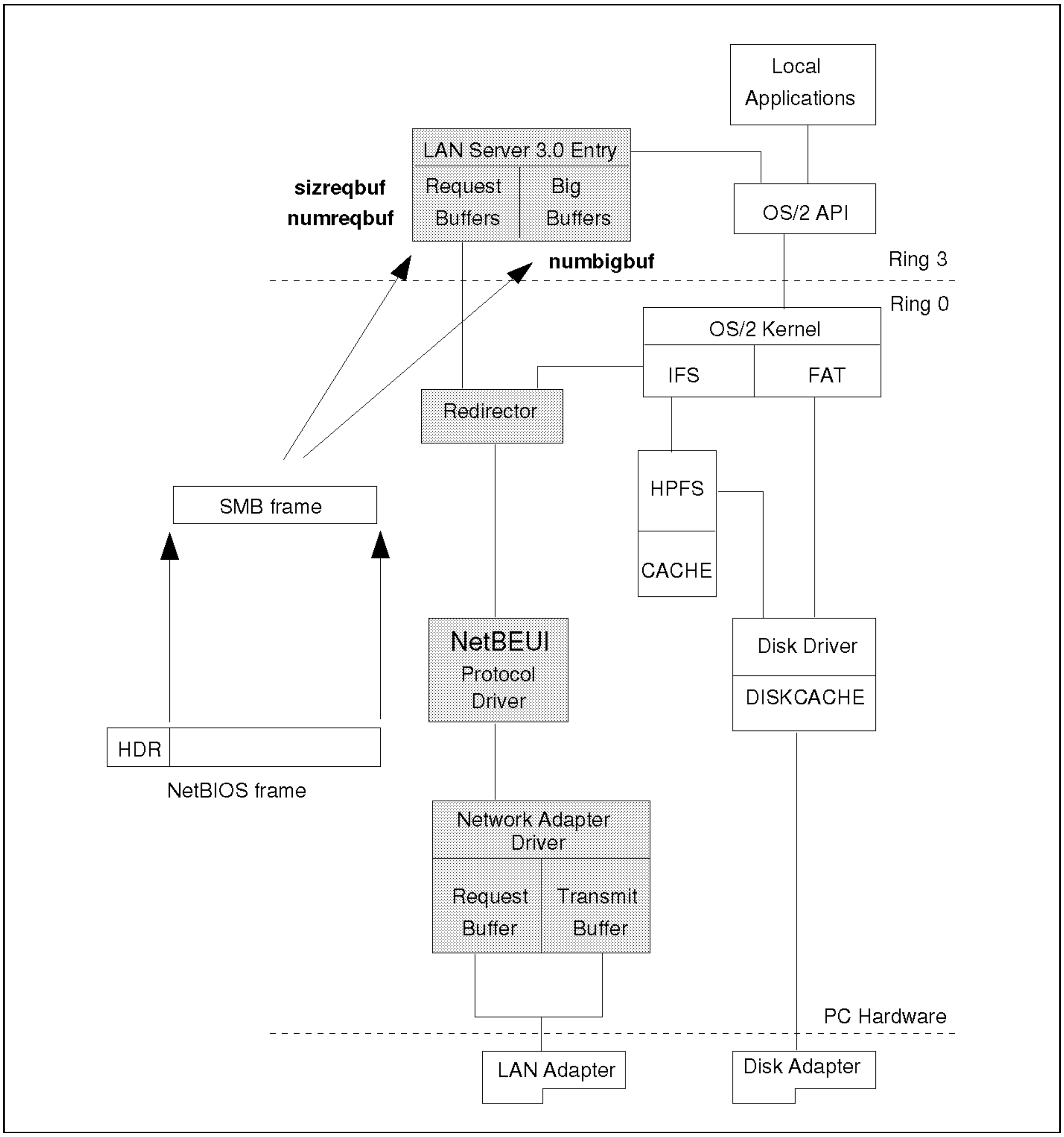

Figure 11. LAN Server Entry - Components

LAN Server Entry supports both the file allocation table (FAT) file systems and the high performance file system (HPFS). Both the FAT file system and an HPFS have a cache that is used to improve performance by keeping frequently used data in memory. A cache is a memory block that will contain frequently accessed data from the disk. Cache for Entry server is managed by OS/2. Due to a 2MB cache size limit, the OS/2 HPFS has a default threshold value of 2KB, so cache will be flushed by large file transfers. This value can be changed by the user if OS/2 2.1 or higher is used.

For Entry server, I/O requests to the file system are made using OS/2 APIs. Once the API call finishes, the server returns the SMB to the requester as an SMB response.

Requests from the requester are normally handled through request buffers. The default size of request buffers for the server is 4096 bytes. If the request (SMB request or response) is larger than the request buffer size, then big buffers will be used. The size of a big buffer is 64KB, so big buffers can satisfy a large SMB request. In anticipation of the next request, the Entry server performs read-ahead independently of the file system using its big buffers regardless if the RAW SMB protocol is used or not. RAW SMB protocol is one of the SMB protocols type and a detail is described in 3.7, "Logon Performance Tuning" on page 83.

Ring 3 server uses a unique algorithm to allocate big buffers at the initialization time. Three or four big buffers will be allocated during the initialization time and **numbigbuf** parameter doesn't affect this initial allocation. Ring 3 server will dynamically allocate additional big buffers if required. The NET STATISTICS command shows how often the server had exhausted the big buffers. However, this only means that DosAlloc failed to get a memory space for the dynamic big buffers. If the number specified as **numbigbuf** is not enough, Ring 3 server will just add more by requesting extra memory to OS/2. So, if NET STATISTICS indicates big buffers are exhausted, you need to add more memory to the system.

On the other hand, request buffers are not dynamically extended. The NET STATISTICS command can check how often the server has used all the request buffers allotted. If the request buffers resource is never exhausted, there may be too many request buffers defined. In this case it is possible to reduce the **numreqbuf** parameter value on the server and free the unused memory for the system. The best setting of **numreqbuf** is to use all the request buffers when the server is in a peak workload. The value that is set for the **numreqbuf** parameter involves a *memory-versus-performance* trade-off. Request buffers and big buffers are the places where SMB frames are stored.

Requesters have corresponding buffers to the server's request buffer. It is called work buffer and the parameters in IBMLAN.INI are **numworkbuf** and **sizworkbuf**. The default number of **sizworkbuf** is 4096. When the server's

sizreqbuf and the requester′s **sizworkbuf** is not matched, the smaller size will be used for the session and memory resources will be wasted. There are situations in which the 4KB default value for the server and requester buffers can be decreased such as 2KB, however you will see more big buffers to be used to satisfy the large SMB requests.

LAN Server incorporates several design features that attempt to improve the LAN performance based on expected behavior of software applications used on the network. However, the behavior of all possible applications cannot be predicted and handled in an optimum manner. The **srvheuristics** parameter for the server section and the **wrkheuristics** parameter for the requester section provides the ability to adjust the server′s behavior for specific applications.

OS/2 LAN Server Version 4.0, Network Administrator Reference Volume 2: Performance Tuning provides a comprehensive chart and description for these **srvheuristics** and **wrkheuristics** parameters.

Entry Server Cache Tuning

LAN Server Entry relies on OS/2 to provide the necessary caching to speed up disk reads and writes.

Since each file system controls its own cache, sufficient system memory must be available to allocate each cache. If both FAT and HPFS file systems are enabled together, the cache of the file system that is used for the more I/O intensive applications should be increased and the other file system cache size should be decreased, making the best use of the available memory.

To define FAT and HPFS cache, the DISKCACHE (for FAT) and IFS (for HPFS) statements must be configured within the CONFIG.SYS file.

DISKCACHE Statement in CONFIG.SYS

This parameter specifies the size of memory (in KB) to allocate for the FAT file system′s disk cache. The proper setting for cache and threshold depends largely on the particular environment. To determine the best size for DISKCACHE it is worth experimenting with different values.

If applications perform better with larger caches, then assigning a large DISKCACHE to the FAT partition would be beneficial. The maximum amount of DISKCACHE that can be assigned in memory is 8MB and to achieve best server performance it is advisable to specify as much DISKCACHE as possible. In general, the greater the DISKCACHE, the better the cache hit ratio which improves the server performance. If no FAT partitions are present then the amount of DISKCACHE is minimized to the default setting (64KB).

The threshold setting specifies the number of sectors (512 bytes) that will be placed into cache for read operations. If the threshold parameter is not specified, it defaults to a value of 4 (2KB). It should not exceed 10% of the total cache size or the specified value will be ignored and the default value will be used instead.

Any read operation that is less than the threshold is read into the DISKCACHE first. Therefore subsequent read operations will probably find the needed data in cache, thus improving performance. With this in mind, the threshold should be assigned depending on the applications that are running. If, for example you want to cache shared programs, it may be worth using a threshold of 64 (128KB), or else if you want to cache only random I/O, use a threshold of 2 (4KB).

Enabling the DISKCACHE lazy write option will improve performance in many cases.

BUFFERS Statement in CONFIG.SYS

For the FAT file system there is a parameter called BUFFERS that can be set to configure the disk buffering system. For HPFS and the HPFS386 file system with the Advanced server this is automatically configured by it′s own device drivers, so the CONFIG.SYS parameter is not used.

The BUFFERS parameter determines the number of disk buffers that OS/2 keeps in memory. Each disk buffer is a 512-byte (sector) portion of storage that OS/2 uses to hold I/O information temporarily. By implementing disk buffers, the operating system can read and write blocks of information more efficiently. It provides a means of flow control between the I/O device and the processor, enabling the processor to see a continuous flow of data being generated from the I/O device, preventing it from unnecessarily waiting for new pieces of information.

If you simultaneously run a large number of programs, you could increase the speed of the system by increasing the buffer value. This is largely true if there are a great number of reads and writes that are less than a sector.

Note

For OS/2 2.x the disk buffers provides memory to cache FAT directory information which can improve performance for an Entry Server especially if it has a complex FAT directory structure. No benefits are experienced for HPFS386.

CACHE Parameter in HPFS IFS Statement

`/CACHE:` parameter in HPFS statement specifies the necessary HPFS cache and threshold sizes. If the cache size is not specified then the default value for CACHE is assigned to 10% of the total physical memory available. A cache size of at least 512KB is normally recommended, however, unless the system is memory constrained, use up to the maximum size for cache (2MB) to obtain best performance.

By default, HPFS assigns a threshold value of 4KB, which means the request smaller than 4KB will be cached. The maximum threshold for cache is 64KB, and should be used if the applications performs better under these conditions and if a sufficient cache size is allocated. It should be noted that if the threshold value exceeds 25% of the total cache size then the specified value will be ignored and the default value will be used instead.

To initialize the cache for HPFS partitions the IFS parameter must be used. HPFS cache is a separate memory area from BUFFERS and DISKCACHE parameters.

Entry Server Memory Optimization

There is always a trade-off between the performance of the server system and the amount of memory installed. You must determine the values that best satisfy your network. If the server is configured with less than the ideal amount of memory then a number of actions can be taken.

1. Adjust the DISKCACHE, BUFFERS and CACHE for HPFS to get the best combination or best use of system memory.

2. The number of request buffers (**numreqbuf**) allocated should be adequate so that the server occasionally uses all the big buffer resource during peak workload times. Configure the **numreqbuf** parameter to 1.5 or 2 times of maxusers.

3. The number of big buffers (**numbigbuf**) allocated should be adequate so that the server occasionally uses all the big buffer resource during peak workload times.

4. The size of the request buffers (**sizreqbuf**) can be reduced from the 4KB default value to 2KB without causing a significant performance degradation in some environments. Compare the network performance for both settings before making the final choice. Refer to Chapter 6, "Further Analysis of Tuning" on page 135 for more details.

Note

Setting the **sizreqbuf** parameter to 2KB can affect the number of big buffers needed since a request to read or write with a record size greater than 2KB, instead of 4KB, will now attempt to use big buffers. It is strongly recommended that the **sizreqbuf** parameter value on the server matches the **sizworkbuf** parameter value on the requester.

3.2 Design Overview of LAN Server - Advanced

LAN Server Advanced consists of an SMB processor component and an HPFS file system component. These two components are tightly coupled and work as a one system called HPFS386. HPFS386 runs at the system privilege level, Ring 0, so minimal OS/2 services is required to satisfy network file I/O requests. HPFS386 supports DMA data transfers. Performance is enhanced by the use of busmaster disk and network adapters, which use DMA functions. HPFS386 uses read-ahead and write-behind logic, allowing network file I/O to achieve a high rate close to the network bandwidth, when data is fully cached. It also has an efficient cache system. You might feel the naming (HPFS**386**) is legacy, but in fact this means the system works in Ring 0 and utilizes 32-bit instructions. So, HPFS386 works on any 386/486 or Pentium machines. HPFS386 of LAN Server 4.0 even utilizes some of the Pentium features.

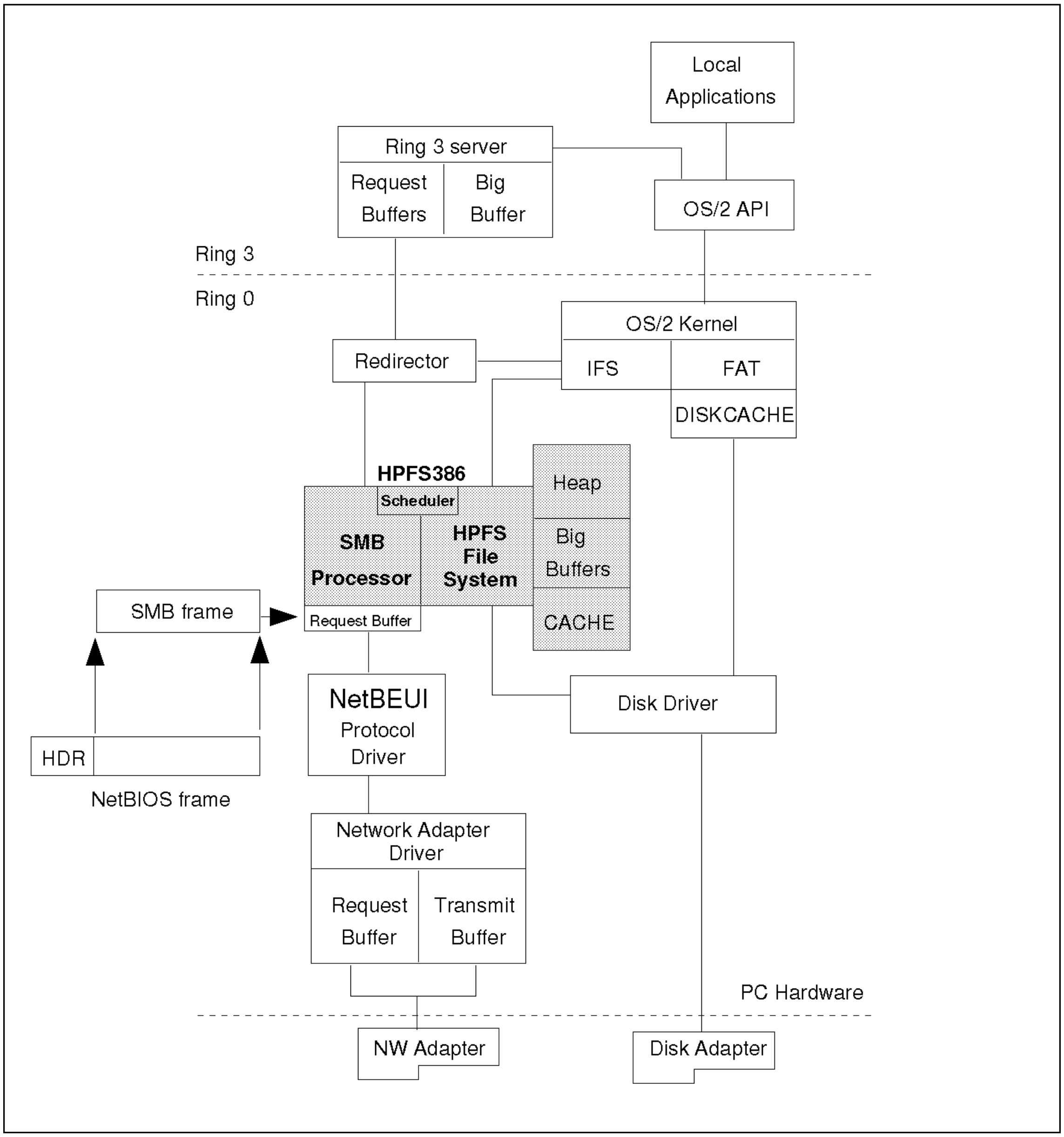

Figure 12. LAN Server Advanced - Components

File I/O performance is enhanced over the Entry Server because of less ring transition due to the direct entry to HPFS386 and a large cache capacity. SMB requests for HPFS file I/O are passed to the SMB processor by NetBEUI. Requests destined for non-HPFS resources such as the FAT file system, character devices, and named pipes are passed by the SMB processor to Ring 3 server, which is included with LAN Server Advanced. The non-HPFS requests are satisfied through OS/2 APIs.

The HPFS386 file system may send any of the SMB protocol types (RAW, SMB and MPX), refer to 3.7, "Logon Performance Tuning" on page 83. In addition, HPFS386 uses *scatter/gather logic* to enhance performance if it is supported by the network and disk device drivers. Scatter/gather is a technique to send the data which is scattered to several memory areas, without issuing a separate I/O or combining spread data frames into a single large area. If an adapter supports this interface along with DMA capability, it is highly efficient.

LAN Server 4.0 has a similar architecture to LAN Server 3.0, plus several new features such as the capacity limit enhancements.

Sideband

Sideband is a set of enhancements designed to reduce the path length for small random reads and writes to files on LAN Server 3.0/4.0. DOS LAN Requester 3.0, DOS LAN Services and OS/2 LAN Requester support this interface to the Advanced server. The latest release of NTS/2 (Network Transport Services) or MPTS (Multiple Protocol Transport Services) is required.

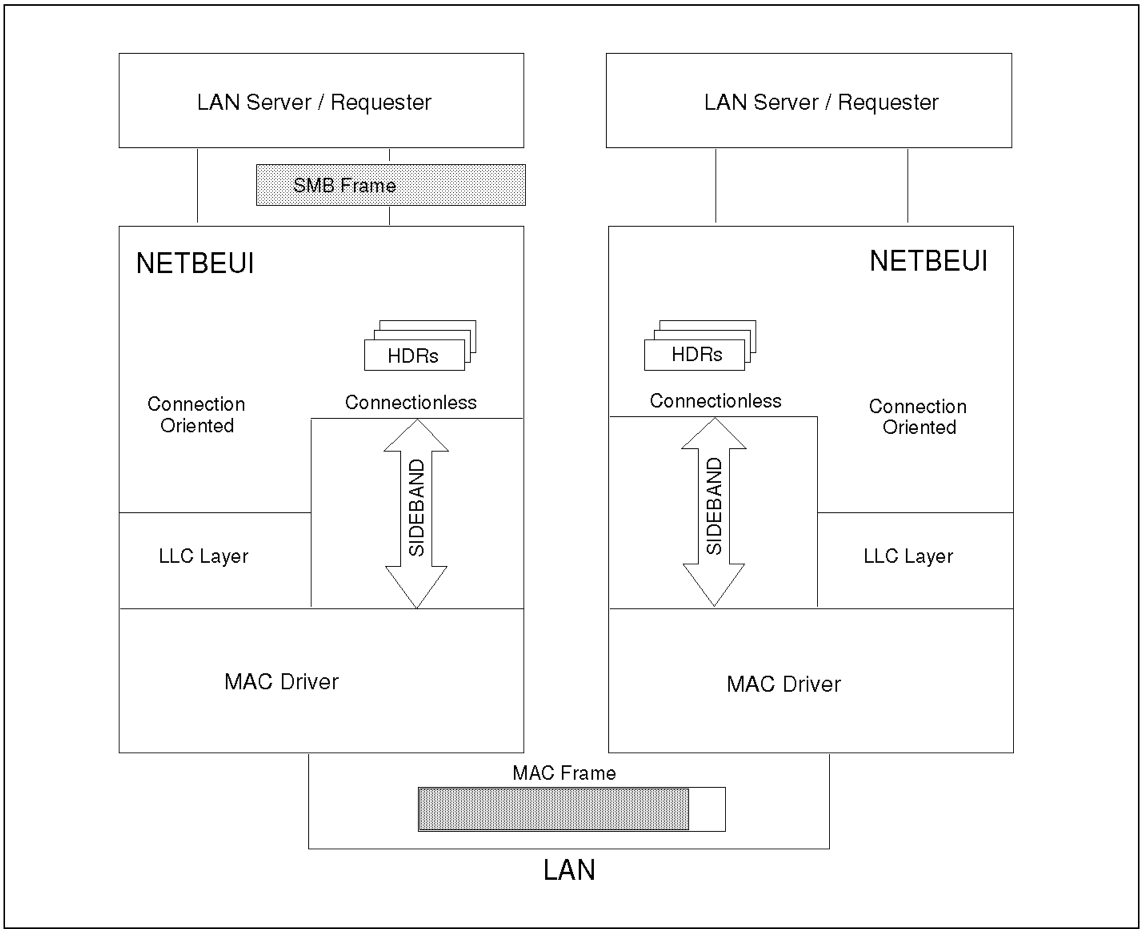

Figure 13. Sideband Mechanism

The Sideband code will use a combination of connectionless communication (See 4.1, "Network Transport Considerations" on page 103) and the reuse of duplicate header information to improve the performance for the transmission of small network packets. If the Sideband code is active on a session, it determines whether the data to be transmitted will be sent over the Sideband path or not. If too many transmission error occur, the sideband will be disabled automatically and can only be enabled by ending and reestablishing a session. At times this error may be logged in the error log file which is viewable with NET ERROR command.

The primary benefit of Sideband is the reduction of CPU usage in a server by eliminating the some processing codes in NetBIOS. The key for Sideband is to use a large MAC (Media Access Control) frame size. Of course, the

Sideband assumes LAN is reliable, so you must keep your LAN as reliable as you can.

Design Changes after LAN Server Version 3.0

Several LAN Server 3.0 components have been changed for the LS Ultimedia Version 1.0 product. The next figure shows the LS Ultimedia server architecture. Shaded components are the changes or additions done by the LAN Server Ultimedia 1.0.

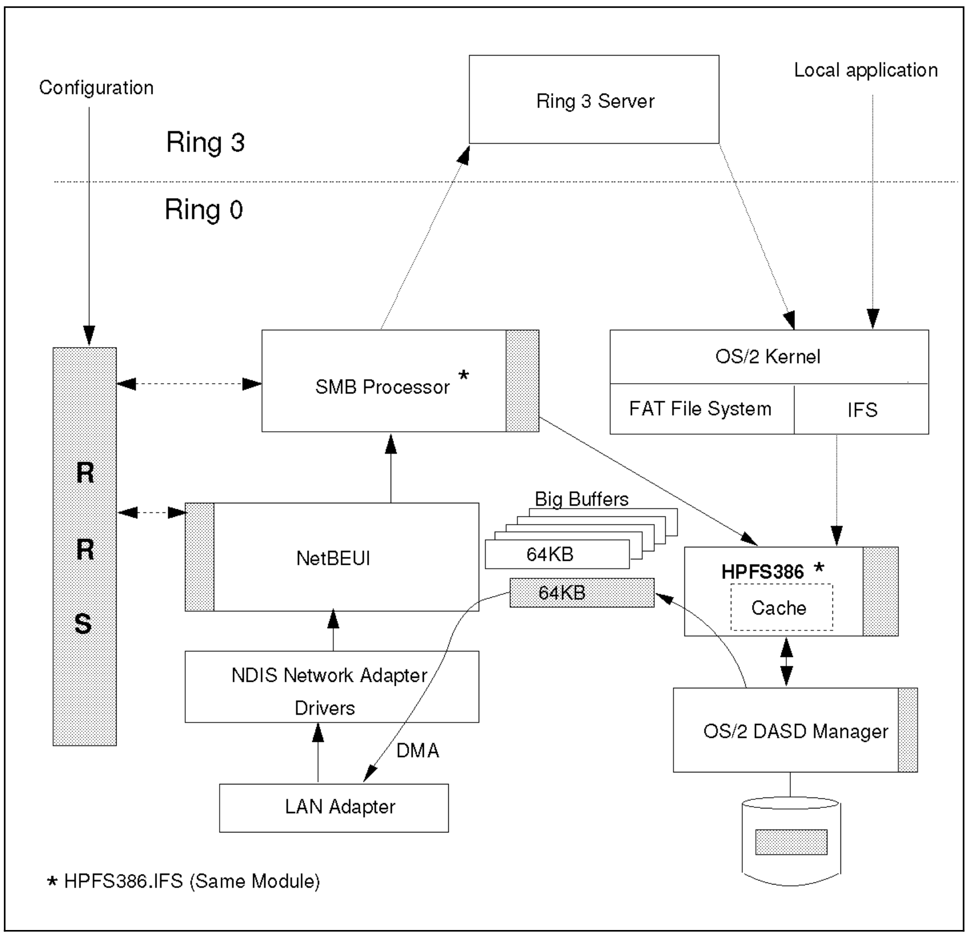

Figure 14. LAN Server Ultimedia Architecture

There are several changes done for the Ultimedia request processing. The first change was to enable the preallocation of big buffers.

Big Buffer Allocation Logic - LS Ultimedia

In the original LAN Server 3.0 Advanced, big buffers (64K) were not preallocated. Initially, when a server is started, two big buffers are allocated. If there is no big buffer available when needed, HPFS386 will allocate an extra one and keep it in the pool after the read request is processed. They are not returned to the system. The original HPFS386 will allocate up to 16 buffers. This was the original design.

After installing LS Ultimedia Version 1.0 or LAN Server 3.01, HPFS386 is able to use its big buffers for multimedia file transfer. Since a multimedia playback typically involves a number of big reads on the same file, HPFS386 allocates one of its big buffers to the multimedia file as soon as it is opened. The buffer will be used only by that file until the file is closed and the buffer is returned to the pool.

If there are no buffers in the pool, HPFS386 will allocate one from the system. Up to 64 buffers are allocated (as opposed to 16 for LAN Server 3.0). These buffers are returned to HPFS′s pool when the multimedia file is closed. They are not returned to the system.

However, the longer the system is running without rebooting, the more memory fragmentation will occur, and the harder it will be for HPFS386 to allocate a big buffer when it needs one. To ensure availability of big buffers, the new HPFS386 allows you to specify the number of buffers that are to be preallocated. This will cause the buffers to be allocated at system startup, when HPFS386.IFS is loaded, or when the server is started by NET START . There are new switches for the HPFS386.IFS statement in CONFIG.SYS:

/FSPREALLOC:*n* Specifies that **n** 64KB buffers are to be preallocated by HPFS386 when the system starts. Requests for physical RAM buffers are most likely to succeed when done during system start. These buffers are never freed.

/SRVPREALLOC:*n* Specifies that **n** 64KB buffers are to be preallocated by HPFS386 when the server starts, in other words, when NET START SERVER is run. Requests for physical RAM buffers have a better chance of succeeding during server start than when made after the server is started. These buffers are freed when the server is stopped.

In addition to the big buffer preallocation mechanism, HPFS386 now provides a priority file access to the Ultimedia requests. It provides a separate queue for the Ultimedia clients. RRS (Resource Reservation System) controls the bandwidth reservation within the system, disk and LAN I/O. When the Ultimedia file is open, HPFS386 calls RRS to reserve the required throughput rate (KB/sec) for the session. HPFS386 will allocate one big buffer to the session and it will proceed a 63.5KB read operation. HPFS386 cache is not used for multimedia files. Since multimedia files are read sequentially using big read requests, use of the cache will degrade performance rather than improve it. A 64KB buffer is dedicated to the multimedia file as soon as it is opened. The I/O requests from that session will be treated as a higher priority than the regular file I/O. NetBEUI is also enhanced to provide a priority NetBIOS interface.

Big Buffer Allocation Logic - LAN Server 4.0

The design is further enhanced by LAN Server 4.0. The **fsprealloc** and **srvprealloc** switches are moved to a new file called HPFS386.INI. HPFS386.INI is located under IBM386FS directory. The following is an example of HPFS386.INI file:

```
;  [filesystem]        ; General file system parameters
;  [lazywriter]        ; Lazy writer parameters
;  [DASD_Limits]       ; DASD Limits parameters
;  [UltiMedia]         ; UltiMedia parameters (added when LAN Server
;                      ;  Ultimedia 1.01 is installed)
;
[filesystem]
useallmem  = YES
lanroot    = C:\IBMLAN
fsprealloc = 40
; srvprealloc = nn

[lazywriter]
lazy       = *: ON
maxage     = *: 5000
bufferidle = *: 500

[DASD_Limits]
ThreshAlertNames  = *: ADMINS
ThreshAlertDelay  = *: 10
ThreshAlertUser   = *: yes
DirFullAlertNames = *: ADMINS
DirFullAlertDelay = *: 10
DirFullAlertUser  = *: yes

[UltiMedia]
buffers    = 4,1
fsprealloc = 40
; srvprealloc = nn
```

Figure 15. HPFS386.INI Example

LAN Server 4.0 HPFS386 can read-ahead 256KB if `buffers=4,k` is specified (where k=1,2,3,4). If `buffers=1,k` is specified, it will operate just like LAN Server 3.01 or LS Ultimedia 1.0. 64KB read-ahead buffers for Ultimedia clients are allocated from the system memory area. This is a separate area from the file system's big buffers. Both the regular file system purpose and Ultimedia purpose, the same keyword `fsprealloc` for example is used, but they are separate areas.

HPFS386 Code Reorganization

In the original LAN Server 3.0 the code for the 386 Advanced server was delivered in two files: HPFS386.IFS and HPFS200.386. The HPFS386.IFS was merely used by OS/2 as a loader for HPFS200.386 which provided all functionality. With the LS Ultimedia product, these two files have been combined into HPFS386.IFS, and converted to a real 32-bit code. This improves performance and decreases the code size. The HPFS200.386 file is still there, but it is a dummy file which is not used anymore. Also, the file HPFS386.DLL has been added. It provides the interface between HPFS386, the RRS configuration program, the disk calibration and the MMUTIL and PROFILER utilities.

In the LAN Server 3.01 manufacturing refresh version, the HPFS386 code is the same as the LS Ultimedia version. LAN Server Version 4.0 has the same structure but with the enhancements described above, and a new directory limit function.

3.3 Inside OS/2 File Systems

A file system is a component of an operating system which manages the creation, deletion and retrieval of permanent data from a physical device. For OS/2, it may also manage network requests, named pipes, printing and other types of resources. It has the ability to support the coexistence of multiple, active file systems on a single workstation.

The choice of the most appropriate file system is important for the performance of a LAN Server. There are several types of file systems, such as the file allocation table (FAT) or installable file systems (IFS), such as HPFS, and HPFS386. The HPFS file system provides a much better performance than the FAT, because it is designed to provide extremely fast access on very large disk volumes.

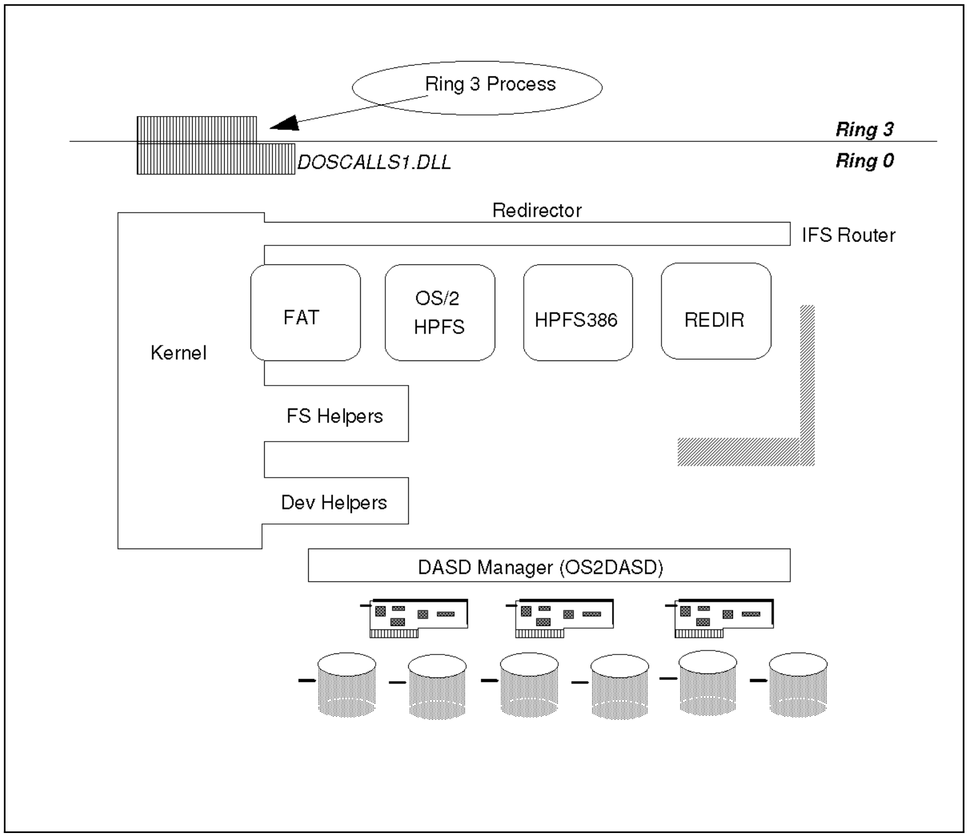

Figure 16. File Systems - Multiple File System Coexistence

The following explains installable file system (IFS):

- An IFS is a file system which is not imbedded into the kernel.
- OS/2 HPFS and HPFS386 are examples of File System Drivers (FSDs) which are IFS.
- FAT is a file system, but not an IFS - it exists as part of the kernel.
- Redirector is an IFS which redirects file system requests at a server.

File System Process

During normal operation, the file system drivers (FSDs) are loaded in the CONFIG.SYS, using an IFS statement. They are loaded in order of their appearance in the CONFIG.SYS file. The FSDs act like DLLs in that they provide entry points which the kernel can call.

An FSD is first called at IPL time at the FS_INIT entry point, and initialization takes place at a Ring 3 protection level. At this point, the appropriate file system will either call the kernel to do I/O or will create a request list and call the disk device driver directly.

Usually the applications do not know the location of a file and their method of storage, so they use APIs to locate the data. These APIs can either be path-based or handle-based.

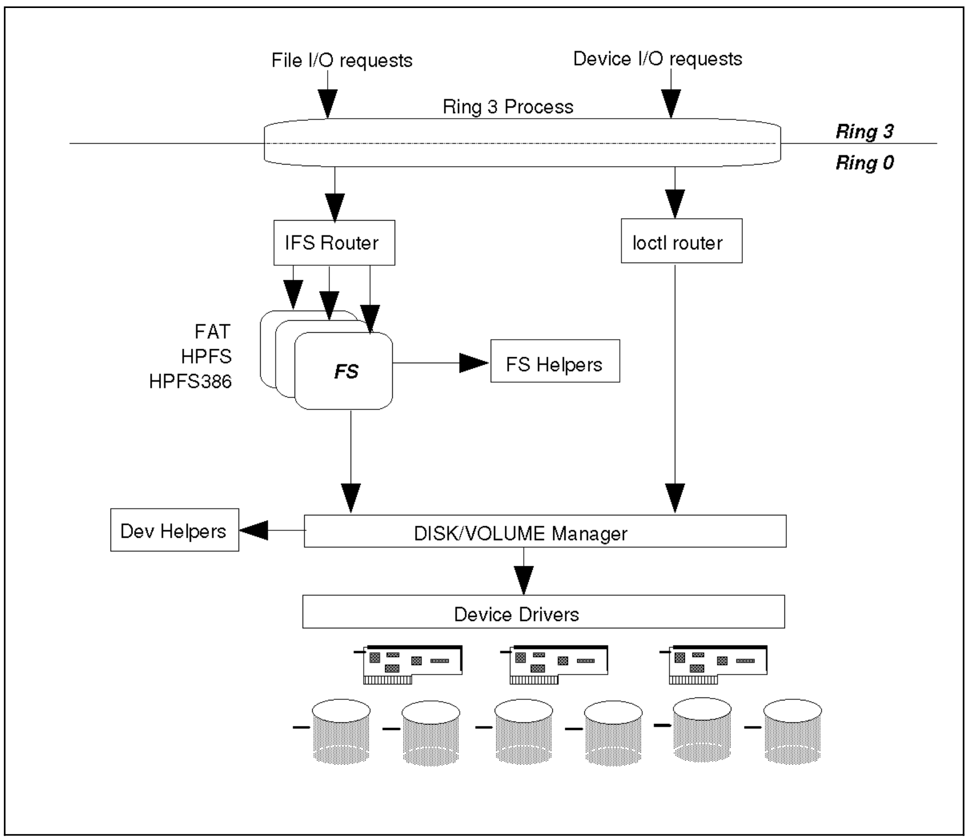

Figure 17. Operation of a File System

On a file I/O request, the IFS router decides which file system should process the request for data. When ascertained, the chosen file system provides the appropriate system services for the request with the assistance of a corresponding FSHelp module. The Device/Volume Manager then maps the physical devices to the device drivers and the file system, and the local router allows access to the devices without the use of the file system. For example, when using modems.

HPFS386 Data Path

For the Advanced server data transfer, a different data path is used based on its file type (FAT or HPFS) and user type (local or remote).

When a remote user accessing data on FAT, the Ring 3 SMB processor is used, and ring transitions between Ring 0 and Ring 3 will be experienced. For HPFS partitions all the data is manipulated within the Ring 0 privilege level and no transitions to the Ring 3 level is required. However, local application on the Advanced server must go through OS/2 kernel and IFS interface to access the HPFS partition, and it causes ring transitions.

The following diagram gives a comprehensive overview into the various data paths that can be taken for each type of file system.

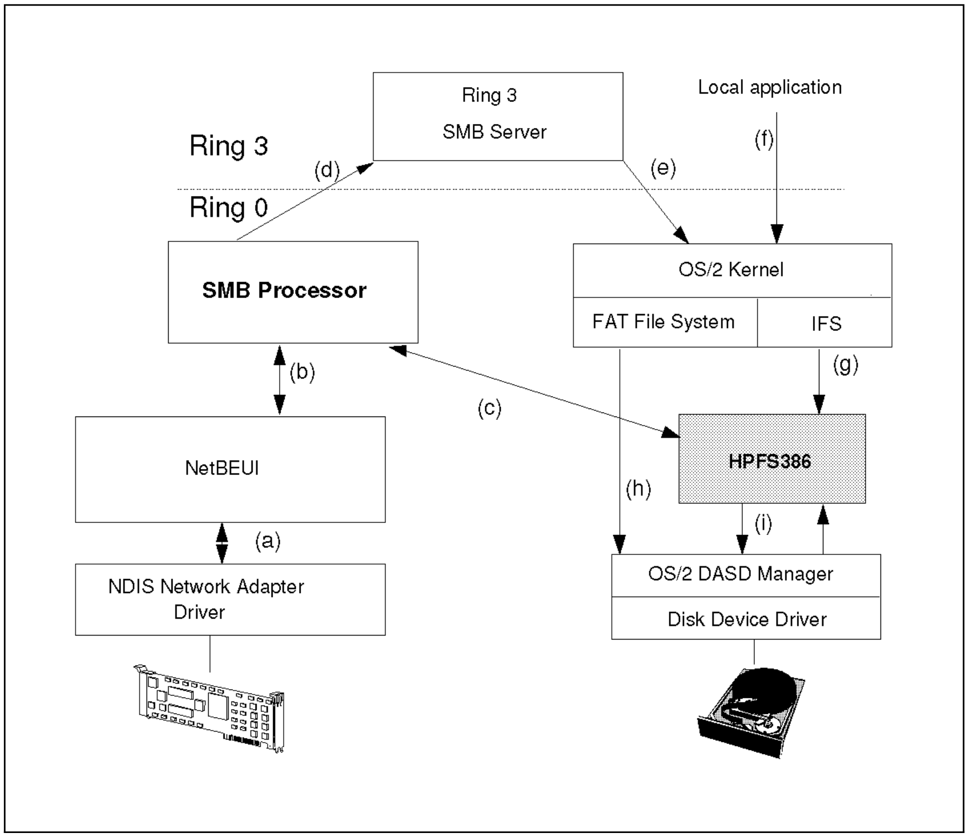

Figure 18. File System Data Transfer

Examples of paths traced through the system are given below:

- Local access to HPFS file in HPFS386 cache: f - g
- Local access to HPFS file, not in cache: f - g - i
- Local access to FAT file: f - h
- Network access to HPFS file, in HPFS386 cache: a - b - c
- Network access to HPFS file, not in HPFS386 cache: a - b -c -i
- Network access to FAT file: a - b - d- e - h

3.4 Inside HPFS386

The 386 High Performance File System (HPFS386) is highly optimized and designed for 80386SX** and above based platforms with large disk systems. It optimizes the performance in the server environment, in which many files are open simultaneously. The HPFS386 is an enhancement of the regular HPFS and represents the logical evolution of LAN Server technology. The server consists of an optimized Ring 0 SMB processor tightly coupled with a bootable installable file system (IFS). Following figure shows how HPFS863 interfaces to the other system components. As you see here, HPFS386 is not only a fast file system, but also it is an SMB processor. Since incoming SMBs are processed within the Ring 0 module and routed to a file system, HPFS386 gives the faster file I/O services to remote users.

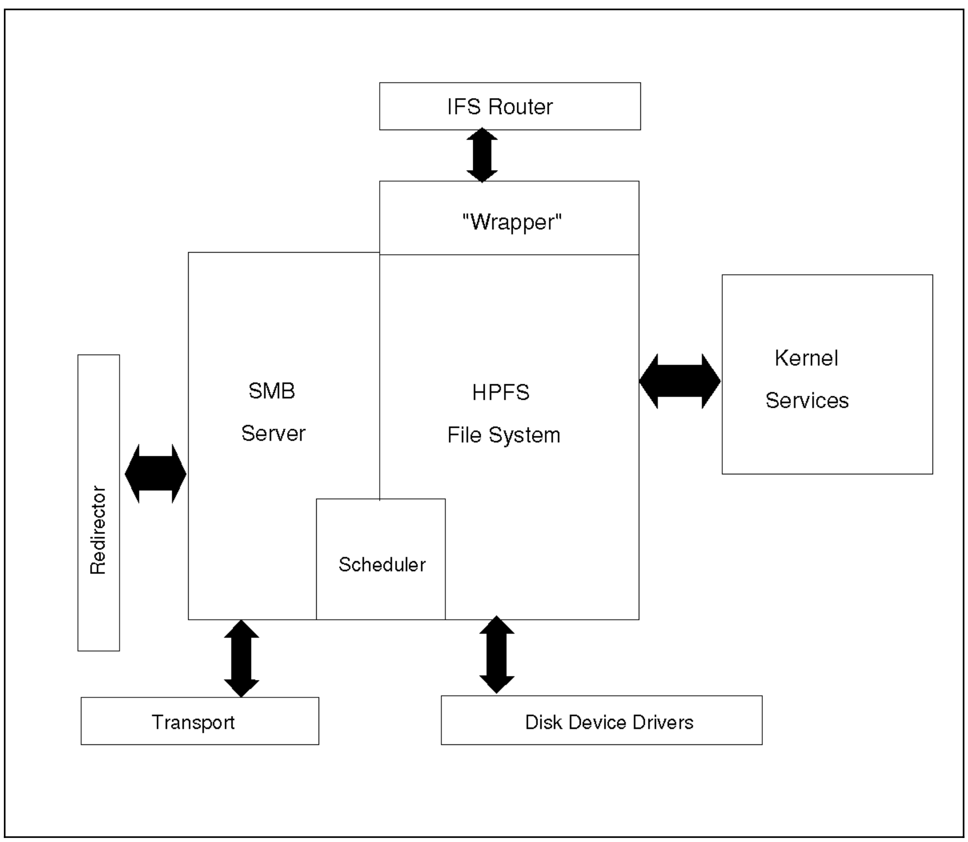

Figure 19. HPFS386 Component Layout

In comparison to the OS/2 HPFS file system, which is written in C Code, the HPFS386 file system is written in 32:32 assembler code using 386 instructions. It uses 16:16 addressing to access external components. With assembler code, it is not easy to optimize the code for the Pentium super-scalar feature.

HPFS386 Cache Memory

HPFS386 file system makes extensive use of caches which are separate from the caches used by Entry server or OS/2 itself. The default cache value for the LAN Server 3.0 Advanced is 20% of the available memory. However, LAN Server 4.0 sets the default cache value to 20% of the available memory if the server has less than 20MB of memory, and 60% of the available memory if the server has 20MB or more. *Available memory* means the total system memory minus the memory spaces which are already used by device drivers and OS/2, before HPFS386 is started. For cache to be allocated above 16MB then the **useallmem** parameter must be set in the IFS statement in CONFIG.SYS for LAN Server 3.0 or HPFS386.INI for LAN Server 4.0. The parameter **useallmem** tells HPFS386 that LAN adapter and disk adapter can support the memory area above 16MB. Actually the LAN Server system will not use all the memory spaces. Cache usage is limited to 60% of the available memory as a default and big buffers apace and request buffers space are also defined by the parameters.

Increasing the cache size can lead to significant performance improvements on the server system and improve performance of applications with high disk I/O requirements.

Lazy-Write

To minimize the frequency with which the system ties up resources writing cached data to disk, both FAT and HPFS (standard and HPFS386) file systems take advantage of the lazy write feature. This can provide significant performance improvement when writing to disk. Lazy-write (or write-behind) defers writing data to the disk until the operating system is idle or when the update is a maximum of 5 seconds old. This allows control to be returned to the application without having to wait for completion of I/O operations.

Read-Ahead

Read-ahead can increase disk I/O performance. This is achieved by extending the read operation (read-ahead) and placing that data in the file system buffer or cache. This is an internal function to the file system and is not configurable by the end user.

Memory Usage Contention

The major users of the system memory are the following:

- Operating system
- HPFS386 cache
- Request buffers
- Big buffers

The more memory that is defined for one component, the less memory is available for the other remaining components. For example, if you define too much memory for the cache, there may not be enough remaining memory for the operating system and big buffers.

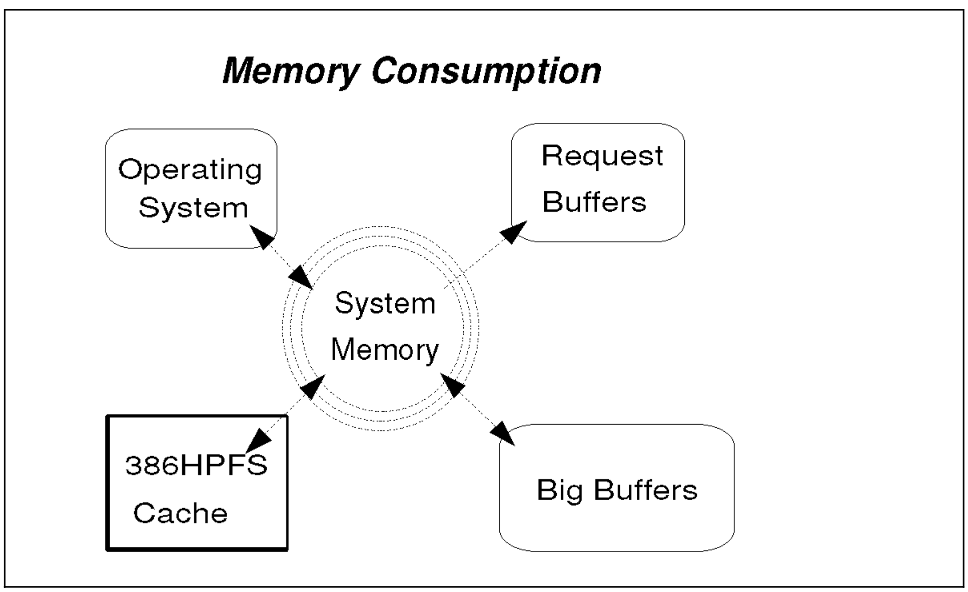

Figure 20. Memory Consumption

The following is a basic requirement:

- 12MB of memory in the server for OS/2 and LAN Server to perform basic server functions.
- Between 0.5-1M of free memory for efficient handling of peak workloads.
- About 3MB to accommodate for the new GUI interface introduced by LAN Server 4.0.

  For this reason, it is best to use GUI from the requester and allow this memory to be used as cache.

In addition to these, you should consider the size of request buffers area, big buffers area and CACHE386 memory spaces. 3.6, “Capacity Tuning Parameters (LAN Server 4.0)” on page 71 and 3.5, “LAN Server Buffer Tuning Summary” on page 68 describe the detail.

There is an overhead for addressing and reservation of the cache memory. For example, a server that has 32MB of memory, there is an overhead of 1.2MB.

Advanced Server Cache Tuning

The ultimate benefit for using LAN Server Advanced is to take advantage of the sophisticated HPFS386 file system and 386 SMB processor. This completely by-passes the OS/2 caching and uses a more intelligent LAN Server caching algorithm which enables it to provide levels of self-tuning.

The 386 SMB (ring 0) server is an optimized network server designed for 386 and higher workstations with large disk sub-systems. It is used primarily on systems that require intensive data / application sharing.

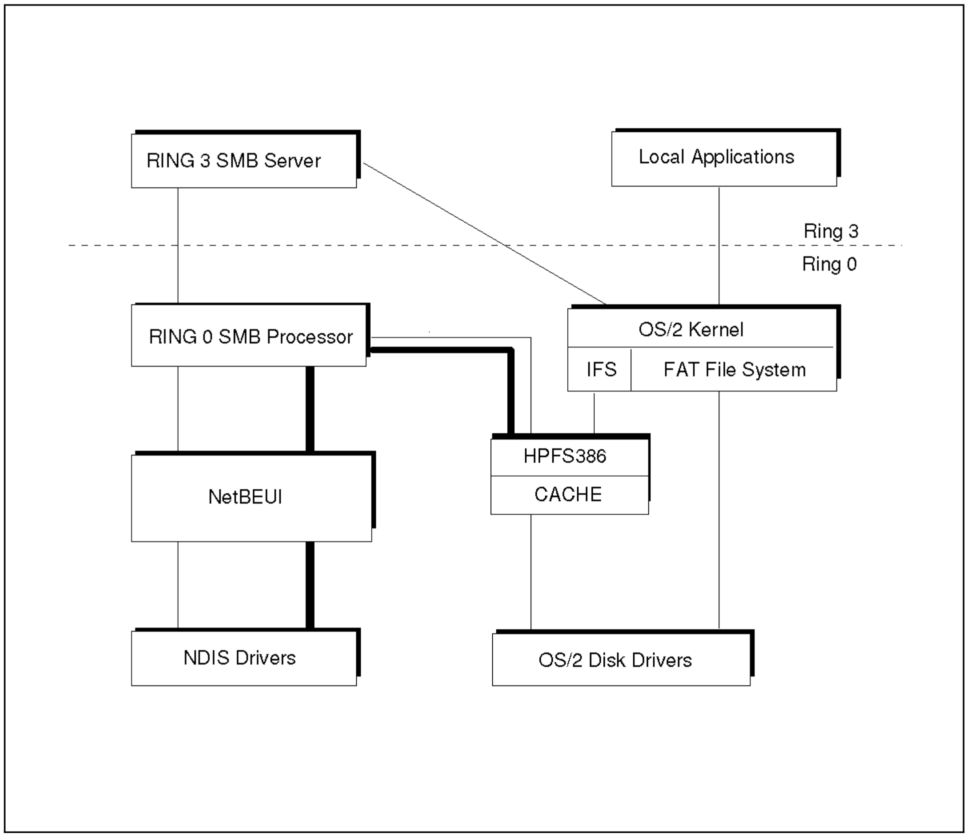

Figure 21. HPFS386 Caching

The HPFS386 allows any incoming data from the LAN to go directly into the HPFS386 cache. The data flow can avoid the longer way through the Ring 3, where additional operations have to be made, such as ring transitions. This architecture provides a much better file I/O performance, because the whole data flow operates within the Ring 0 privilege level.

Although the Advanced server performs well with HPFS386 partitions it can also be used, to access FAT partitions. To do this it uses the Ring 3 SMB processor, which continues to be available even if you have the Advanced server installed.

Assigning FAT cache, if required, is similar to the procedure laid out for the Entry server, however in this situation as much memory as possible should be dedicated to HPFS386. If FAT is not installed, or access is minimal, it is recommended that the minimum DISKCACHE parameter is used.

The following parameters are used for defining and monitoring HPFS386.

HPFS386 IFS Statement (LAN Server 3.0)

This parameter defines the HPFS386 installable file system. If the cache option is not specified in the IFS statement then by default 20% of the available memory is assigned. The minimum cache size that can be specified for HPFS386 cache is 256KB , but implementing this will provide minimal performance improvements during disk I/O operations. It is generally recommended that a cache of at least 512KB be assigned. The maximum allowable cache is determined by the amount of physical memory available within the system after OS/2 and LAN Server are loaded. It is advisable to allocate as much cache as possible to obtain the best server performance.

If for example you are using a 16MB dedicated server try allocating a cache size of 7MB, or if you have a 32MB server try using as much as 20MB of cache on the machine. The current version of the spreadsheet tool, LAN Server Tuning Assistant, can calculate a recommended value for you.

As HPFS386 is a more sophisticated file system there is no need to specify any threshold limits as this is dynamically managed by LAN Server.

The heap option is required for HPFS386 to store it′s internal data structures such as file, find and search handles. If the heap is omitted, the size of the heap is limited only by the size of the physical memory available. The minimum value is 16KB and the maximum value is determined by the amount of physical memory available. If the heap is specified, then only 25% of the heap is initially allocated and any further memory is assigned as required, up to the size specified.

By using the cache monitoring facility provided with LAN Server the cache performance can be finely tuned to maximize the cache read hit ratios with the minimum sized cache. This is essential within a memory constrained environment.

For LAN Server Advanced the CACHE386 utility can help you analyze server performance. The parameter /STATS displays the statistic concerning cache usage. To achieve the maximum benefit of the cache use the cache hit ratio to maximize the hit ratio with a minimum sized cache. The following is an example of CACHE386 output:

```
CACHE386 Statistics

Read Requests:              7937022    Disk Reads:        1025032
Cache Hit Rate (Reads):         87%    Cache Reads:       6911990

Write Requests:             1028634    Disk Writes:        216009
Cache Hit Rate (Writes):        79%    Lazy Writes:        812625

Hot Fixes:           0
```

Figure 22. CACHE386 Output Example

HPFS386 Cache Allocation above 16MB

HPFS386 cache is divided into two objects:

- A LoCacheObject which address below the 16MB limit
- A HiCacheObject which address above the 16MB limit

To implement the HiCacheObject the USEALLMEM parameter must be set to YES in the IFS statement in CONFIG.SYS for LAN Server 3.0 or 386HPFS.INI for LAN Server 4.0.

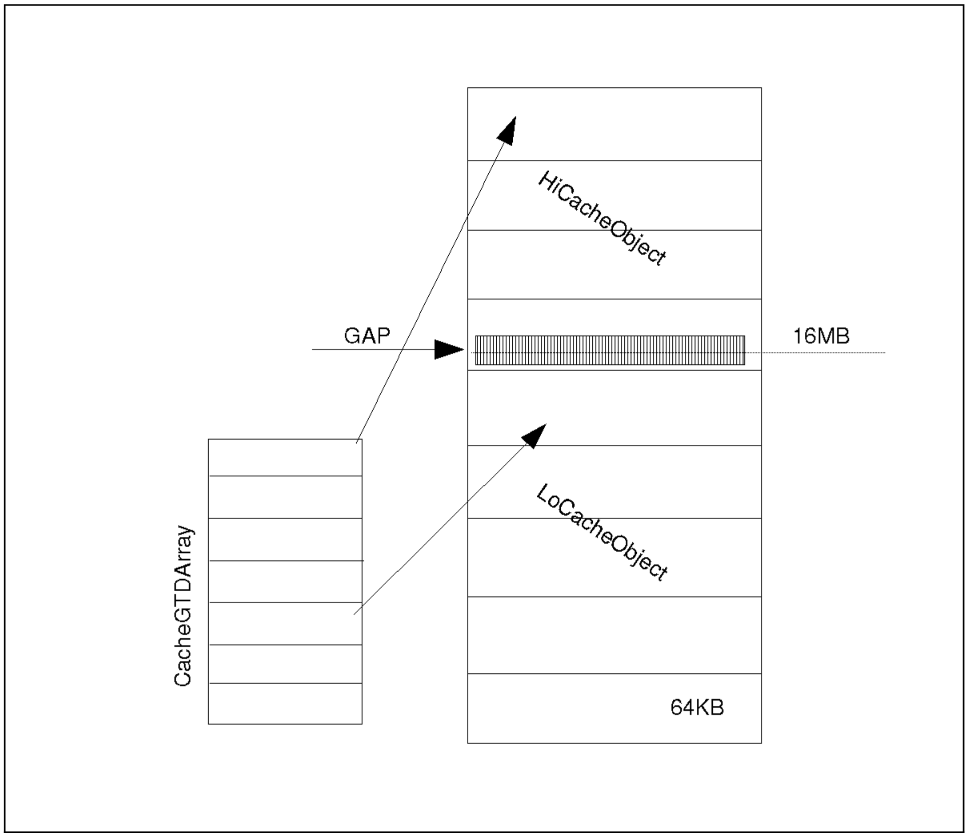

Figure 23. HPFS386 Allocation

When assigning HPFS386 cache there is a specific consideration that should be taken into account. Due to the HPFS386 structure, there is a memory area near the 16MB address boundary which it is not possible to allocate contiguously. This gap ensures that there should not be one HPFS386 cache object spanning across the 16MB region.

If for example you have a machine running with 32MB of RAM, then you should not define a cache size that will cross the 16MB boundary, that is greater than 15MB. There is a possibility that errors can be generated as a result. If the **useallmem** parameter is used, it is important you have hardware components within your machine that have the ability to address memory above 16MB. The IBM SCSI adapter and LANStreamer adapter have this capability. IBM 16/4 token ring adapter is a shared RAM adapter and it doesn′t have this addressing capability. However, NetBEUI moves the data on behalf of the adapter.

Heap Allocation

The heap space is used for dynamic memory allocation and caters to small objects and file handling. It contains various objects to satisfy a quick memory allocation. The heap size can be left at the default value. This will allow the heap to grow as the system runs, limited only by the available physical memory in the system.

On a very busy server system, it is possible to run out of heap space. If this occurs there may be a noticeable degradation in performance, and service requests may fail. Errors returned to the requester that specify the server is out of resource, especially on DOS searches, may indicate that the server has run out of heap space. If this happens you will need to freeup some memory on the server.

Initially the heap is set to 25% of its specified size and it starts growing when many files are open on the server. When these files are closed the heap gives the memory back to the heap manager and never frees the memory back to the system. From the servers point of view, this means the heap can only grow to its defined size and never shrink unless the machine is rebooted.

3.5 LAN Server Buffer Tuning Summary

LAN Server uses two types of buffers to handle read/write requests on the server systems for LAN Requesters - request buffers and big buffers.

Request Buffers

The request buffers are the most frequently used buffers in LAN Server and account for all the read/write requests that are equal to or smaller than the size defined by the parameter **sizreqbuf**. They are used by the server to receive SMBs from the network and to hold SMB data being transferred to/from the network adapter card buffers. The request buffers are configured by assigning appropriate values to **numreqbuf** and **sizreqbuf**. Ring 3 server will allocate the request buffers from the system memory and they are shared by Ring 0 server.

- **numreqbuf**

  This parameter specifies the number of buffers, with size determined by **sizreqbuf**, the server uses to take concurrent requests from requesters.

  For sequential file access, that is an application reading data, a server will go into read-ahead mode, transmitting 4KB buffers of data to a requester while at the same time reading ahead and filling another 4KB buffer ready for the next transmission. For this reason there should be at least two request buffers per concurrent requester. Since the probability of all requesters concurrently sending requests is relatively low, then the **numreqbuf** parameter is usually set to twice the **maxusers** parameter. Migrate to the LAN Server 4.0 if you have many concurrent users logging on to the server. LAN Server 4.0 has a new design to enable 2000 request buffers, depending on system memory size. Note that you need 32-bit DMA capable adapters for disk and LAN interface and need to specify **useallmem** in order to allocate such a large number of request buffers. However, as the buffers are pooled among all requesters, then the figure of two buffers per requester could be reduced as the number of requesters increase.

- **sizreqbuf**

  This parameter sets the size, in bytes, of the request buffers the server uses to take requests from requesters.

  This parameter should be set to the same value for every server on the network. It should also correspond to the value of the **sizworkbuf** parameter on each requester. By default this is set to 4096 bytes, but there are situations in which this value is not the optimum value for the server and requester buffers.

  If for example most of the frames transmitted on the network are less than the 4096 byes, for example 1980 bytes, then the server and request buffers could be reduced accordingly. This would provide the flexibility of either assigning a greater number of smaller request buffers or just using the extra memory for application resources.

A server should have sufficient request buffers available to handle the peak request workload. You can check on the level of usage of the request buffers allocated by using the NET STATISTICS command on the server system. This will indicate how often all of the request buffers are allocated. If they are never exhausted then reduce the number of buffers and continue to monitor. Repeat this procedure until the server uses all the request buffers only during peak workload.

Care should be taken to match the server and requester buffers sizes. Failure to do so will just waste memory. Smaller sizes will be used for the SMB data size during the session negotiation time.

Big Buffers

Big buffers are used for large sequential data transfers such as program loads and large files transfers and printing. Ring 3 server will allocate big buffers for FAT and OS/2 HPFS usage. HPFS386 allocates separate big buffers. Big buffers have a fixed size of 64KB. As each big buffer takes up 64KB of system memory it is important to prepare enough memory space for big buffers.

Big buffers are used in association with the RAW SMB protocol and for read-ahead. Read-ahead will be done in most cases depending on the srvheuristics parameter. If the requester has a limited size of buffers, the multiplexed SMB format will be used instead. Refer to 3.7, "Logon Performance Tuning" on page 83 for details.

To set up the maximum number of 64KB buffers, the **numbigbuf** parameter is used for Entry server or Ring 3 server function of Advanced server. Advanced server′s big buffers for HPFS386 is defined with **fsprealloc** or **srvprealloc** parameters on IFS statement or in the HPFS386.INI file. Refer to Figure 15 on page 54 for an example of HPFS386.INI file.

Until LAN Server 3.0, three big buffers are allocated at initialization time and any additional big buffers are allocated on demand up to the maximum value of **numbigbuf** parameter in IBMLAN.INI (Entry server), or up to 16 big buffers (Advanced server).

For the Entry server, the dynamic allocation of big buffers can be controlled using the **srvheuristics** parameter, positions 17 and 18. Digit position 17 of the **srvheuristics** parameter specifies the amount of time that dynamically allocated big buffers stay in memory. Digit position 18 of the **srvheuristics** parameter specifies the amount of time that the server waits after failing to allocate additional big buffers before trying again.

For LAN Server 4.0 Advanced, big buffer related parameters are controlled by the HPFS386.INI file.

3.6 Capacity Tuning Parameters (LAN Server 4.0)

In addition to server performance tuning, server capacity tuning is also required. LAN Server 3.0 is set by default at installation to support an average of 20 to 30 users. If you want to make a large domain such as supporting over 100 requesters, you need to change some parameters.

LAN Server 4.0 Advanced is set by default to support 100 users. It is difficult to say what suitable number of requesters that LAN Server can provide good performance. It depends on many factors such as the system resources, the applications characteristics, the server′s network adapter card, the requester′s speed and so on. Some of these parameters may affect the performance, since a lack of resources forces a user to wait for a resource to become available. The users need to understand the meaning of parameters and related parameters to set servers efficiently.

However, there is a useful spread sheet for Lotus 1-2-3** and Microsoft Excel**. It is called CNFGLS30 and it automatically determines suitable values for a specified environment. It is a tool for LAN Server 3.0 but LAN Server 4.0 includes the equivalent tool as a PM application called LAN Server 4.0 Tuning Assistant.

LAN Server 4.0 Capacity Improvements

LAN Server 4.0 has a very important change called capacity improvements. This is a capability to support large number of requesters, such as 1000 users. Actually, 1000 users cannot be connected to LAN Server 3.0. In addition to this change, LAN Server 4.0 Advanced is set by default to support an average LAN with around 100 users. In fact, the structure of the **numreqbuf** related control block is redesigned to support this large number of users.

LAN Server 4.0 Tuning Assistant

A very visible change in LAN Server 4.0 is the addition of the PM utility program, which assists in fine tuning the resources of the LAN Server 4.0 system to match the users′ configuration requirements. This utility is, with minor modifications, a stand-alone version of the configuration spreadsheet tool, CNFGLS30, which has been in use by LAN Server administrators for a long time.

It goes beyond the spreadsheet tool by actually updating the servers′ configuration files such as CONFIG.SYS, PROTOCOL.INI, HPFS386.INI, and IBMLAN.INI automatically. User will find it in the LAN Service folder. It can also be executed at any time after the install process has been completed. It is recommended that this utility be used in order to achieve the good performance with LAN Server 4.0.

Note

This tool automatically calculates the number for **numreqbuf** as twice the value specified for the number of users. You can increase or decrease the value if you want.

Request Buffer Design Change in LAN Server 4.0

In LAN Server 3.0, the internal control block that keeps pointers (this pointer is called NB in the Figure 24 on page 73) to each request buffer is designed as 16-bit code that uses segment addressing. Its 64KB space limit affects the number of request buffers. By this 64KB restriction and the size of each NB (about 100 bytes per NB structure), the maximum value of **numreqbuf** is limited to around 350 and the maximum concurrent users is below 300, because the server needs more than one request buffer for each user.

With LAN Server 4.0, the design of NB structures has been changed from the 64KB segment to a linear addressing mode, 32-bit memory object, so that the number of NB structures will not be restricted to 64KB. In this design, all 16-bit (unsigned short) offsets which are used in LAN Server 3.0 to address NB structures are changed to support a 32-bit (either 16:16 or global linear address) addressing mode. Other data structures related to the NB structures have changed accordingly. Now the maximum values for **numreqbuf** is increased in proportion to the increase in the number of NB structures.

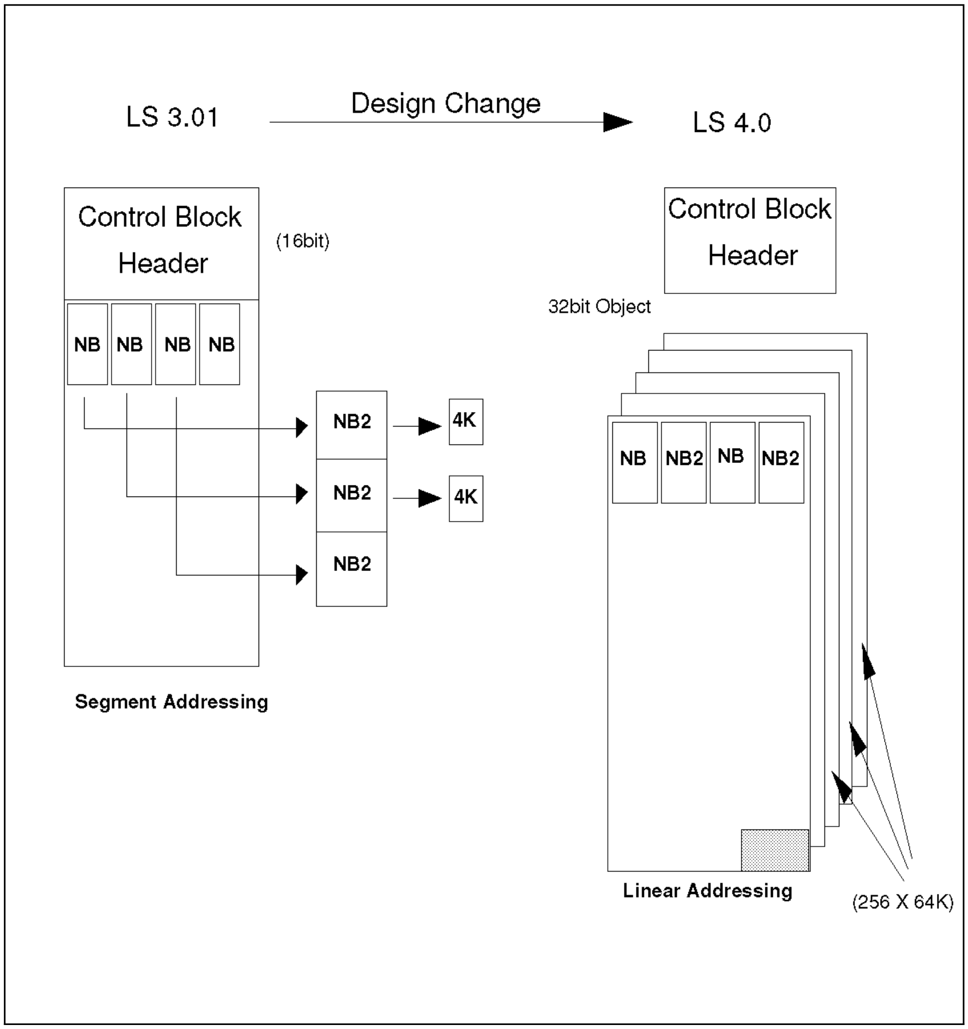

Figure 24. Request Buffer Allocation Design Change

Since configuring **numreqbuf** = 2000 to support 1000 requesters results in 8-16 MB of locked memory, you may need to set **useallmem** option in IBMLAN.INI to Yes. Please be careful that the value that you set for the **numreqbuf** parameter involves a memory-versus-performance trade-off. In the actual environment, NetBIOS commands and sessions resources would be exhausted if **numreqbuf** is set to 2000. That is because each active user needs one or more NetBIOS commands resource and sessions resources. With four LAN adapters, the maximum number of these resources is 1012 (253 X 4).

The next table can be used to determine the maximum **numreqbuf** for each value of **sizreqbuf**.

Table 5. Memory Requirement vs Request Buffer Size and Number				
SIZREQBUF	**MAX NUMREQBUF**	**REQBUF/SEG**	**SELECTORS**	**MEMLOCKED**
1K + 260 = 1284	2000	51	39	2.4MB
2K + 260 = 2308	2000	28	71	4.4MB
4K + 260 = 4356	2000	15	133	8.3MB
8K + 260 = 8452	1792	7	256	14.4MB
16K + 260 = 16644	768	3	256	12.1MB
32K + 260 = 33028	256	1	256	8.0MB

New Configuration Defaults

The Advanced version of LAN Server 3.0 is being used in much larger configurations than the default settings of many parameters allow, necessitating changes by the administrator before the server can be ready for use by all intended clients. LAN Server 4.0 addresses this point by providing a new set of parameter defaults which provide for 100 workstations out of the box. The person who installs LAN Server 4.0 should run the Tuning Assistant that was mentioned before.

The Entry server will have defaults to support only 20 to 30 clients so that it can run in a smaller memory machine than the Advanced server. The Tuning Assistant should be run for Entry server also so that all available system memory will be used to its performance and capacity optimization.

Refer to the "IBMLAN.INI Parameters" on page 78 for the default value of each server.

Large Domain Considerations

There is an increasing demand for the use of a very large domains within LAN Server networks. Domains provide a means of defining one large logical server which is physically spread over a number of server systems. This allows any user who is defined to a domain to access resources also defined to the domain without having knowledge of the physical location of that resource (location independent naming).

LAN Server domains were designed to cater to logical groups of users of around 50 to 100 people. Larger organizations can often be broken down into logical groups of around this size. However, with the large growth in LANs implemented in large organizations, it is now necessary to configure domains of much larger sizes.

The following are the recommendations that are made when attempting to use a very large domain. Use the LAN Server 4.0 Tuning Assistant or CNFGLS30 spreadsheet as a base and make further adjustments as necessary, by following the guidelines below.

Capacity Considerations

1. All heavily used server systems should have multiple LAN adapters installed to increase the number of NetBIOS sessions available to that server. Up to 1012 NetBIOS sessions can be supported per server system if the maximum of four network adapters are installed. Each NetBIOS session established can have multiple connections (logons and NET USE commands). As a general rule, one NetBIOS session will be used by each user accessing any one server.
2. Set **numreqbuf** to two times the number of NetBIOS sessions (x1 in the netx line of IBMLAN.INI) up to a maximum value of 2000. If you have multiple adapters in your server, use the sum of all x1 settings in each of net1, net2, and so on. See Figure 25 on page 76.
3. Set NetBIOS commands (x2 in the netx line of IBMLAN.INI) to 2 times the number of NetBIOS sessions, up to a maximum value of 1012 distributed equally over multiple adapters (x2 in the netx lines).

Sample IBMLAN.INI File and Calculation

```
OS/2 LAN Server initialization file

 [networks]
 ; netx = drivername$, a, drivertype, x1, x2, x3

 net1 = NETBEUI$,0,LM10,50,125,14
 net2 = NETBEUI$,1,LM10,50,100,14
 net2 = NETBEUI$,2,LM10,50,100,14

   :
```

Figure 25. IBMLAN.INI Example of Multiple Netx

Calculation in this case is:

- **Numreqbuf** should be twice of the total number of sessions, so it should be larger than 300.
  2 (net1 + net2 + net3) = 2 x (50 + 50 + 50) = 300
- Total number of NetBIOS commands is 325 and is larger than the twice of 125 and is spread over 3 adapters.
  (net1 + net2 + net3) = (125 + 100 + 100) = 325 > 125

Domain Design Considerations

Besides the specific tuning recommendations made above, the design of the domain itself, including resource location and user load, should be carefully examined to achieve optimal performance in this type of environment. Most of these recommendations are aimed at making optimal use of NetBIOS session connections to each server. Each server can support up to 1012 NetBIOS sessions, or 1000 simultaneous user connections.

The domain controller need not become a bottleneck restricting the number of active user logons if the following design guidelines are followed:

- As a general rule to increase network performance for larger domains, resources should be spread over the domain through multiple servers. Administrators should make sure this is performed in such a way that user access is distributed over a number of servers in order to avoid congestion on single servers at times of peak demand.

- Wherever possible, allocate resources residing on the same server to any one user. This will reduce the total number of NetBIOS sessions established by that user to the domain.
- Do not allocate home directories residing on the domain controller to users, as this will create a bottleneck as the domain controller reaches the limit of NetBIOS sessions. Eliminating the LAN Server messaging function will also conserve NetBIOS names.
- To ensure that users can logon without undue delay, a number of additional servers should be defined as backup domain controllers so that user logons can be distributed to the backup domain controllers when the primary domain controller is busy. Care should be taken to activate the DCDB (Domain Control Data Base) replicator service so that any changes made to the DCDB are automatically shadowed to the additional servers. If you plan to implement backup domain controllers on your LAN we recommend that you install LAN Server 3.01 or 4.0.
- The **T1** timer located in PROTOCOL.INI, and accessible through the LAPS configuration interface, should be increased to ensure that active sessions do not time out as network load increases. However, if you need this change, your LAN is something wrong and the network specialist should analyze the LAN.
- As domain size increases, so do the number of sessions established with each server. To increase the chances of a successful connection with the required server the **autodisconnect** parameter in IBMLAN.INI should be changed to a low value to ensure that inactive sessions are disconnected quickly, freeing sessions for other users.
- If possible, do not use the DOS full screen interface on DOS requester. This means you can disconnect all DOS users from the DCNAMEIBMLAN$ session automatically established (NET USE x: /D).

Summary of Capacity Parameters

In this section we will discuss all relevant parameters for tuning a server system. These parameters reside in three main files. The CONFIG.SYS parameters can set the values for memory, cache, heap space, disk space, swapping activities and write time. The PROTOCOL.INI parameters are relevant for adapter buffers and protocol buffers. The IBMLAN.INI parameters have an influence on several subservices which can run in a server system.

IBMLAN.INI Parameters

The following table shows the parameters and the related subservice from the IBMLAN.INI file. A brief description of all parameters follows after the table which shows the meaning of the parameter and capacity characteristics when increasing or decreasing the values. The default value of this table is for LAN Server 4.0. To see the default value for LAN Server 3.0, you can refer to the Entry section in this table.

The table describes the parameters that mostly affect LAN Server′s capacity. It offers recommendations for adjustment, and lists related parameters. The parameters are located in the Networks and Server sections of the IBMLAN.INI file. Some of these parameters are related to adding users to the network. Others are considered to be capacity parameters, but they may also affect performance, because a lack of server resources forces a user to wait for a resource to become available. The default values for these parameters are set for an average LAN with 20 to 30 users for LAN Server 3.0 and LAN Server 4.0 Entry and around 100 users for LAN Server 4.0 Advanced.

If the server fails to start successfully after IBMLAN.INI parameter values are changed, record the error reported and use the NET ERROR command to determine the cause of the problem.

Table 6 (Page 1 of 2). OS/2 LAN Server Capacity Tuning Parameters - IBMLAN.INI

Subservice / Section	**Parameter**	**Default Value**		**Minimum Value**	**Maximum Value**
		Entry	**Adv**		
Network	**x1** *a	32	102	2	254
	x2 *a	50	225	16	255
	x3 *a	14	14	5	254

Table 6 (Page 2 of 2). OS/2 LAN Server Capacity Tuning Parameters - IBMLAN.INI					
Subservice / Section	**Parameter**	**Default Value**		**Minimum Value**	**Maximum Value**
		Entry	**Adv**		
Server	**maxconnections**	128	300 *b	1	2000
	maxlocks	64	64 *c	1	8000
	maxopens	160	256 *d	1	8000
	maxsearches	150	350 *e	1	11250 *f
	maxsessopens	80	256	1	8000
	maxsessreqs	50	50	1	65535
	maxshares	32	192	2	1000
	maxusers	32	101	1	1000
	numreqbuf	48	250	5	2000
Lsserver	**srvpipes**	3	3	1	20

- *a. This parameter is set by per LAN adapter card.
- *b. This parameter does not count connections to HPFS386 server shares. There can be up to 2048 connections to HPFS386 server shares, in addition to the maxconnections parameter value.

  For FAT file system accesses, this default is now 300.
- *c. This parameter specifies the maximum number of locks the server can have on non-HPFS386 files. The number of locks permitted on HPFS386 files is bounded by the amount of heap space the HPFS386 has available. Each HPFS386 file lock requires at least 30 bytes of heap space.
- *d. The HPFS386 servers ignore this parameter and allocate handles dynamically. The maximum number of opens permitted on HPFS386 files are as follows:
  - Opens for files: 64K
  - Opens for finds: 8192
  - Opens for searches: 6144
- *e. The first open file instance takes approximately 300 bytes from the heap. Each additional instance of the file opened takes approximately 60

bytes. For large numbers of open files, the lack of available physical memory may reduce the maximum number of open files.

The space for the finds and searches comes out of the same 8KB table. So if you have allocated 6KB for searches, and the system uses all of that 6KB, there will only be 2KB left for finds.

For FAT file system accesses, this default is now 350.

The HPFS386 servers ignore this parameter. The maximum number of searches on HPFS386 files is equal to the maximum number of opens for finds plus the maximum number of opens for searches as defined in maxopens.

- *f. This is a practical limit for maxsearches.

x1

This variable indicates the number of NetBIOS sessions the requester or server allocates. Changing this value increases or decreases the number of users and servers on the network defined by the netx statement with which this workstation can communicate. Each session can have multiple connections. Connections include logons and NET USE commands. This parameter must be set to a value that is less than or equal to the maximum sessions (sessions) parameter value in the NETBEUI_NIF section of the PROTOCOL.INI file.

x2

This variable indicates the number of simultaneous NetBIOS commands (network control blocks) a requester or server can post. For a server, changing this value can increase or decrease the number of requester requests it can process at once. If a value less than 16 is specified, the value of 16 is used. This parameter must be set to a value that is less than or equal to the maximum commands (ncbs) parameter value in the NETBEUI_NIF section of the PROTOCOL.INI file.

x3

This variable indicates the number of NetBIOS names the requester allocates. The requester uses NetBIOS names for the computer name and messaging names. To add more messaging names, increase the value of this variable. If this value is greater than 10, start OS/2 LAN Requester and the Messenger service before other NetBIOS applications. This parameter must be set to a value that is less than or equal to the maximum names (names) parameter value in the NETBEUI_NIF section of the PROTOCOL.INI file.

maxconnections

This parameter specifies the maximum number of connections that requesters can have to the server. This is the number of NET USE commands the server can handle.

For example, a user issuing five NET USE commands needs five connections. Five users who each issue one NET USE command need five connections. Increase this parameter value if many users access the server. This parameter value must be greater than or equal to the **maxusers** parameter value.

maxlocks

This parameter specifies the maximum number of file locks on the server. This is the maximum number of byte ranges (records) that may be locked by users on the server. Increase the value of this parameter if there is a large number of heavily used files. This parameter applies only to lock requests issued by DOS requesters.

maxopens

This parameter specifies the maximum number of files, pipes, and devices the server can have open at one time.

For example, the value of this parameter must be greater than or equal to five for a user opening five files. The value of this parameter must also be greater than or equal to five for five users opening the same file. If many users access the server simultaneously, increase the value of this parameter.

Each DOS remote IPL workstation requires three open files at the remote IPL server workstation.

Note: The maximum number of open files is 8000. However, the maximum number of unique open files is 1279. The first opening of a file counts against the maximum of 1279. Additional openings of the same file count against the maximum of 8000.

maxsearches

This parameter specifies the maximum number of directory searches the server can do simultaneously. These searches are executed when a user does a wild-card search of a directory; for example,

```
DIR Z:TEXTFILE.*
```

If the server′s files are heavily used, increase the value of this parameter. See digit position 7 of the **srvheuristics** parameter for more information about searches.

maxsessopens

This parameter specifies the maximum number of files, pipes, and devices one requester can have open on the server. If many of the server resources are used simultaneously, increase the value of this parameter.

Note: The server uses some of the value specified with the **maxsessopens** parameter for internal processing, so the entire value specified with this parameter is not available to the user.

maxsessreqs

This parameter specifies the maximum number of resource requests one requester can have pending on the server. If users need to perform multiple tasks simultaneously on the server, increase the value of this parameter.

maxshares

This parameter specifies the maximum number of resources the server can share with the network. For example, if one user is using five resources on the server, the value of this parameter must be at least 5; but if five users are using the same server resource, the value of this parameter need only be set to 1. If the server shares many resources, increase the value of this parameter.

Note: The number of shared resources displayed by the NET CONFIG SRV command will be different from the number specified with the maxshares parameter. This is because the number of shared resources displayed by the NET CONFIG SRV command also includes default system shares (ibmlan$, admin$, and so on), and one share for each partition on the server (a$, b$, and so on).

maxusers

This parameter sets the maximum number of users who can use the server simultaneously. This equals the number of users who might issue a NET USE command to the server. A user who issues five NET USE commands counts as one user. Five users, each issuing a NET USE command to the same resource, count as five users. This value is the number of NetBIOS sessions on the server.

Note: The **maxusers** parameter value cannot exceed the **maxconnections** parameter value.

srvpipes

This parameter sets the maximum number of pipes that the server uses. If many users log on simultaneously, increase this value. The rule is:
 srvpipes > maxuser / 40

3.7 Logon Performance Tuning

Performance of the network at logon time can be a problem if most of your users logon to the LAN at the same time. This has historically been an area of weakness in the LAN Server product, but significant enhancements have now been made in LAN Server 3.01 and 4.0. The logon process is a fairly complex procedure and is not just a case of sending a user ID and password to a server to be validated. Before talking about logon performance tuning, an SMB overview is described because this is the base knowledge needed to understand what is going on the wire.

SMB Overview

SMB (Server Message Block) is the name of protocol over NetBIOS and it is used by server and requester to communicate. SMB is the basic protocol between server and requester and by understanding SMB′s role, you can learn more about LAN Server.

SMB protocols are categorized in three groups described here. In the reference publication *OS/2 LAN Server, Network Administrator Reference Volume 2: Performance Tuning*, these same terminologies are used. There are three types of SMB protocols that are used for transferring data between a requester and a server:

Core SMB Type
Header and data reside within one network buffer. This SMB protocol is used to transfer amounts of data that are less than equal to the buffer size specified by the **sizreqbuf** or **sizworkbuf** parameters.

RAW SMB Type
RAW means the frame is not cooked because most frames have no SMB header. SMB layers trust the quality of LAN and there is no sequence control in the SMB layer for the frames over LAN. Header and data reside within the first network buffer only. Subsequent data is transferred through big buffers to the user′s data area directly without header information. This flow continues until all requested data has been received; no additional SMB requests are required. RAW SMB protocol is used if the amount of data to transfer exceeds the network buffer size and if big buffers are available.

Multiplexed SMB Type
For example, the Read Block Multiplexed SMB flow operates with the following style:

1. Header and the part of data reside within the first network buffer.
2. Subsequent data is sent without header information along with the SMB response protocol header.
3. This flow continues until all requested data has been received; no additional SMB requests are required.

Multiplexed SMB protocol is used if the amount of data to transfer exceeds the requester buffer size and if either RAW SMB protocol is not supported or big buffers are not available. Both RAW and Multiplexed protocols are used to transfer large amounts of data very quickly.

Random file access is characterized by a request for a small amount of data that may reside anywhere in the file. Core SMB protocol is typically used by the requester in this case. The data is cached in the file system to minimize disk seeks.

Sequential file access is characterized by successive requests for data that is contiguous in the file. Core SMB protocol is used by the requester in this case, as long as the requested amount of data is less than or equal to the requester′s network buffer size (**sizworkbuf**).

Large file transfer is characterized by a request for an amount of data that is greater than the size of the requester′s network buffer (**sizworkbuf**). Multiplexed or RAW SMB protocol is used by the requester in this case.

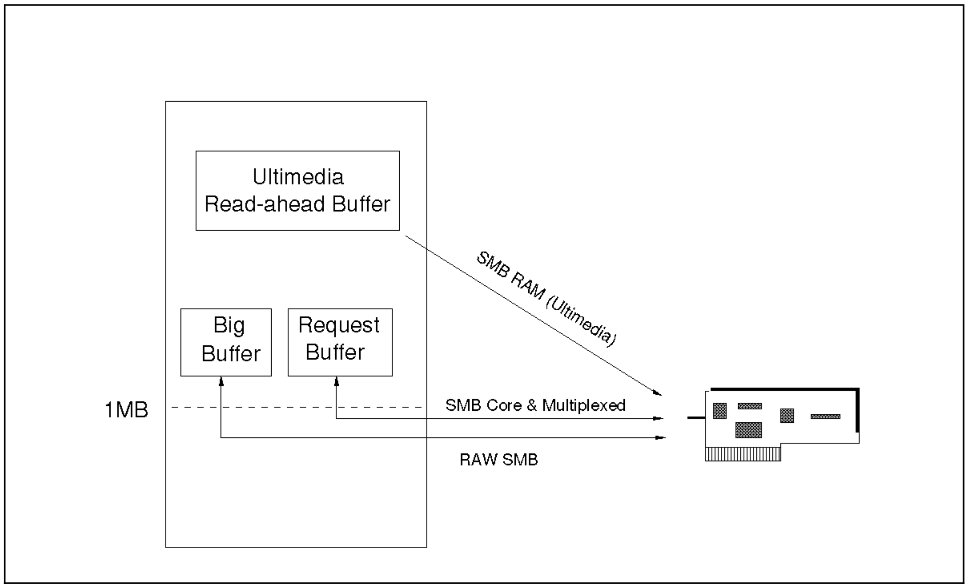

Figure 26. SMB Types

LAN Server 4.0 provides an excellent tool to work with SMB, called SMB Tool. SMB Tool (SMBTOOL.EXE) is zipped in the Productivity Diskette 1. It can be used to interpret various types of network traces. SMB Tool also can activate its internal trace facility to capture the SMBs in the server or the requester. It includes extensive formatting and filtering features for SMB trace data. It can be used for advanced problem determination too. It can analyze TAP (Trace and Performance) format traces and interpret some SMB frames that DatagLANce* cannot understand.

IBM added to the industry standard SMB protocol and defined the SMB extensions for IBM LAN Server 3.0. SMB commands have been used for most common file oriented network requests such as open, close, read and write, as well as for notifying a network server of various presentation and application layer events occurring on the network client. Various levels (dialects) of the SMB protocol are used in network products from IBM, Microsoft and DEC** and others. X/Open has published a reference, *Protocols for X/Open PC Interworking: SMB Version 2*, that describes the base protocol used by many SMB based network products. This is also referred to as the LM1.2 SMB dialect. IBM enhancements are additions to the LM1.2 dialect designed to solve design problems associated with that dialect as well as to add features required to fully support new state of the art operating systems.

SMB Flow Example

Server and requester (also called consumer) communicate with SMB request and SMB response frames. The legacy servers use basic SMB protocol described in the following figure as an example. Consumer sends an OPEN SMB, then receive a response, then sends a READ SMB, and so on. This is slow process because each request is serialized and waits for a response.

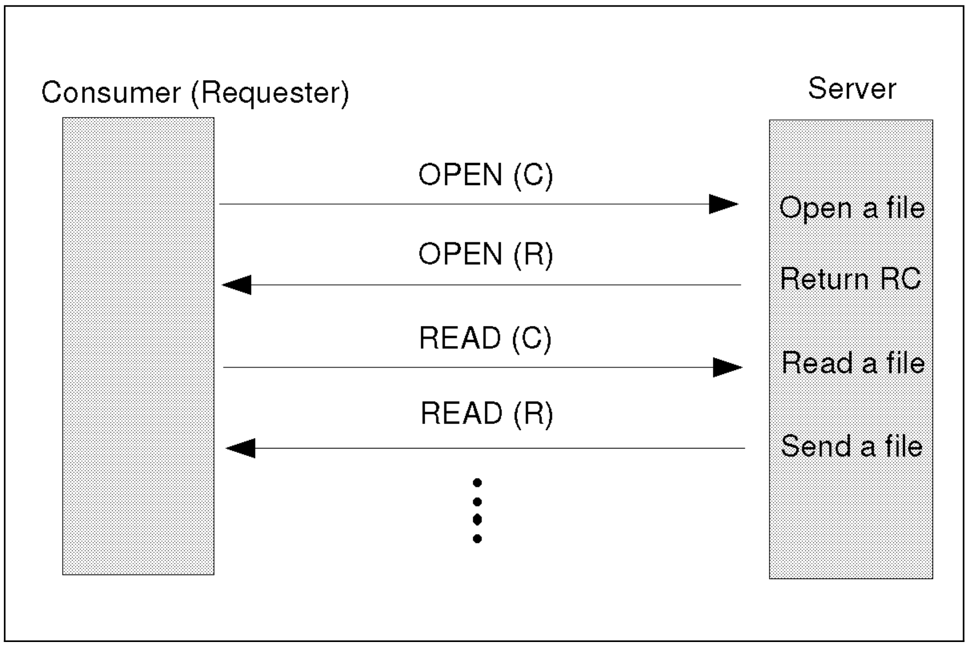

Figure 27. Example of Simple SMB Flow

With the latest evolution of the protocol, the same work can be done with a single interaction using a new SMB type called OPEN_and_X. OPEN_and_X SMB can have another SMB subcommand and in this case it is READ_and_X. Then again READ_and_X can have another SMB subcommand which could be CLOSE.

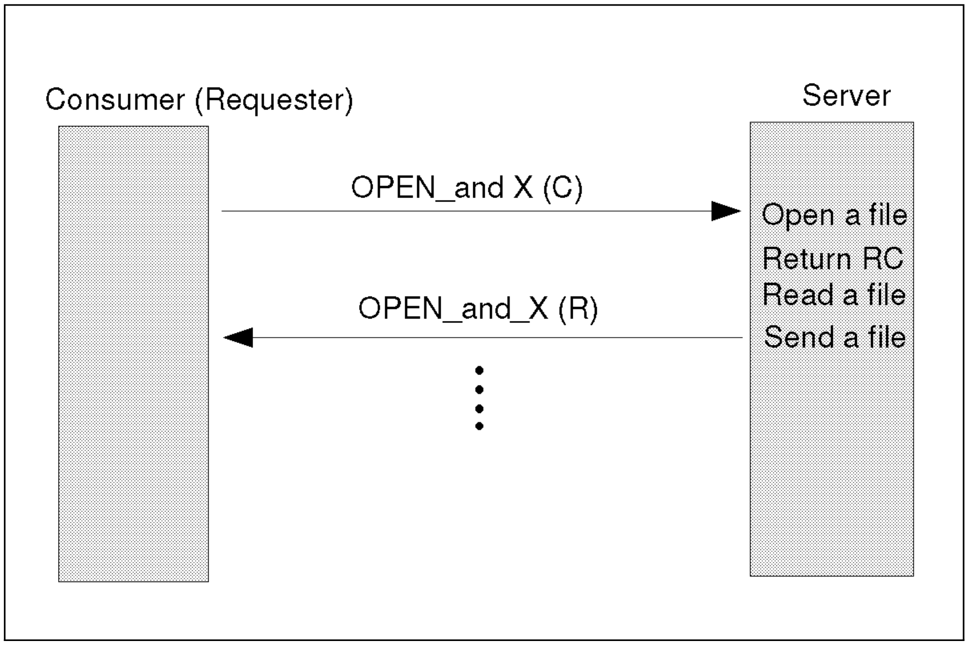

Figure 28. Example of Advanced SMB Flow

If the data is small and can be fit into the request buffer, core protocol will be used. If it is large and cannot fit into one request buffer, RAW or multiplexed protocol is used. With this way, SMB header length could be longer than basic length of 32 bytes. The following figure shows a rough structure how OPEN_and_X (and_X type) SMB is organized.

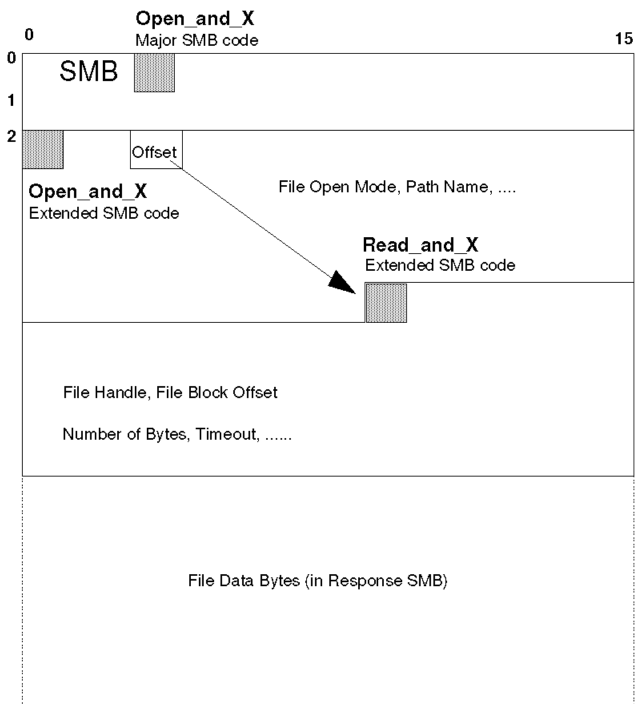

Figure 29. Example of And_X Type SMB Header

With this in mind, redirector assumes 260 bytes as a frame overhead area to **sizreqbuf** or **sizworkbuf**. For a 4096 bytes default value of these parameters, 4356 bytes is allocated and it is the maximum data length which could be sent to the LAN.

The assumption of 260 bytes comes from the following calculation:

```
  LAN Header     -----  30 bytes (DA+SA+Maximum RIs)
  LLC Header     -----   4 bytes
  NetBIOS Header -----  14 bytes
  SMB Header     ----- 212 bytes (as a maximum length)
 -----------------------------------------------------
  Total Header Length  260 bytes
```

However, the average SMB length is less than 212 bytes when a large data transfer is used, so smaller buffer size can be specified in the NetBEUI **maxdatarcv** or LAN adapter′s **xmitbufsize**. The default value of **maxdatarcv** is 4168 bytes is less than 4356 but it is probably a reasonable size for the most of transactions. You will see the real trace examples in “Example of Tuning an Application” on page 153.

Analyzing Logon Process

This section examines the actual logon process by analyzing the trace date taken at the LAN level, by using the SMB Tool utility. You will see what can be tuned to get a shorter logon time.

LAN Server has three logon types. They are domain logon, local logon and non-validated logon. The difference is the type of verification that occurs during LAN Server logon as follows:

- **Domain Logon**

  Validates user ID and password on the domain logon server. The user ID and password are sent to a domain controller or backup server to be validated using the domain copy of the NET.ACC.

- **Local Logon**

  Validates your user ID and password on the local workstation using the local copy of the NET.ACC file.

- **Non-validated logon**

  Allows users to log on without user ID and password being verified. The user ID and password are only stored.

In the case of local and non-validated logon, the definition in DCDB such as the logon assignment, the logon profile, and the public application are not used at all. Each time the requester accesses to the server′s resource, the user′s authority for the resource is checked by the server. With LAN Server 3.0, the logon option is defined as verification= parameter in the Netlogon section of IBMLAN.INI. With LAN Server 4.0, this is set in the 37th parameter in **wrkheuristics** in the Requester section of IBMLAN.INI.

The average domain logon procedure will consist of the following steps:

1. Validation of user ID and password: This is normally handled at the domain controller. The user ID and password is registered with the client code at this stage.

2. Enforce a single logon: This involves adding a special NetBIOS name (domain name/user ID) to the network and checking that it does not already exist.

3. Add additional NetBIOS names to the network: These are the client workstation and messaging names.

4. Establish user logon attributes: This allocates the users logon resource assignments and LAN Server applications at logon.

5. Register the user ID and password with UPM (OS/2 requesters only): This allows a user to logon at a workstation with only local validation, which defers authentication until the client attaches to a resource. This may initially be viewed as a security liability but access control is enforced when a request for a resource is made. When a user issues a NET USE command to access a resource, the user ID and password are checked before the user is permitted access to the resource. This allows users to access resources on domains that they are not logged on to, cross domain access.

Figure 30 on page 91 is a sample data flow for the logon procedure.

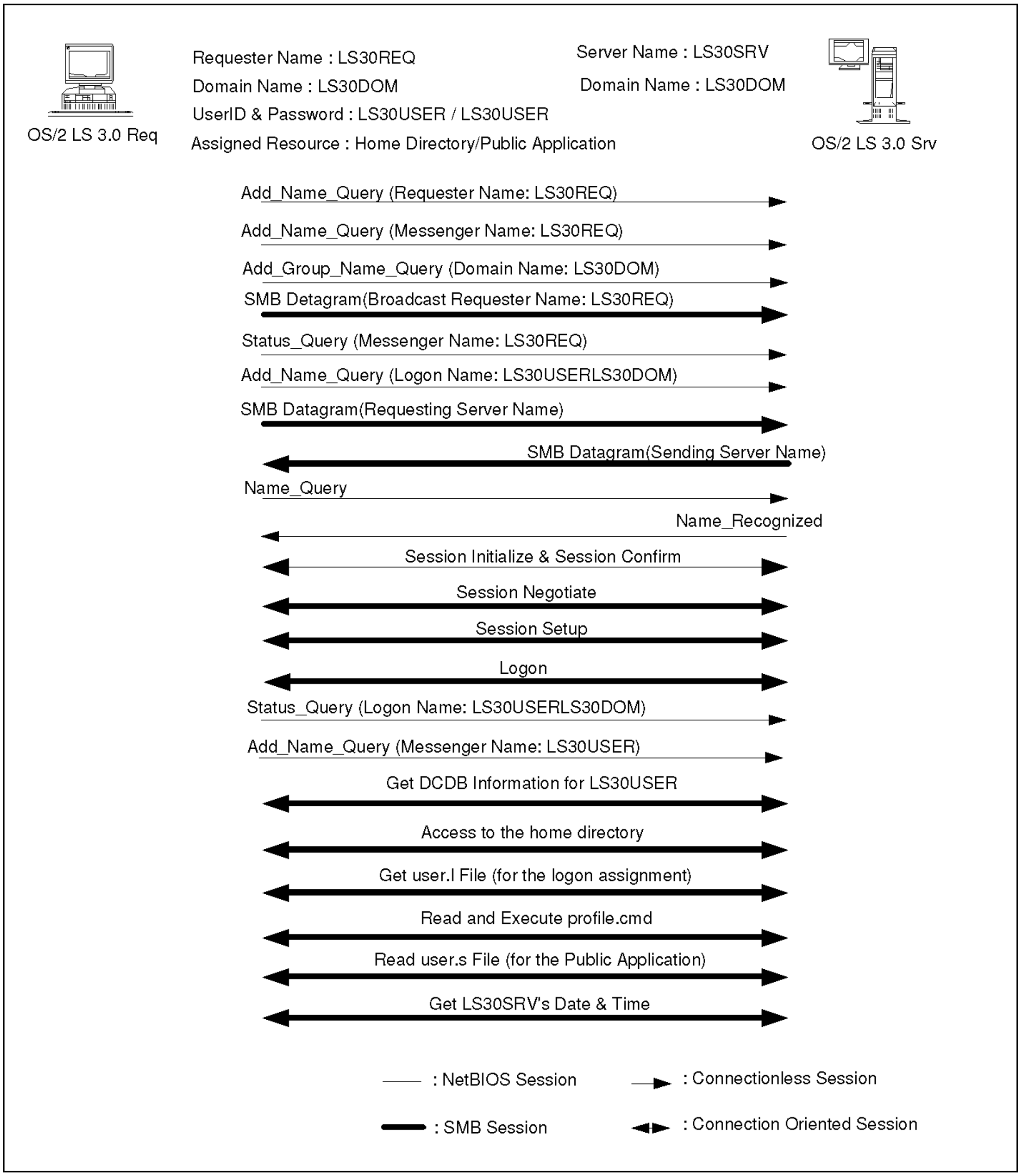

Figure 30. LAN Server 3.0 Logon Sequence

The following describes the detailed sequence of the logon process. Steps which have a pointer such as (*a) have more detailed consideration in the consideration section, after these steps.

1. Add_Name_Query

   The requester broadcasts the Add_Name_Query command for adding the Requester name, LS30REQ to the network. (*a)

2. Add_Name_Query

   The requester broadcasts the Add_Name_Query command for adding the Messenger name, LS30REQ to the network. (*a)

3. Add_Group_Name_Query

   The requester broadcasts the Add_Group_Name_Query command for adding the Domain name, LS30DOM. (*a)

4. SMB Datagram (Broadcast Requester Name)

   The requester broadcasts the Requester name, LS30REQ, for requesting the logon acceptable servers to announce themselves using a Mailslot. (*b, *f)

5. Status_Query

   The requester broadcasts the Status_Query commands to check for a duplicate Messenger name, LS30REQ. (*c,*d)

6. Add_Name_Query

   The requester broadcasts the Add_Name_Query command to add the Logon name, LS30USERLS30DOM. (*a,*e)

7. SMB Datagram (Request Server Name)

   The requester broadcasts the Logon Name, LS30USER, for trying to find the server name to logon. (*b,*f)

8. SMB Datagram (Send Server Name) The server broadcasts the server name, LS30SRV, looking for a response from the requester. (*b,*f)

9. Name_Query

   The requester broadcasts the Name_Query command to find out server′s LAN address using the name, LS30SRV. (*a,*g)

10. Name_Recognized

    The server sends the Name_Recognized command to the requester to notify its address. (*g)

11. Session_Initialize & Session_Confirm

The requester sends the Session_Initialized frame and the server returns the Session_Confirm command. In this initialization, the NetBIOS frame size for the session is determined. (*g)

12. Session Negotiate

    The requester sends the Negotiate SMB command to tell the server its acceptable SMB version. The server responds to the requester with its maximum SMB data size and big buffer capability.

13. Session Setup

    The requester sends its maximum SMB data size and determines the buffer size for the session. The requester sends its logon name, LS30USER. It then connects LS30SRVIPC$. (*g, *h)

14. Logon

    The requester sets the parameter and issues an internal UserLogon API to logon the server. (*h)

15. Status_Query

    The requester broadcasts Status_Query commands to check for a duplicate logon Name, LS30USERLS30DOM. (*a,*e)

16. Add_Name_Query

    The requester broadcasts the Add_Name_Query commands for the messenger name, LS30USER. (*a,*e)

17. Get DCDB information for the logon user

    The requester tries to connect to LS30SRVIBMLAN$ to open DCDB directory and tries to search LS30SRVDCDBUSERSLS30USER directory by issuing an internal UserGetInfo API for getting the information for LS30USER. (*h)

18. Access to the Home Directory

    The requester will establish the access to LS30USER′s home directory.

19. Read a file USERS.L for the logon assignment

    The requester tries to read LS30SRVDCDBUSERSLS30USERUSER.L file to get the logon assignment information.

20. Read and Execute the profile.cmd

    The requester will read and execute the profile.cmd.

21. Read a file USERS.S for the public application

    The requester will read LS30SRVDCDBUSERSLS30USERUSER.S file to get the public application information.

Logon Script

If the user ID has a logon script, it is executed at this point.

22. Get the server's date and time

    The requester tries to get the LS30SRV's date and time. (*h)

Considerations:

*a: The frame number of this command is (1 + **netbiosretries** (default:8)). The **netbiosretries** parameter is specified in the PROTOCOL.INI file and you can reduce the number of it. (For DLR use DXMT0MOD.SYS, dlcretricount (**RC=**) parameter in CONFIG.SYS.) In all but the most complex bridged LAN environments, this is unnecessarily high. Reducing this value to 1 can save up to 3.5 seconds for every NetBIOS name that the workstation adds to the network. If this value is too low, you will experience a successful logon followed by a session termination. Should this occur, increase the value until the session is no longer terminated.

LAN Server 4.0

LAN Server 4.0 has a default setting of **netbiosretries** as 2.

*b: Since these frames are not connection oriented but connectionless, discarding of them at a bridge or at the network adapters could be invisible on the LAN. NetBIOS traces (TRACE ON 164 from the command line before the failure and OS2TRACEMASK=0x7FF in PROTOCOL.INI) on the target machine could be used to see if the adapter actually received the datagrams and passed them to NetBEUI. If it seems that these datagrams are dropped (especially for the step 8 and 9), you need to increase the value for the related parameters, **numdgrambuf** in IBMLAN.INI and **datagrampackets** in PROTOCOL.INI.

*c: This step can be eliminated if you do not use the messaging utility since this reduces the number of NetBIOS names which need to be added to the network. This is achieved by removing the messenger from the Services section of the OS/2 workstation's IBMLAN.INI file or on DOS workstations by starting DOS LAN Requester with the NET START RDR command.

*d: The frame number of this command is the value of **netbiosretries** in PROTOCOL.INI mentioned above.

*e: This step can be eliminated if you set the **multilogon** parameter in IBMLAN.INI to YES, (**/MLO** in DOSLAN.INI). This eliminates the need for the adapter to check for the existence of duplicate NetBIOS names.

LAN Server 4.0

The **multilogon** parameter is in the 40th entry in the **wrkheuristics** parameter.

*f : If you change one or more additional servers to backup domain controllers, it can also receive the requester′s request and send their names. The fastest server to communicate with the logon requester becomes the logon server for it. These can be configured to share the load of the logon process during busy periods. Make sure you have LAN Server V3.01 up if you intend to use backup domain controllers.

*g: If the requester has the logon assignment resources in several servers, these steps are repeated for each server.

*h: The named_pipe is used for these steps. In the case many users logon simultaneously, increase the value of **srvpipes** parameter in IBMLAN.INI. The following equation can be used to calculate a value for it. Do not use the calculated value if it is less than the default value, which is 3, or more than maximum value, which is 20.

srvpipes = **maxusers** / 40

DatagLANce Trace for Complex Logon

The following is another logon scenario with more resource connection. The trace was taken and formatted by DatagLANce. In the trace output, `NetB` is NetBIOS functional address, `Req` indicates a requester, `DC` means domain controller, `AS` means additional server, and `EXT` means external server. The scenario is the following. Number is a corresponding trace frame number:

1. The user ID SHIMIZU logs on to the domain ITSCAUS. (0-41) The domain controller name is ITSCSV00. This was known during the requester startup and before this trace.
2. The user has several logon assignments and they are processed. (42-66)
3. The user also has a printer assignments from additional server ITSCSV01. (94-108)
4. The user has `NET USE` command to other domain′s server CSPPLAN. (144-157)
5. After logon, the user went to T:TOSHI and executed `Type Notice.Txt` command. (160-176)
6. The user logged off. (179-219)

You will see that the access to each server results a sequence of Negotiate and Session Setup and X protocol, along with NetBIOS level session establishment.

```
Num Dest Sour Size Interpretation
-------------------------------------------------------------------------------
  0 NetB Req   204 SMB C Transaction \MAILSLOT\NET\NETLOGON
  5 NetB Req    63 NETBIOS Find name ITSCSV00
  6 Req  DC     61 NETBIOS Name ITSCSV00         recognized
  7 DC   Req    17 LLC D=F0 S=F0 C U SABME P
  8 Req  DC     17 LLC D=F0 S=F0 R U UA F
  9 DC   Req    18 LLC D=F0 S=F0 C S RR NR=0 P
 10 Req  DC     18 LLC D=F0 S=F0 R S RR NR=0 F
 11 DC   Req    32 NETBIOS D=56 S=81 Session initialize
 12 Req  DC     32 NETBIOS D=81 S=56 Session confirm
 13 DC   Req    18 LLC D=F0 S=F0 R S RR NR=1 F
 14 DC   Req   136 SMB C Negotiate Protocol PC NETWORK PROGRAM 1.0 (more)
 15 Req  DC     18 LLC D=F0 S=F0 R S RR NR=2
 16 Req  DC    109 SMB R Negotiated Protocol 4
 17 DC   Req   187 SMB C Session Setup and X Account SHIMIZU
 18 Req  DC     18 LLC D=F0 S=F0 R S RR NR=3
 19 Req  DC    115 SMB R Session Established and X
 20 DC   Req   204 SMB C Transaction \PIPE\LANMAN
 21 Req  DC     18 LLC D=F0 S=F0 R S RR NR=4
 22 Req  DC    194 SMB R Transaction Completed
 23 NetB Req    63 NETBIOS Status Query
 24 DC   Req    32 NETBIOS D=56 S=81 Data ACK
 25 Req  DC     18 LLC D=F0 S=F0 R S RR NR=5
 26 NetB Req    63 NETBIOS Check name SHIMIZU        <03>
 27 NetB Req    63 NETBIOS Check name SHIMIZU        <03>
 28 DC   Req   101 SMB C Tree Connect and X Path=\\ITSCSV00\IBMLAN$ Device=?????
 29 Req  DC     84 SMB R Tree Connected and X Service=A:
 30 DC   Req   134 SMB C Transact2 Find First \DCDB\USERS\SHIMIZU
 31 Req  DC    132 SMB R Transact2 Completed
 32 DC   Req    67 SMB C Tree Disconnect T=703D
 33 Req  DC     67 SMB R Tree Disconnected
 34 DC   Req    96 SMB C Tree Connect and X Path=\\ITSCSV00\IPC$ Device=IPC
 35 Req  DC     77 SMB R Tree Connected and X Service=IPC
 36 DC   Req   154 SMB C Transaction \PIPE\LANMAN
 37 Req  DC    315 SMB R Transaction Completed
 38 DC   Req   352 SMB C Transaction Call \PIPE\IBMLAN\SERVER.RNS
 39 Req  DC     32 NETBIOS D=81 S=56 Data ACK
 40 DC   Req    18 LLC D=F0 S=F0 R S RR NR=10 F
 41 Req  DC    231 SMB R Transaction Completed
 42 DC   Req    98 SMB C Tree Connect and X Path=\\ITSCSV00\SHIMIZU Device=A:
 43 Req  DC     84 SMB R Tree Connected and X Service=A:
 44 DC   Req   147 SMB C Open and X \DCDB\USERS\SHIMIZU\USER.L
 45 Req  DC    924 SMB R Opened and X F=1F62
 46 DC   Req   147 SMB C Open and X \DCDB\USERS\SHIMIZU\USER.L
 47 Req  DC    924 SMB R Opened and X F=1F63
 48 DC   Req    73 SMB C Close File F=1F62
 49 Req  DC     67 SMB R Closed File
```

Figure 31 (Part 1 of 5). DatagLANce Trace Output for Complex Logon Scenario

```
50 DC   Req    73 SMB C Close File F=1F63
51 Req  DC     67 SMB R Closed File
52 DC   Req   133 SMB C Transaction \PIPE\LANMAN
53 Req  DC    129 SMB R Transaction Completed
54 DC   Req    99 SMB C Tree Connect and X Path=\\ITSCSV00\DISK-COM Device=A:
55 Req  DC     84 SMB R Tree Connected and X Service=A:
56 DC   Req   133 SMB C Transaction \PIPE\LANMAN
57 Req  DC    129 SMB R Transaction Completed
58 DC   Req    99 SMB C Tree Connect and X Path=\\ITSCSV00\DISK-TRN Device=A:
59 Req  DC     84 SMB R Tree Connected and X Service=A:
60 DC   Req   132 SMB C Transaction \PIPE\LANMAN
61 Req  DC    128 SMB R Transaction Completed
62 DC   Req    98 SMB C Tree Connect and X Path=\\ITSCSV00\LSADMIN Device=A:
63 Req  DC     84 SMB R Tree Connected and X Service=A:
64 DC   Req   133 SMB C Transaction \PIPE\LANMAN
65 Req  DC    129 SMB R Transaction Completed
66 DC   Req    99 SMB C Tree Connect and X Path=\\ITSCSV00\PROJ-PER Device=A:
67 Req  DC     84 SMB R Tree Connected and X Service=A:
68 DC   Req   133 SMB C Transaction \PIPE\LANMAN
69 Req  DC     32 NETBIOS D=81 S=56 Data ACK
70 Req  DC    129 SMB R Transaction Completed
71 NetB Req    63 NETBIOS Find name ITSCSV01
72 Req  AS     61 NETBIOS Name ITSCSV01         recognized
73 AS   Req    17 LLC D=F0 S=F0 C U SABME P
74 Req  AS     17 LLC D=F0 S=F0 R U UA F
75 AS   Req    18 LLC D=F0 S=F0 C S RR NR=0 P
76 Req  AS     18 LLC D=F0 S=F0 R S RR NR=0 F
77 AS   Req    32 NETBIOS D=89 S=85 Session initialize
78 Req  AS     32 NETBIOS D=85 S=89 Session confirm
79 AS   Req    18 LLC D=F0 S=F0 R S RR NR=1 F
80 AS   Req   136 SMB C Negotiate Protocol PC NETWORK PROGRAM 1.0 (more)
81 Req  AS     18 LLC D=F0 S=F0 R S RR NR=2
82 Req  AS    109 SMB R Negotiated Protocol 4
83 DC   Req    32 NETBIOS D=56 S=81 Data ACK
84 AS   Req   191 SMB C Session Setup and X Account SHIMIZU
85 Req  AS     18 LLC D=F0 S=F0 R S RR NR=3
86 Req  AS     32 NETBIOS D=85 S=89 Data ACK
87 AS   Req    18 LLC D=F0 S=F0 R S RR NR=3 F
88 Req  AS    117 SMB R Session Established and X
89 DC   Req   132 SMB C Transaction \PIPE\LANMAN
90 Req  DC     18 LLC D=F0 S=F0 R S RR NR=27
91 AS   Req    32 NETBIOS D=89 S=85 Data ACK
92 Req  AS     18 LLC D=F0 S=F0 R S RR NR=4
93 Req  DC    128 SMB R Transaction Completed
```

Figure 31 (Part 2 of 5). DatagLANce Trace Output for Complex Logon Scenario

```
  94 AS   Req   101 SMB C Tree Connect and X Path=\\ITSCSV01\IBM4029 Device=LPT1:
  95 Req  AS     18 LLC D=F0 S=F0 R S RR NR=5
  96 Req  AS     79 SMB R Tree Connected and X Service=LPT1:
  97 DC   Req   133 SMB C Transaction \PIPE\LANMAN
  98 Req  DC    129 SMB R Transaction Completed
  99 AS   Req    32 NETBIOS D=89 S=85 Data ACK
 100 Req  AS     18 LLC D=F0 S=F0 R S RR NR=6
 101 AS   Req   102 SMB C Tree Connect and X Path=\\ITSCSV01\4039-300 Device=LPT1:
 102 Req  AS     18 LLC D=F0 S=F0 R S RR NR=7
 103 Req  AS     79 SMB R Tree Connected and X Service=LPT1:
 104 DC   Req   132 SMB C Transaction \PIPE\LANMAN
 105 Req  DC    128 SMB R Transaction Completed
 106 AS   Req    32 NETBIOS D=89 S=85 Data ACK
 107 Req  AS     18 LLC D=F0 S=F0 R S RR NR=8
 108 AS   Req   101 SMB C Tree Connect and X Path=\\ITSCSV01\IBM4079 Device=LPT1:
 109 Req  AS     18 LLC D=F0 S=F0 R S RR NR=9
 110 Req  AS     79 SMB R Tree Connected and X Service=LPT1:
 111 DC   Req   100 NETBIOS
 112 DC   Req    32 NETBIOS D=56 S=81 Data ACK
 113 Req  DC     67 NETBIOS
 114 DC   Req   147 SMB C Open and X \DCDB\USERS\SHIMIZU\USER.S
 115 Req  DC    192 SMB R Opened and X F=1F64
 116 DC   Req    73 SMB C Close File F=1F64
 117 Req  DC     67 SMB R Closed File
 118 DC   Req   147 SMB C Open and X \DCDB\USERS\SHIMIZU\USER.S
 119 Req  DC    192 SMB R Opened and X F=1F65
 120 DC   Req    73 SMB C Close File F=1F65
 121 Req  DC     67 SMB R Closed File
 122 DC   Req   138 SMB C Open and X \DCDB\DATA\DCDB.A
 123 Req  DC   1944 SMB R Opened and X F=1F66
 124 Req  DC   1944 NETBIOS D=81 S=56 Data, 1912 bytes
 125 AS   Req    32 NETBIOS D=89 S=85 Data ACK
 126 DC   Req    18 LLC D=F0 S=F0 R S RR NR=35 F
 127 Req  DC    396 NETBIOS D=81 S=56 Data, 364 bytes
 128 Req  AS     18 LLC D=F0 S=F0 R S RR NR=10
 129 DC   Req    77 NETBIOS
 130 DC   Req    32 NETBIOS D=56 S=81 Data ACK
 131 Req  DC   1165 NETBIOS
 132 DC   Req    73 SMB C Close File F=1F66
 133 Req  DC     67 SMB R Closed File
 134 DC   Req    32 NETBIOS D=56 S=81 Data ACK
 135 DC   Req   267 SMB C Transaction Call \PIPE\IBMLAN\SERVER.RNS
 136 Req  DC     18 LLC D=F0 S=F0 R S RR NR=39
 137 Req  DC    242 SMB R Transaction Completed
```

Figure 31 (Part 3 of 5). DatagLANce Trace Output for Complex Logon Scenario

```
138 DC   Req    67 SMB C Tree Disconnect T=783D
139 Req  DC     67 SMB R Tree Disconnected
140 DC   Req    67 SMB C Tree Disconnect T=D83D
141 Req  DC     67 SMB R Tree Disconnected
142 DC   Req    32 NETBIOS D=56 S=81 Data ACK
143 Req  DC     18 LLC D=F0 S=F0 R S RR NR=42
144 NetB Req    63 NETBIOS Find name CSPPLAN
145 Req  EXT    75 NETBIOS Name CSPPLAN          recognized
146 EXT  Req    31 LLC D=F0 S=F0 C U SABME P
147 Req  EXT    75 NETBIOS Name CSPPLAN          recognized
148 Req  EXT    31 LLC D=F0 S=F0 R U UA F
149 EXT  Req    32 LLC D=F0 S=F0 C S RR NR=0 P
150 Req  EXT    32 LLC D=F0 S=F0 R S RR NR=0 F
151 EXT  Req    46 NETBIOS D=87 S=86 Session initialize
152 Req  EXT    46 NETBIOS D=86 S=87 Session confirm
153 EXT  Req   150 SMB C Negotiate Protocol PC NETWORK PROGRAM 1.0 (more)
154 Req  EXT   115 SMB R Negotiated Protocol 3
155 EXT  Req   165 SMB C Session Setup and X Account SHIMIZU
156 Req  EXT    46 NETBIOS D=86 S=87 Data ACK
157 Req  EXT    97 SMB R Session Established and X
158 EXT  Req    46 NETBIOS D=87 S=86 Data ACK
159 Req  EXT    32 LLC D=F0 S=F0 R S RR NR=4
160 DC   Req    75 SMB C Check Directory Path \toshi
161 Req  DC     67 SMB R Checked Directory Path
162 DC   Req    32 NETBIOS D=56 S=81 Data ACK
163 Req  DC     18 LLC D=F0 S=F0 R S RR NR=44
164 DC   Req   132 SMB C Transact2 Find First \toshi\notice.txt
165 Req  DC    139 SMB R Transact2 Completed
166 DC   Req    86 NETBIOS
167 DC   Req    32 NETBIOS D=56 S=81 Data ACK
168 Req  DC     87 NETBIOS
169 DC   Req   115 SMB C Open and X \toshi\NOTICE.TXT
170 Req  DC     97 SMB R Opened and X F=1F67
171 DC   Req    77 NETBIOS
172 DC   Req    32 NETBIOS D=56 S=81 Data ACK
173 Req  DC    943 NETBIOS
174 Req  DC     18 LLC D=F0 S=F0 R S RR NR=48
175 DC   Req    73 SMB C Close File F=1F67
176 Req  DC     67 SMB R Closed File
177 DC   Req    32 NETBIOS D=56 S=81 Data ACK
178 Req  DC     18 LLC D=F0 S=F0 R S RR NR=50
179 DC   Req    67 SMB C Tree Disconnect T=F840
180 Req  DC     67 SMB R Tree Disconnected
181 DC   Req    67 SMB C Tree Disconnect T=C840
182 Req  DC     67 SMB R Tree Disconnected
```

Figure 31 (Part 4 of 5). DatagLANce Trace Output for Complex Logon Scenario

```
183 DC   Req   67 SMB C Tree Disconnect T=D040
184 Req  DC    67 SMB R Tree Disconnected
185 DC   Req   67 SMB C Tree Disconnect T=F040
186 Req  DC    67 SMB R Tree Disconnected
187 DC   Req   67 SMB C Tree Disconnect T=D840
188 Req  DC    67 SMB R Tree Disconnected
189 DC   Req   71 SMB C User Logoff and X
190 Req  DC    71 SMB R User Logged-Off and X
191 DC   Req   32 NETBIOS D=56 S=81 Data ACK
192 DC   Req   32 NETBIOS D=56 S=81 Session end
193 Req  DC    18 LLC D=F0 S=F0 R S RR NR=58
194 DC   Req   17 LLC D=F0 S=F0 C U DISC P
195 Req  DC    17 LLC D=F0 S=F0 R U UA F
196 EXT  Req   81 SMB C Tree Disconnect T=F840
197 Req  EXT   81 SMB R Tree Disconnected
198 EXT  Req   85 SMB C User Logoff and X
199 Req  EXT   85 SMB R User Logged-Off and X
200 EXT  Req   46 NETBIOS D=87 S=86 Data ACK
201 EXT  Req   46 NETBIOS D=87 S=86 Session end
202 Req  EXT   32 LLC D=F0 S=F0 R S RR NR=8
203 EXT  Req   31 LLC D=F0 S=F0 C U DISC P
204 AS   Req   67 SMB C Tree Disconnect T=1040
205 Req  AS    67 SMB R Tree Disconnected
206 AS   Req   67 SMB C Tree Disconnect T=1840
207 Req  AS    67 SMB R Tree Disconnected
208 Req  EXT   31 LLC D=F0 S=F0 R U UA F
209 AS   Req   67 SMB C Tree Disconnect T=0840
210 Req  AS    67 SMB R Tree Disconnected
211 AS   Req   67 SMB C Tree Disconnect T=3040
212 Req  AS    67 SMB R Tree Disconnected
213 AS   Req   71 SMB C User Logoff and X
214 Req  AS    71 SMB R User Logged-Off and X
215 AS   Req   32 NETBIOS D=89 S=85 Data ACK
216 AS   Req   32 NETBIOS D=89 S=85 Session end
217 Req  AS    18 LLC D=F0 S=F0 R S RR NR=17
218 AS   Req   17 LLC D=F0 S=F0 C U DISC P
219 Req  AS    17 LLC D=F0 S=F0 R U UA F
```

Figure 31 (Part 5 of 5). DatagLANce Trace Output for Complex Logon Scenario

Chapter 4. LAN Server Network Tuning

The amount of traffic on the LAN and the servers ability to handle requests coming off the LAN are critical in determining the overall performance of the server system. A considerable amount of time is spent sending and receiving data at the server depending on the LAN adapter type and protocol used. The optimization of this process can give considerable performance benefits to the server.

This chapter discusses the number of actions that can be taken to improve network I/O and relieve bottlenecks within the server.

4.1 Network Transport Considerations

LAN Server 3.0 requires NTS/2 LAPS. NTS/2 LAPS has a full implementation of the Network Device Driver Interface Specification (NDIS) Version 2.0. This allows a server to run several protocols simultaneously, such as NetBIOS, TCP/IP and SNA. LAN Server 4.0 has MPTS and it is basically the same transport services except NetBIOS over TCP/IP is included and available as an LM10 protocol.

NetBIOS Interface and Protocol

NetBIOS API is a well known communication interface. It provides a communication interface between the application programs thru the physical medium and LAN. A NetBIOS session is a logical connection between two names on the network. A session can be established by issuing a NCB.LISTEN (Station A) and a NCB.CALL (Station B). Once this session is established, two-way guaranteed delivery communication is possible between these stations. Data transfer is possible in two ways:

1. Connection oriented

   Data transfer is provided by the session layer. This is the most reliable connection. If data is lost or some errors occur, NetBIOS takes care of the error recovery and returns an error code to the application through the NCB (NCB_RETCODE).

2. Connectionless

   Data transfer goes directly to the link layer. This is the fastest connection in the NetBIOS environment, but the receipt of data is not guaranteed. This interface is also called Datagram. LAN Server uses this type of connection for example, for its message system or server announcement. The Sideband technology also uses this interface.

In a LAN Server/Requester environment NetBIOS is provided in the LAN Adapter and Protocol Support (LAPS) for OS/2 and in the LAN Support Program (LSP) for DOS/Windows Clients.

NetBIOS interface is described in the *IBM Local Area Network Technical Reference* and it is called as an NB30 interface. LAN Server uses a lower level interface called LM10 which is only available for subsystems. LM10 interface is restricted but it runs on Ring 0 and is faster than NB30. The program component which provides LM10 interface is called NetBEUI. For NB30 and LM10, NetBIOS protocols on the wire is the same.

NetBEUI - Faster NetBIOS

IBM LAN Server (2.0 Advanced, 3.0 Entry/Advanced, 4.0 Entry/Advanced) uses NetBEUI as the transport services. NetBEUI provides APIs for redirector or Ring 0 SMB server. NetBEUI is implemented as a Ring 0 code and gives a much better performance for the server, due to running in the kernel privilege level, and shorter path length than NetBIOS.

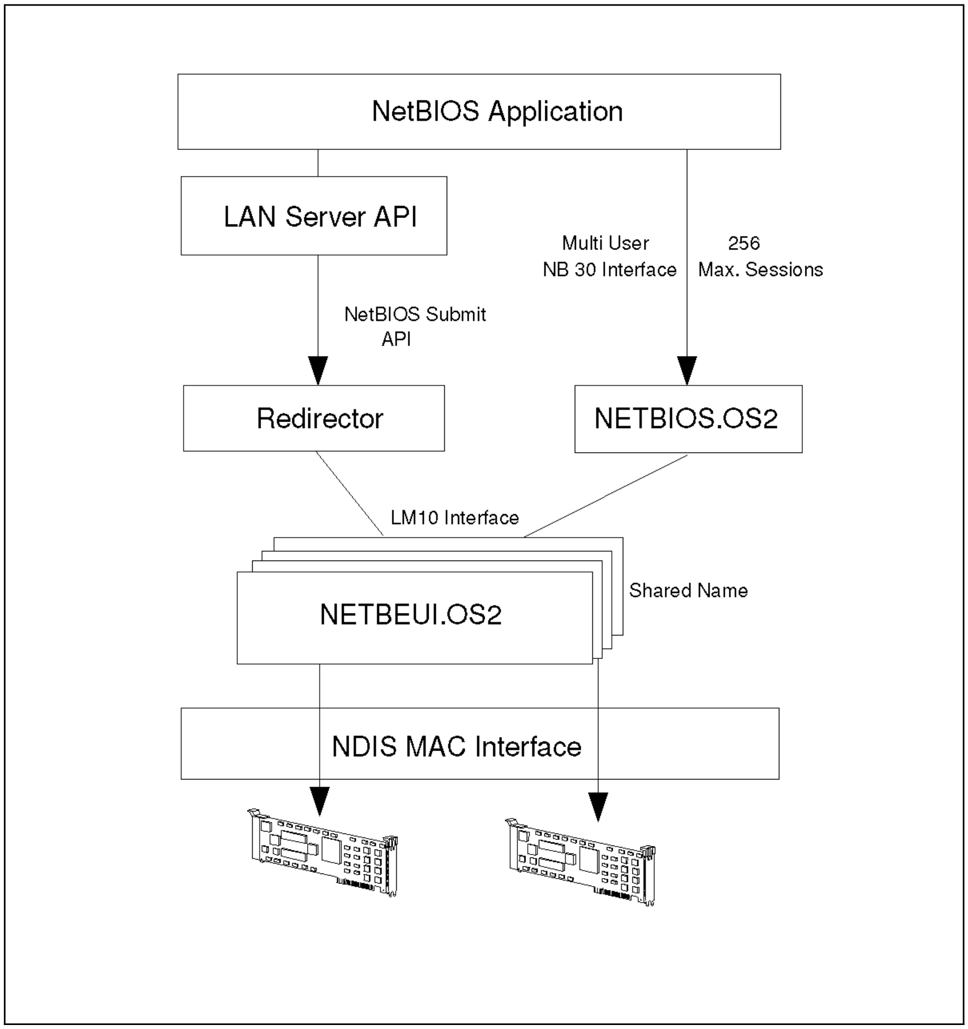

Figure 32. NetBEUI and NetBIOS Relationship

You can have better performance for your system by using:

- Multiple adapters for multiple network applications
- Multiple adapters for a single network application (for example LAN Server)

NetBIOS / NetBEUI Control Block and Limitations

MAC (Media Access Control) data is passed to NetBEUI over the NDIS interface protocol manager. NetBEUI is responsible removing the NetBIOS header information and sending the SMB data to the prime user which is either HPFS386 or redirector NETWKSTA.

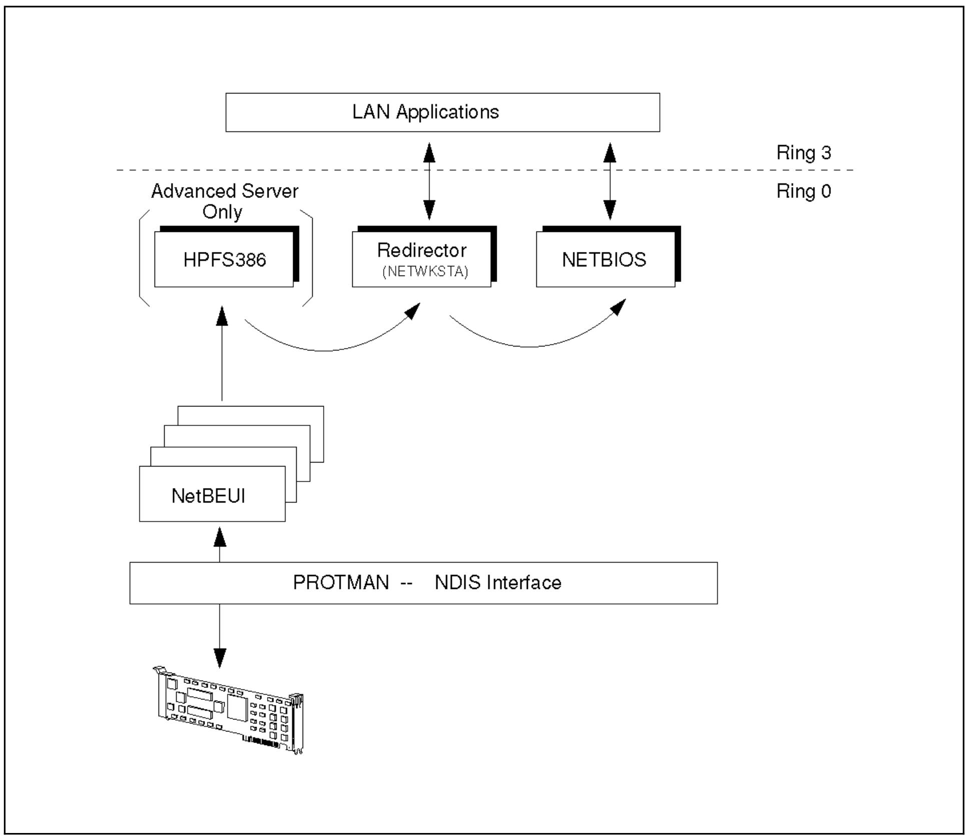

Figure 33. NetBEUI - Single User Interface

NetBEUI is a single-user interface and is limited to seeing only one application. LAN Server extends this limitation to three application interfaces, sitting on top of NetBEUI (HPFS386 / Redirector (NETWKSTA) / NetBIOS). It is important that these three interfaces come to an arrangement. If the Advanced Server is loaded then the 386SMB server (HPFS386) gets priority over the NETWKSTA and NetBIOS. This means, NetBEUI sends the incoming frame to the 386SMB server first, then the frame is routed to the redirector and finally to the NetBIOS API. This rule has general validity, except for acknowledgments that come from these applications.

There is only one 64KB segment for NetBIOS, however NetBEUI can have up to four 64KB segments of memory. LAN Server (3.0 Entry/Advanced, 4.0 Entry/Advanced) allows you to install up to four LAN adapters in the system. Each adapter corresponds to one 64KB NetBEUI segment. Therefore the 64KB limit of the NetBIOS API is restricted now to the segments of the NetBEUI interface, because NetBIOS can map onto each of the NetBEUI segments.

NetBIOS / NetBEUI has some limitations:

- For multiple adapters a corresponding NetBEUI 64KB control block is generated. Each of the control blocks can be configured separately for each adapter.
- LAN Server uses this control information (through the LM10 Interface) to translate any NetBIOS operations.
- There is only one NetBIOS control block and it maps onto each of the NetBEUI control blocks depending on the application being used. Each application should have a fixed adapter number to communicate with.

NetBIOS for TCP/IP (LAN Server 2.0 / 3.0)

NTS/2 LAPS enables you to run both NDIS conformed TCP/IP protocol stack and NetBEUI protocol stack on the same LAN adapter. In addition to the NDIS conformed TCP/IP, NetBIOS for TCP/IP provides NetBIOS emulation on top of the TCP/IP socket interface. NetBIOS for TCP/IP is provided either as a separate product for TCP/IP Version 1.2 or a NetBIOS Kit for TCP/IP Version 2. With a proper system setup, LAN Server and LAN Requester can use both the TCP/IP protocol stack and NetBEUI. The requester can access the servers through the TCP/IP network. This means the servers or the requesters can be on different LANs over a wide area network TCP/IP connection.

A reason for implementing NetBIOS for TCP/IP is to go through the IP router. NetBIOS is not a routable protocol. The internet protocol (IP), sitting on level 3 of the OSI reference model (network layer), has routing information. The NetBIOS information will now be packed into an IP frame and can be routed through a network, as seen in Figure 34 on page 108.

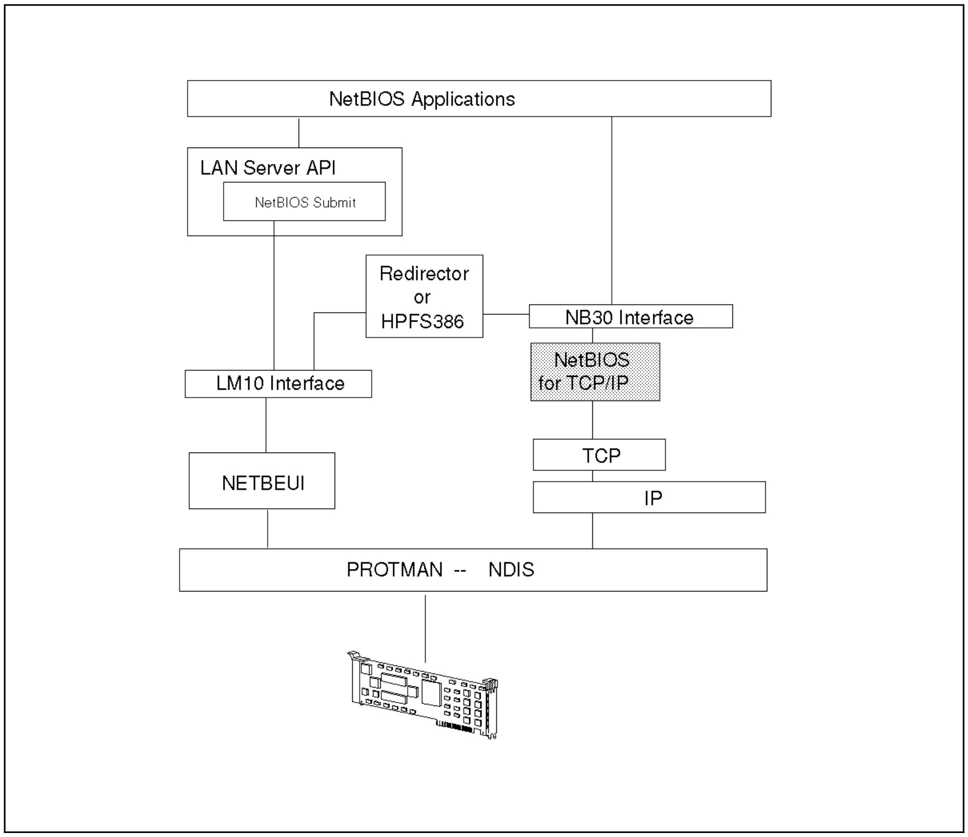

Figure 34. NetBIOS for TCP/IP

IBM NetBIOS for TCP/IP for OS/2 is an implementation of the RFC 1001/1002 NetBIOS and it has been designed to operate with IBM TCP/IP for OS/2. It provides a standard NetBIOS programming interface. Since NetBIOS is an application programming interface (API) and not a protocol, applications using NetBIOS are not bound to a particular network medium or a particular network protocol. However, for NetBIOS to actually move the information through the network, it must use an underlying protocol and the partner application wishing to communicate must use the same protocol. NetBIOS for TCP/IP uses TCP/IP as the transport protocol.

The NetBIOS for TCP/IP uses TCP/IP socket interface to encapsulate NetBIOS frames into TCP/IP packets. The NetBIOS for TCP/IP allows peer-to-peer communication over the TCP/IP network with other computers which have compatible services.

This NetBIOS interface is compatible with IBM′s NetBIOS specification. The NetBIOS program fully supports the Broadcast node (B-node) operation specified by the RFC 1001 and 1002. B-node operation is the simplest and most widely implemented node type for NetBIOS for TCP/IP. It uses broadcasting to exchange information between hosts and works well in isolated LANs. The NetBIOS program also provides extended facilities to allow internet operation through Internet Protocol (IP) routers. These extensions provide a subset of the point-to-point (P-node) functionality specified in the RFCs. P-node operation is used in environments with no broadcasting. The M-node or mixed node operation is a combination of the B- and P-node operation, mentioned above. IBM NetBIOS for TCP/IP is a B-node implementation with routing information.

Two levels of OS/2 interfaces exist for NetBIOS. An application program can use a dynamic link routine interface or a device driver interface. An application program may use either type of interface, but cannot use both interfaces at the same time. Resources provided to and obtained from one of the interfaces cannot be used at the other interface.

In order for an application program to use a device driver interface, the application program itself must be a device driver or have a device driver as one of its components. The application program device driver must be set up to support communication between device drivers. In doing this, the application program device driver can be called by the NetBIOS for the posting of events.

There are several considerations you must resolve before using NetBIOS for TCP/IP:

- An application using the NetBIOS for TCP/IP *cannot* communicate with another application using the native NetBIOS. Even if the applications are written to the NetBIOS programming interface, TCP/IP protocol stacks cannot talk with NetBIOS protocol stacks. A partner must use the same communication protocol stack.
- NetBIOS for TCP/IP replaces the NetBIOS programming interface.

  When you install NetBIOS for TCP/IP, it replaces the IBM standard NetBIOS programming interface called NB30 interface. The NB30 interface is defined in the *IBM Local Area Network Technical Reference*. The NB30 interface is called IBM standard NetBIOS API and there are two level of interfaces: Dynamic Link Library (DLL) level and Device Driver (DD) level. After you installed NBDRIVER.SYS in CONFIG.SYS, all the NetBIOS calls based on NB30 interface are directed to NetBIOS for TCP/IP.

LAN Server/Requester uses a modified NetBIOS interface called LM10 Interface. LM10 is specially developed for LAN Server/Requester to use and is faster than NB30.

NetBIOS for TCP/IP Performance

In comparison to the native NetBIOS transport, the performance using NetBIOS for TCP/IP is generally slow. This is because the workstation must encapsulate the NetBIOS frame from/to the IP frame. This causes extra mapping overhead. There is also a ring transition overhead between Ring 0 and Ring 3 each time a frame is processed by NetBIOS for TCP/IP. This Ring 0 / Ring 3 transition impacts the performance of the workstation. Assuming there is a native TCP/IP communication, the performance using NetBIOS for TCP/IP would be better, because the server code is designed to understand NetBIOS calls.

NetBIOS over TCP/IP of LAN Server 4.0

LAN Server 4.0 along with MPTS has implemented the similar architecture for NetBIOS over TCP/IP but it is more integrated in a Ring 0. It is called NetBIOS over TCP/IP or **TCPBEUI** and is much faster than the previous implementation of NetBIOS for TCP/IP product. In this document we call this function as TCPBEUI.

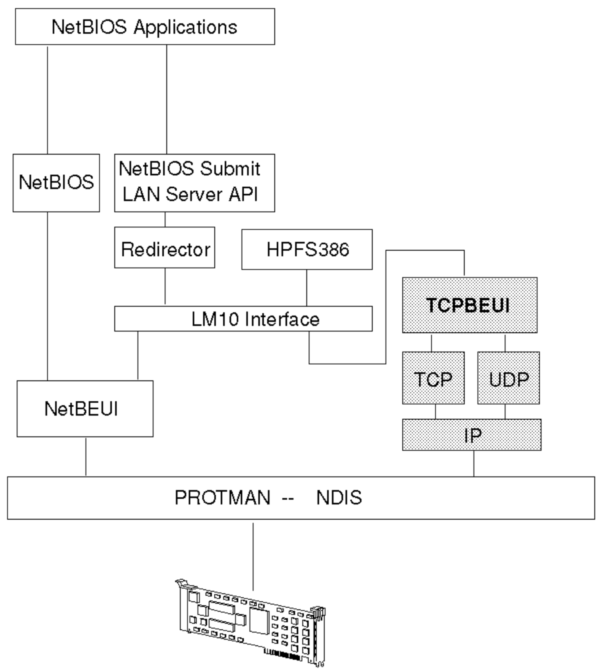

Figure 35. TCPBEUI Layout

TCPBEUI is designed to talk through LM10 interface. It has three different routing extensions to span subnets through IP routers:

- The *Broadcast file* is an ASCII file which contains the IP broadcast addresses. The first broadcast is in the local LAN segment, then the next address from the broadcast file is taken and is sent to the next LAN segment. LAN Server uses broadcast frames, for example to inform the participating nodes in the network about its presence, known as a server announcement. LAN Requester, for example uses a broadcast frame to find out a domain controller.
- The *Names file* is an ASCII file which contains the NetBIOS name and the IP address per line. The example is:

  ```
  CODESRV     9.13.32.79
  ```

  On any discovery operation, TCPBEUI will search the names file first, before broadcasting to the network.

- At the *domain name server* the NetBIOS name can be written in directly with its IP address. This is probably the fastest way.

In comparison to the NetBEUI protocol, Figure 32 on page 105, the TCPBEUI has only one memory block, because TCPBEUI does not have an equivalent function of name sharing to four adapter cards. So TCPBEUI has a maximum limitation of 254 sessions. It has no possibility of balancing the workload over several adapters, because there is a TCP/IP protocol stack between the MAC layer and the TCPBEUI.

The TCPBEUI has its own cache for looking up an address. Ten minutes after booting the system, the TCPBEUI cache dumps its information to a temporary file on the disk. This file is updated every following hour. After the next startup of the system, the TCPBEUI loads the information of the temporary file in its cache. When the TCPBEUI is searching for an address, the TCPBEUI cache is checked first, then the name file and then the broadcast file.

In general TCPBEUI is slower than NetBEUI. It can be compared with the performance of the NetBEUI communication of the Entry server. The performance of the TCPBEUI can be improved by defining a *Names file* on every workstation, so that a requester doesn't need to send any preliminary broadcast frames to locate the destination.

Note

Performance comparison between NetBEUI and TCPBEUI:

- NetBEUI is 30% faster than TCPBEUI
- TCPBEUI is approximately twice as fast as NetBIOS for TCP/IP

These statistics will vary by the nature of the application workload that is present over the wide area network, WAN. In general, the performance is better for sequential traffic, which is normally the case when data is accessed over a WAN, rather than for small random reads.

As a summary, it is not recommended to replace NetBIOS protocol stack with TCP/IP, because of some disadvantages of TCPBEUI. It is OK to have a controlled number of remote LAN users over TCP/IP, but it is not wise to use the TCP/IP protocol within local bridged LANs.

4.2 LAN Adapter Considerations

Essentially the LAN adapter can be compared to a nozzle which physically limits the amount of traffic flowing to and from the server as a nozzle limits the flow of water from a hose. Depending on the number of users, the volume and type of data transactions and other factors, server performance may be limited by the ability of the network adapter. Identification of a bottleneck at the network adapter is not always a simple task. However if you feel the server response time is not what it should be, or if the CPU utilization is high, then this may be the problem. Poor network design or network traffic congestion can have an adverse effect on the server and workstation performance.

Packet Size

Different network topologies, for example token-ring and Ethernet, will use different frame sizes for data transmission. The server software device drivers should be configured so that the transmit buffer size matches the adapter frame size as closely as possible. Under LAN Server, this configuration can be done from the LAPS configuration utility.

A number of factors can effect the transmit buffer size:

- Adapter type
- Ring speed
- Bridge′s maximum frame size
- Receive buffers on partner

Note: The recommended value for the token-ring 16/4 adapter running at 16Mbps is *4224 bytes*, (4096 + 128 byte frame header). This assumes the server and requesters use default value of **sizreqbuf** and **sizworkbuf** which is 4096 bytes. The 128 bytes header length assumption is not aiming for the maximum or worst case, but it is a good value for the average transactions.

The number of transmit buffers should be set to more than two to allow for overlapped buffering. These are specified in the IBM Token-Ring Network Adapter (IBMTOK_NIF) section of the PROTOCOL.INI file, or through the LAPS configuration utility.

LAN Adapter Selection

There are many different types of LAN adapter design, but for file servers these fall into two main categories; busmaster and non-busmaster. The following discusses the benefits of using busmaster LAN adapters, although for small, lightly loaded LANs, non-busmaster LAN adapters should be quite adequate.

In tests completed at IBM LAN Systems Performance LAB results indicated that upgrading from the IBM 16/4 Token-Ring busmaster adapter to the new IBM LANStreamer MC32 gave an increase in performance of 50%.

LAN adapter technology has developed to include new high performance adapters which utilize Asynchronous Transfer Mode (ATM), which will be an important performance enhancement to consider when using future network environments.

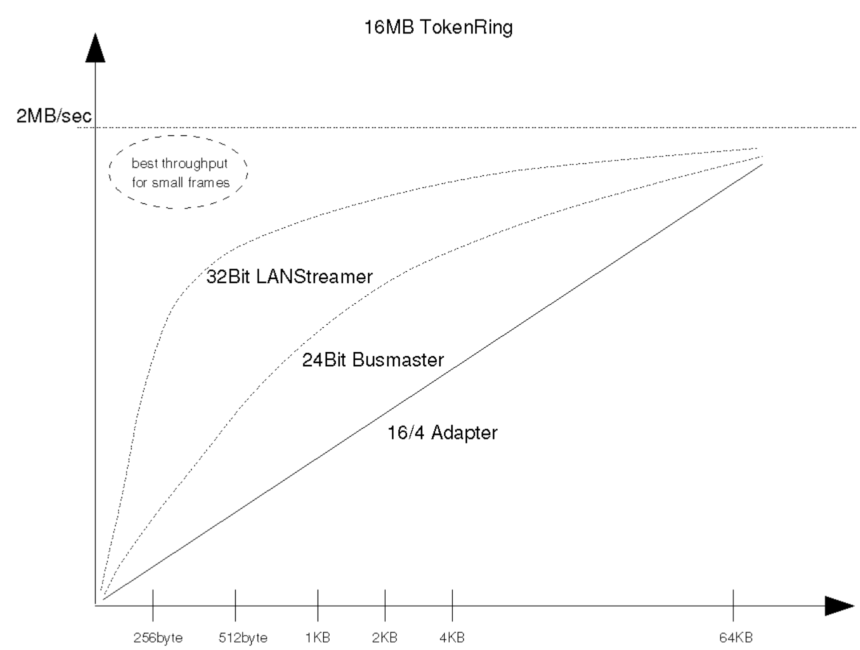

Figure 36. Comparison Between Adapters

Shared RAM Adapter

Shared RAM adapters derived their name from the fact they carry on-board RAM that is shared with the system processor. The memory on the adapter card is *mapped* into an unused block of system memory above the 640KB line in the *upper memory area* on DOS systems. The upper memory area is the 384KB of memory immediately above the 640KB line. The starting address of the shared RAM area is determined by the adapter device driver unless the adapter is an MCA adapter, in which case the address is determined by the reference diskette.

The physical memory in the server that exists at the chosen shared RAM location is automatically disabled by the memory controller and the RAM on the network adapter is enabled. Thus, the server processor can effectively talk to the network adapter by addressing the shared RAM location.

On the server side there is only an address in the shared RAM which points directly to the adapters RAM. Shared RAM can be 8, 16, 32, or 64KB in size depending on which adapter is used and how it is configured. Adapter cards with 64KB support RAM paging which allows the system to view the 64KB of memory on the card in four 16KB pages. This scenario only requires 16KB of contiguous system memory instead of the 64KB required when not using RAM paging. All IBM NetBIOS products support RAM paging.

The main disadvantage of shared RAM architecture is that any data movement between the shared RAM area and other areas of system memory must be done under direct control of the system′s CPU. Move instruction from/to the shared RAM memory address is much slower than the same MOVE instruction from/to the normal system memory area. This means that the CPU may be tied up doing the primitive task of moving data.

For example, a server with multiple shared RAM network adapters can only serve one adapter at a time. Busmaster adapter allows the CPU to be free to initiate disk I/O or to serve other CPU tasks.

This movement of data to and from the shared RAM must be done because applications cannot operate on data while that data resides in the shared RAM area. This obviously has an adverse impact on file server performance, but in small LANs with traditional file serving type workstation applications (word-processing and printing), this is not really a problem. For anything such as database application or 30 users using a variety of applications this can be a major problem. The cost difference in these two types of adapter is very little when compared to the overall investment made in the file server.

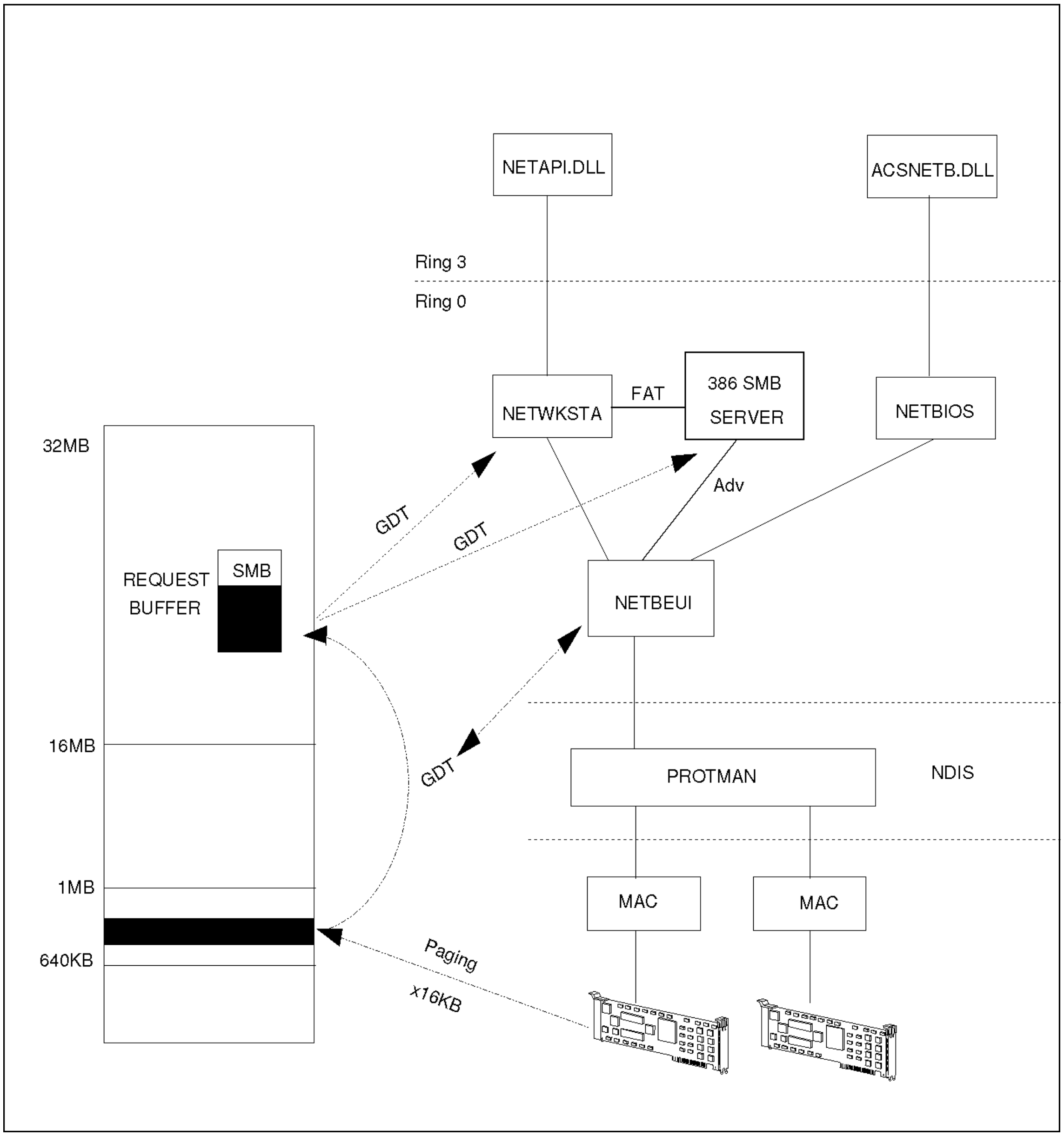

Figure 37. 16/4 Token Ring Adapter Card Data Flow

When using the Token-Ring 16/4 adapter with OS/2 LAN Server, it is recommended that *a multiple of 16KB shared RAM size* should be used for best performance and memory utilization. This can be configured with the system reference disk or DIP switches.

There are a number of parameters that can be used to alter the behavior of the LAN adapter. Typically the shared RAM area itself contains various *status* and *request* blocks, Service Access Points and Link Station control blocks, receive buffers and transmit buffers. These can be defined through the LAN Access Protocol Support (LAPS) utility on an OS/2 LAN Server system.

When data is transmitted from a client to a server the frame will reside in the adapters memory. The system has a pointer to this memory block. For uploading and transforming this data, the NetBEUI gives a memory address using the Global Descriptor Table (GDT). Every SMB block read from the redirector or the SMB server uses the GDT selectors.

Busmaster Adapter

Busmaster adapters utilize on-board Direct Memory Access (DMA) controllers to transfer data directly between the adapter and the system memory without involving the system processor. The primary advantage of this architecture is that it frees up the system processor to perform other tasks, which is especially important in the server environment.

What is the approach for making an adapter faster? There is a 2MB/sec limit on the wire, when the token-ring is running 16Mbit/sec. The best state would be the maximal throughput (2MB/sec) for every frame size. This is not possible, but the busmaster adapter has the peculiarity of a better throughput with small frame sizes than other adapters (see Figure 36 on page 114). This is beneficial for loading applications, spread sheet type applications, logon process and other. In comparison to a 16/4 adapter it is not a big difference when copying large frames over the wire.

24-bit busmaster Adapter Card: The 24-bit busmaster adapter card has the capability to load the incoming data directly into the systems RAM (DMA transfer). The data resides in memory lower than 16MB, because the adapter can only address up to 16MB. It is not beneficial to use 24-bit busmaster adapter cards if any LAN Server memory resources are allocated above 16MB. However memory above this region can still be used by other applications on the server.

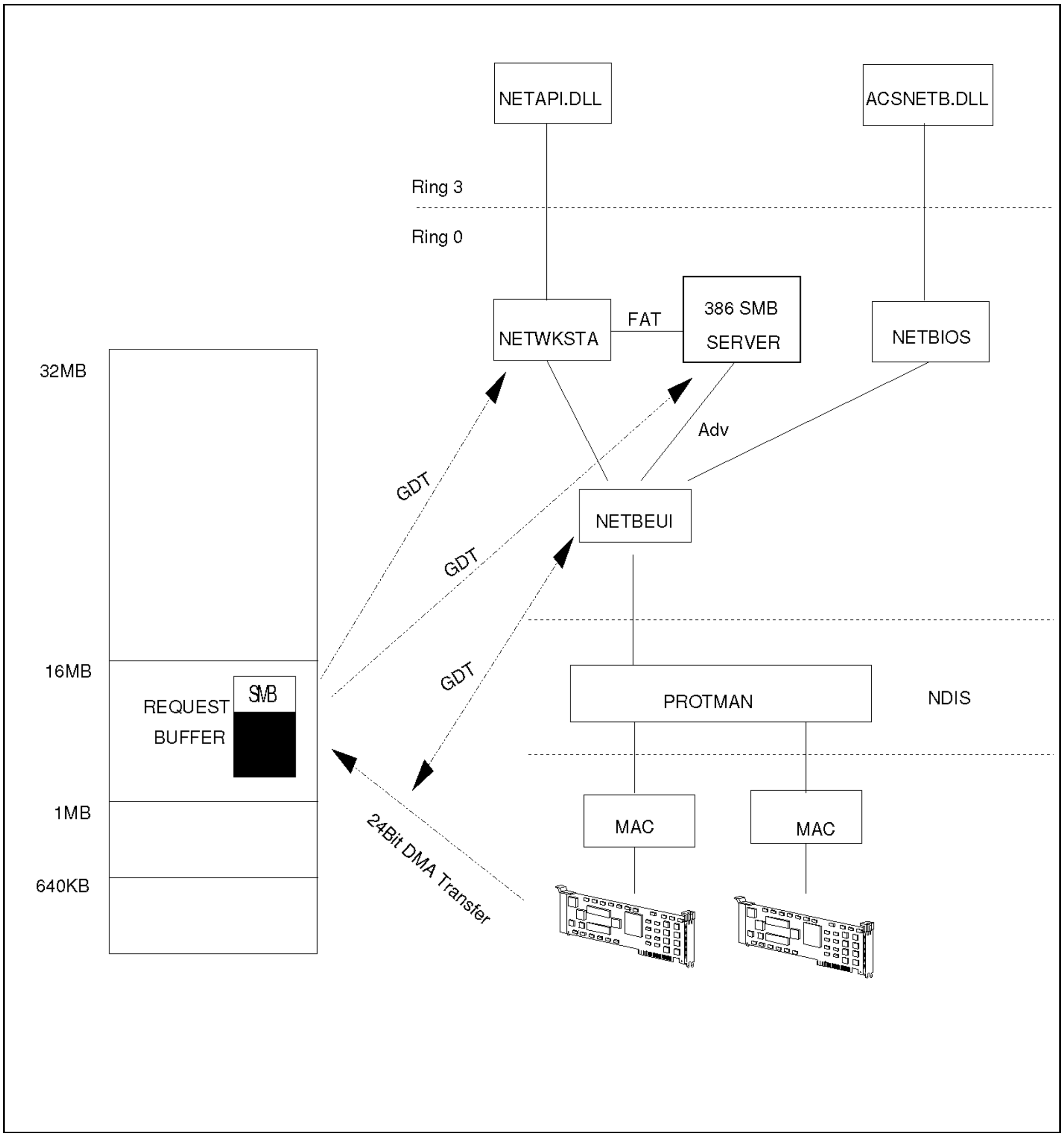

Figure 38. 24-bit Busmaster Adapter Data Flow

This ability to transfer large amounts of data on the bus without direct system processor involvement has led to the term Burst mode data transfer.

The LAN device drivers written for these types of cards can take advantage of the burst mode which, in turn, provides high performance levels on the file server,with data transfers of up to 40MB/sec. One such LAN adapter is the IBM Token-Ring LANStreamer MC32.

IBM LANStreamer Adapter Card

The IBM LANStreamer adapter employs a completely different design to previous IBM LAN adapters. Until the LANStreamer was introduced, all token-ring busmaster adapters employed adapter memory as a frame buffer, typically 64K bytes in size. The buffer was used to assemble frames before they were sent to the server or sent from the server to the network. The time elasticity provided by this buffer allowed the token-ring chip set to complete its processing and forwarding of the frame before the frame was lost - a condition known as overrun (receive) or underrun (transmit).

The LANStreamer utilizes a revolutionary chip set that is capable of processing token-ring frames without using memory as a frame buffer. This chip set also does not incur significant overrun or underrun errors. Therefore, the latency of assembling frames into an on-card buffer is eliminated. Furthermore, adapter memory buffers are now no longer required. This makes the design cheaper to manufacture. This low latency chip set is the key to the small-frame performance characteristics of the LANStreamer adapter. This technology is also used in the design of the EtherStreamer* MC 32 LAN adapter, in that it does not need to buffer the assembled frames into on-card memory, before passing the frames onto the network operating system.

In addition the EtherStreamer LAN adapter supports full duplex mode that allows the adapter to transmit as well as receive at the same time, thus providing a throughput of 10MB/sec on the receive channel and 10MB/sec on the transmit channel, an effective adapter throughput of 20MB/sec. To implement this feature an additional electronic switching unit is required in the installation.

Another characteristic of the LANStreamer adapter could be the higher CPU utilization. The CPU utilization could be higher than slower adapter because the LANStreamer adapter can pass significantly more data to the server than earlier adapters. This corresponds to more frames per second that must be processed by the server network operating system.

A consequence of the high LANStreamer throughput is that the LAN adapter is now not usually a bottleneck in the LAN system. Of course, the network itself can become a bottleneck if throughput requirements overwhelm the ability of the network technology being used. For example, if an application required 3MB/sec of throughput, then token-ring at 16Mbps will not perform up to the task. In this case multiple network adapters or a different network technology must be employed. However, while the LAN adapter itself may no longer be a bottleneck, some other component will emerge in this role as network loading increases.

In the following figure, the LANStreamer adapter transfers the incoming data directly into the system memory above 1MB, without interrupting the CPU. NetBEUI moves this data block into the user′s area, using the global descriptor table selectors (GDT). In this case it is the request buffer. Now the redirector or the SMB server can access the SMB buffer area using the GDT selectors as well. When sending frames to the LAN adapter, NetBEUI can directly send the user′s data in the request buffer to the adapter. This movement is done by the adapter′s DMA function, and this dramatically reduces the server′s CPU utilization. Scatter send is used to the adapter, so the data area doesn′t have to be the contiguous area. This again eliminates the move instruction and extra buffer area.

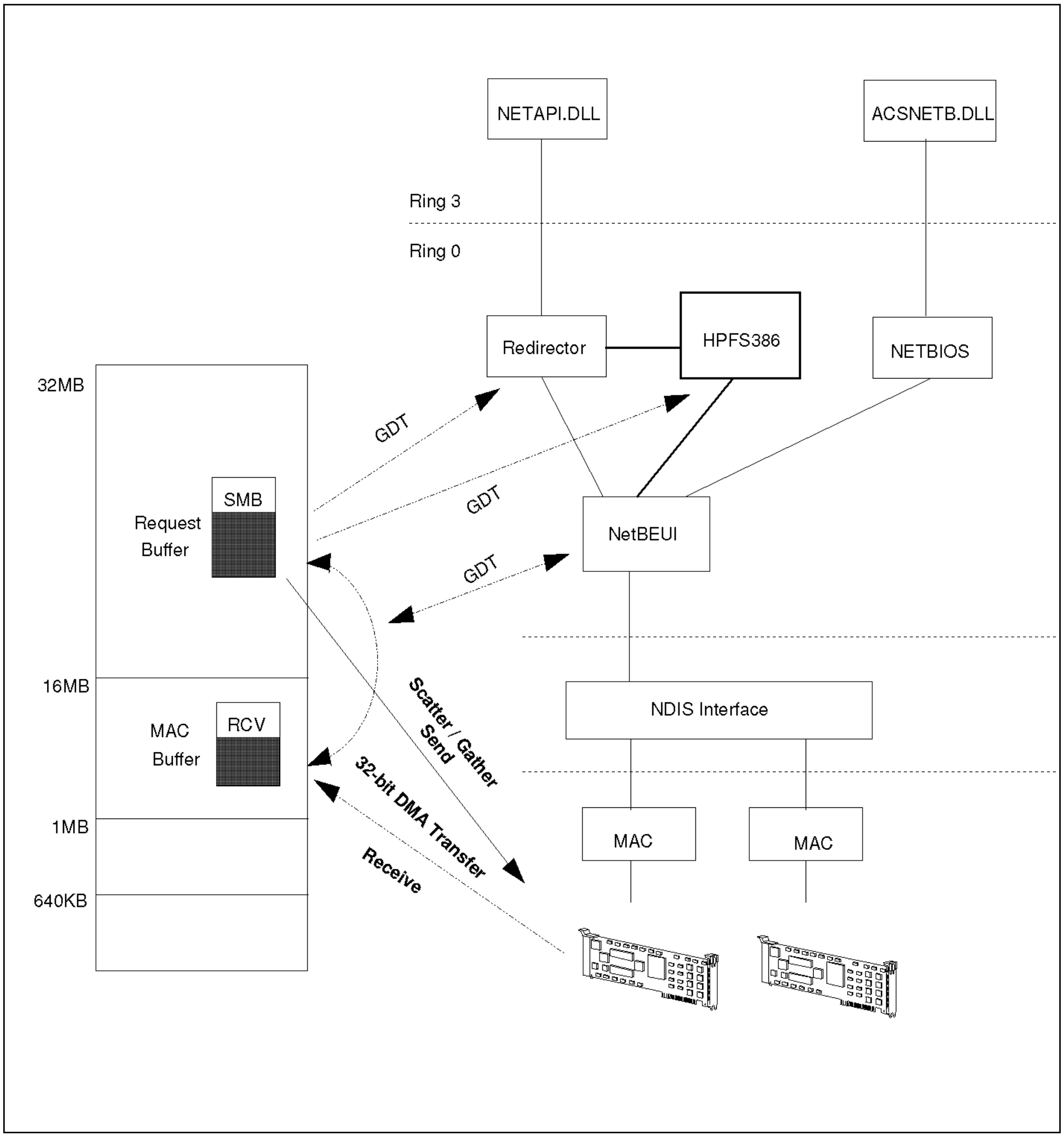

Figure 39. 32-bit LANStreamer Adapter Data Flow

LAN Server and Multiple Network Adapters

The use of multiple network adapters within an OS/2 LAN Server system can offer significant performance improvements. Up to four network adapters may be configured within one system with automatic load balancing across all network adapters at session initialization. Network adapters may be connected to one LAN segment (path A in the Figure 40) or to multiple LAN segments (path B in the Figure 40).

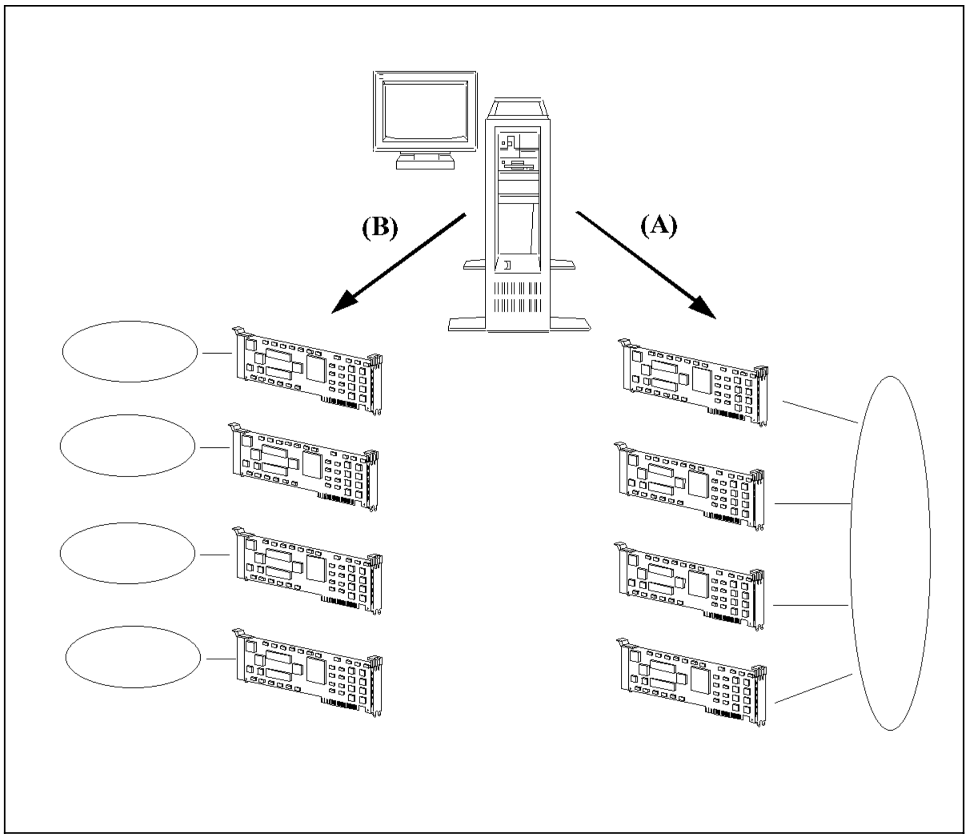

Figure 40. Multiple Adapter Support

However it should be noted that when non-busmaster network adapters are used in this way, the load on the CPU can be so high that the CPU replaces the LAN I/O system as the bottleneck.

Note: As only two non-busmaster adapters can be used at one time on a server then four adapters cannot be connected to the server if they are all non-busmaster adapters.

A faster processor may help in some cases, but a better solution is to use multiple busmaster adapters which will not cause such a dramatic rise in CPU utilization.

4.3 LAN Server Network Tuning Summary

In a network which is subjected to heavy LAN traffic, the load put on the file server can be excessive and cause degradation in performance. When it has been determined that this is the case, adding a faster or additional LAN adapter can help to improve performance. Once it is understood that the LAN adapter is loaded with too much network traffic, and this is not due to any error condition on the LAN, segmenting the LAN topology into multiple segments helps improve performance considerably.

The following points describe the principle considerations that should be understood when dealing with LAN Server network I/O performance:

1. Check that the LAN adapter is not causing a bottleneck. The use of a busmaster adapter, for example, a LANStreamer MC 32 adapter, will improve network I/O considerably while freeing the CPU for other server functions.
2. Limit the use of TCPBEUI for the requesters which only can be reached through TCP/IP routers. NetBEUI is faster than TCPBEUI and TCPBEUI can only support 254 maximum sessions. If you are still using NetBIOS for TCP/IP from TCP/IP Version 2 for OS/2 package, migrate to LAN Server 4.0 with a new MPTS to get TCPBEUI.
3. There are a number of buffers that can affect performance on both the server system and the requester. Using large size of NetBEUI buffer and MAC level buffer, for example 8192 bytes, will improve the file transfer type performance, however it might degrade the response time for other application types and the performance gain is not significant. You could stay with the default value.
4. Use sideband as much as you can, so we don′t recommend to disable Sideband by specifying SIDEBAND = 0 in NetBEUI section. The implementation of the Sideband subprotocol provides performance enhancements for the small size of read/write requests. Rather than sending each read/write request over the network as a separate SMB, Sideband allows multiple requests to be strung together, without the

need for an acknowledgement of receipt of each individual request. Sideband is automatically enabled on LAN Server Advanced installation. If the error rate is detected by the network as being too high, Sideband is automatically turned off. Once the Sideband has been turned off, it can only be activated by terminating and re-establishing the session. The key to use Sideband is to increase MAC frame size both in the server and in the requesters.

Chapter 5. LAN Requester Performance Tuning

In this chapter we will consider actions that can be taken to improve requester performance. This chapter is divided into sections for each requester component: OS/2 requester and DOS requesters (now we have two requester components, DOS LAN Requester for LAN Server 3.0 and DOS LAN Services for LAN Server 4.0).

5.1 OS/2 Requester

OS/2 Requester has two types of buffers, a work buffer and a big buffer. You need to know the function of these buffers before you change the default parameters. Do not change these parameters unless you have made changes to the server′s parameter, **sizreqbuf** and network parameters discussed in 3.5, “LAN Server Buffer Tuning Summary” on page 68.

Buffers Tuning Parameters

OS/2 Requesters use two types of memory spaces to provide data buffering. They are work buffers and work cache. The following parameters for each buffer are specified in IBMLAN.INI file.

Work Buffers

The **numworkbuf** parameter sets the number of buffers the requester can use to store data for transmission both to and from the server. These buffers are used in constructing the SMBs that are sent to the server. They also provide data buffering between applications running in the requester and the network adapter card.

This buffer is used effectively in the following cases:

- Read-ahead (for small sequential reads)
- Write-behind (for small sequential writes)
- Search-ahead
- Lock-behind

If a requester is running multiple applications, or accessing multiple files sequentially or simultaneously or using remote IPL service on the server then increasing this value may improve requester performance. If however the requester is not running multiple OS/2 applications then this value could be reduced to 5 to conserve memory, without a large degradation in performance. You can use the NET STATISTICS command to check if work buffers are exhausted or not. (Request buffers is the name returned from this command line interface.)

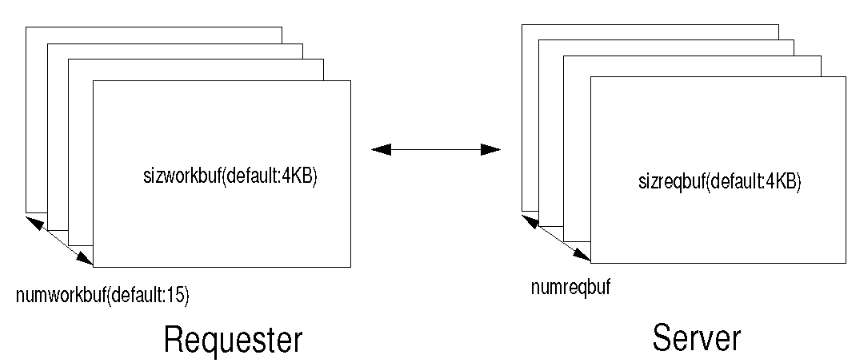

Figure 41. OS/2 LAN Requester Work Buffer Parameters

The **sizworkbuf** parameter, which specifies the size of the work buffer, should be set equal on all requesters on the network and equal to **sizreqbuf** on all the servers. If they are not matched, the memory in the larger of the mismatched buffers is wasted. There are situations in which the 4KB default value for these buffers is not optimal. For example, if the network administrator monitors the traffic on the ring and determines that most frames are less than the transmit buffers on the requesters and servers. In this example, the server and requester buffers could be reduced, providing a memory reduction and they can increase the number of buffers.

Work Cache Buffers

The **maxwrkcache** parameter specifies the maximum size, in KB, of the requester′s large-transfer buffers. The work cache memory space is used for receiving large data transfers from big buffers on the server. Data transfers to and from the server that are greater than the size of a work buffer use the work cache. The value of this parameter must be a multiple of 64 in order to match the size of the corresponding big buffer on the server. Increasing this parameter may only be effective if the requester is running multiple applications which are file intensive, for example copying large files to/from the server. If this is not the case then do not change this value, since each increment of this value uses up 64KB of memory. The NET STATISTICS requester command is also available for this buffer.

Note

Some applications will use their own application buffers for large reads and writes instead of using work cache.

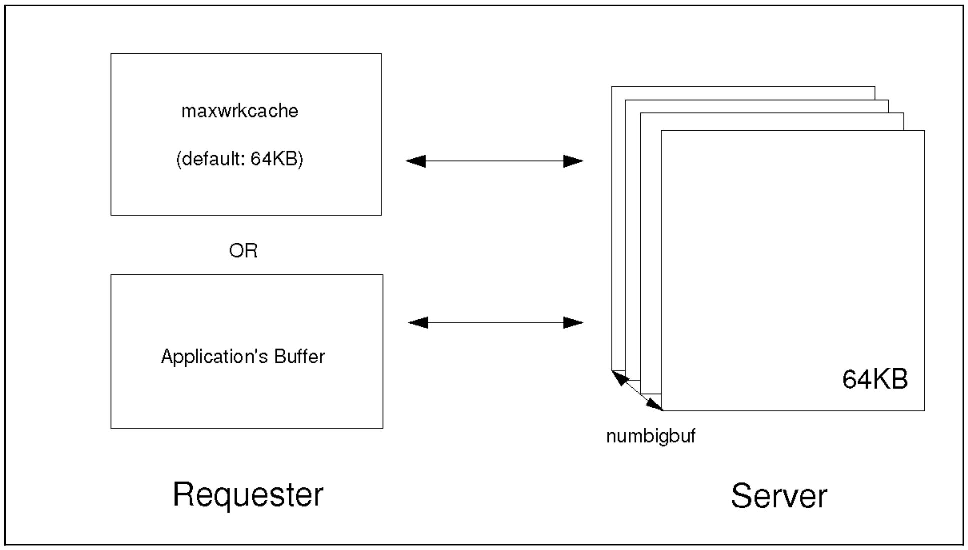

Figure 42. OS/2 LAN Requester Big Buffer Parameters

Reducing Memory Requirement

Following are the recommendations for reducing the memory size for OS/2 requesters:

- Reduce **numworkbuf** to 5 if not running multiple OS/2 sessions
- Reduce **sizworkbuf** to 2KB, if small data I/O is the main application on the requester.

- Comment out **netbios.os2** in CONFIG.SYS, if no other NetBIOS applications are running.
- Set **protectonly** statement in CONFIG.SYS to yes, if DOS support is not required.
- Set **memman** to **swap**, **move** (in CONFIG.SYS).

5.2 DOS LAN Requester

The default values set at DOS LAN Requester installation have been chosen to provide good performance for most users, with minimal memory requirements. You need only change these parameters if a performance problem is identified.

Buffers Tuning Parameters

DOS LAN Requesters use two types of memory spaces to provide data buffering. They are network buffers and big buffers. The following parameters for each buffer are specified in DOSLAN.INI file.

Network Buffers

The DOS LAN Requester uses these buffers to construct the SMBs sent to the server.

The **/NBS** parameter, whose default value is 1KB, specifies the size of request buffers. Setting **/NBS** to 4KB will normally match the value of the requester buffers in the server providing efficient data transfer. However, it may be necessary due to memory availability to reduce the work buffer size and **sizreqbuf** on the server to provide optimum performance.

The **/NBC** parameter defines the number of network buffers on the DOS requester and should not be set below the default value. If you increase the value of **/NBC** parameter, you should also increase the value specified for the NetBIOS command resource. See Figure 43 on page 129.

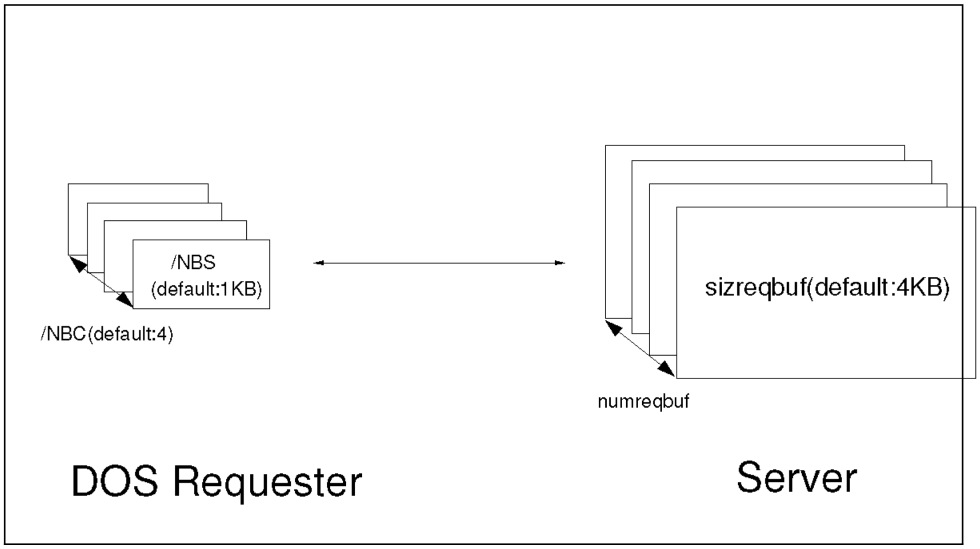

Figure 43. DOS LAN Requester Request Buffer Parameters

Use the following formula to obtain the required memory for request buffers.

(**/NBC** x (**/NBS**+70))

70 is a header size for the request buffers.

Big Buffers

The big buffers are only important if applications are performing large sequential writes to the server (printing or large file transfer). This plays a less important role in the DOS requester than workcache does in the OS/2 requester. They are used for transfer greater than **NBS** but less than or equal to **BBS**. The DOS requester uses a User Memory Transfer option that can handle the transfer of large files without permanently allocating system memory to this function as is required by big buffers.

When the data being requested is greater than the size of the big buffer, the User Memory Transfer function moves the data straight from the network adapter card into the user memory space allocated for data, bypassing the big buffers. If the network buffer size is set to 4KB, then it is worth setting the number of big buffers (**/BBC**) to zero to conserve memory. Big buffers′ principal function is to enable a user to keep the network buffer size (**/NBS**) small while retaining one large buffer to handle applications with a common data block size similar to the size of the big buffer. In most environments it is recommended to set **/NBS** to 4K and disable **/BBC** to save memory while maintaining network performance. For the environment where memory is very important, set **/NBS** to 1K and set **/BBC** = 0.

Use the following formula to obtain the required memory for big buffers.

(**/BBC** x (**/BBS**+70))

70 is a header size for the big buffers.

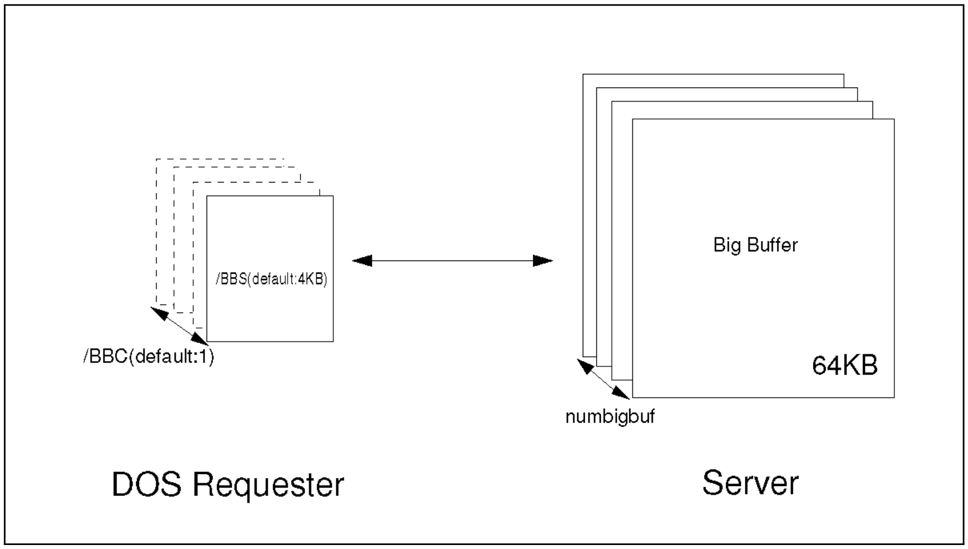

Figure 44. DOS LAN Requester Big Buffer Parameters

NetBIOS Tuning Parameters

If you increase the value of the **/NBC** parameter, you should also increase the value specified for the NetBIOS commands resource. The minimum NetBIOS requirements are:

- For a redirector:
  - Sessions = **/SVR** + 1
  - Commands = **/NBC** + 5
  - Names = 6 (+1 for remote IPL)
- For a requester:
  - Sessions = **/SVR** + 2
  - Commands = **/NBC** + 8

- Names = 8 (+1 for remote IPL)

When you use the standard DOS NetBIOS (DXMT0MOD.SYS) these parameters can be changed in the CONFIG.SYS statement. When you use DOSBEUI with NDIS drivers (DXMJ0MOD.SYS) these parameters can be changed in DXMJ0MOD_MOD section of PROTOCOL.INI file.

Reducing Memory Requirement

In a DOS environment memory is normally scarce and allocation of memory to meet both LAN and application demands is complex. However, there are some actions to minimize the memory requirement.

- Load the client components in high memory on i386 (or higher) workstations. Use one or more of the following parameters from DOSLAN.INI: **/HIM**, **/UMB**, or **/EMS**.
- Set the **lastdrive** parameter in CONFIG.SYS to the minimum drive letter. Each unused drive letter requires approximately 80 bytes of memory.
- Use the standard DOS NetBIOS (DXMT0MOD.SYS) instead of DOSBEUI with NDIS drivers(DXMJ0MOD.SYS).This must be specified in the DXMAID configuration within the LAN Support Program.
- Adjust the request buffer (**/NBS**) and big buffer (**/BBS**) parameters for optimal use of system memory.

5.3 DOS LAN Services

DOS LAN Services is a new code base which provides many new features including peer file and print sharing, reduced RAM requirement for both DOS and Windows, a new Graphical User Interface (GUI), as well as several other significant enhancements. DOS users will observe a significant performance improvement over past network experience in many of their applications. This is accomplished primarily by retaining information read from the server in workstation buffers more effectively.

DLS Buffers Tuning Parameters

DOS LAN Services use two types of memory spaces to provide data buffering as a requester. They are work buffers and big buffers. Peer server function uses transmit buffers. These parameters are specified in the NETWORK.INI file. DOS LAN Services can use big buffers as cache space and also has a read-ahead function. It provides a significanct performance imporovement compared with DOS LAN Requester, especially for random read access.

Work Buffers

DOS LAN Services uses these buffers to construct the SMBs communicate with the server. The **numworkbuf** parameter specifies the number of work buffers. Changing the size and number of these buffers may or may not improve performance. This is because the big buffers effect may override the work buffers.

Big Buffers

DOS LAN Services uses big buffers for two purposes, large file transfer and file caching. The **sizbigbuf** parameter specifies the size of big buffers and the **numbigbuf** parameter specifies the number of big buffers. DOS LAN Services uses a User Memory Transfer option for the request beyond the big buffer size. However, big buffers should still be configured because they are used as the cache space.

Using a cache at the requester provides good influence to the server′s activity because it reduces the workload of the server. DOS LAN Services can keep the data in big buffers as a cache area. You do not need to change the values for big buffers. If you specify the **autocache** parameter to Yes (Yes is a default setting), DOS LAN Services automatically allocates **numbigbuf**, **sizbigbuf** and **extraheap** based on the amount of XMS memory and overrides these parameters′ value specified in NETWORK.INI to provide suitable parameters for each machine to improve its performance.

Note

DOS LAN Services does not have a **wrkheuristics** parameter like DOS LAN Requester, so all DOS LAN Service requesters are affected by the configuration of the **srvheuristics** setting on the server.

Transmit Buffers

DOS LAN Services uses transmit buffers for transferring data for Peer Service. This size should be set to the same size as **sizbigbuf** on DOS requesters and **sizworkbuf** on OS/2 requesters.

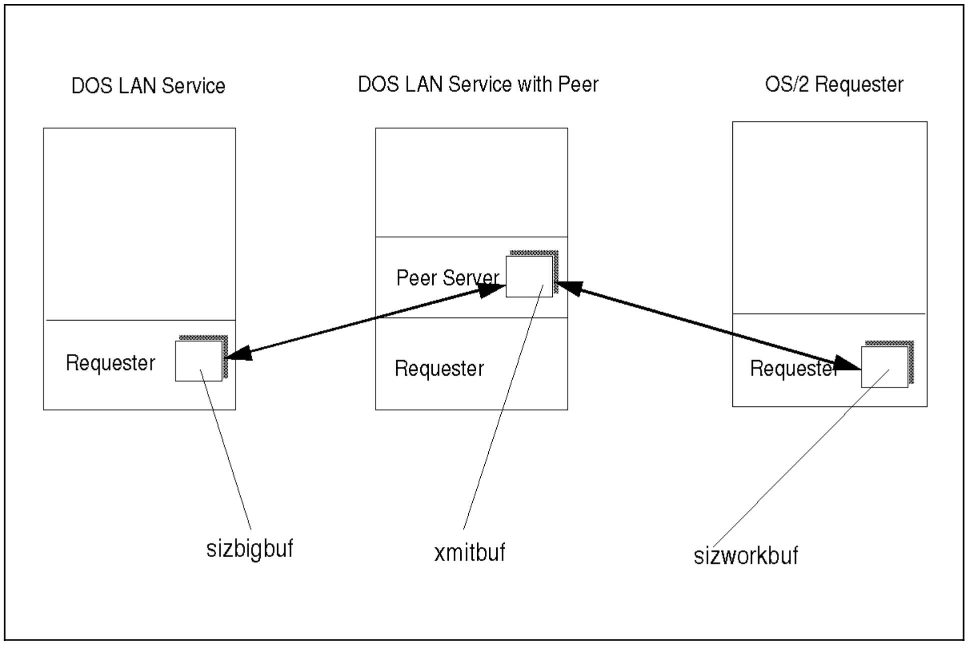

Figure 45. Peer Requesters' Buffers

DLS NetBIOS Tuning Parameters

DOS LAN Services use the NetBEUI driver internally. This driver is the same as DXMJ0MOD.SYS and it also has an unload function. You can save your memory space for NetBEUI until you load DOS LAN Services. You can specify parameters for this NetBEUI driver as well as DOS LAN Requester in PROTOCOL.INI.

Reducing Memory Requirement

DOS LAN Services has the useful memory reduction function called Virtual Redirector, that can be used with Windows 3.1.

- Configure DOS so that it can use UMB and XMS. DOS LAN Services will load some components to UMB first, and load the rest of the components to XMS and then it use the conventional memory area. DOS LAN Services doesn′t use EMS.
- Set **autocache** to **yes**. It automatically allocates **numbigbuf**, **sizbigbuf** and **extraheap** based on the amount of XMS memory.

> **Note**
>
> If memory is more important than performance set **autocache** to **no** and let **sizbigbuf** and **numbigbuf** default.

- Use the virtual redirector function when you use DOS LAN Services with Windows 3.1. The redirector can run as a virtual device driver thus not using any of the low memory.
- Set the **lastdrive** parameter in CONFIG.SYS to a minimum drive letter. Each unused drive letter requires approximately 80 bytes of memory.

Chapter 6. Further Analysis of Tuning

This chapter discusses the issues related to performance tuning a server for various specific environments. Included within this chapter is a section that provides an example on how to tune an application and details the necessary performance and analysis tools required for this process.

6.1 Performance and Analysis Tools

When tuning an application it is necessary to identify the behavior of the network workload. This can only be achieved by using appropriate tools that provide a detailed representation of the frames that are being sent. The following utilities are examples of performance / analysis tools that can be used to achieve a comprehensive overview of the data transfer and the effects it causes on the server.

- DatagLANce - operates at the physical layer and monitors the actual frames being transmitted on the network.
- SMB Trace - operates at the transport layer and monitors the SMB message blocks that are being transmitted on the network.
- System Performance Monitor/2, SPM/2 - operates at the application layer and monitors the behavior of the server as the server functions.

DatagLANce

You can use the DatagLANce Analyzer to get an accurate picture of the current activity on your network or a historical record of network activity over a specified period of time. You can design your own screens and save them from the menu. If you wish, you can create 32 different bar charts of real-time statistics, Or launch an analysis session and call up frame summary views, protocol interpreted views, and hexadecimal views-all color coded, highlighted, and tracked simultaneously. Rearrange the statistics, add to them, and save them under a new name. Use the DatagLANce Analyzer′s alarms to let you know when certain statistical thresholds, like network utilization, are reached. While the DatagLANce Analyzer is monitoring your network, you can even use the Personal System/2 (PS/2) computer for other applications because it is based on OS/2. The following is the summary of the features:

- 88 source and destination address pairs filtered in real time
- Real-time frame capture while monitoring
- Eight fast, super-powerful, programmable event detectors
- Flexible, user-definable interface
- Reliable, accurate information
- Continuous reports of top talkers, ring map (token-ring only), error conditions, statistics, and selected network data
- Broad data import and export support
- Extensive protocol decode coverage
- 10-millisecond time stamp (token-ring) or 32-millisecond time stamp (Ethernet)
- Optional 840-nanosecond, high-resolution time stamp
- Fully windowed, graphical, multitasking user interface

This release of the DatagLANce Analyzer supports all or portions of the following protocol suites:

- FDDI Protocol Suite
- Token-Ring Protocol Suite
- Ethernet/802.3 Protocol Suite
- IBM Protocol Suite
- TCP/IP Protocol Suite
- SUN NFS** Protocol Suite
- XNS Protocol Suite
- Novell** NetWare** Protocol Suite
- DECnet** Protocol Suite
- AppleTalk** Protocol Suite
- Banyan** VINES** Protocol Suite
- ISO Protocol Suite
- X.25 Protocol Suite

You can see the trace file of DatagLANce in the following section. This is a very useful trace tool for LAN Server, because it can interpret not only NetBIOS commands, but also some SMB commands. You can save the captured trace as a different format such as TAP (Token-Ring Trace and Performance Program) or Sniffer and analyze them.

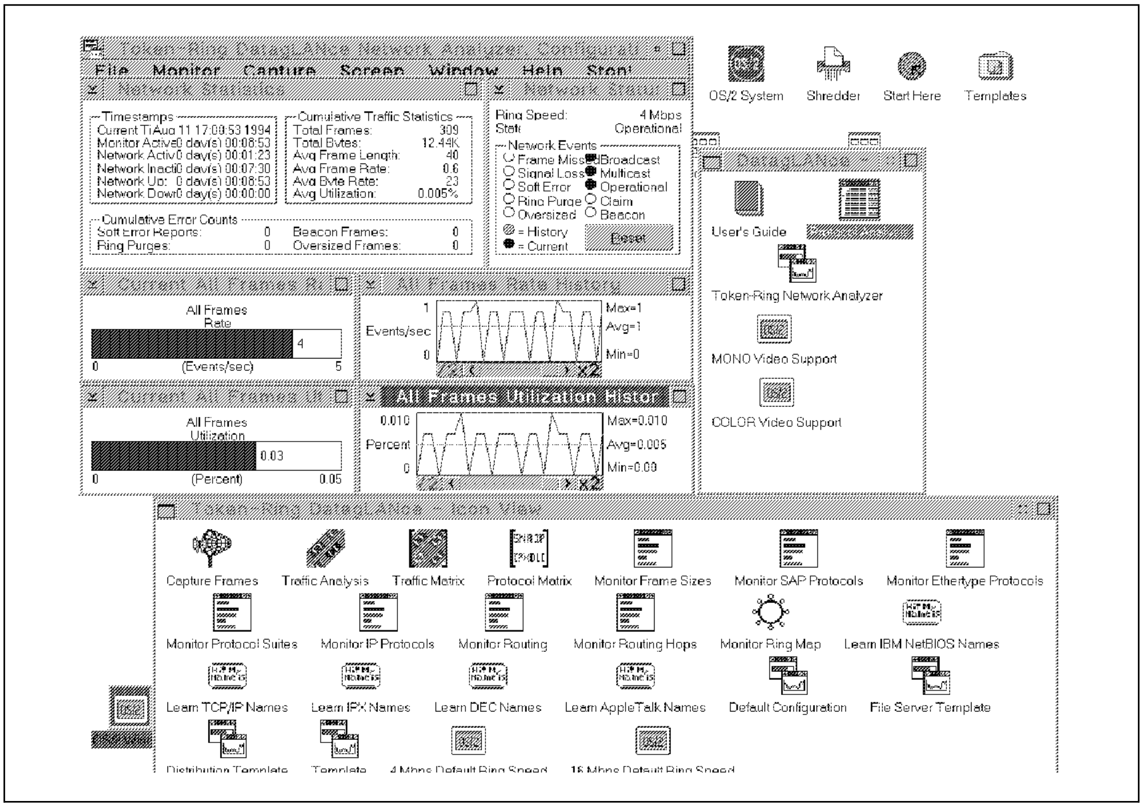

Figure 46. DatagLANce

SMB Trace Tool

This Presentation Manager* utility, SMBTOOL.EXE can be used to capture various types of network traces. It is very useful to capture SMB flows in the server or in the requester. It also includes extensive formatting and filtering features for SMB trace data. It can be used for advanced problem determination. It can analyze TAP (Trace and Performance) format traces and also interpret some SMB that DatagLANce cannot understand. The utility takes no parameters and requires OS/2 2.1 or later.

It is zipped in LAN Server 4.0 Productivity Aids 1 as the SMBTOOL.ZIP file.

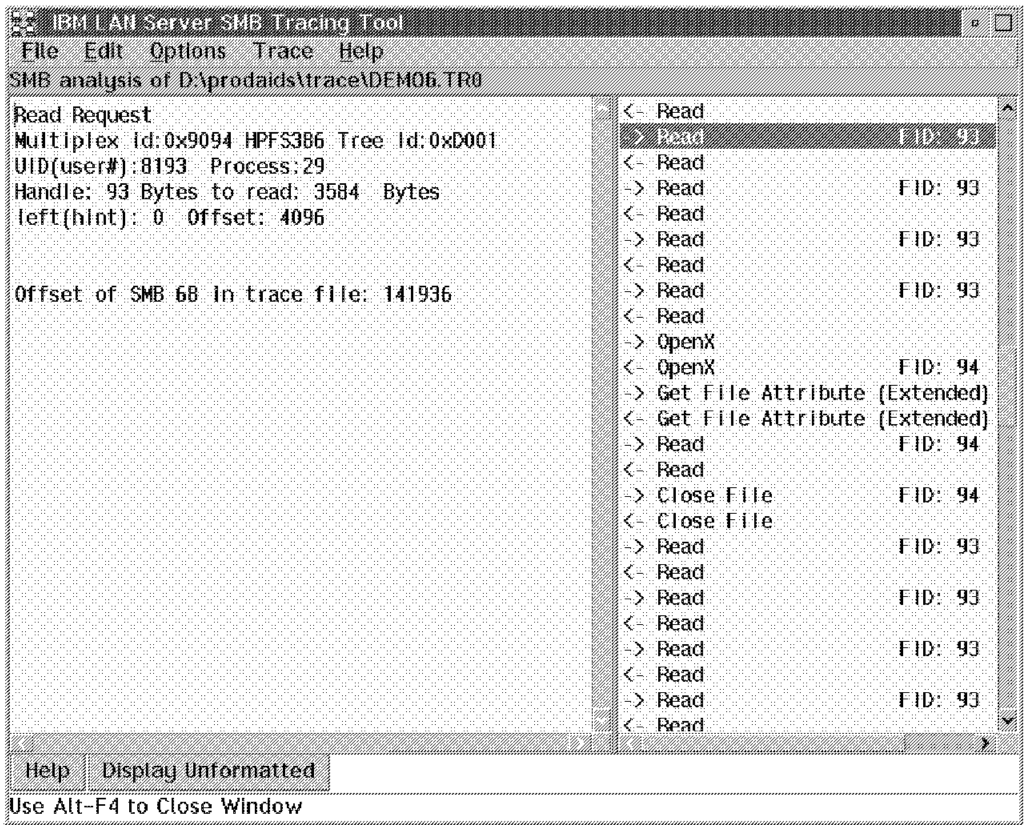

Figure 47. SMB Tracing Tool

SPM/2

The IBM System Performance Monitor/2, Version 2.0 (SPM/2) application is designed to assist in analyzing the performance of the hardware and software in an OS/2 environment. It is useful to system administrators who must manage resources closely to achieve greater efficiency from their servers. The type of information generated is vital for administrators to assess the current performance of a particular server and help identify any performance changes when performance related modifications are done.

Introducing System Performance Monitor

The SPM/2 application provides the following features:

- Data Collection Facility
- SPM/2 Monitor
- Logging Facility
- Report Facility

Data Collection Facility: This collects information about the system resource usage. The information can be displayed with the SPM/2 Monitor in a Presentation Manager window as a graphic, real-time output, or the information can be logged to a file using the Logging Facility.

SPM/2 Monitor: This is a Presentation Manager application that shows utilization of critical system resources. It assists network administrators in monitoring resources such as the central processing unit (CPU), fixed disks, and random access memory (RAM) activity. A resource monitor displays one or more resource data lines for each of the resources being monitored. Each resource data line shows the percentage of activity (for CPU, disk, or memory usage) on the vertical axis of the resource monitor. The data is summarized in continuously updated graphs on the display.

Logging Facility: This accesses the data collected by the Data Collection Facility and writes the data to log files. The log files are then accessed by the Report Facility to generate a variety of reports according to a format desired by the administrator.

Report Facility: This produces reports from collected performance data related to the CPU, fixed disks (physical and logical), RAM activity, and memory swapping. The reports provide more detailed information than is provided by the SPM/2 Monitor, and they may be studied by administrators to analyze any server performance problems. This can be represented in one of three report formats - summary, tabular or dump.

A useful feature for LAN administrators that is built-in to SPM/2 is the Distributed Feature. This allows an administrator to monitor the performance of multiple servers located on the LAN from a single workstation. In addition, this ensures that the bulk of the processing is executed on a requester rather than on the server itself. Essentially data is sent from the Data Collection Facility on a monitored LAN Server to the Logging Facility or SPM/2 Monitor on a managing LAN Requester. SPM/2 supports data collection from remote systems using a remote named-pipe interface.

Remote Server Monitoring with SPM/2

The SPM/2 Distributed Feature allows an administrator to execute the Data Collection Facility on an OS/2 LAN Server and execute the Logging Facility or the SPM/2 Monitor on a remote OS/2 LAN Requester that acts as a managing system. This minimizes the overhead of the Logging Facility and SPM/2 Monitor on the LAN Server which ensures that SPM/2 has no effect on the overall performance results.

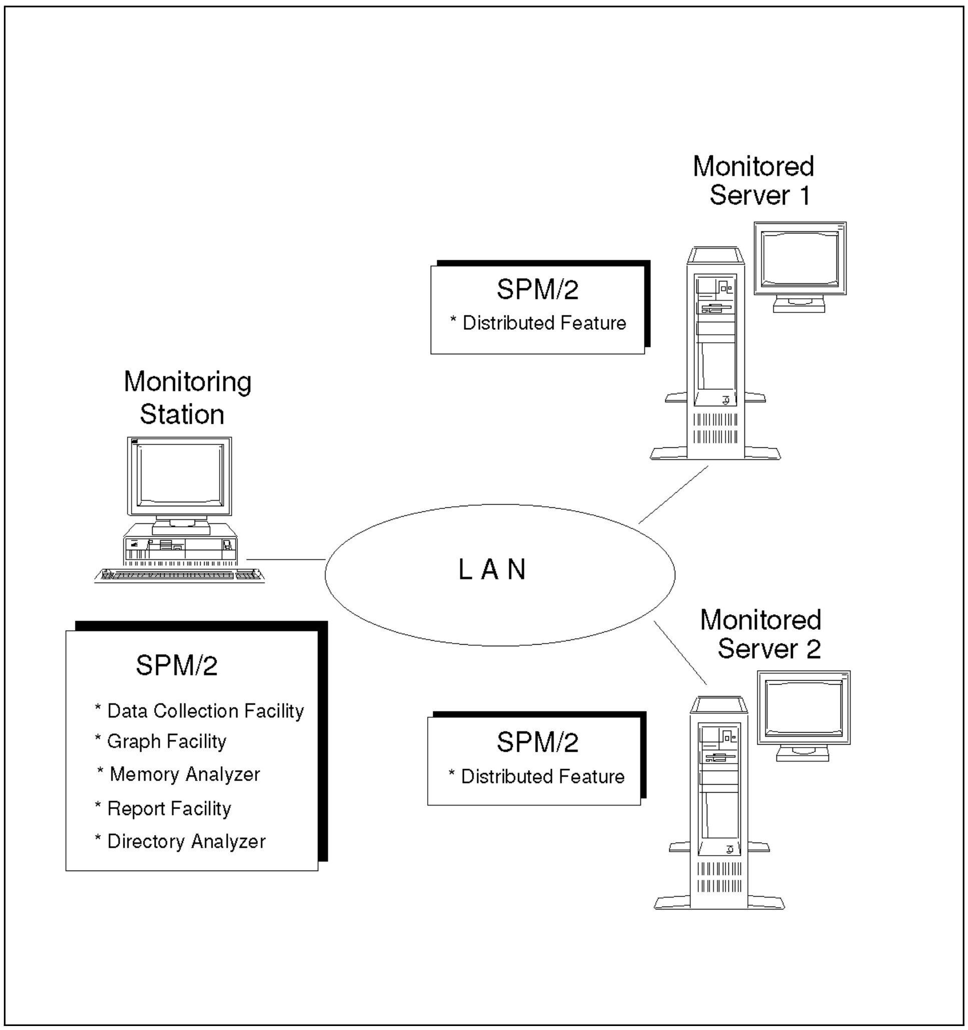

Figure 48. SPM/2 Distributed Feature

To perform remote monitoring, the OS/2 LAN Server or Peer Requester requires either an additional copy of SPM/2 or the SPM/2 Distributed Feature.

In order to use this remote feature, the Data Collection Feature must be started on the LAN Server before the Logging Facility or SPM/2 Monitor is started on the LAN Requester (the managing system). Then, on the managing system, the SPM/2 Monitor can be initiated from the OS/2 command line or the Presentation Manager interface. The SPM/2 Monitor window opens and establishes communication with the server. Several SPM/2 Monitors can be opened concurrently, with each SPM/2 Monitor monitoring a separate server. The SPM/2 Monitor title bar displays the server name to distinguish which server is being monitored. In addition to this, the LAN Requester must be logged on to a domain and must have authority to access named pipes on the LAN Server to be monitored (access is enabled by the system administrator).

When installing the SPM/2 application on an OS/2 LAN Server and performing remote monitoring, increase the value specified by the **srvpipes=** parameter in the IBMLAN.INI file (located in the IBMLAN directory). SPM/2 requires three additional named pipes to function. Also the NetBIOS resource parameters Maximum Sessions, Maximum Commands and Maximum Names have to be modified respectively.

Using SPM/2

The following is the examples of the monitor tool for CPU, DISK and memory.

SPM/2 CPU Monitor

The CPU Monitor measures total CPU usage by plotting the percentage of time that the CPU is busy against the total CPU time. If CPU activity remains high for an extended period of time, the system may have too many CPU-intensive tasks running at once. This can slow the system response time. If CPU activity is low, the system may not be heavily loaded and can probably handle more tasks.

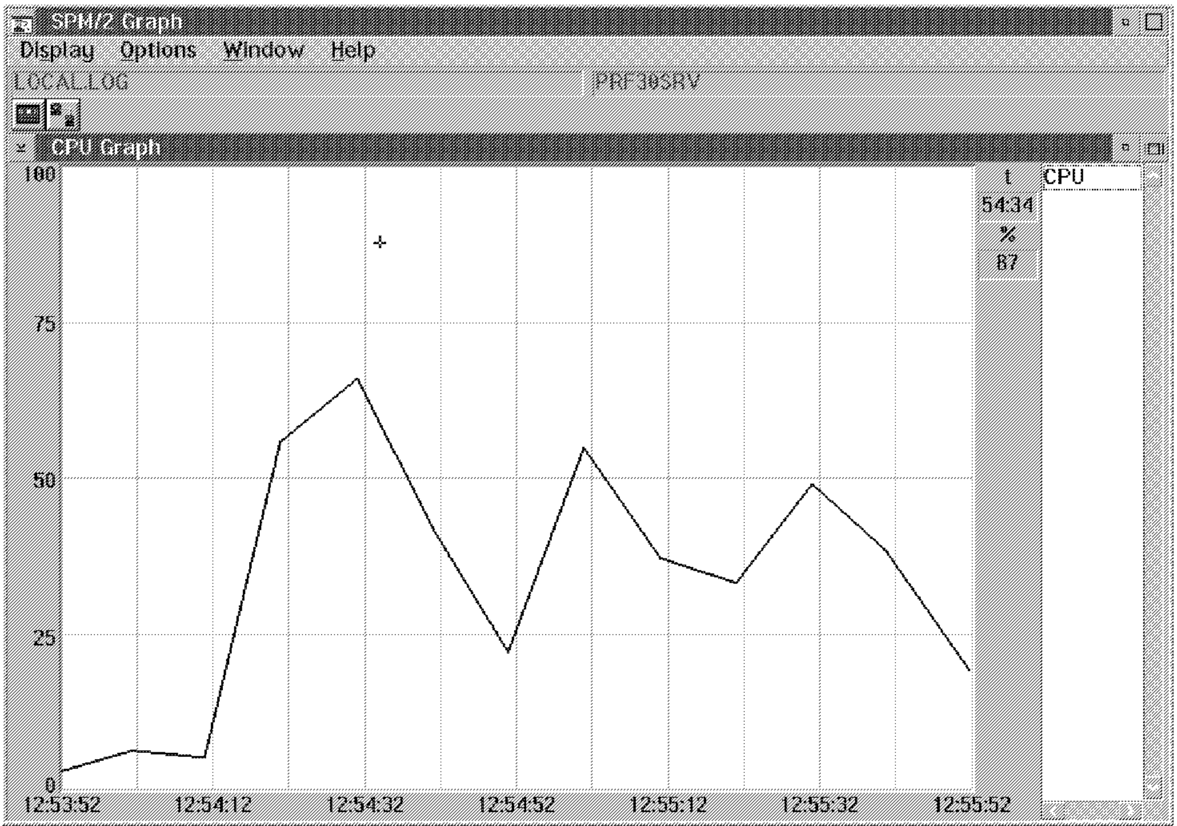

Figure 49. CPU Monitor

SPM/2 Disk Monitor

The Disk Monitor measures the percentage of time the disk device driver is handling requests. The Disk Monitor displays a data line for each physical disk in the system. If the disk activity remains high for an extended period of time, processes may have to wait significant periods of time in a queue to access the disk before continuing. This can slow system response time.

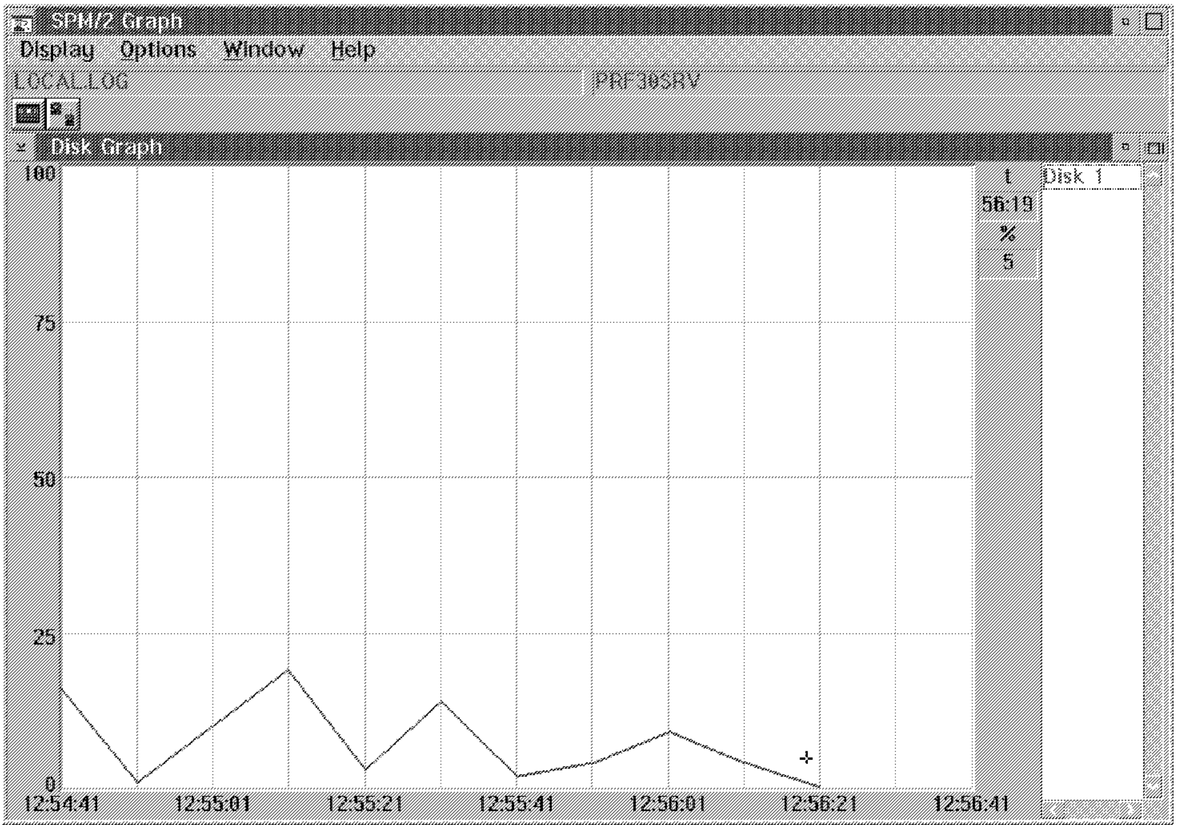

Figure 50. Disk I/O Monitor

SPM/2 RAM Monitor

The RAM Monitor uses five resource data lines to display information about two types of memory activity:

- Memory allocation and use
- Swapping activity

If swap-in and swap-out activity are high and the working set is near 100%, more memory may need to be installed.

The five memory resources are as follows:

1. *Fixed Memory*

Memory that cannot be swapped-out or discarded is called fixed. It will remain in physical RAM as long as the application that owns it is loaded. Examples of applications and their fixed memory components are:

- OS/2 - Parts of kernel - VDISK - DISKCACHE
- Database Manager - Buffer pool
- LAN Server - Buffers

Another application can load only if there is enough free memory for loading.

2. *Working Set Memory*

   The working set resource data line shows the percentage of data RAM that is in the working set. This information is useful for determining if the physical memory in the computer is sufficient for the applications that are currently running. Each point on the working set resource data line represents the percent of RAM in the working set during the previous working set period.

   The reported working set is bounded by the fixed and used memory. It is always more than the fixed and less than used memory. If the actual working set exceeds physical memory, it is reported at or near 100% and swapping may occur. You may want to reduce the number of active applications or install more physical memory if the RAM Monitor consistently reports a stable working set near 100%.

3. *Used Memory*

   Used memory is RAM allocated by the OS/2 system and present in physical memory. The used memory line shows the memory that is allocated and not swapped-out.

4. *Swap-out Activity*

   The RAM Monitor measures the amount of memory swapping activity. When demand for memory exceeds the capacity of physical memory, the OS/2 system swaps memory segments to disk in order to free physical memory. Specifically, the swapped-out resource data line shows the percentage of time the OS/2 system is busy swapping memory out to disk, including time it is blocked by other processes or waiting for disk input/output. (I/O)

5. *Swap-in Activity*

   When memory that has been previously swapped-out is required in physical memory, it is swapped-in from disk. Specifically, the swap-in line shows the percentage of time the OS/2 system is busy swapping-in memory, including time it is blocked by other processes or waiting for disk I/O.

If swapping activity remains high for an extended period of time, the system is probably spending more time swapping memory segments than it spends doing useful work. This may be caused by the computer having insufficient physical memory to support the applications currently running.

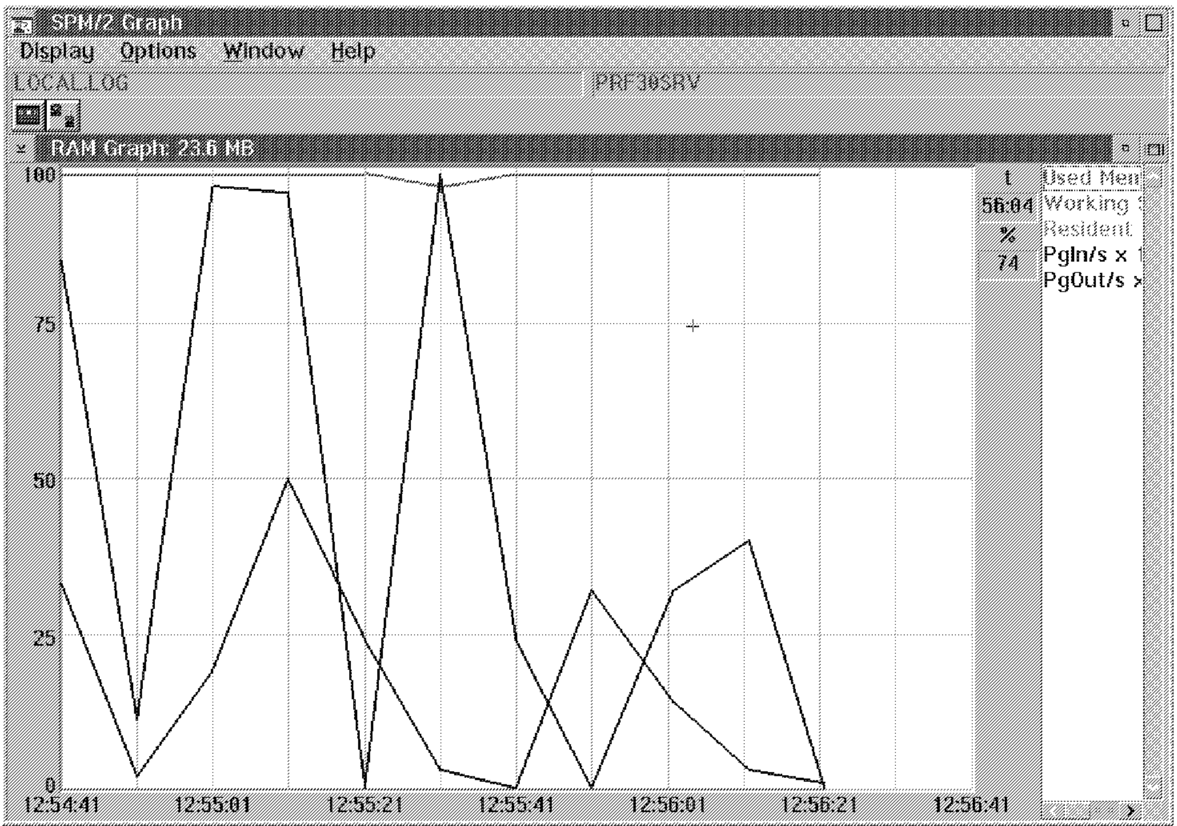

Figure 51. RAM Monitor

6.2 Tuning an Application

The default performance parameter settings for LAN Server are based on values obtained from a representative sample of LANs. They are calculated to ensure they are suitable for most customer environments. However in some cases this may not apply, and it is necessary to change these settings for particular LAN environments. For example, if the majority of the LAN workload consists of mainly copying large amounts of data, then it is necessary to know which parameter should be modified to improve network performance. However, alterations to these performance parameters must be carefully considered so that there is not much of an impact on any of the other tasks executed within the LAN environment.

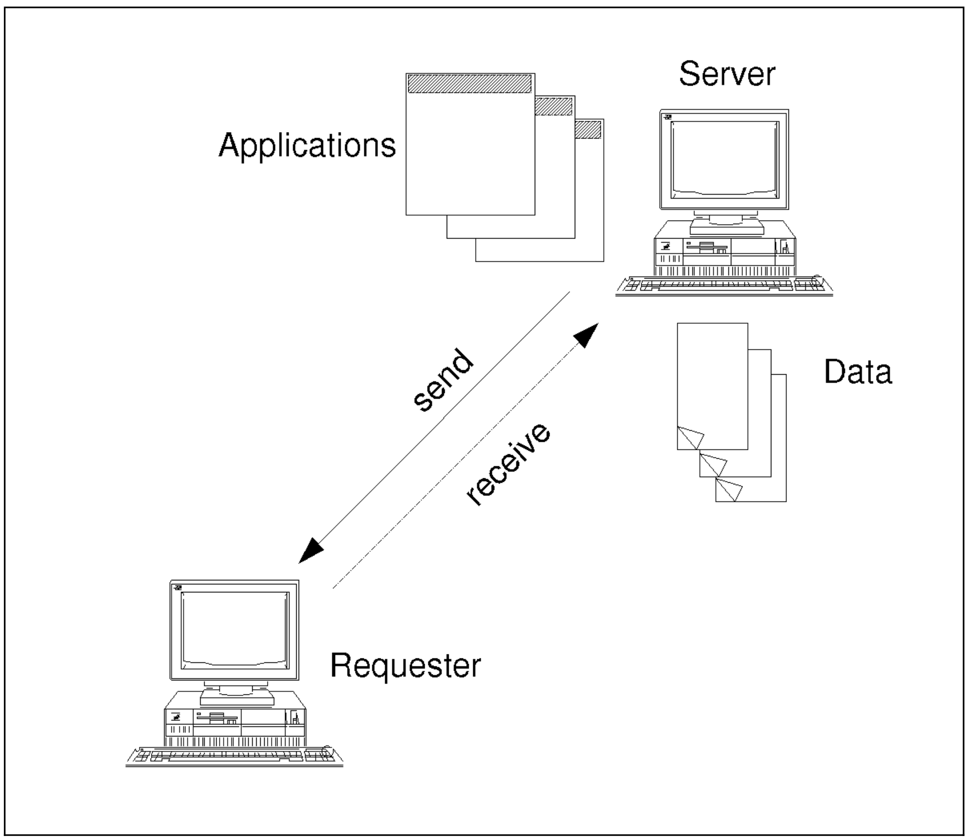

Figure 52. Different Tasks in a LAN Environment

Depending on the nature of the data transfer within LAN Server the appropriate data transfer protocol, Core, Multiplexed or RAW, is used. By identifying the protocol transfer that occurs the most frequently it would be possible to make certain changes that would ensure that the data transfer occurs more swiftly. For example:

1. The Core SMB protocol is implemented when copying small amounts of data. This predominantly can be used, because a typical PC application uses small size of read/write to the disk in general. An example of this would be when a secretary loads a spread sheet and working on the sheet during the day.
2. The Multiplexed SMB is used when one request buffer is not enough for the data. This would normally be experienced while loading application programs, such as Lotus Freelance** or Microsoft Word**. However, it is only useful to optimize performance for this transfer, if these applications are loaded several times a day and not used for long periods throughout the day. The Multiplexed SMB protocol will always be used if there are no big buffers present, and if the data transfers are greater than the size of **sizworkbuf**.
3. The Raw SMB protocol is used when copying large sequential amounts of data and there are big buffers present. This normally occurs during data backing up or copying situations.

In LAN Server the data flow from a server to a requester is dependent on several buffers situated at different layers within the OSI 7-layer reference model. Each of these buffers have a corresponding parameter that can modify their specification, and by limiting their sizes in a specific ways, they can provide optimum data transfers with minimal resource usage.

These buffers comprise of the:

- Transmit/receive buffers which are located in the adapter RAM
- NetBIOS receive buffers which are present in physical RAM
- Request/work buffers and big buffers which are also present in physical RAM

From a servers point of view, the buffer settings should be configured taking into consideration the most frequent block size that is transferred by all the applications that are being shared on it. Also depending on the nature of the applications it may be worth tuning the server to perform better when transmitting data, say if the primary role of the server is to download applications to the requesters, or strike a compromise if the data transactions occur equally both ways.

For requesters however, they can be tuned independently depending on the particular applications that are being used. They can be more finely performance tuned than a server as the optimization is simply for its own benefit.

Essentially as a LAN Server transaction is made, say from the server to the requester, data is passed from the application using request buffers to the NetBIOS request buffer area. Here it is moved to the adapter buffers where the data is placed into a frame and sent to the requester. At the requester the opposite process occurs until the application receives the data from the requester′s work buffers. Figure 53 on page 149 shows the necessary parameters that need to be tuned to set the values of each of the buffers. They are all defined in their respective initialization files. The adapter buffers and NetBIOS buffers are defined in the PROTOCOL.INI file and the server/requester parameters are all defined in the IBMLAN.INI file.

When copying data from a requester to a server rather than the other way around, there are other modifications that have to be made.

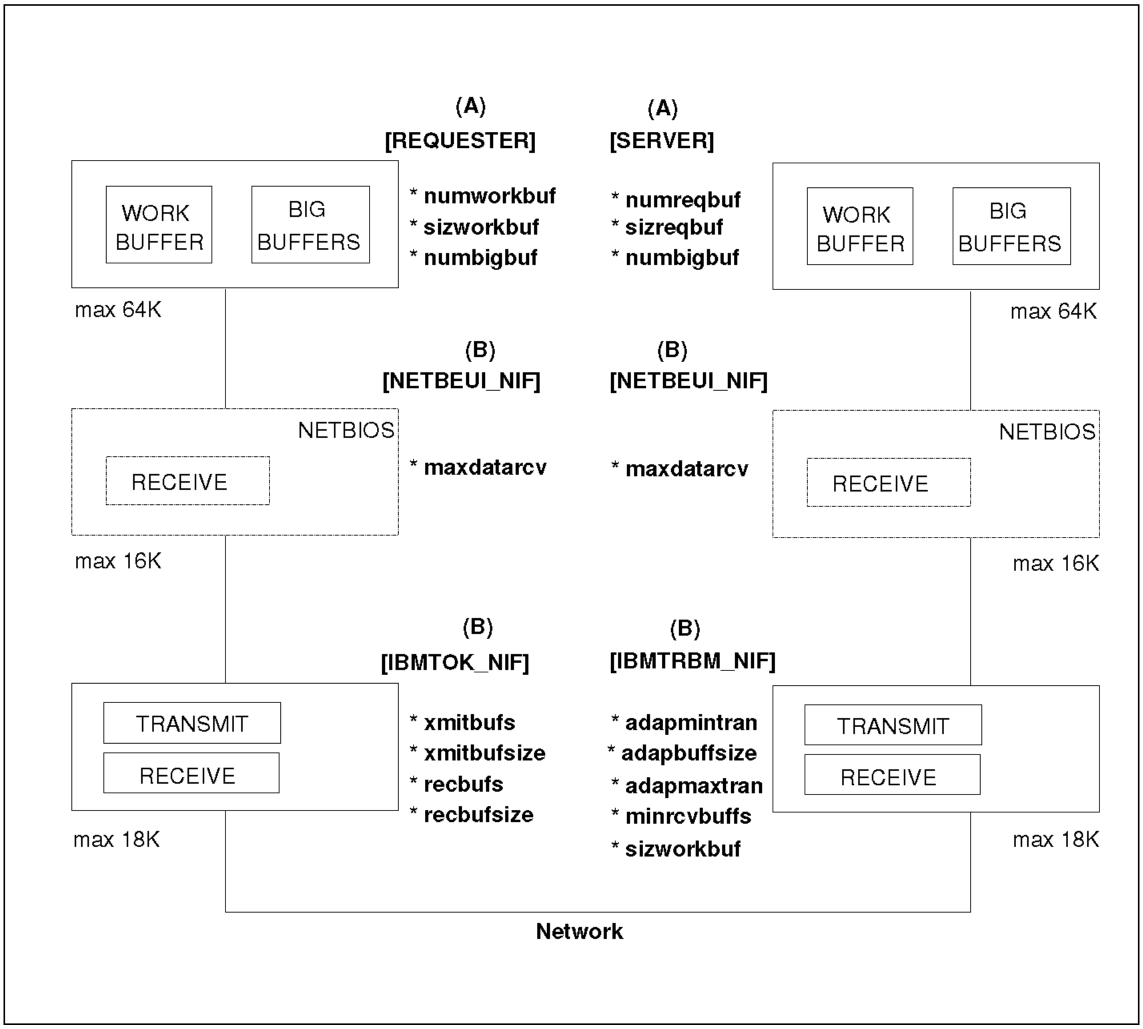

Figure 53. Network Data Flow with Related Parameters

When deciding the sizes of each of these buffers certain considerations have to be taken. In theory all the buffers in the system should be matched to ensure that there is a one-to-one relationship between every data transfer between buffers. However, this is not necessarily the best approach. To conserve resources or even due to resource restrictions, it is possible to have buffers of different sizes which are multiples of each other. If they are not multiples of each other then resources can be wasted and the data transfer would not be as efficient. The buffers consist of:

- Adapter buffering
- NetBIOS buffering
- LAN Server buffering

Here are a few guidelines that should be used when determining the size of each buffer.

16/4 Token-Ring Adapter Buffering

By default each adapter has 15 buffers each comprising of 256 bytes. As data arrives from the network it is copied from the frame and placed into this buffer area. Any data that is greater than 256 bytes is placed in multiple receive buffers and passed on to the next level of NetBIOS buffering. If a 16Mbps token ring is used with early token release activated, there is a possibility that the frame size can be as large as 18KB. Implementing the above receive buffer configuration, data can still be handled efficiently as the multiple buffers can generate sufficient delay for the buffers to be continually processed and updated.

Careful consideration should be taken when changing the receive buffer size in adapter RAM. If the receive buffers are over allocated then the other adapter resources will be restricted to whatever memory is remaining.

The transmit buffers on the server side do not have to be configured explicitly as they are allocated during the session setup.

It is not necessary to change the transmit buffer parameter on the server side, because the server has the ability to adjust its transmitting frame size to each requester. This is a dynamic process which optimizes data exchange between the server and the requester.

NetBIOS Buffering

The receive buffers, **maxdatarcv**, defined in the NetBEUI section of the PROTOCOL.INI file are used to transfer data through NetBIOS. By default they are set to 4168 bytes which is suitable for most circumstances. However, if the network workload is such that the data can be transferred in blocks greater than 4KB then this value can be increased accordingly to match the transfer. The maximum value for **maxdatarcv** is 16KB.

In the majority of cases this value will be greater than the receive buffers on the adapter and for improved performance it should be a multiple of the receive buffer size. This will ensure that multiple data transfers from the receive buffers would be needed before the NetBIOS buffers are filled with data to pass it onto LAN Server's buffering. The actual data manipulation at this stage is controlled by the NetBEUI drivers.

LAN Server Buffering

LAN Server buffering is used to pass data to and from a particular application to NetBIOS. Therefore, they should be configured to a size that corresponds to the most used block size from the application and NetBIOS buffering. By default the request/work buffers are set to 4KB which is acceptable for most cases as applications and NetBIOS tend to use transfers of around of 4KB. If the size of **sizworkbuf**/**sizreqbuf** parameters have to be changed for any reason then you must ensure it is a multiple of the NetBIOS buffer size, **maxdatarcv**, or performance could be degraded.

The only consideration that should be taken into account for big buffers is the number of big buffers that are present. To check if there is a sufficient number of them, the NET STATISTICS command can be issued to determine how often the requester is using all available big buffers. The default for requesters is one big buffer and typically this is sufficient to cope with most heavy network workloads.

Note

When configuring buffers you cannot be certain that the actual values are used during the data transfer. Depending on the amount of free memory space remaining on the machine, the machine decides the size of the buffers at the session negotiation during the logon process. If you want to determine the actual size of data that is being transferred it is possible to obtain the information by tracing the logon process. For more information see 3.7, "Logon Performance Tuning" on page 83.

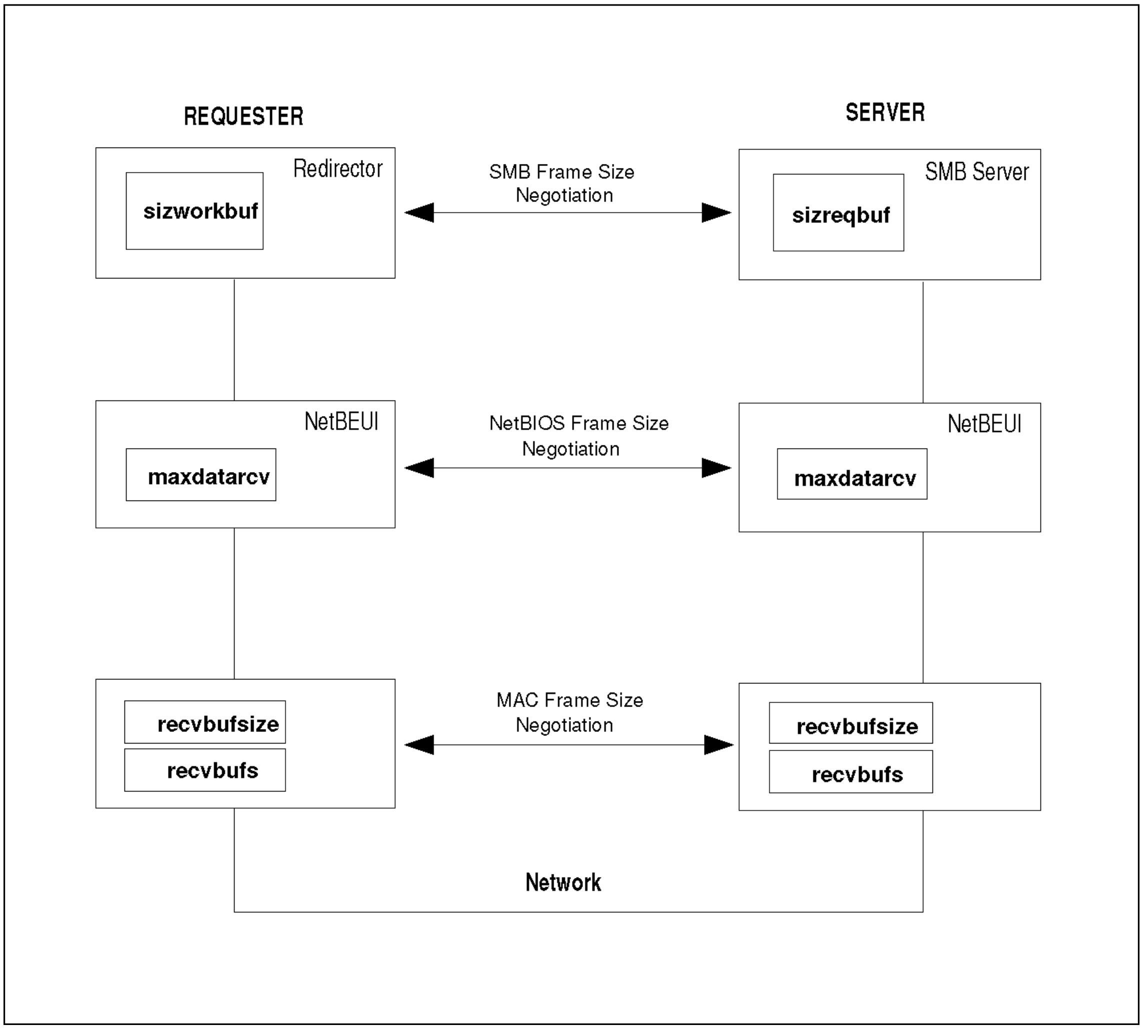

Figure 54. Negotiation during Session Establishment

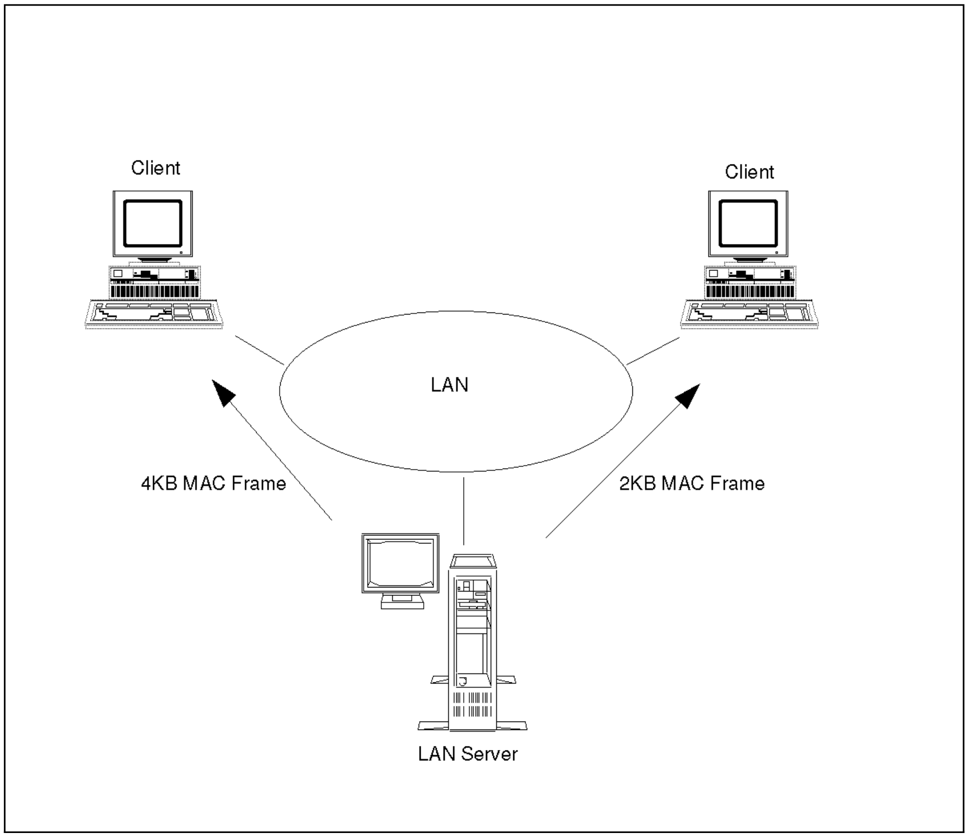

Figure 55. Dynamic Adapter Frame Size

Example of Tuning an Application

To help you understand more clearly how to tune an application to improve performance, the examples are presented. In this example, we used Lotus Freelance** Graphics, however the procedure would be the same even if a DOS application is to be used.

As mentioned earlier, to tune an application, knowledge of its workload characteristics is necessary. To obtain this information traces can be performed to investigate the typical data block transfer that is being transferred.

The two most important scenarios that should be considered are:

1. Loading an application
2. Loading data within an application

Depending on the function of the server these considerations can be optimized accordingly.

Throughout this example it is assumed that a requester is being tuned and subjected only to the workload generated by Freelance Graphics. If, however, a server needs to be optimized instead, then the same methodology still applies.

The following diagram shows the configuration of the test setup. All the buffers values are left to their default values.

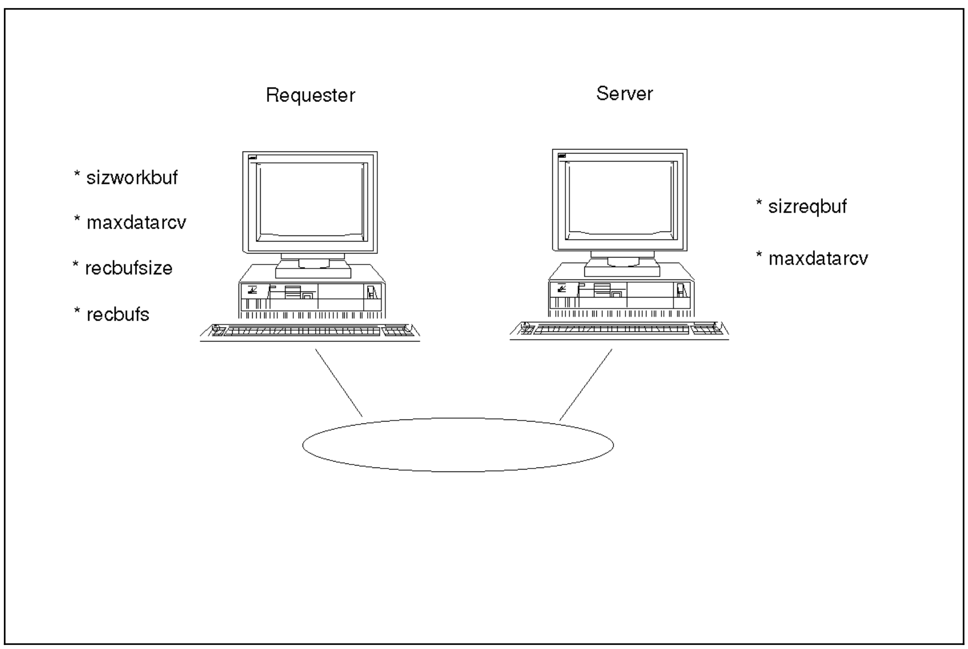

Figure 56. Test Scenario

Three traces are used to describe the behavior of the data transfer for Freelance Graphics under various conditions.

Trace 1 - SMB Core Protocol

The Core SMB protocol is implemented as the data transfers are smaller than **sizworkbuf**. The trace has been taken while loading a presentation file (LS06.PRS) into the Freelance Graphics application using a 16/4 adapter.

No	Dest	Source	Size	Interpretation
2	Server	Requester	124	SMB C Transact2 Find First \LS06.PRS
3	Requester	Server	133	SMB R Transact2 Completed
4	Server	Requester	116	SMB C Transact2 Qry Path Inf \LS06.PRS
5	Requester	Server	111	SMB R Transact2 Completed
6	Server	Requester	116	SMB C Transact2 Qry Path Inf \LS06.PRS
7	Requester	Server	111	SMB R Transact2 Completed
8	Server	Requester	32	NETBIOS D=10 S=07 Data ACK
9	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=46
10	Server	Requester	77	NETBIOS
11	Requester	Server	4176	NETBIOS
12	Server	Requester	116	SMB C Transact2 Qry Path Inf \LS06.PRS
13	Requester	Server	111	SMB R Transact2 Completed
14	Server	Requester	32	NETBIOS D=10 S=07 Data ACK
15	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=48
16	Server	Requester	124	SMB C Transact2 Find First \LS06.PRS
17	Requester	Server	133	SMB R Transact2 Completed
18	Server	Requester	124	SMB C Transact2 Find First \LS06.PRS
19	Requester	Server	133	SMB R Transact2 Completed
20	Server	Requester	107	SMB C Open and X \LS06.PRS
21	Requester	Server	97	SMB R Opened and X F=009C
22	Server	Requester	77	NETBIOS

When a presentation file is loaded in Freelance Graphics there are many small SMB frames generated to handle the data transfer. DosFindFirst and DosQryPathInfo are the examples of this trace.

In this scenario it is useful to have small transmit and receive buffers on the adapter at the requester. The default size of 256 bytes for the receive buffers is sufficient for good performance. However, for better data throughput at the the NetBIOS layer, the **maxdatarcv** parameter can be reduced from 4176 bytes to 2KB. This should free up memory that can be used for the more memory hungry resources. This also applies to the **sizworkbuf** parameter for determining the size of the work buffers. If they are configured to 2KB as well, then work buffers will be used to its full potential.

Trace 2 - SMB Multiplexed Protocol

This is the case that the application uses the Multiplexed SMB protocol to transfer the data from requester to server as data is being transmitted continuously without using big buffers.

The trace is taken while downloading the Freelance graphics application from a server to a requester using a 16/4 adapter.

No	Dest	Source	Size	Interpretation
0	Requester	Server	4200	NETBIOS D=07 S=0C Data, 4168 bytes
1	Requester	Server	822	NETBIOS D=07 S=0C Data, 790 bytes
2	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=0
3	Requester	Server	4200	SMB R Opened and X F=00EE
4	Requester	Server	52	NETBIOS D=07 S=0C Data, 20 bytes
5	Requester	Server	4200	NETBIOS D=07 S=0C Data, 4168 bytes
6	Requester	Server	1445	NETBIOS D=07 S=0C Data, 1413 bytes
7	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=3
8	Requester	Server	4200	SMB R Opened and X F=00EF
9	Requester	Server	52	NETBIOS D=07 S=0C Data, 20 bytes
10	Requester	Server	3014	NETBIOS
11	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=5
12	Requester	Server	4200	SMB R Opened and X F=00F0
13	Requester	Server	52	NETBIOS D=07 S=0C Data, 20 bytes
14	Requester	Server	4200	NETBIOS D=07 S=0C Data, 4168 bytes
15	Requester	Server	4200	NETBIOS D=07 S=0C Data, 4168 bytes
16	Requester	Server	356	NETBIOS D=07 S=0C Data, 324 bytes
17	Requester	Server	18	LLC D=F0 S=F0 R S RR NR=8
18	Requester	Server	4176	NETBIOS
19	Requester	Server	4176	NETBIOS
20	Requester	Server	4176	NETBIOS
21	Requester	Server	4176	NETBIOS
22	Requester	Server	4176	NETBIOS
23	Requester	Server	4176	NETBIOS
24	Requester	Server	4176	NETBIOS
25	Requester	Server	4176	NETBIOS
26	Requester	Server	4176	NETBIOS
27	Requester	Server	4176	NETBIOS
28	Requester	Server	4176	NETBIOS
29	Requester	Server	3152	NETBIOS
30	Requester	Server	4176	NETBIOS
31	Requester	Server	4176	NETBIOS
32	Requester	Server	4176	NETBIOS
33	Requester	Server	4176	NETBIOS
34	Requester	Server	4176	NETBIOS
35	Requester	Server	4176	NETBIOS

The maximum MAC frame size including NetBIOS transaction is 4176 bytes which was negotiated at session setup time. There are no dedicated acknowledgement frames sent, as the acknowledgement information is combined into the NetBIOS data frames themselves. If program loading of large size is frequently done during a day, it would be useful to tune the system for this type of transfer and leave all the buffer values to their defaults.

Trace 3 - SMB Multiplexed Protocol (Small Mac Size)

This trace is taken using a 4Mbps LAN adapter which has a limited amount of RAM on the adapter.

No	Dest	Source	Size	Interpretation
0	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
1	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
2	Server	Requester	77	NETBIOS
3	Requester	Server	2040	SMB R Read From File Successful
4	Requester	Server	2040	NETBIOS D=08 S=26 Data, 2008 bytes
5	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
6	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
7	Server	Requester	77	NETBIOS
8	Requester	Server	2040	SMB R Read From File Successful
9	Requester	Server	2040	NETBIOS D=08 S=26 Data, 2008 bytes
10	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
11	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
12	Server	Requester	77	NETBIOS
13	Requester	Server	2040	SMB R Read From File Successful
14	Requester	Server	2040	NETBIOS D=08 S=26 Data, 2008 bytes
15	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
16	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
17	Server	Requester	77	NETBIOS
18	Requester	Server	2040	SMB R Read From File Successful
19	Requester	Server	2040	NETBIOS D=08 S=26 Data, 2008 bytes
20	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
21	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
22	Server	Requester	77	NETBIOS
23	Requester	Server	2040	SMB R Read From File Successful
24	Requester	Server	2040	NETBIOS D=08 S=26 Data, 2008 bytes
25	Requester	Server	160	NETBIOS D=08 S=26 Data, 128 bytes
26	Server	Requester	32	NETBIOS D=26 S=08 Data ACK
27	Server	Requester	77	NETBIOS
28	Requester	Server	2040	SMB R Read From File Successful

```
29 Requester     Server       2040 NETBIOS D=08 S=26 Data, 2008 bytes
30 Requester     Server        160 NETBIOS D=08 S=26 Data, 128 bytes
31 Server     Requester         32 NETBIOS D=26 S=08 Data ACK
32 Server     Requester         77 NETBIOS
33 Requester     Server       2040 SMB R Read From File Successful
34 Requester     Server       2040 NETBIOS D=08 S=26 Data, 2008 bytes
35 Requester     Server        160 NETBIOS D=08 S=26 Data, 128 bytes
36 Server     Requester         32 NETBIOS D=26 S=08 Data ACK
37 Server     Requester         77 NETBIOS
38 Requester     Server       2040 SMB R Read From File Successful
39 Requester     Server       1144 NETBIOS D=08 S=26 Data, 1112 bytes
40 Server     Requester         32 NETBIOS D=26 S=08 Data ACK
41 Server     Requester         77 NETBIOS
42 Requester     Server       2040 SMB R Read From File Successful
43 Requester     Server       1656 NETBIOS D=08 S=26 Data, 1624 bytes
44 Server     Requester         32 NETBIOS D=26 S=08 Data ACK
```

Due to the limited RAM available on the 4Mbps adapter, the size of the NetBIOS transactions is reduced to 2040 bytes. This is due to the fact that the NetBIOS data block size is dependent on the amount of memory available on the adapter and the MAC driver can only allow 2040 bytes to be passed to NetBIOS. As a result the actual NetBIOS buffer size which is set up during session initialization is reduced to 2040 bytes.

As the data being transmitted in this trace is smaller than the size of the **sizworkbuf** parameter, then the Core SMB protocol is adopted. NetBIOS expects acknowledgements to be sent when 2300 bytes of data is sent, but as the NetBIOS transfers are restricted to 2040 bytes, extra frames needs to be transmitted before NetBIOS can send an acknowledgement. This can be visualized by looking at frames 29 and 30 in the trace.

If a 16/4 adapter was used, the network traffic could be reduced as only one frame needs to be sent instead of two. This means frame 29 and 30 can be combined to one frame. Because the NetBIOS frame size is limited to 2040 bytes length, sending 4KB of SMB frame requires the segmentation of the frame.

Trace 4 - SMB RAW Protocol

This trace is taken to illustrate the function of big buffers. The data transfer is done using RAW SMB protocol, and this was done by performing an XCOPY of data from the server to the requester. (XCOPY of a subdirectory with several megabytes of data.)

No	Dest	Source	Size	Interpretation
1	xxxxxx	Server	170	SMB C Transaction \MAILSLOT\LANMAN
2	xxxxxx	xxxxx	25	LLC D=00 S=04 C U TEST P
3	xxxxxx	xxxxx	25	LLC D=00 S=04 C U TEST P
4	xxxxxx	xxxxx	35	LLC D=00 S=04 C U TEST P
5	Server	Requester	83	SMB C Read Bl. Raw 64512 at 17920
6	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
7	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
8	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
9	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
10	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
11	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
12	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
13	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
14	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
15	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
16	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
17	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
18	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
19	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
20	Requester	Server	2608	NETBIOS D=0F S=19 Data, 2576 bytes
21	Server	Requester	32	NETBIOS D=19 S=0F Data ACK
22	Server	Requester	83	SMB C Read Bl. Raw 64512 at 82432
23	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
24	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
25	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
26	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
27	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
28	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
29	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
30	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
31	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
32	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
33	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
34	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
35	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
36	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
37	Requester	Server	2608	NETBIOS D=0F S=19 Data, 2576 bytes
38	Server	Requester	32	NETBIOS D=19 S=0F Data ACK
39	Server	Requester	83	SMB C Read Bl. Raw 64512 at 146944
40	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
41	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
42	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
43	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes
44	Requester	Server	4456	NETBIOS D=0F S=19 Data, 4424 bytes

As expected the server uses the RAW SMB protocol to transfer the data from the server to requester. This is due to the fact that a large sequential data transfer is taking place. The acknowledgement is present after every 64KB has been sent to the LAN Server from the requester, and can be illustrated by frames 21 and 38.

For LAN Server Entry it is useful to increase the number of big buffers (**numbigbufs**) in the IBMLAN.INI file if they are prone to get exhausted at peak levels. For LAN Server Advanced Version 3.01 or 4.0 it allocates big buffers with the fsprealloc or srvprealloc parameters.

To obtain an approximate view of the workload characteristics of an application there is a more straight forward approach that could be adopted. This technique essential checks if the application is using small or large record transfers to transmit data. This is achieved by comparing two scenarios:

1. Retrieving data with the application on the LAN
2. Retrieving the same data using an OS/2 COPY command.

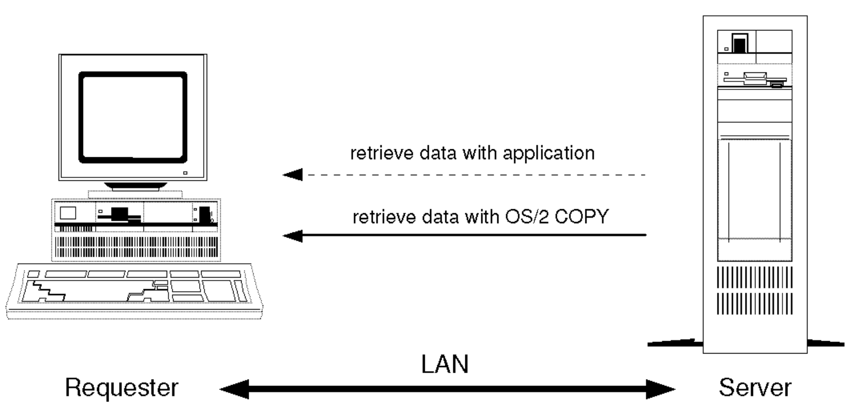

Figure 57. Large Data Transfer Benchmark

Using the OS/2 COPY command will initiate a RAW SMB protocol transfer if big buffers are present, or a Multiplexed SMB protocol transfer. This should increase the data throughput on the network. Therefore, if the application runs noticeably slower than an OS/2 COPY, then the application is probably transferring small records rather than large records.

Appendix A. Net Errors

In this appendix, we will pick up typical errors related to the performance and capacity problems. Most of them are based on the ASKQ library and LAN Server forum.

We start this appendix by describing how to read the NCB data that is sometimes appended with net error files to provide you with the basic knowledge to understand it.

A.1 How to Read NCB Data

To understand NET ERRORs, you need to know NCB data format. You can get this information from the manual *IBM Local Area Network Technical Reference*.

Now, as a real-life example, lets assume you received the following error in the LAN Server error log.

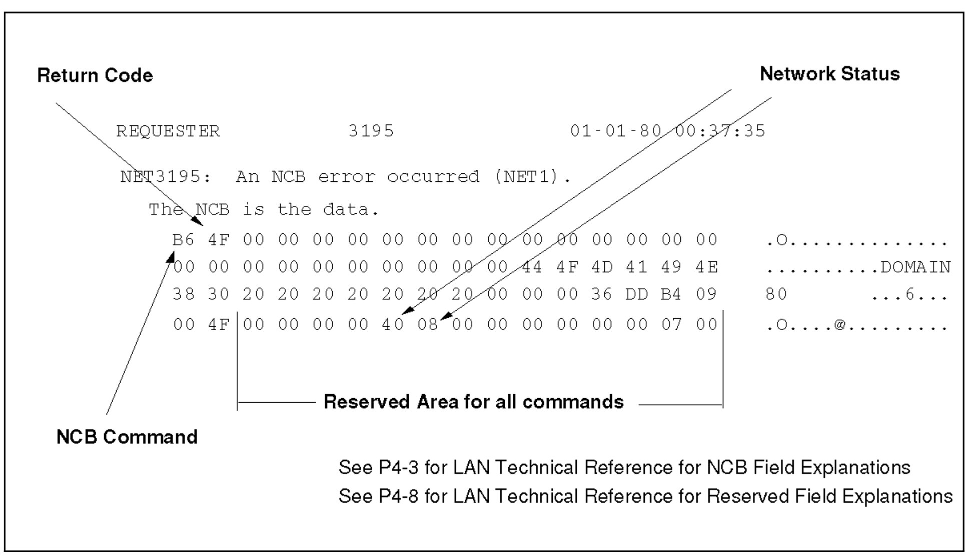

Figure 58. NCB Data Format

The first byte of the NCB is the command. X′B6′ is the code for NCB.ADD.GROUP.NAME. This function adds a NetBIOS group name to the adapter. For the server, this is normally the domain name. This command failed with a return code of X′4F′ (second byte).

Looking up this return code, we see that there was a problem with the network status bits - one or more of bits 8 - 11 in the network status field were set on. This is one of the fields in the NCB protocol.

In Figure 58 on page 163, it is the 54th and 55th bytes. These are read in low-high order, so the value of the network status field is 0840. This breaks down to Figure 59:

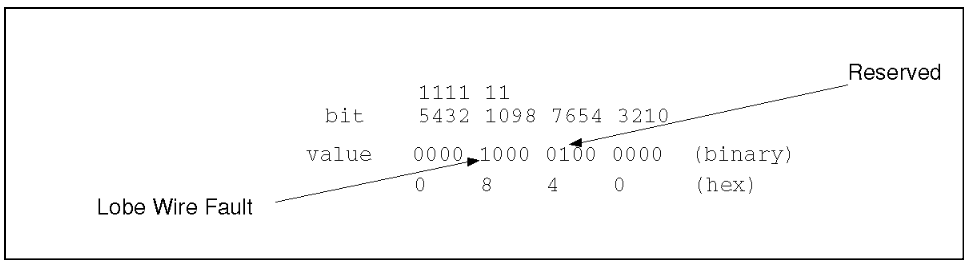

Figure 59. Network Status Data Format

Now, looking at the interpretation of these bits on page B-55 of *IBM Local Area Network Technical Reference*, we see:

- Bit 11
  - Lobe wire fault
  - An open or short circuit has been detected in the lobe data path. The adapter will be closed.
- Bit 6 - reserved

From this interpretation we see that a hardware error exists, stopping the server from starting.

A.2 Net Error Examples

The followin is the examples of the NET ERRORs and recommended actions.

NET3101

Case1

Cannot complete the database application due to the resource lack of **numreqbuf**.

Environment:

- The server runs a database application that uses a lot of resources.
- The server is LAN Server 3.0 Advanced and using Server 295 - model 8600 that has 32MB of RAM, 50MHZ processor, and a LANStreamer Card.
- Raised **numreqbuf** to the maximum of 300 but received the same error after a short time.

Returned Error:

- `NET3101E The system has run out of NUMREQBUFFs.`

Description:

300 is the maximum value you can set **numreqbuf** for LAN Server 3.0. This is a limitation of the LAN Server code not the hardware. The general rule of thumb for **numreqbuf** setting is to allocate from 1.5 to 2 **numreqbuf**s per workstation.

Action:

- Upgrade LAN Server 3.0 to 4.0. At same time, add more memory to allow enough number of request buffers allocated as well as allowing database applications to use required memory.
- Make smaller domains or have the application on more servers so that all the resources of one machine are not being tied up.

Case2

The domain controller runs out of resource controlled by **numdgrambuf**.

Environment:

- A large domain with 700+ DLR & OS/2 users and 23 additional servers.

Returned Error:

- NET3101E The system has run out of NUMDGRAMBUFFs.

Description:

The parameter, **numdgrambuf** sets the number of buffers for processing datagrams. Servers use datagrams to broadcast their presence on the domain. Datagrams are also used for domain-wide broadcasts. Basically what has occurred is you have encountered a situation where you did not have enough datagram buffers to handle all the datagram processing that was occurring.

If you are on a network with many servers or with a large amount of domain-wide broadcasts, you may want more datagram buffers to handle incoming announcements.

NetBIOS traces (TRACE ON 164 from the command line before the failure and OS2TRACEMASK=-0x7FF in PROTOCOL.INI) on the server could be used to see if the adapter actually received the datagrams and passed them to NetBEUI.

Action:

- Their is no absolute correct setting for this parameter. The only thing that can be suggested is to increase the value of the parameters, **numdgrambuf** in IBMLAN.INI and **datagrampackets** in PROTOCOL.INI until you are not receiving the error anymore.
- If you are looking to decrease the amount of datagrams that go out on the network, you may want to increase the value of NUMBER OF NAMES IN REMOTE NAMES DIRECTORY which can be changed in the NetBIOS section of LAPS. Increasing this parameter saves the address to different machines (for instance, the servers), so that a datagram does not have to be sent to find a machines location. It will already be stored in this name directory. Therefore, the amount of datagram traffic on the network will be reduced, and this may help alleviate the NET3101 error you are receiving.

NET3193

Case

LAN Server/Requester connection problems over the bridge. The large files between LAN Server and Requesters cannot be copied although logon is established and small data can be copied.

Environment:

- Connecting 2 offices via OEM (Proteon) routers and 9600bps lines.
- Using LAN Server and Requester between each side.

Returned Error:

- SYS0055: The specified network resource is no longer available.
- SYS0240: The network connection is disconnected.
- NET3193: A virtual circuit error occurred on the session to (srv_id) The NCB command and return code is the data.
  96 18
- NET3195: An NCB error occurred: \\srv_name\resource_accessed. The NCB is the data.

Description:

LAN Server/Requester uses NetBIOS session which has NetBIOS response timer T1. This error seems to have occured due to some timer expiration.

Action:

Increase NetBIOS response timer, T1 in NETBEUI_NIF section of PROTOCOL.INI from default 500 milliseconds to 4000 milliseconds, for example. You can try more to find a suitable timer value. Note that you should change both the server and requester parameter.

NET3199

Returned Error:

- NET3199 - Sideband transmissions have been disabled for the session with server ***.

  Cause: An excessive number of frames have been lost on the connection with the indicated server. This could be caused by a busy server or defective network hardware.

  Action: In order to re-enable Sideband transmissions for this session, the session must be deleted and then re-established. In the case of Sideband transmissions being disabled repeatedly, contact your network administrator regarding the instability of the network configuration.

Description:

This error is probably occurring because of some sort of adapter congestion at the server.

Action:

Reboot the server and use a network monitoring tool such as DatagLANce to check for any type of adapter congestion. Also check the LANTRAN.LOG, net error log, and perhaps the ACSLAN.LOG to see if you can find any related errors.

NET5305

Case1: Requester cannot copy large files (1MB or more) from network to local drive. The large files between LAN Server and Requesters cannot be copied although logon is established and small amount of data can be copied.

Environment:

- Using LAN Server and Requester between each side.
- Both machines have 16/4 token-ring cards operating at 4Mbps.

Returned Error:

- NET3193: A virtual circuit error occurred on the session to M800350
  96 18
- NET5305: An NCB command timed out. The session may have terminated abnormally. The NCB is the data.
- NET5380: A network adapter malfunction has occurred. The NCB request was refused. The NCB is the data.
- NET5380: A network adapter malfunction has occurred. The NCB request was refused. The NCB is the data.

```
12 05 37 00 00 90 7D 00 33 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 12 DD 00 00
00 05 00 00 00 00 43 30 00 00 00 00 0D 6A C0 17
```

Description:

- NET3193

  The connection between your requester and the specified server was unexpectedly disconnected. The server may have been started again, or a network problem may have occurred.

  The Hex 96 18 return code specifies SESSION_END. The Hex18 frame is transmitted as a result of a HANGUP command, a SEND command that timed out, or some abnormal condition. Its function is to indicate the termination of a session to the remote partner.

- NET5305

  A network control block (NCB) command to a remote workstation failed because the remote workstation did not respond in time. The remote workstation is not listening. The session to the remote workstation may have been dropped.

There is a parameter located in the IBMLAN.INI file that may solve this problem with the server timing out while transmitting the file. The parameter **srvheuristics** bit 15.

This parameter provides the NetBIOS timeout. The NetBIOS timeout is the length of time the server waits for an acknowledgement response from a requester. If a response is not received prior to this timeout, the server will disconnect the session to that requester. The default value of 34 seconds may not be enough if the requester/server are separated by a congested bridge or slow telecommunications lines.

Action: Set **srvheuristic** bit 15 to 9. This will prevent a NetBIOS timeout. Then decrease the value to get a proper timeout value.

Case2: The server is sharing two applications to 30 DOS LAN Requesters on the LAN and is experiencing timeout NET5305 NCB error. This problem occured by setting the digit position 0 of **wrkheuristic** and **srvheuristic** to 1 which means opportunistic locking is active.

Returned Error:

- `NET5305: An NCB command timed out. The session may have terminated abnormally. The NCB is the data.`

Description:

- Opportunistic locking works fine in case the requester needs a large file with sequential read/write and there is very small user access to a particular file. This assumes that there will be no access contention at same time. The server can lock an entire file and read-ahead a large block of file for caching. Opportunistic locking doesn′t give any advantage to random I/O type applications which will read/write small records at one time.

  In this case, 30 requesters share two applications in the server. That is not too bad for opportunistic locking.

Action:

Turn off the opportunistic locking both server and requester for this case.

NET5333

Cannot start LAN Server, received the message, SYS0054.

Environment:

- LanStreamer MC 32 PN 92F8942 NDIS device driver version 1.20.07
- OS/2 2.1 XR02010

- OS/2 LAN Server / Requester 3.00 IP07001
- LAPS 2.20.2 WR07045

Returned Error:

- ```
  C:\net start server
  The REQUESTER service is starting............
  The REQUESTER service could not be started.
  ```
- NET3056: A system error has occurred.
- NET3502: OS/2 error 54 has occurred.
- SYS0054: The network is busy, or is out of resources.
- NET5333: The NETBIOS interface is busy, or NETBIOS is out of linkstation

```
The NCB request was refused.  The NCB is the data.
  20 21 00 02 10 50 77 00 80 01 00 00 00 00 00 00        ....P
w.......
  00 00 00 00 00 00 00 00 00 00 50 53 32 38 30 20       ......
  20 20 20 20 20 20 20 20 20 00 00 00 20 DD 00 00
    ... .
```

Action:

To fix this problem, try setting the **wrkheuristic** bits 12, 14 and 15 to zero in the requester section of the IBMLAN.INI.

Also in the CONFIG.SYS file there is a line that looks like this:

```
RUN=C:\IBMCOM\PROTOCOL\NETBIND.EXE
```

Change this to read:

```
CALL=C:\IBMCOM\PROTOCOL\NETBIND.EXE
```

NET5380

Case: Losing communication sessions via LUA services

Returned Error:

- NET3195: An NCB error occurred (NET1). The NCB is the data.
```

```
96 F6 7D 05 DC D6 1D 00 04 11 44 00 E0 53 24 00
00 00 00 00 00 00 00 00 00 00 1C 77 00 0A 00 00
00 00 00 00 00 00 00 00 00 00 00 00 96 DD B4 09
00 F6 00 00 00 00 00 20 00 00 00 00 00 00 1E 5C
```

- NET5380: A network adapter malfunction has occurred. The NCB request was refused. The NCB is the data.

```
96 F6 7D 05 DC D6 1D 00 04 11 44 00 E0 53 24 00
00 00 00 00 00 00 00 00 00 00 1C 77 00 0A 00 00
00 00 00 00 00 00 00 00 00 00 00 00 96 DD B4 09
00 F6 00 FF 00 00 00 20 00 00 00 00 81 F2 4B 00
```

- NET5380: A network adapter malfunction has occurred. The NCB request was refused. The NCB is the data.

```
96 F6 7D 05 CC 92 1D 00 04 11 28 00 E0 53 24 00
```

Description:

- NET5380

  A network control block (NCB) command to a remote workstation failed because the remote workstation did not respond in time. The remote workstation is not listening. The session to the remote workstation may have been dropped.

Action:

- Solution 1:

  You are running out of resources on the type 2 card. The type 2 card only has 16K of RAM whereas the 16/4 card has 64K of RAM. It would be possible to decrease some parameters to use the type 2 card, but you would then run into problems by severely limiting the number of users possible. Therefore, the only viable solution in such an environment is to use 16/4 token-ring card.

- Solution 2:

  This error is usually generated when one set of parameters are set either too high or too low (for example, sessions, commands and names in the IBMLAN.INI file are set higher than those set in NetBIOS in PROTOCOL.INI).

  Set the number of sessions, commands and names you need for X number of users you want on the network. If maxusers is set to 50, then sessions should be at least 50, commands should be the same as sessions, and names can be left at default. These are values in the

NetBEUI parameters in PROTOCOL.INI. Maxsessions should be equal to or greater than sessions in the IBMLAN.INI file, the same goes for maxcommands and maxnames. You should allot one linkstation per workstation and additional server. The NET3195 errors indicate an inconsistency between your IBMLAN.INI file and PROTOCOL.INI file parameters.

Appendix B. Performance Parameters

The following will list all relevant parameters related to tuning a server system. Although this is redundant from the product publication, this may be convienient for you as a quick reference. The parameters reside in three main files:

- IBMLAN.INI
- CONFIG.SYS
- PROTOCOL.INI

B.1 IBMLAN.INI

The IBMLAN.INI parameters have an influence on several sub-services which can run in a server system. The following table shows the parameters and the related sub-services from the IBMLAN.INI file. A brief description of all parameters is followed after the table.

Table 7 (Page 1 of 2). OS/2 LAN Server Tuning Parameters - IBMLAN.INI

Sub-service / Section	Parameter	Default Value	Minimum Value	Maximum Value
File Server	**alertsched**	5	0	65535
	numbigbuf	12	0	80
	numfiletasks	1	1	8
	numreqbuf	36	5	300
	sizreqbuf	4096	1024	32768
	srvanndelta	3000	0	65535
	srvannounce	60	0	65535
	srvheuristics	NAR 2	NAR 2	NAR 2

Table 7 (Page 2 of 2). OS/2 LAN Server Tuning Parameters - IBMLAN.INI				
Sub-service / Section	**Parameter**	**Default Value**	**Minimum Value**	**Maximum Value**
Replicator	**guardtime**	2	0	interval / 2
	interval	5	1	60
	pulse	3	1	10
	random	60	1	120
Netlogon	**pulse**	60	60	3600
	randomize	30	5	120
	scanpause	0	0	15
	scanperiod	15	0	1440
	scantime	0:00	0:00	23:59
UPS	**batterymsg**	600	30	3600
	batterytime	60	0	28800
	messdelay	5	0	120
	messtime	120	30	300
	recharge	100	5	250
RIPL	**maxthreads**	10	0	dependent on system
LSserver	**srvpipes**	3	1	20
DCDB Replicator	**guardtime**	2	0	interval / 2
	interval	5	1	60
	pulse	3	1	10
	random	60	1	120

alertsched: Interval time (minutes) at which the server checks for alert conditions. This parameter uses memory and CPU power. Reduce the value only if you need more frequent checking.

numbigbuf: Number of 64K buffers used for large data transmission. This parameter uses memory. Increase the value only if you have a lot of large amount of data transfer.

numfiletasks: Number of concurrent file handling processes and print requests. This paramteter reserves threads in the system. Increase this parameter only when many requesters access the same range of files.

numreqbuf: Number of request buffers the server uses to take requests from the requester. This parameter impacts memory requirements. Increase the value from 1.5 to 3 per requester. Refer to 3.6, "Capacity Tuning Parameters (LAN Server 4.0)" on page 71 for LAN Server 4.0.

sizreqbuf: Size of a request buffer (bytes). Refer to 3.6, "Capacity Tuning Parameters (LAN Server 4.0)" on page 71 for LAN Server 4.0. Increase the value only, if you have large amounts of data transferring over the wire. The **sizworkbuf** should have the same value.

srvanndelta: Time (in milliseconds) the server uses to vary its announce rate. To avoid any unnecessary tasks, don′t decrease this parameter.

srvannounce: Rate (in seconds) at which the server announces its presence on the network.

srvheuristics: The detailed parameter which the server′s behaviour is represented as a bit representation, off or on. The new IBMLAN.INI file of LAN Server 4.0 has a description for each digit.

guardtime: Time interval (in minutes) at which the export path must be stable before importers can connect to it. Decreasing this parameter uses more CPU.

interval: Time interval (in minutes) at which subdirectories and files in the export path are checked for changes. Decreasing this parameter uses more CPU.

pulse: Time how often the exporter sends updates to the importers. Increasing this parameter will use more CPU.

random: Interval (in seconds) at which an importer can use to connect. If this parameter is increased, it gives better load balancing for the server.

pulse (Netlogon): Time interval (in seconds) between the update of NET.ACC notices. Decreasing this value needs more CPU power.

randomize: Time period (in seconds) at which the member or backup servers should send a request at random to get changes after receiving a change notice. Decreasing this parameter can overload the domain controller with multiple update requests.

scanpause: Amount of time (in seconds) at which the domain controller waits before polling each user. Increase this parameter to avoid too many datagrams on the network.

scanperiod: Time interval (in minutes) at which the domain controller polls all users. If you increase this parameter to 1440 (24hours) the server only polls once a day.

scantime: Time at which polling occurs. Set this parameter to a time where you have no traffic (for example 0:00=midnight)

batterymsg: Number of seconds a server waits between low battery alerts. You can increase this parameter.

batterytime: Number of seconds the server can run on the battery power before shutdown. Decrease this parameter if you have many open files on your server.

messdelay: Number of seconds between power failure and the first message to the user. If you have a lot of open files, decrease this parameter.

messtime: Time interval (in seconds) at which a message will be sent to the users. When you decrease the **batterytime** you should also decrease this value.

recharge: Number of minutes of recharge time required by the UPS to gain one minute of battery time. Increase the parameter if you have many open files to close first before a shutdown can be issued.

maxthreads: Number of threads the RIPL service starts in order to perform asynchronous reading. Increase the parameter only if you have many RIPL workstations.

srvpipes: Maximum number of pipes a server will use. Increase this parameter if you have many users log on simultaneously.

guardtime (DCDB Replicator): Time duration (in minutes) at which the export path must be stable before servers can connect to it. Increase the parameter for better performance (but for less security).

interval (DCDB Replicator): Time interval (in minutes) at which the DCDB directory is checked for the changes. Increase the parameter for better performance, however DCDB changes will not be reflected in time.

pulse (DCDB Replicator): Time interval the domain controller sends DCDB updates to other servers. Decreasing this parameter has less redundant pulse updates.

random (DCDB Replicator): Time interval (in seconds) for which the server has time to connect. Increase this parameter for avoiding an overload on the DCDB Replicator.

B.2 CONFIG.SYS

The CONFIG.SYS parameters can set the values for memory, cache, heap space, disk space, swapping activities and write time. The following table shows the parameters and the related component of the CONFIG.SYS file.

Table 8 (Page 1 of 2). OS/2 CONFIG.SYS Tuning Parameters

Component	**Parameter**	**Default Value**
Memory	**BUFFERS**	30
HPFS.IFS	**/CACHE**	20%
386HPFS.IFS	**/C**	20%
	/H	20% of remaining memory
	/USEALLMEM	xx
CACHE.EXE (Entry)	**/LAZY**	ON
	/MAXAGE	5000 (ms)
	/DISKIDLE	xx
	/BUFFERIDLE	500
CACHE386.EXE (Advanced)	**/LAZY**	ON
	/MAXAGE	5000
	/BUFFERIDLE	500
HPFS386.INI (LS 4.0)	**/FSPREALLOC:x**	n
	/SRVPREALLOC:x	n
Memory / Disk	**DISKCACHE**	64K

Table 8 (Page 2 of 2). OS/2 CONFIG.SYS Tuning Parameters		
Component	**Parameter**	**Default Value**
Disk I/O	**SWAPPATH**	path=C:....
	PRIORITY I/O	YES

BUFFERS: Number of disk buffers that OS/2 keeps in memory. The disk buffer is a 512 byte (sector) portion of storage OS/2 holds I/O information temporarily. If you simultaneously run a large number of programs, you should increase the value to have better performance.

HPFS.IFS - /CACHE: Amount of memory (KB) the system reserves for cache memory. If this parameter is not specified, the system will use 20% of the system memory.

HPFS386.IFS - /C: HPFS386 cache size (KB), managed by CACHE386. If this parameter is not specified, 20% of the system memory will be used. LAN Server 4.0 uses 60% of the system memory, when more than 20MB of memory is installed.

HPFS386.IFS - /H: Maximum dynamic allocation for heap (KB). The heap option is required for HPFS386 to store its internal data structure such as file, find and search handles. If no heap space is defined, the system will use 20% of the remaining memory (after subtracting cache, diskcache, buffers amd other memory using components.)

HPFS386.IFS - /USEALLMEM: Indication to HPFS386 that disk and LAN adapter can support 32-bit DMA, so HPFS386 can allocate its buffers beyond 16MB.

CACHE.EXE - /LAZY: Switch to enable or disable lazy writing for the specified drive. It defers writing data to the disk until the operating system is idle or when the update is a maximum of five seconds old.

CACHE.EXE - /MAXAGE: Maximum time (in milliseconds) that a dirty cache block can be in memory before being flushed.

CACHE.EXE - /DISKIDLE: Minimum time (in seconds) that a disk must be idle before it can accept data from the cache.

CACHE.EXE - /BUFFERIDLE: Minimum time (in milliseconds) that a dirty cache block for a drive must be idle before it is written opportunistically when the disk subsystem is idle.

HPFS386.IFS - /FSPREALLOC:n: Number of 64KB buffers which have to be preallocated by the HPFS386 when the system is started.

HPFS386.IFS - /SRVPREALLOC:n: Number of 64KB buffers which have to be preallocated be the HPFS386 when the server is started.

DISKCACHE Size of memory (KB) to allocate for the FAT file system. Increase this parameter, if you have DOS applications on a FAT Drive.

SWAPPATH: Size and the location of the swap area on the hard disk. To improve performance, the SWAPPER.DAT file could be placed on a separate logical drive which limits the size of the SWAPPER.DAT and prevents disk fragmentation.

PRIORITY I/O: Disk I/O priority for applications running in the foreground. Set this parameter to NO and the background task disk I/O (for example, server application) has priority over the foreground task.

B.3 PROTOCOL.INI

The PROTOCOL.INI parameters are relevant for adapter buffers and protocol buffers. The following table shows the performance parameters concerning the protocol stacks and the MAC drivers.

Table 9 (Page 1 of 2). OS/2 PROTOCOL.INI Tuning Parameters

Component	Parameter	Default Value
NETBEUI.NIF	**MAXDATARCV**	4168
	Ti	30000
	T1	500
	T2	200
	NETBIOSRETRIES	8
	PIGGYBACKACKS	1
	MAXIN	1
	MAXOUT	1
	MAXTRANSMITS	6
	MINTRANSMITS	2

Table 9 (Page 2 of 2). OS/2 PROTOCOL.INI Tuning Parameters		
Component	**Parameter**	**Default Value**
IBMTOK.NIF	**RECBUFS**	2
	RECBUFSIZE	256
	XMITBUFS	1
	MAXTRANSMITS	6
IBMTRBM.NIF	**ADABUFFSIZE**	1032
	ADAPMINTRAN	4
	ADAPMAXTRAN	48
	SIZWORKBUF	2048

IBM OS/2 NetBIOS (NetBEUI) Parameter Settings

This section is an extraction from the *MPTS Configuration Guide*.

Use the following information to configure the protocol driver for the IBM OS/2 NetBIOS (NetBEUI) interface. You must calculate the RAM usage of the following parameters so that total system RAM usage does not exceed the capacity of the workstation.

- Maximum sessions
- Maximum commands
- Maximum names
- GDT selectors
- Number of names in remote directory
- UI-frame descriptors
- I-frame descriptors
- Loopback frame descriptors

Network Adapter Address (netaddress): This parameter causes the IBM OS/2 NetBIOS protocol driver to attempt to configure the local address, with the following constraints:

- Some network adapters do not allow local addresses to be set.
- Multiple protocols may attempt to set a single network adapter local address. If multiple protocols set the local address, the values set by each protocol should match. It is recommended that the local address be set in one location.
- Some network adapters allow the local address to be set in the network adapter configuration.

- The current station address is initially set to the permanent station address by the network adapter. However, the network adapter address overrides this value.
- The address must be unique among all other network adapter addresses on the network.

The default address format depends on the network adapter driver type in the network adapter common characteristics table (CCT). The default address format is:

Network Adapter Driver Type	**Address Format**
802.5	IBM Token-Ring Network format
802.3	IEEE standard notation, Ethernet address format
DIX Version 2.0 + 802.3	IEEE standard notation, Ethernet address format
DIX Version 2.0	IEEE standard notation, Ethernet address format

A letter precedes the local address specified to indicate the format of the local address:

Letter	**Meaning**
I	IEEE standard notation, Ethernet address format
T	IBM Token-Ring Network format

Default value Blank (uses the universally administered address)

Range The following ranges are valid:

- I followed by X¢020000000000¢ – X¢FEFFFFFFFFFF¢
- T followed by X¢400000000000¢ – X¢7FFFFFFFFFFF¢

Local or global Local

Type of Ethernet Driver Support (etherand_type): This parameter specifies the default protocol conventions to use when the IBM OS/2 NetBIOS protocol driver is bound to an Ethernet network adapter that supports both IEEE 802.3 and DIX 2.0 conventions.

If the Ethernet network adapter does not support both IEEE 802.3 and DIX 2.0 conventions, this parameter is ignored with no indication to the user. If this parameter is not specified, the default is IEEE 802.3.

Default value I

Range I (IEEE 802.3) or D (DIX Version 2.0)

Local or global Global

Universally Administered Address Reversed (useaddrrev): This parameter specifies whether the address used for the permanently encoded address is reversed. If this parameter is set to Yes, the permanently encoded address is reversed and then stored. For example, an address of 010203040506 is saved as 060504030201.

Default value Yes
Range Yes or No
Local or global Global

Notes:

1. The actual hardware address is not reversed.
2. Some NetBIOS applications may require this parameter to be set to No.

NetBIOS Trace Level (os2tracemask): This parameter specifies the type of information to trace.

Default value 0
Range 0 – X¢FFFF¢
Local or global Global

Maximum Sessions (sessions): This parameter specifies the maximum number of NetBIOS sessions that can be open simultaneously. Each session uses about 300 bytes.

The IBM OS/2 NetBIOS protocol driver uses a session each time it adds or finds a NetBIOS name. Increase the value of this parameter if ADD.NAME or FIND.NAME requests occur simultaneously. If a request fails due to lack of sessions, the IBM OS/2 NetBIOS protocol driver returns an interface-busy error.

Default value 40
Range 1–254
Local or global Global

Maximum Commands (ncbs): This parameter specifies the number of network control block (NCB) descriptors to allocate for managing NCBs submitted to the IBM OS/2 NetBIOS protocol driver over any single network adapter.

Default value 95
Range 1–255
Local or global Global

Maximum Names (names): This parameter specifies the maximum number of NetBIOS names that can be defined over any single network adapter. One name is reserved for defining the station address of the network adapter. If the IBM OS/2 NetBIOS protocol driver is bound to multiple network adapters, this parameter specifies the maximum number of NetBIOS names in each network adapter name table. For more information about adding names, refer to the **Maximum Sessions** parameter.

Default value	21
Range	2–254
Local or global	Global

GDT Selectors (selectors): This parameter specifies the number of internal data descriptors to allocate for global descriptor table (GDT) selectors. GDT selectors are a limited OS/2 resource. If too many GDT selectors are allocated, performance of other application programs and protocol drivers may be hindered.

The IBM OS/2 NetBIOS protocol driver uses GDT selectors to copy data into user buffers on RECEIVE, RECEIVE-ANY, and RECEIVE-ANY-ANY NCB operations. Increase the value of this parameter if many concurrent receive-type NCBs are in progress. Estimate one GDT selector for each concurrent active session.

Default value	5
Range	2–100
Local or global	Global

Full Buffer Datagrams (usemaxdatagram): This parameter specifies whether to request the full transmit buffer size for datagrams. A *datagram* is a type of data encapsulation in the network layer. If this parameter is set to No, the length of a datagram is equal to the transmit buffer size minus the associated overhead, with a maximum of 512 bytes. The overhead equals 44 bytes for the NetBIOS header, up to 36 bytes for the LAN header, and up to 6 bytes of data hold buffer (DHB) overhead. If this parameter is set to No, large messages are truncated.

Set this parameter to Yes if you require the full transmit buffer size for datagrams. The maximum length of a datagram is the transmit buffer size minus the associated overhead.

Default value	No
Range	Yes or No
Local or global	Global

Adaptive Windowing Interval (adaptrate): This parameter specifies the time, in milliseconds, between runs of the adaptive window algorithm. For each link, the IBM OS/2 NetBIOS protocol uses the adaptive window algorithm to change the **Maximum Receives Outstanding** and **Maximum Transmits Outstanding** parameter values to match the values set on the remote workstation. The adaptive window algorithm also considers the conditions of the link, including adapter receive buffers and transmission load.

When no dropped packets are detected, the adaptive window algorithm increases the **Maximum Transmits Outstanding** parameter value. If the number of dropped packets detected is more than the **Window Errors** parameter value, the adaptive window algorithm decreases the **Maximum Transmits Outstanding** parameter value. Similarly, the adaptive window algorithm changes the **Maximum Receives Outstanding** parameter value based on the time-out expiration specified by the **Acknowledgment Timer - T2** parameter value.

The **Adaptive Windowing Interval** parameter value should be large with respect to the **Response Timer - T1** and **Acknowledgment Timer - T2** parameter values. However, the **Adaptive Windowing Interval** parameter value can be less than the **Inactivity Timer - Ti** parameter value. A value of 0 disables the algorithm so that the **Maximum Receives Outstanding** and **Maximum Transmits Outstanding** parameter values are not dynamically changed.

Default value	1000
Range	0–65535
Local or global	Global

Window Errors (windowerrors): This parameter specifies the number of dropped packets that the adaptive window algorithm allows before decreasing the **Maximum Transmits Outstanding** parameter value. For example, if the **Window Errors** parameter is set to 1, then one packet can drop between runs of the algorithm without having any effect. In this example, if two packets drop, then the algorithm decreases the **Maximum Transmits Outstanding** parameter value. Increase the value of the **Window Errors** parameter on a network with a heavy workload. Decrease the value of the **Window Errors** parameter on a network with a light workload. Refer to the **Adaptive Windowing Interval** parameter for more information about the adaptive window algorithm.

Default value	0
Range	0–10
Local or global	Global

Maximum Receive Data Size (maxdatarcv): This parameter specifies the maximum size of the user data field in any frame that this node receives on a session, in bytes. The partner limits the size for the user data field in frames transmitted over the session to the smaller of this size, or the size available in its transmit buffer. NetBIOS takes into account the maximum size that is forwarded by bridges in the path. A frame is never sent that is larger than a bridge can forward.

Default value	4168
Range	512–16384
Local or global	Global

Inactivity Timer - Ti (ti): This parameter specifies the inactivity timer value, in milliseconds. The inactivity timer determines how often the IBM OS/2 NetBIOS protocol driver checks an inactive link to verify that the link is still operational. This parameter value should usually be set to 30000 to minimize unnecessary link checks. Refer to the **Response Timer - T1** parameter for more information about relationships between the timer parameters **Response Timer - T1**, **Acknowledgment Timer - T2**, and **Inactivity Timer - Ti**.

Default value	30000
Range	1000–65535
Local or global	Global

Response Timer - T1 (t1): This parameter specifies the transmission timer value, in milliseconds, for NetBIOS links. The transmission timer determines the delay before transmitting a link-level frame again if no acknowledgment is received. The three timer parameters must have the following relationship:

T2 ≤ T1 ≤ Ti

The **Response Timer - T1** parameter value should be approximately two to five times larger than the **Acknowledgment Timer - T2** parameter value.

Default value	500
Range	50–65535
Local or global	Global

Acknowledgment Timer - T2 (t2): This parameter specifies the delayed acknowledgment timer value in milliseconds. The delayed acknowledgment timer determines the length of the delay before acknowledging a received frame, when the number of frames sent is less than the **Maximum Receives Outstanding** parameter value.

Usually, the receiver of the NetBIOS message packets collects the packets until the number of packets specified by the **Maximum Receives Outstanding** parameter are received. The receiver then sends an acknowledgment to the sender. However, when the sender sends fewer frames than specified by the **Maximum Receives Outstanding** parameter, the sender waits for an acknowledgment before sending more frames.

If the **Acknowledgment Timer - T2** parameter value is too high, there can be long delays between transmissions. If the **Acknowledgment Timer - T2** parameter value is too low, the receiver may send unnecessary acknowledgments that hinder performance. Increase the value of this parameter on a busy network. Decrease the value of this parameter on a network that is not busy. Refer to the **Response Timer - T1** parameter for more information about relationships between the timer parameters **Response Timer - T1**, **Acknowledgment Timer - T2**, and **Inactivity Timer - Ti**.

Default value	200
Range	50–65535
Local or global	Global

Maximum Receives Outstanding (maxin): This parameter specifies the number of NetBIOS message packets to receive before sending an acknowledgment. This parameter value should be less than or equal to the **Maximum Transmits Outstanding** parameter value. If the **Maximum Receives Outstanding** parameter value is greater than the **Maximum Transmits Outstanding** parameter value, the **Acknowledgment Timer - T2** parameter times out frequently and wastes link bandwidth.

Refer to the **Adaptive Windowing Interval** parameter for information about dynamically changing the **Maximum Receives Outstanding** parameter value.

Default value	1
Range	1–127
Local or global	Global

Maximum Transmits Outstanding (maxout): This parameter specifies the number of NetBIOS message packets to send before expecting an acknowledgment. The **Maximum Transmits Outstanding** parameter value should be greater than the **Maximum Receives Outstanding** parameter value. If the **Maximum Transmits Outstanding** parameter value is less than the **Maximum Receives Outstanding** parameter value, the **Acknowledgment Timer - T2** parameter frequently times out and wastes link bandwidth.

Refer to the **Adaptive Windowing Interval** parameter for information about dynamically changing the **Maximum Transmits Outstanding** parameter value.

Default value	1
Range	1–127
Local or global	Global

Query Timeout (netbiostimeout): This parameter specifies the time, in milliseconds, that the IBM OS/2 NetBIOS protocol driver waits between transmission attempts. Refer to the **NetBIOS retries** parameter for more information.

Default value	500
Range	500–10000
Local or global	Global

NetBIOS Retries (netbiosretries): This parameter specifies the number of times the IBM OS/2 NetBIOS protocol driver attempts transmissions at the NetBIOS level before assuming that the receiver is not present. The transmission activities include name claims, session setups, and other similar actions. Refer to the **Query Timeout** parameter for more information.

Default value	8
Range	1–50
Local or global	Global

Number of Names in Remote Name Directory (namecache): This parameter specifies the number of remote names that the workstation contacts. If this parameter is set to 0, the remote name directory is not used. The remote name directory reduces the number of frames broadcast to the NetBIOS functional address on the network. The remote name directory sends a frame to a specific node whenever possible.

Default value	0
Range	0–255
Local or global	Global

Piggybacked Acknowledgments (piggybackacks): This parameter specifies whether the IBM OS/2 NetBIOS protocol driver sends and receives acknowledgments *piggybacked* with incoming data. This technique improves LAN performance by combining an acknowledgment for received data with a request for more data. If this parameter is set to 1, the workstation sends and requests piggybacked acknowledgments. If this parameter is set to 0, the workstation neither sends nor requests piggybacked acknowledgments.

If this parameter is set to 1 on your workstation and your partner workstation does not support piggyback acknowledgments, data acknowledgments are not piggybacked. You may want to set this parameter to 0 if your partner workstation supports piggybacked acknowledgments but is not returning a sufficient number of packets to send piggybacked acknowledgments at a satisfactory rate.

Default value	1
Range	0 or 1
Local or global	Global

UI-Frame Descriptors (datagrampackets): This parameter specifies the number of data descriptors to allocate for packeting NetBIOS datagrams into UI-frames. Increase this parameter value when the IBM OS/2 NetBIOS protocol driver sends a large number of datagrams.

Default value	2
Range	2–1000
Local or global	Global

I-Frame Descriptors (packets): This parameter specifies the number of I-frame packet descriptors that the IBM OS/2 NetBIOS protocol driver can use to build data link control (DLC) frames from NetBIOS messages.

Default value	350
Range	1–1000
Local or global	Global

Loopback Frame Descriptors (looppackets): This parameter specifies the number of internal loopback packet descriptors. Loopback packets are used when the same node is sending and receiving I-frames or UI-frames.

Default value	1
Range	1–1000
Local or global	Global

Pipeline Packets (pipeline): This parameter specifies the number of NetBIOS message packets that are prebuilt and waiting in a pipeline for each session. Increase the value of this parameter if long streams of packets are usually sent. Decrease the value of this parameter if short, occasional groups of packets are sent.

Default value	5
Range	1–200
Local or global	Global

Maximum Transmits (maxtransmits): This parameter specifies the number of packets that the IBM OS/2 NetBIOS protocol driver can send at once to a network adapter. The IBM OS/2 NetBIOS protocol queues the packets internally when this parameter value is low. The network adapter must queue the packets when this parameter value is high. Therefore, the value of this parameter depends on the capabilities of the bound network adapters.

Several performance considerations exist when a network adapter is bound to more than one protocol. The total number of packets that can be sent by the protocols to the network adapter should not exceed the capacity of the network adapter. Therefore, ensure that the total of the **Maximum Transmits** parameter values for all the bound protocols does not exceed the capacity of the network adapter.

For example, suppose that a network adapter is bound to both the IBM IEEE 802.2 protocol driver and the IBM OS/2 NetBIOS protocol driver. If the network adapter can handle 20 packets, then the sum of the **Maximum Transmits** parameter value set for the IBM IEEE 802.2 protocol driver plus the **Maximum Transmits** parameter value set for the IBM OS/2 NetBIOS protocol driver should not be greater than 20. The packet-handling capacity of each network adapter should be checked. Refer to the documentation supplied with the network adapter for more information.

Default value	6
Range	1–10000
Local or global	Global

Minimum Transmits (mintransmits): When a network adapter returns an out-of-resource condition, the IBM OS/2 NetBIOS protocol driver stops sending packets. This parameter specifies the number of transmission confirmations the IBM OS/2 NetBIOS protocol driver must receive from the network adapter before sending additional packets. The value of this parameter depends on the capabilities of the bound network adapters. This parameter value should be less than the **Maximum Transmits** parameter value. A value of 0 and a value of 1 have the same effect.

Default value 2
Range 0–9999
Local or global Global

DLC Retries (dlcretries): This parameter specifies the number of additional transmission attempts that the IBM OS/2 NetBIOS protocol driver makes before assuming that the receiving workstation DLC layer is not responding. The value of this parameter can be low on a reliable network that does not drop many packets. Increase this parameter value on a network that drops a large number of packets.

The types of network adapters on the network affect reliability. Some network adapters may drop packets because of limited buffering capabilities. For another limit on transmission attempts, refer to the **netbiosretries** parameter

Default value 5
Range 1–65535
Local or global Global

Adapter MAC Drivers

Each network adapter in LAPS has a corresponding initialization (.NIF) file containing information about the network adapter driver, including the default values for parameters associated with the network adapter driver.

To change a network adapter you can use the LAPS utility or change the value directly in PROTOCOL.INI.

IBM Token-Ring 16/4 Adapters

Use the following information to configure the network adapter driver for the IBM Token-Ring Network adapters. This parameter specifies the early token release option for the IBM Token-Ring Network 16/4 adapters. The early token release option reduces the average time that another network adapter must wait to gain access to the network. Network adapters that do not support the early token release option ignore this parameter. Specify Yes if you want this option.

Default value: No
Range: Yes or No

Adapter Mode (adapter): This parameter identifies the network adapter assignment if more than one IBM Token-Ring Network adapter is installed in the workstation.

.

Default value: Primary
Range: Primary or Alternate

Notes:

1. You must change the default value for the alternate network adapter.

2. If you install and configure using the unattended method with response files and you have an existing PROTOCOL.INI file from an earlier version of LAPS, you must specify alternate with the adapter keyword for an alternate IBM Token-Ring Network adapter.

Network Adapter address (netaddress): This parameter overrides the universally administered address. The address must be unique among all other network adapter addresses on the network. Specify the network adapter address in IBM Token-Ring Network format. If the protocol driver specifies a network adapter address for the network adapter driver, that protocol driver value becomes the default value for the network adapter driver

.

Default: Blank (uses the universally administered address)
Range: X¢400000000000¢ – X¢7FFFFFFFFFFF¢

Shared RAM Address (ram): This parameter specifies the physical RAM location on the network adapter if the network adapter default location is not adequate. This parameter is for the IBM Token-Ring Network Adapter and the IBM Token-Ring Network Adapter II only. This parameter value is a hexadecimal number, located on an 8KB boundary for the IBM Token-Ring Network Adapter or a 16KB boundary for the IBM Token-Ring Network Adapter II. The network adapter default RAM location immediately follows the read-only memory (ROM) on the next appropriate boundary. For example, if the ROM is at the default location (X¢CC00¢), the network adapter RAM default is X¢CE00¢ for the IBM Token-Ring Network Adapter and X¢D000¢ for the IBM Token-Ring Network Adapter II. For compatibility with IBM network adapter defaults, set this parameter to X¢D800¢ for the primary network adapter or X¢D400¢ for the alternate network adapter.

The IBM Token-Ring Network 16/4 Adapter and the IBM Token-Ring Network 16/4 Adapter/A do not use RAM paging. RAM size is determined by the switch settings on the network adapter.

.

Default: Blank (uses the network adapter default settings of X¢D800¢ for the primary network adapter or X¢D400¢ for the alternate network adapter)

Range: X¢A000¢ – X¢F000¢ in increments of X¢200¢

Note: You can specify only four hexadecimal digits for this parameter in LAPS configuration. Do not specify the least significant 0 of a 5-digit address. For example, specify an address of X¢D8000¢ as X¢D800¢.

The following network adapters require shared memory for a shared RAM region starting at the selected RAM location and 8KB of memory for the ROM region:

Table 10. Shared RAM for IBM Token-Ring Network Adapters

Network Adapter	Amount of Shared Memory Required
IBM Token-Ring Network Adapter	X¢2000¢ (8KB)
IBM Token-Ring Network Adapter II	X¢4000¢ (16KB)
IBM Token-Ring Network 16/4 Adapter	X¢2000¢ (8KB) or X¢4000¢ (16KB) or X¢8000¢ (32KB) or X¢10000¢ (64KB)

The RAM location that is selected depends on the RAM page-size switch setting on the network adapter. If 8KB or 16KB is set, then all locations are valid. If 32KB or 64KB is set, then the address must be on a 32KB or 64KB boundary, respectively.

The specified location must not conflict with the address of any other adapter configured and installed in the workstation. The selection specified cannot overlap with the ROM region used by the IBM Token-Ring Network adapters. The ROM region is determined by the switch settings on the network adapter. It is suggested that the ROM region for the primary network adapter be set to X¢CC000¢ and that the ROM region for the secondary network adapter be set to X¢DC000¢. However, this region can be set to another available 8KB location in the range from X¢C0000¢ through X¢DC000¢.

Use the guidelines in the following table to select the shared RAM location for the appropriate network adapter:

Table 11 (Page 1 of 2). Shared RAM Location Guidelines

Network Adapter	Guidelines
IBM Token-Ring Network 16/4 Adapter	• Depending on the setting of the shared RAM page-size switches on the network adapter, the following number of bytes of shared RAM are required: – 8KB (X¢2000¢ bytes) – 16KB (X¢4000¢ bytes) – 32KB (X¢8000¢ bytes) – 64KB (X¢10000¢ bytes) The **Shared RAM address** parameter cannot conflict with any of the following areas, which also require shared memory: – The fixed 24KB region that is required (X¢C0000¢ – X¢C5FFF¢) if a VGA adapter is also being used – The fixed 16KB region that is required (X¢C0000¢ – X¢C3FFF¢) if an EGA adapter is also being used. The location that you specify must not have been previously specified, and it must reside on its corresponding address boundary (or larger). For example, X¢D800¢ is a 32KB address boundary. • Use the default of 16KB for the network adapter switch setting. If you choose this setting, the network adapter mapping function of 64KB to 16KB is invoked. You have the advantage of more storage with less shared RAM used in PC memory. 16KB pages from the 64KB are mapped into the 16KB RAM. • You must use X¢D000¢ as the shared RAM location if you set the page-size switch settings to 64KB. • When the switch settings on the IBM Token-Ring Network 16/4 Adapter are set to 64KB, there is a conflict if the switch settings for the ROM region are set to the last 8KB within that same 64KB region. In this case, the shared RAM region used will be reduced to 56KB, allowing the 8KB necessary for the ROM area (X¢DE000¢).

Table 11 (Page 2 of 2). Shared RAM Location Guidelines	
Network Adapter	**Guidelines**
IBM Token-Ring Network Adapter II	Select any option that has not been used. Make sure that the starting address you select was not specified for ROM; 16KB are used.
IBM Token-Ring Network Adapter	Select any option that has not been used. Make sure that the starting address you select is not specified for ROM; 8KB are used.
IBM PC Network Adapter	Do not set the location to X¢D0000¢ or X¢D4000¢ if an IBM PC Network adapter is configured or if you intend to configure an IBM PC Network adapter. The IBM PC Network adapter always uses X¢D0000¢ – X¢D7FFF¢. It also uses X¢CC000¢ – X¢CDFFF¢ if the IBM PC Network adapter was configured as the primary network adapter. If the IBM PC Network adapter was configured as the alternate network adapter, location X¢DC000¢ – X¢DDFFF¢ is used.

Maximum Number of Queued Transmits (maxtransmits): This parameter specifies the maximum number of transmit queue entries for the network adapter driver. For example, suppose that a network adapter is bound to both the IBM IEEE 802.2 protocol driver and the IBM OS/2 NetBIOS protocol driver. If the network adapter can handle 20 packets, then the sum of the **Maximum Transmits** parameter value set for the IBM IEEE 802.2 protocol driver plus the **Maximum Transmits** parameter value set for the IBM OS/2 NetBIOS protocol driver should not be greater than 20. The packet-handling capacity of each network adapter should be checked. Refer to the documentation supplied with the network adapter for more information.

.

Default: 6
Range: 6–50

Number of Receive Buffers (recvbufs): This parameter specifies the number of receive buffers. Any memory remaining on the network adapter after other storage requirements have been satisfied is configured as extra receive buffers.

.

Default: 2
Range: 2–60

Receive Buffer Size (recvbufsize): This parameter specifies the length of the data part of each receive buffer in the shared RAM area of the network adapter, in bytes. This area of a receive buffer holds the data part of an incoming packet. It does not include the 8-byte overhead needed by the network adapter. The value of this parameter must be a multiple of eight. The maximum size of a receive buffer is 2048 bytes minus an 8-byte overhead. Since the incoming packet can span multiple receive buffers, it is not usually necessary to change this parameter default value. If this parameter value is set too high for the network adapter, a configuration error occurs.

.

Default: 256
Range: 256–2040 in increments of 8

Number of Adapter Transmit Buffers (xmitbufs): This parameter specifies the number of transmit buffers to allocate on the network adapter. Allocating a second transmit buffer may improve transmission performance, but it also reduces the amount of memory available for storing received packets.

.

Default: 1
Range: 1–16

Transmit Buffer Size (xmitbufsize): This parameter specifies the length of the data part of each transmit buffer in the shared RAM area of the network adapter in bytes. This area of a transmit buffer holds the data part of an outgoing packet. It does not include the 8-byte overhead needed by the network adapter but does include the entire frame that is to be transmitted. The value of this parameter must be a multiple of eight. The maximum size of a transmit buffer depends on the network adapter in use. The older IBM Token-Ring Network Adapter, IBM Token-Ring Network Adapter II, and IBM Token-Ring Network Adapter/A allow only 2040 bytes. The newer IBM Token-Ring Network 16/4 Adapter and IBM Token-Ring Network 16/4 Adapter/A allow 4456 bytes at the 4Mbps setting and 17952 bytes at the 16Mbps setting.

.

Default: Blank (uses the lesser of 17952 or 25% of available RAM on the network adapter)
Range: 256–17952 in increments of 8

Note: Specify a value for the transmit buffer that is large enough to contain the following items:

- LAN header
- DLC header

- Largest data field

IBM PS/2 Adapter for Ethernet Networks

Use the following information to configure the network adapter driver for the IBM PS/2 adapter for Ethernet networks.

Micro Channel Slot Number (slotnumber): This parameter specifies the slot number of the network adapter in the workstation. Specify this parameter only when more than one IBM PS/2 adapter for Ethernet networks is installed in the workstation.

.

Default: Blank (no default)
Range: 1– 8

Maximum Number of Queued General Requests (maxrequests): This parameter specifies the maximum number of concurrently outstanding general request queue entries.

.

Default: 8
Range: 6– 24

Maximum Number of Queued Transmits (maxtransmits): This parameter specifies the maximum number of concurrently outstanding TransmitChain commands that the network adapter driver can queue.

.

Default: 12
Range: 6– 24

Number of Receive Buffers (receivebuffers): This parameter specifies the number of receive buffers allocated in workstation memory.

.

Default: 12
Range: 6– 24

Receive Buffer Size (receivebufsize): This parameter specifies the size, in bytes, of each receive buffer. Specify a value large enough to hold the *average* size of received frames.

.

Default: 256
Range: 256–1536 in increments of 256

Receive Chain Queue Entries (receivechains): This parameter specifies the number of entries in the receive chain header queue.

.
Default: 12
Range: 6–24

B.4 RAM Usage for IBM OS/2 NetBIOS Protocol Driver (NetBEUI)

This section describes how to calculate RAM usage for NetBEUI.

Note: IBM Token-Ring Network support in OS/2 Extended Edition allocated resources for all the drivers from shared RAM on the IBM Token-Ring Network 16/4 Adapter/A. However, with LAPS, the MAC driver for IBM Token-Ring Network 16/4 Adapter/A still uses shared RAM for allocation of its resources, but IBM OS/2 NetBIOS protocol drivers (NetBEUI) use system memory for allocation of their resources. With each protocol driver having its own area of system memory, limitations of sharing resources between the protocol drivers, such as links, are eliminated.

To determine the values to specify for the capacity parameters, you need to know the RAM usage of the IBM OS/2 NetBIOS protocol driver so that total system RAM usage does not exceed the capacity of your system. The following table lists the RAM usage (in bytes) per item as well as the parameter corresponding to each item as it is displayed on the Parameters for IBM OS/2 NetBIOS window.

Table 12 (Page 1 of 2). IBM OS/2 NetBIOS (NetBEUI) RAM Usage for 386/486/Pentium

Data Area Item	Related Parameter	RAM Usage (bytes)
Overhead for adapter work and communication area	Not applicable	6650
Each command	**Maximum commands**	70

Table 12 (Page 2 of 2). IBM OS/2 NetBIOS (NetBEUI) RAM Usage for 386/486/Pentium		
Data Area Item	**Related Parameter**	**RAM Usage (bytes)**
Each name	**Maximum names**	100
Each session	**Maximum sessions**	700
Each selector	**GDT selectors**	10
Each name in remote name directory	**Number of names in remote name directory**	60
Each I-packet	**I-frame descriptors**	110
Each UI-packet	**UI-frame descriptors**	130
Each loopback packet	**Loopback frame descriptors**	150

The total system RAM usage is the sum of the RAM usage for all data area items.

For 386/486/Pentium-based workstations, the sum of RAM usage for all data area items except maximum sessions must be less than or equal to 64KB, as illustrated in Figure 60 on page 201.

6650 bytes overhead
70 bytes × **Maximum commands**
100 bytes × **Maximum names**
10 bytes × **GDT selectors**
60 bytes × **Number of names in remote name directory**
110 bytes × **I-frame descriptors**
130 bytes × **UI-frame descriptors**
150 bytes × **Loopback frame descriptors**

≤ 64KB

6650 bytes overhead
70 bytes × **Maximum commands**
100 bytes × **Maximum names**
700 bytes × **Maximum sessions**
10 bytes × **GDT selectors**
60 bytes × **Number of names in remote name directory**
110 bytes × **I-frame descriptors**
130 bytes × **UI-frame descriptors**
150 bytes × **Loopback frame descriptors**

Total system RAM usage

Figure 60. System RAM Usage Calculation for a 386/486/Pentium-Based Workstation. Using the IBM OS/2 NetBIOS Protocol Driver

Appendix C. RAID Technology and Disk Performance

RAID stands for Redundant Arrays of Inexpensive Disks, and provides a method of classifying the different ways of using multiple disks to increase availability and performance.

Disk Arrays

The capacity of single large disks has grown rapidly, but the performance improvements have been modest, when compared to the advances made in the other subsystems that make up a computer system. The reason for this is that disks are mechanical devices, affected by delays in seeks and the rotation time of the media.

In addition, disks are often among the least reliable components of the computer systems, yet the failure of a disk can result in the unrecoverable loss of vital business data, or the need to restore a tape backup with consequent delays. The use of arrays of inexpensive disks can offer a solution to these concerns.

There is nothing unusual about connecting several disks to a computer to increase the amount of storage. Mainframes and minicomputers have always had banks of disks. The disk subsystem is called a disk array when several disks are connected and accessed by the disk controller in predetermined patterns designed to optimize performance and/or reliability.

The driving force behind disk array technology is the observation that it is cheaper to provide a given storage capacity or data rate with several small disks connected together than with a single disk.

RAID Classifications

Disk arrays seem to have been invented independently by a variety of groups. One specifically, the Computer Architecture group at the University of California, Berkeley invented the term *Redundant Arrays of Inexpensive Disk* (RAID).

The original RAID classification described five levels of RAID (RAID-1 through 5). To these have been added RAID-0 (data-striping), RAID-1 Enhanced (data stripe mirroring) and Orthogonal RAID-5 (which includes extra redundancy of components such as disk adapters). RAID-0 is not a pure RAID type, since it does not provide any redundancy.

Different designs of arrays perform optimally in different environments. The two main environments are those where a high I/O rate is needed. That is:

.

1. High transfer rates are very important
2. High I/O rates is needed - that is for applications requesting short length random records

The following table shows the RAID array classifications.

Table 13. RAID Classifications

RAID Level	**Description**
RAID-0	Block Interleave Data Striping without Parity
RAID-1	Disk Mirroring/Duplexing
RAID-1 Enhanced	Data Strip Mirroring
RAID-2	Bit Interleave Data Striping with Hamming Code
RAID-3	Bit Interleave Data Striping with Parity Check
RAID-4	Block Interleave Data Striping with One Parity Disk
RAID-5	Block Interleave Data Striping with Skewed Parity
Orthogonal RAID-5	RAID-5 with Additional Redundancy (such as disk adapters)

In a LAN Server environment the relevant RAID configurations are RAID-0, RAID-1 and RAID-5 and function as follows:

RAID-0: RAID-0 is the data organization term used when striping data across multiple disk drives, without parity protection. Data striping improves performance with large files since reads/writes are overlapped across all disks. However, reliability is decreased as the failure of one disk will result in a complete failure of the disk subsystem.

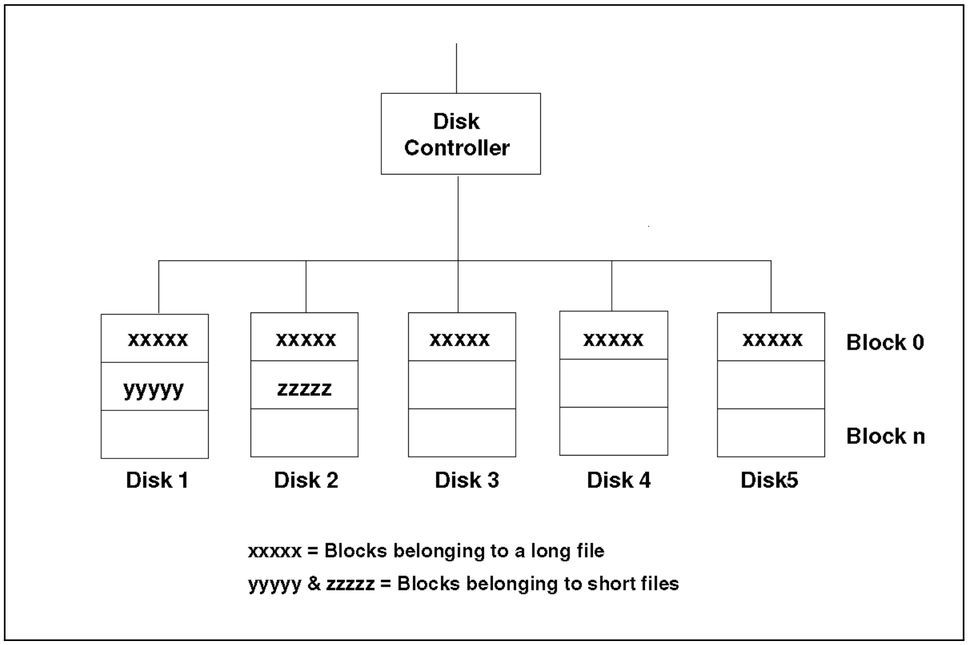

Figure 61. RAID-0 (Block Interleave Data Striping without Parity)

RAID-1: RAID-1 is the term used with disk mirroring or duplexing at the hardware level. Whenever the computer makes an update to a disk, it can arrange to duplicate that update to a second disk, thus mirroring the original. This level of RAID is implemented in LAN Server fault tolerance.

Either disk can fail, and the data is still accessible. Additionally, because there are two disks, a read request can be serviced from either disk, thus leading to improved throughput and performance on the disk subsystem. However, the down side is the cost of using 50% of disk storage space for mirroring.

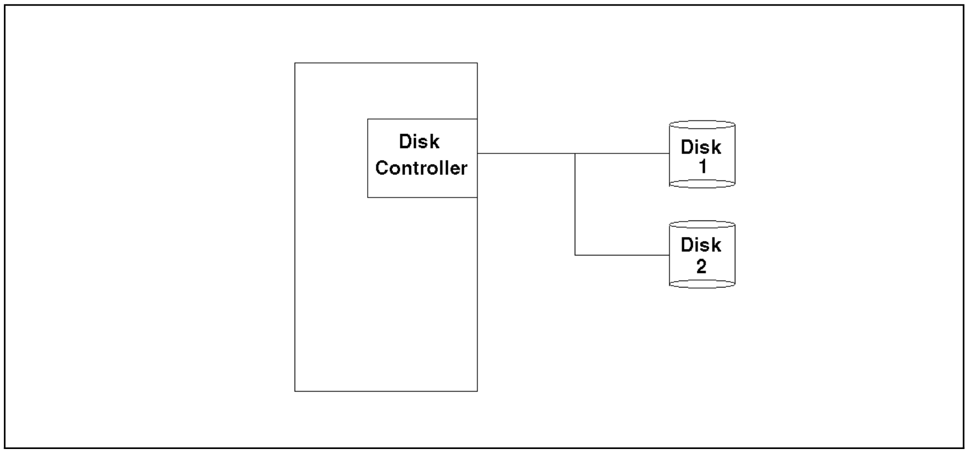

Figure 62. RAID-1 (Disk Mirroring)

In the case of the IBM RAID controller there are two separate disk SCSI channels available from one card and so duplexing is possible from one disk controller card. Some might argue this is not true duplexing, but is mirroring on separate disk control channels. In any case, the IBM RAID controller provides an extremely cost effective and adaptable method for arranging data on a disk subsystem using RAID-1.

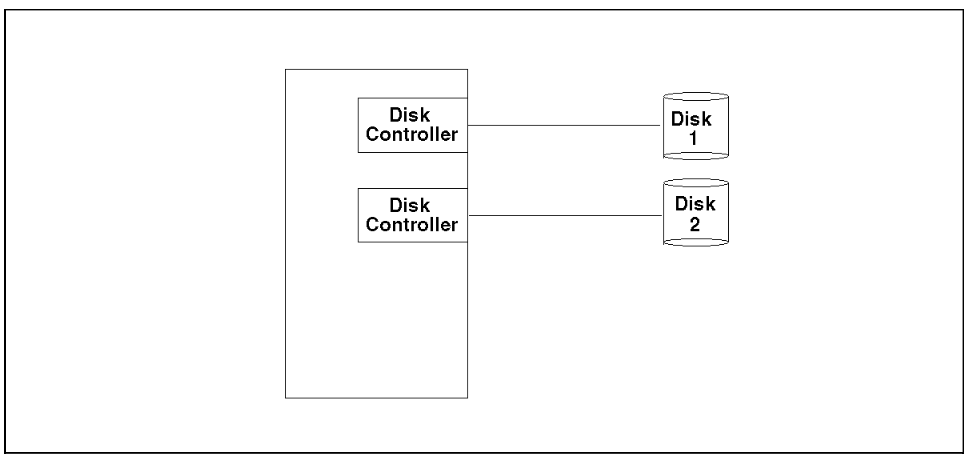

Figure 63. RAID-1 (Disk Duplexing)

RAID-5: RAID-5 is the term used when striping data across three or more disks with skewed parity. This means the data organization is essentially the same as RAID-0, but there is an additional element of parity checking. The parity checking is used to encode the data and guard it against loss, and is referred to as a checksum, disk parity or error correction code (ECC). The principle is the same as memory parity, where the data is guarded against the loss of a single bit of data. In RAID-5, the parity information is stored on the disk array to guard against data loss, and skewing is used to remove the bottleneck that would be created by having all the parity information stored on a single drive.

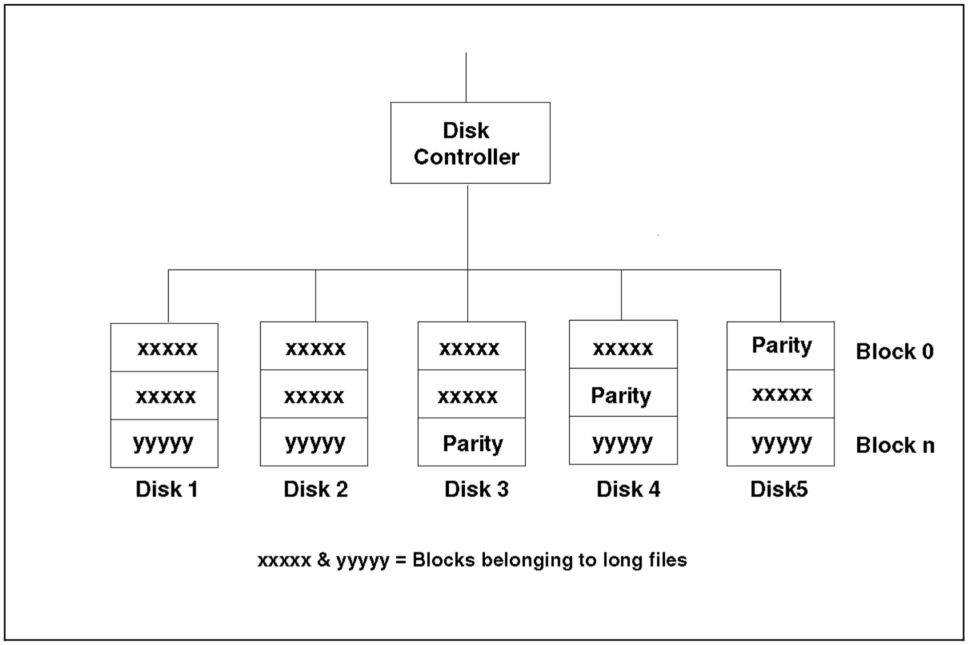

Figure 64. RAID-5 (Block Interleave Data Striping with Skewed Parity)

Using RAID technology to provide striping of data across multiple disks will often improve performance as well as enhance data integrity in an environment where data is predominantly read off the disk without a subsequent update (write).

RAID Performance Characteristics

The following is a summary of each RAID type:

RAID-0: Block Interleave Data Striping without Parity

.

- Fastest data-rate performance
- Allows seek and drive latency to be performed in parallel
- Significantly outperforms single large disk

RAID-1: Disk Mirroring/Disk Duplexing and Data Strip Mirroring (RAID-1, Enhanced)

.

- Fast and reliable, but requires 100% disk space overhead
- Data copied to each set of drives
- With LAN Server, performance almost as fast as RAID-0 due to split seeks (split seeks send half of the operation to one side of the mirrored drives, and the other half to other side)
- No performance degradation with a single disk failure
- RAID-1 Enhanced provides mirroring with odd number of drives

RAID-5: Block Interleave Data Striping with Skewed Parity

.

- Best for random transactions
- Poor for large sequential reads if request is larger than block size
- Better write performance than RAID-3 and RAID-4
- Block size is key to performance, must be larger than typical request size
- Performance degrades in recovery mode that is when a single drive has failed

Orthogonal RAID-5: RAID-5 with multiple orthogonal disk adapters

.

- All the benefits of RAID-5
- Improved performance (due to load being spread across disk adapters)
- Improved reliability due to redundancy of disk adapters as well as disks

Table 14. Summary of RAID Performance Characteristics

RAID Level	**Capacity**	**Large Transfers**	**High I/O Rate**	**Data Availability**
Single Disk	Fixed (100%)	Good	Good	
RAID-0	Excellent	Very Good	Very Good	Poor
RAID-1	Moderate (50%)	Good	Good	Good
RAID-5	Very Good	Very Good	Good	Good
Orthogonal RAID-5	Very Good	Very Good	Good	Very Good

LAN Server Fault Tolerance and Performance

The OS/2 LAN Server 3.0/4.0 Advanced provides a disk fault tolerance subsystem. It implements RAID-1, which works in conjunction with the 386 HPFS file system and the OS/2 DASD Manager. It provides the capabilities to perform disk mirroring and disk duplexing.

.

- Disk mirroring - The duplication of a single logical drive or volume on two partitions which do not reside on the same physical disk. The physical disks on which the partitions reside are controlled by the same disk controller.
- Disk duplexing - A superset of drive mirroring, with the additional restriction that the two disks on which the two partitions reside are controlled by two different disk controllers.

The only way performance improvements can be experienced when using LAN Server fault tolerance is through the read and write operations made to disk.

- When a 386 HPFS read request is performed within a fault tolerance system then the drive with the least activity is used. This provides improved performance as the data scheduled on the more busy drive can now be processed while new data is be continually handled by the free drive.That is to say, the server will perform split read operations. This will ensure that will be read from the secondary mirrored partition if the primary partition is busy or unavailable. With a larger cache present this is not so significant as most of the read requests will be satisfied by the caching mechanism.

- When write requests are made to 386 HPFS using fault tolerance the data has to be written to two drives rather than a single drive. The two write requests are asynchronously processed so that the application program′s original request to the operating system doesn′t have to wait longer than a single disk configuration. However, this may not generate a notable delay but some CPU resource would be required to process an extra I/O request.

If the lazy write feature is activated in the 386 HPFS caching then the extra write request is not significant to impact the CPU performance as disk writes are performed when the CPU is idle. In general, you should not expect a performance improvement in using the LAN Server 3.0/4.0 fault tolerance feature.

List of Abbreviations

CCT	Common Characteristics Table
DCDB	Domain Controller Database
DLC	Data Link Control
DMA	Direct Memory Access
DRAM	Dynamic RAM
ECC	Error Correcting code
ESDI	Enhanced Small Device Interface
FAT	File Allocation Table
FSD	File System Driver
GDT	Global Descriptor Table
HPFS	High Performance File System
IDE	Intelligent Drive Interface
I-Frame	Information Frame
I/O	Input/Output
KB/s	Kilo bytes per second
LAPS	LAN Adapter and Protocol Support
L1	Level 1
LSP	LAN Support Program
LS Ultimedia	LAN Server Ultimedia
MAC	Medium Access Control
MB/s	Mega bytes per second
MCA	Micro Channel Architecture
MPTS	Multi Protocol Transport System
NCB	Network Control Block
NIC	Network Interface Card
PCI	Peripheral Component Interconnect
RAID	Redundant Arrays of inexpensive Disk
RRS	Resource Reservation System
RFC	Request for Comments
SCSI	Small Computer System Interface
SMB	Server Message Block
SRAM	Static RAM

Index

Special Characters

Numerics

A

B

C

D

J

L

Q

R

S

T

U

V

W

X

Printed in U.S.A.

GG24-4430-00

Table Definitions

id	File	Page	References
C3HEAD	CH3NEW	78	78
C3ROW1	CH3NEW	78	78
C3ROW2	CH3NEW	78	79
C3ROW3	CH3NEW	78	79
HEAD	APX3	175	175, 176, 176
ROW1	APX3	175	175
ROW2	APX3	175	176, 176
ROW3	APX3	175	176, 176
CHEAD	APX3	179	179, 179, 179, 179
CROW1	APX3	179	179, 179
CROW2	APX3	179	179
CROW3	APX3	179	179, 180
PHEAD	APX3	181	181
PROW1	APX3	181	181
PROW2	APX3	181	182, 182
PROCES	APX3	199	199

Figures

id	File	Page	References	
TOP10A	PERFPREF	xvi	1	
BEGIN2	PERFPREF	xvii	2	
INTRO1	CH1NEW	4	3	
PERFMCE	CH1NEW	6	4	6, 7
PRFMA	CH1NEW	8	5	9
PERFME2	CH1NEW	9	6	9, 9
PROBMET	CH1NEW	11	7	
SYNCHRO	CH2NEW	24	8	24
PCIBUS	CH2NEW	28	9	27
DISK2	CH2NEW			

		34	10	
LSENTRY	CH3NEW	42	11	
LSADV	CH3NEW	48	12	
SIDEB	CH3NEW	50	13	
LSULTI	CH3NEW	51	14	
INI386	CH3NEW	54	15	70
FLSYST	CH3NEW	56	16	
FILESY	CH3NEW	57	17	
SMBSRV	CH3NEW	59	18	
386HPFS	CH3NEW	60	19	
CCHE	CH3NEW	62	20	
CACHING	CH3NEW	64	21	
CACHE1	CH3NEW	67	23	
NBS	CH3NEW	73	24	72
NETX	CH3NEW	76	25	75
SMB	CH3NEW	85	26	
SMBFL1	CH3NEW	86	27	
SMBFL2	CH3NEW	87	28	
SMBCB	CH3NEW	88	29	
LOGON30	CH3NEW	91	30	90
NB10	CH4NEW	105	32	112
NCBLIM	CH4NEW	106	33	
NBOTCP	CH4NEW	108	34	107
NBTCP4	CH4NEW	111	35	
ALLADP	CH4NEW	114	36	117
ADAPTER	CH4NEW	116	37	
BMADP	CH4NEW	118	38	
LSADPT	CH4NEW	121	39	
ADPTER	CH4NEW	122	40	122, 122
WORKBUF	CH5NEW	126	41	
BIGBUF	CH5NEW	127	42	

DLRBUF	CH5NEW	129	43	128
DLBGBU	CH5NEW	130	44	
PEERBUF	CH5NEW	133	45	
DTAGLAN	CH6NEW	137	46	
SMBTOOL	CH6NEW	138	47	
SMBTLS	CH6NEW	140	48	
CPU	CH6NEW	142	49	
DISK	CH6NEW	143	50	
RAMI	CH6NEW	145	51	
PERFCE2	CH6NEW	146	52	
DTAFLW	CH6NEW	149	53	148
SESSION	CH6NEW	152	54	
SRV2CL	CH6NEW	153	55	
TESTS	CH6NEW	154	56	
TRANSF	CH6NEW	160	57	
NCBFORM	APX1	163	58	164
BIT	APX1	164	59	164
NBIOS	APX3	201	60	200
RAID0	APX4	205	61	
RAID1A	APX4	206	62	
RAID1B	APX4	206	63	
RAID5	APX4	207	64	

Headings

id	File	Page	References
NOTICES	PERFENV	xiii	Special Notices ii
BIBL	PERFPREF	xviii	Related Publications
CHAP01	CH1NEW	1	Chapter 1, Introduction xv
UNSEPER	CH1NEW	3	1.2, Understanding Server Performance in General
PETUAPP	CH1NEW	10	1.3, General Performance Tuning Approach

APXE	APX3	175	Appendix B, Performance Parameters
INIS	APX3	175	B.1, IBMLAN.INI
INIS2	APX3	179	B.2, CONFIG.SYS
INIS3	APX3	181	B.3, PROTOCOL.INI
IPROT	APX3	182	IBM OS/2 NetBIOS (NetBEUI) Parameter Settings
RAI	APX4	203	Appendix C, RAID Technology and Disk Performance
LASEFAT	APX4	209	LAN Server Fault Tolerance and Performance

Index Entries

I038	PERFVARS		
		i	(1) SMB 47, 70, 83, 83, 84, 125, 147, 147, 147, 154
I047	PERFVARS		
		i	(1) Throughput 6, 125, 125, 125, 125, 128, 129
I048	PERFVARS		
		i	(1) Tuning Tools 72, 72, 135, 135, 137, 138, 141, 142, 143, 154
I046	PERFVARS		
		i	(1) Response Time

Spots

id	File	Page	References
PAGENO	GG244430 SCRIPT		
		213	(no text)

Tables

id	File	Page	References
PARAM1	APX3		
		199	12

Processing Options

Runtime values:

- Document fileid .... GG244430 SCRIPT
- Document type .... USERDOC
- Document style .... NEWSTYLE
- Profile EDFPRF30
- Service Level 0029
- SCRIPT/VS Release 4.0.0
- Date 94.11.02
- Time 14:47:38
- Device 3820A
- Number of Passes 4
- Index .... YES
- SYSVAR D .... YES
- SYSVAR G .... INLINE
- SYSVAR V .... ITSCEVAL

Formatting values used:

- Annotation .... NO
- Cross reference listing .... YES
- Cross reference head prefix only .... NO
- Dialog .... LABEL
- Duplex .... YES
- DVCF conditions file .... (none)
- DVCF value 1 .... (none)
- DVCF value 2 .... (none)
- DVCF value 3 .... (none)
- DVCF value 4 .... (none)
- DVCF value 5 .... (none)
- DVCF value 6 .... (none)
- DVCF value 7 .... (none)

DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id′s	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	.,
Read cross-reference file	(none)
Running heading/footing rule	NONE
Show index entries	NO
Table of Contents (maximum level)	3
Table list on new page	YES
Title page (draft) alignment	RIGHT
Write cross-reference file	(none)

Imbed Trace

Page 0	PERFGLOB
Page 0	PERFVARS
Page 0	PERFENV
Page i	PERFEDNO
Page ii	PERFABST
Page xiii	PERFSPEC
Page xiv	PERFTMKS
Page xiv	PERFPREF
Page xix	PERFACKS
Page xx	CH1NEW
Page 18	CH2NEW
Page 39	CH3NEW
Page 101	CH4NEW
Page 124	CH5NEW
Page 134	CH6NEW
Page 162	APX1
Page 173	APX3
Page 201	APX4
Page 210	PERFABRV
Page 221	PERFEVAL