OS/390

Network File System Performance Tuning Guide

SC26-7255-00

OS/390

Network File System Performance Tuning Guide

SC26-7255-00

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

First Edition, September 1998

This edition applies to Version 2 Release 6 of OS/390 (5647-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

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

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
RCF Processing Department
M86/G26 050
5600 Cottle Rd.
SAN JOSE, CA 95193–0000
United States of America

Or you can send your comments electronically to Internet: starpubs@vnet.ibm.com

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:
- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Notices

References in this publication 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 that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the responsibility of the user.

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, U.S.A.

This document is intended to help use the OS/390 Network File System. This document contains tuning information that helps to diagnose performance problems of the OS/390 Network File System.

Measured performance data contained in this document was determined in a controlled and isolated network environment which is not necessarily representative of customer installations. Further, every actual installation is unique in its product mix, configurations, and use of its systems. **Therefore, the same performance results obtained in this document may not be obtained in actual installations or under different laboratory situations.**

Trademarks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

3090	OpenEdition
AIX	OS/2
DFSMS/MVS	PS/2
DFSMShsm	RACF
ESA/390	RAMAC 3
ESCON	Resource Measurement Facility
IBM	RMF
MVS/DFP	RS/6000
MVS/ESA	System/390
MVS/SP	

The following terms are trademarks of other companies:

DECstation	Digital Equipment Corporation
DEC ULTRIX	Digital Equipment Corporation
HP 9000 Series 700	Hewlett-Packard Corporation
HP/UX	Hewlett-Packard Corporation
Macintosh	Apple Computer Inc.
Microsoft	Microsoft Corporation
NFS	Sun Microsystems, Incorporated

PC-NFS	Sun Microsystems, Incorporated
Portmapper	Sun Microsystems, Incorporated
POSIX	Institute of Electrical and Electronics Engineers
Solaris	Sun Microsystems, Incorporated
Sun	Sun Microsystems, Incorporated
SunOS	Sun Microsystems, Incorporated
UNIX	Unix System Laboratories, Inc.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Summary of Changes

Summary of Changes for OS/390 Version 2 Release 6 Network File System Performance Tuning Guide, order number SC26-7255–00.

New Information

In this release, OS/390 Network File System (OS/390 NFS) is enhanced with the following functions:

- WebNFS server and internet

  WebNFS eliminates the overhead of PORTMAP and MOUNT protocols, making the protocol easier to use through corporate firewalls. It also reduces the number of LOOKUP requests that are required to identify a particular file on the server.
- File locking over the OS/390 NFS server

  The OS/390 Network File System Network Lock Manager (OS/390 NFS NLM) and the OS/390 Network File System Network Status Monitor (OS/390 NFS NSM) are Remote Procedure Call (RPC) based servers that execute as autonomous *daemon* servers on NFS server system. NSM and NLM work together to provide file locking and access control capability with OS/390 NFS server.
- File name extension mapping support

  The OS/390 Network File System server is enhanced to provide file extension mapping for members of a Partition Data Set (PDS) and Partition Data Set Extended (PDSE) that are mounted via the Network File System from a client machine. The file name extension mapping capability is provided by the use of *side files* on the MVS host specified by the system administrator and the user during the MOUNT operation.
- Sun NFS Version 3 protocol support for both client and server functions

  The NFS Version 3 protocol is a revision of the NFS version 2 protocol. It supports larger files and file systems by allowing 64-bit sizes and offsets. The NFS Version 3 protocol enhances performance by returning attributes for every procedure, thus reducing the number of calls requesting modified file attributes. The NFS version 3 also enhances performance by adding support to allow the NFS server to do asynchronous writes and by relaxing the limitations on transfer sizes.
- TCP support

  In addition to User Datagram Protocol (UDP), OS/390 NFS server supports connection-oriented reliable Transmission Control Protocol (TCP). Any client platform with TCP support can choose to access the OS/390 NFS server using TCP.

Changed Information

- The title of this publication has changed from *DFSMS/MVS Network File System Performance Tuning Guide* to *OS/390 Network File System Performance Tuning Guide*.
- As part of the name change of OpenEdition to OS/390 UNIX System Services, occurrences of OS/390 OpenEdition have been changed to OS/390 UNIX System Services or its abbreviated name, OS/390 UNIX. OpenEdition can continue to appear in messages, panel text, and other code with OS/390 UNIX System Services.

This publication is a revision in support of the functional changes introduced with OS/390 NFS. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change. For a book that has been updated in softcopy only, the vertical lines indicate changes made since the last printed version.

This revision also includes maintenance and editorial changes.

Preface

About This Book

This publication provides system programmers and operators with methods to assess the performance of their NFS** environment, determine their OS/390 NFS performance requirements, and to evaluate the trade-offs between cost and performance. It also provides examples of methods to evaluate the performance of OS/390 NFS.

In this document, NFS client or NFS server refers to NFS client or NFS server in general, which includes OS/390 NFS. When necessary, OS/390 will be specifically specified.

Required Product Knowledge

To use this publication effectively, you should be familiar with the IBM MVS operating system, the IBM Time Sharing Option (TSO), and their commands. In addition, you should be familiar with the basic concepts of the NFS protocol and networking (TCP/IP) as well as the NFS client operating systems accessing the OS/390 NFS.

Related Publications

For general information on the OS/390 NFS, refer to:

Publication Title	Order Number
OS/390 Network File System User's Guide	SC26-7254
OS/390 Network File System Customization and Operation	SC26-7253
DFSMS/MVS General Information	GC26-4900

For performance related information, refer to:

Publication Title	Order Number
TCP/IP: Performance Tuning Guide	SC31-7188
AIX for RISC System/6000 Performance Monitoring and Tuning Guide	SC23-2365

For information about TCP/IP, refer to:

Publication Title	Order Number
AIX Operating System TCP/IP User's Guide	SC23-2309
TCP/IP for MVS: Programmer's Reference	SC31-7135
Introducing IBM's Transmission Control Protocol/Internet Protocol Products for OS/2, MVS, VM	GC31-6080
IBM Transmission Control Protocol/Internet Protocol Version 2.0 for OS/2: Users Guide	SC31-6076

Publication Title	Order Number
IBM Transmission Control Protocol/Internet Protocol Version 2.0 for OS/2: Installation and Administration	SC31-6075
IBM Transmission Control Protocol/Internet Protocol Version 2.1.1 for DOS: Users Guide	SC31-7045
IBM Transmission Control Protocol/Internet Protocol Version 2.1.1 for DOS: Programmer's Reference	SC31-7046
IBM Transmission Control Protocol/Internet Protocol Version 2.1.1 for DOS: Installation and Administration	SC31-7047
TCP/IP for MVS: Customization and Administration Guide	SC31-7134
TCP/IP for MVS: User's Guide	SC31-7136

For information about OS/390 UNIX System Services, refer to:

Publication Title	Order Number
OS/390 UNIX System Services Programming: Assembler Callable Services Reference	SC28-1899
OS/390 UNIX System Services Programming Tools	SC28-1904
OS/390 UNIX System Services Command Reference	SC28-1892
OS/390 UNIX System Services User's Guide	SC28-1891
OS/390 UNIX System Services Planning	SC28-1890

For information about OS/390, refer to:

Publication Title	Order Number
OS/390 MVS JCL Reference	GC28-1757
OS/390 MVS Installation Exits	SC28-1753
OS/390 MVS System Messages, Vol 1 (ABA-ASA)	GC28-1784
OS/390 MVS System Messages, Vol 2 (ASB-EWX)	GC28-1785
OS/390 MVS System Messages, Vol 3 (GDE-IEB)	GC28-1786
OS/390 MVS System Messages, Vol 4 (IEC-IFD)	GC28-1787
OS/390 MVS System Messages, Vol 5 (IGD-IZP)	GC28-1788
OS/390 MVS Auth Assembler Services Guide	GC28-1763

For information about the IBM Time Sharing Option (TSO), refer to:

Publication Title	Order Number
OS/390 TSO/E User's Guide	SC28-1968

For information about SMP/E, refer to:

Publication Title	Order Number
OS/390 SMP/E User's Guide	SC28-1740

For information about RS/6000 communications, refer to:

Publication Title	Order Number
AIX Commands Reference for RISC System/6000, Volume 1	GC23-2376
AIX Commands Reference for RISC System/6000, Volume 2	GC23-2366
AIX Commands Reference for RISC System/6000, Volume 3	GC23-2367
AIX Communications Concepts and Procedures for IBM RISC System/6000	GC23-2203

For information about DFSMS, refer to:

Publication Title	Order Number
DFSMS/MVS DFSMSdfp Diagnosis Reference	LY27-9606
DFSMS/MVS DFSMSdfp Storage Administration Reference	SC26-4920
DFSMS/MVS DFSMShsm Diagnosis Guide	LY27-9607
DFSMS/MVS Macro Instructions for Data Sets	SC26-4913
DFSMS/MVS DFSMSdfp Advanced Services	SC26-4921
DFSMS/MVS Using Data Sets	SC26-4922

For information about RACF, refer to:

Publication Title	Order Number
OS/390 Security Server (RACF) Security Administrator's Guide	SC28-1915
OS/390 Security Server (RACF) System Programmer's Guide	SC28-1913

Chapter 1. Introducing OS/390 NFS

This chapter provides an overview of the remaining chapters and the possible ways to tune the performance of the IBM* OS/390* Network File System (OS/390 NFS). It also briefly explains the Client-Server relationship. The Client-Server relationship and related terminology are provided to clarify references that might be misunderstood and to establish term usage within the context of this document.

Purpose and Structure of Tuning Guide

This tuning guide is geared towards OS/390 V2R6 Network File System. Most of the information presented in this publication is relevant to previous releases. Information pertaining only to OS/390 V2R6 NFS will be marked with parameters where appropriate.

The remainder of this publication is divided into the following five chapters:

- "Chapter 2. Performance Tuning in the NFS Environment" on page 5 explains performance tuning within the context of the NFS Client-Server environment.
- "Chapter 3. Optimizing the NFS Environment" on page 11 explores areas that may impact overall performance within the NFS environment.
- "Chapter 4. Evaluating OS/390 NFS Performance" on page 19 explains the techniques to evaluate the performance of OS/390 NFS.
- "Chapter 5. Tuning the OS/390 NFS Server" on page 23 explains tuning considerations specific to OS/390 NFS server.
- "Chapter 6. Tuning the OS/390 NFS Client" on page 33 explains tuning considerations specific to OS/390 NFS client.

Chapters 2 though 4 provide a brief overview of performance tuning from a total system perspective. Chapters 5 and 6 discuss the performance tuning at the client and server level.

The Client-Server Relationship and Terminology

This section is intended to provide a basic understanding of the NFS Client-Server relationship and the terminology used in later sections. Terminology specific to a limited area will be introduced as necessary. More in depth detail about the Network File System can be found in the *OS/390 Network File System User's Guide* and *OS/390 Network File System Customization and Operation* references.

A *client* is a computer or process that requests services on the network. A *server* is a computer or process that responds to a request for service from a client. A *user* accesses a service, which allows the use of data or other resources.

Figure 1 on page 2 illustrates the client-server relationship. The upper right portion of the figure shows the OS/390 NFS server. The lower right portion of the figure shows the OS/390 NFS client. The left portion of the figure shows various NFS clients and servers which can interact with the OS/390 NFS server and client. The center of the figure shows the Transmission Control Protocol/Internet Protocol (TCP/IP) network used to communicate between the clients and servers.

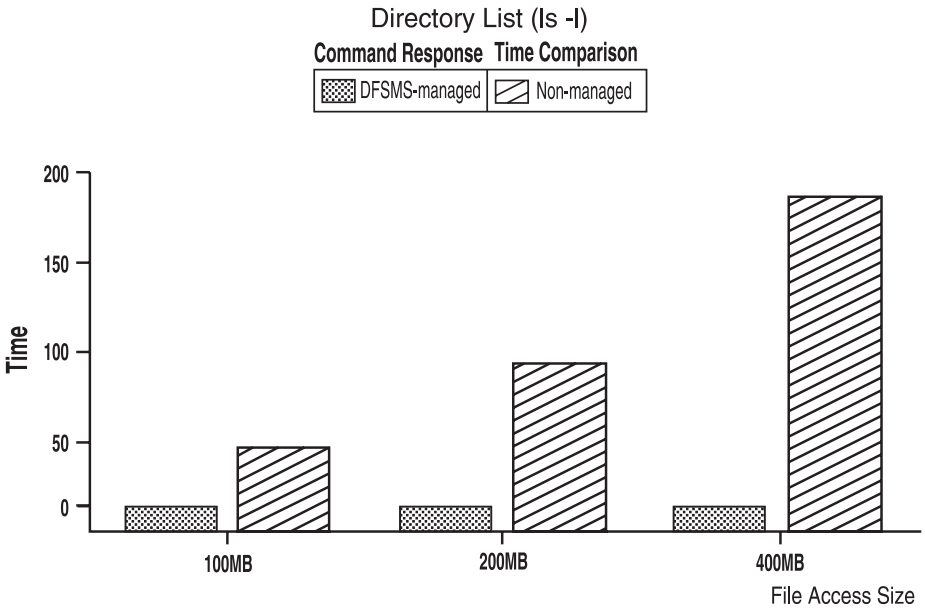


Figure 1. OS/390 NFS Client-Server Relationship

With the OS/390 NFS server, you can remotely access MVS/ESA conventional data sets or OS/390 UNIX files from workstations, personal computers, and other systems that run client software for the Sun NFS version 2 protocols, the Sun NFS version 3 protocols, and the WebNFS protocols over TCP/IP network.

The OS/390 NFS server acts as an intermediary to read, write, create or delete OS/390 UNIX files and MVS data sets that are maintained on an MVS host system. The remote MVS data sets or OS/390 UNIX files are mounted from the host processor to appear as local directories and files on the client system. This server makes the strengths of an MVS host processor — storage management, high-performance disk storage, security, and centralized data — available to the client platforms.

With the OS/390 NFS client you can allow basic sequential access method (BSAM), queued sequential access method (QSAM), virtual storage access method (VSAM), and OS/390 UNIX users and applications transparent access to data on systems which support the Sun NFS Version 2 protocols and the Sun NFS Version 3 protocols. The remote NFS Server can be an MVS, UNIX**, AIX, OS2, or other system. The OS/390 NFS client is implemented on OS/390 UNIX and implements the client portion of the Sun NFS Version 2 protocols and the Sun NFS Version 3 Protocols.

To access the NFS server a **mount** command must be executed on the NFS client system to provide a temporary link (until unmounted) between a local directory (preferably empty), or unused logical drive, and a remote file identifier available on the NFS server. The exact syntax and file naming conventions will depend on the NFS client and server implementations in use. With OS/390 NFS, the data set high level qualifier(s) (HLQ) would be used to access conventional MVS data. The Hierarchical File System (HFS) path name, preceded by a special configurable keyword, would be used to access OS/390 UNIX files.

In addition to providing a link between NFS client and server, the **mount** command also has parameters associated with it. Some of these parameters, such as buffer sizes and timing values, can affect performance. This topic will be

discussed further in Section “NFS Client System Performance Tuning” on page 13. Since the defaults for these parameters vary from one implementation to another, specific details should be obtained from the appropriate client documentation. With the OS/390 NFS implementation there are opportunities to override the MVS file allocation parameters and processing options on the **mount** command as well. Client enablement code is provided with the OS/390 NFS product for many NFS client implementations. File allocation and processing options will be further discussed in Sections “Data Set Creation Attributes” on page 23 and “Processing Attributes” on page 25.

Chapter 2. Performance Tuning in the NFS Environment

This chapter explains performance tuning within the context of the NFS Client-Server environment. It provides guidelines on the performance expectation of the underlying hardware and software products on which an NFS Client-Server implementation is dependent. These guidelines specifically note the limitations to performance tuning and describe processes to tune OS/390 NFS.

What is Performance Tuning?

In general, performance tuning improves the price to performance ratio for a system or set of services by reallocating the available computing, network, or storage resources. The reallocation of these resources not only improves the performance for a particular load of work, but also accommodates an increase in the amount of work to be performed with minimal acquisition of additional resources. The information acquired from performance tuning can also be an important basis for long range capacity planning.

Some possible reasons to do performance tuning might be:
- to access more data over existing networks
- to improve response time for particular applications or groups of users
- to better utilize the available storage capacity
- to minimize cost for additional services or functional capability

How is Performance Characterized?

Performance is the manner in which a process, system, processor, network, or device behaves for a particular load or unit of work. To measure, or quantify, performance, we monitor the length of time for a unit of work to complete. If units of work are being shared, we monitor the amount of time waiting for a resource to be available to perform a unit of work. A unit of work is a specific activity or action that we expect a process, system, processor, network, or device to perform. This could be something as granular as an I/O request, sending or receiving a buffer of data over a network, or processing an NFS request. We are frequently interested in the performance of a particular set of work activities which we will refer to as a load of work, or *workload*.

When a particular workload has been identified for performance measurement, we can determine the performance *metrics*, or units of measurement, that are relevant to that workload. Some examples of the performance metrics that will be used in reference to performance tuning for OS/390 NFS are:

- **throughput**, the number of units of work completed per unit of time
- **response time**, the elapsed time necessary to complete a unit of work
- **CPU time**, the amount of time the processor spends executing instructions
- **Instructions per byte**, the total number of CPU instructions to process data divided by the number of bytes of data processed.
- **aggregate throughput**, the sum of the throughput measurements for multiple processes
- **NFS operations per second (NFSops)**, the number of NFS requests processed by an NFS server divided by the total time in seconds to process the requests

Measuring performance metrics can be as simple as using your watch to time the execution of a particular command or as complex as using specialized hardware and software tools to monitor and extract a diversity of performance metrics. “Chapter 4. Evaluating OS/390 NFS Performance” on page 19 will address some methods that can be used to evaluate the performance of OS/390 NFS. The method selected will depend on the complexity of the workload and the monitoring tools available at your installation.

What is the NFS Environment?

The NFS environment includes the NFS client system(s), the mix of networks available, the NFS server system(s), and the manner in which they are configured. While this guide is intended for OS/390 NFS, it is important to know the performance limitations within the NFS environment to determine the necessity of tuning OS/390 NFS.

NFS client systems range from single user desktop personal computers to large scale processors with many users. These NFS client systems typically support multiple applications as well. Clearly, the NFS client system resources will be shared between its users and/or applications. These resources include available physical storage, memory, processing capability, and network access. The NFS client is one application, with a possibility for many users, that must share the memory, processing, and network resources in exchange for providing access to additional physical storage on other systems. The degree to which the NFS client application must share such resources will affect performance for NFS client users and applications.

The NFS server system, like the NFS client system, must share resources with other users and applications. NFS server application performance will be affected by the amount of contention over system resources and by the priority established for the NFS server application. The overall performance of an NFS server is also influenced by the number of NFS clients for which it provides services.

The network(s) over which NFS clients access NFS servers also affect overall performance in the NFS environment. Such networks can be homogenous, consisting of a single network medium, or heterogeneous, consisting of a mix of network mediums. Each network medium type has an expected maximum capacity, or *bandwidth*. For instance, the capacity of a Token Ring network may be 16 megabits per second (Mbps) or 4 Mbps, and the capacity of a Fiber Distributed Data Interface (FDDI) network is 100 Mbps. When different network mediums are combined in a more complex network environment, the capacity for a fixed route over the network is limited by the network segment with the smallest capacity. For example, a route over a network consisting of both 4 and 16 Mbps Token Rings and a FDDI backbone will have a maximum capacity equivalent to that of the Token Ring, or 4 Mbps. When bridges, routers, and gateways are included in a network configuration, their capacity must also be considered. Such devices must also be considered when tuning performance in a network environment, particularly if a device does not support increased network buffer sizes.

How to Tune for Performance

Given the complexity of the NFS environment, it is important to establish a methodology for tuning performance. The following steps provide a guideline that highlights particular areas relevant to the NFS environment. Implementing the guideline may involve more than one person or support organization.

1. **Identify Performance Requirements**

Before you begin performance tuning in general, and particularly OS/390 NFS, determine those areas where performance is unsatisfactory. This is a good time to establish more precise performance requirements. As users and application requirements are identified, it can be advantageous to rank or group them according to their requirements.

2. **Know the NFS Environment**

   In the previous section, the NFS environment was discussed. It is very important to fully understand the performance of the existing NFS environment, particularly that of the network. Such analysis can eliminate unnecessary tuning of the NFS client and server systems.

3. **Establish Performance Objectives**

   Once the performance requirements and the NFS environment are known, you are in a position to define and prioritize your performance objectives. The performance objectives should be specified in a manner that is quantifiable and measurable. Keep in mind that you will need an executable workload or test scenario to evaluate the effectiveness of your performance tuning.

4. **Define Workloads and Test Scenarios**

   You may already have workloads or test scenarios depending on how requirements and/or objectives were defined. However, these may be unwieldy or impractical to use for your performance tuning purposes. Therefore, it is advantageous to spend some time initially to define some simplified test cases that can be executed in a repeatable fashion and with as much control as is feasible. Simple test cases are also useful in diagnosing performance problems, particularly to locate an area within the NFS environment that may be impacting performance.

5. **Select Monitoring Tools**

   While you may only be interested in the performance of OS/390 NFS, you will find it useful to have access and familiarity with a variety of performance hardware and software monitoring tools. Not only will such knowledge assist you in collecting data to evaluate performance, but it will also help you to identify areas that may be impacting the performance of OS/390 NFS. Minimally, a set of monitoring tools must be identified to collect the data upon which performance tuning decisions will be made and to determine the effectiveness of tuning.

6. **Collect Performance Data**

   At this point you should be ready to begin collecting performance data. Initial measurements will be the starting point, or *baseline*, that will be used to evaluate the effect of your tuning. Since there can be a significant variation in network and system performance, it is prudent to repeat a performance measurement to establish the degree of variation inherent in the measurement. Doing this will provide a sense of whether or not future tuning is really affecting performance or simply normal variation.

   While it may be convenient to collect performance data when systems and networks are idle, this is probably not practical. However, it is useful to collect data during peak and low activity periods. If user or application requirements are related to a specific time period, or sensitive to other system and network activity, data should be collected for these periods of time, as well. Remember that redistributing the workload may be the most cost effective approach to performance tuning.

7. **Evaluate Performance Data**

   This may seem like an obvious step. However, keep in mind that the NFS environment may be quite complex. The more complex the NFS environment

and your test cases are, the more data there will be to evaluate. You may also have both client and server data to evaluate as well as network data.

As you begin evaluating performance data you have collected, look for areas or opportunities where performance could be improved. Before attempting to tune OS/390 NFS, you should determine if performance is primarily impacted by the NFS server system, the NFS client system, or the network itself. In fact, you may determine that additional data must be collected prior to any performance tuning. You may also discover evidence that configurations and parameter settings within the evaluation environment are not optimal.

The output of this step is a list of changes that you believe will positively affect performance. As you make this list, identify the impact or cost associated with a change. It is also useful to identify any resources that are heavily utilized or that have contending requirements. This will help you to prioritize and ultimately select the changes you will make or recommend.

8. **Tune Your NFS Environment**

   Make one change from your list of possible tuning changes at a time. Measure and evaluate the effect of that change before making any other changes. Pay particular attention to any impact on heavily utilized resources that may have been previously identified in addition to newly exposed resources that now appear to be impacted. Since performance tuning typically involves a trade off in resource utilization, make sure you have not inadvertently caused a performance problem elsewhere. Also, before deciding to implement a change, consider whether any observed changes are due to the tuning change you've made or simply a result of normal measurement variation. Repeat this step as necessary.

The Impact of the NFS Protocol on Performance

Command response time is of particular importance to NFS client users. The longer a user waits for the results of a particular command, the more important this will become. The nature of transparent access with the NFS protocol results in users not necessarily being aware of the impact on performance caused by the network and the NFS server system.

Also, while users are generally knowledgeable of the commands supported by the NFS client operating system, they may have no knowledge of the NFS commands, or procedures, that are executed as a result of one simple user command. In fact, one user command typically results in execution of multiple NFS commands.

Another impact on command response time is the NFS protocol itself which is intended to be as stateless as possible. This means that a stateless server operates correctly without maintaining any protocol state information for its clients. A stateless protocol was originally selected to minimize the probability of data losses due to a server crash. The OS/390 NFS strives to be as stateless as possible.

The stateless nature of the NFS server places the responsibility of keeping track of NFS commands on the NFS client. To do this NFS client implementations generally wait a period of time for a response to a particular NFS command. If a response is not received within this period of time, the NFS client will retransmit the NFS command. This process is repeated for a fixed number of times or until a response is received.

NFS servers and clients have typical methods of queuing requests and responses as part of the underlying protocol layers. When these queues are full, incoming requests or responses are dropped. The NFS client and server do not know when

responses or requests are dropped. Both rely on the stateless protocol whereby the client will eventually retransmit the NFS request again. Under these conditions the NFS client is waiting for a response that will never be received. Clearly, waiting to retransmit an NFS command, particularly multiple times, will negatively affect the response time for the initial user command.

To determine whether or not NFS client users are being impacted by the situation previously described, most NFS client implementations provide an **nfsstat** command to monitor NFS statistics. Figure 2 shows the output from the **nfsstat -c** command.

```
USER1:/u/user1:>nfsstat -c

Client rpc:
calls           badcalls        retrans         timeout
51              0               0               0

Client nfs_V2:
calls        badcalls
45           0
null         getattr     setattr         root         lookup
0        0%  21     47%  1          2%   0        0%  13         29%
readlink     read        writecache      write        create
0        0%  0       0%  0          0%   0        0%  1           2%
remove       rename      link            symlink      mkdir
4        9%  0       0%  0          0%   0        0%  0           0%
rmdir        readdir     fsstat
0        0%  3       7%  2          4%


Client nfs_V3:

calls           badcalls
4            0
null         getattr      setattr         lookup       access
0        0%  0        0%  0           0%  0        0%  0        0%
readlink     read         write           create       mkdir
0        0%  0        0%  0           0%  0        0%  0        0%
symlink      mknod        remove          rmdir        rename
0        0%  0        0%  0           0%  0        0%  0        0%
link         readdir      readdirplus     fsstat       fsinfo
0        0%  0        0%  0           0%  2        50%  1       25%
pathconf     commit
1       25%  0        0%
```

Figure 2. Displaying NFS Client RPC and NFS Statistical Information

When using the **nfsstat** command, users should be aware that results are cumulative. These statistics may be reset with the **-z** option of the **nfsstat** command. Also, the **nfsstat -c** command provides statistics for all NFS client activity, which makes it possible to access files on more than one NFS server. When using this command to query the NFS client statistics for OS/390 NFS, make sure that all NFS client access is for OS/390 NFS only.

The **nfsstat -c** command provides two types of statistics, *Client rpc* and *Client nfs*. The *Client rpc* statistics provide an indication of NFS performance from the perspective of that particular NFS client. As a general rule of thumb, if the *timeout* value is more than 0.2% of the total number of RPC calls, then some performance tuning is necessary.

If the *timeout* value is essentially equivalent to the *retrans* value, the NFS client is waiting on the NFS server. While you can increase the **timeout** option of the **mount** command to reduce the number of retransmissions, this will not improve the perceived responsiveness of the NFS server.

The *badcall* and *badxid* statistics are generated when information is lost or dropped somewhere between the NFS client and the NFS server. This can happen when processes are interrupted and are not necessarily indicative of a performance problem unless they are disproportionately high or persist. On the NFS client system the **netstat -s** command provides additional statistics that may be useful in tuning at the network level. For the OS/390 NFS, the MVS TCP/IP **netstat** command provides information that may be useful for tuning as well.

The *Client nfs* statistics provide a distribution of the NFS procedure calls made by users and applications on that particular NFS client. For a typical NFS client you will probably see a larger percentage of *getattr* and *lookup* calls. These calls are made whenever an NFS file is initially accessed. Directory listing is a common user activity that will generate these NFS calls.

The *write* and *read* statistics can also provide insight into the manner of NFS access on an NFS client, such as read or write biases. If the percentage of *read* and *write* NFS calls is higher than the percentage of *getattr* and *lookup* NFS calls, the client is probably accessing relatively large files as opposed to accessing many smaller files. Conversely, if the percentage of *getattr* and *lookup* NFS calls is higher than the percentage of *read* and *write* NFS calls, you might want to investigate whether users are querying directories or accessing relatively small files. In the latter case you might further determine if such files should be stored on an NFS server or the NFS client system.

Chapter 3. Optimizing the NFS Environment

This chapter explores areas that may impact the overall performance within an NFS environment. The focus will be on various network and NFS client and server system parameters which may affect NFS performance. Specific product documentation should be consulted for additional detail.

Network Performance Tuning

SYMPTOM: Data transfer rates significantly less than network capacity

ACTION 1: Know your network topology

Section “What is the NFS Environment?” on page 6 briefly introduced the role of the network in the NFS environment. Clearly, transferring data over congested networks results in poor performance. Therefore, when a performance problem is encountered in the NFS environment, it is useful to determine whether or not the network is the source of the problem prior to investigating other alternatives. While there are products specifically designed to monitor and report on network activity, knowing your particular network topology may be sufficient for initial tuning.

Figure 3 on page 12 shows a simple network topology with 2 16Mbps Token Rings connected to a 100 Mbps FDDI backbone network. In this example, access from NFS client A, on a 16 Mbps Token Ring, to NFS server B, directly connected to the FDDI backbone, would be as limited by network bandwidth as access to NFS Server A. On the other hand, access from NFS Client B, directly connected to the FDDI backbone, would be limited by the load on the FDDI network and the capacity of NFS Server B.

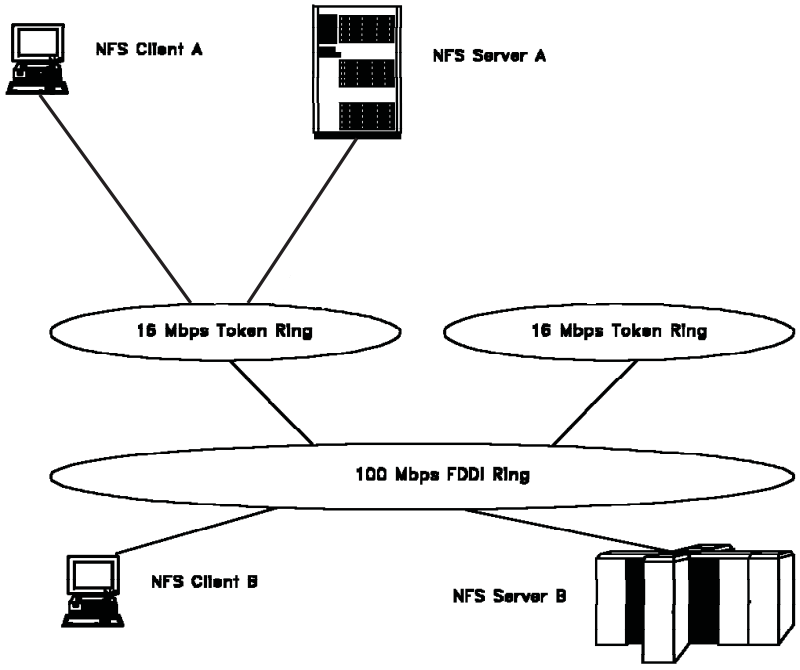


Figure 3. Sample Network Topology

There are other factors to consider in addition to the theoretical capacity of a network configuration. For instance, there are differences in the capability of network adaptors and network controllers. Overall network performance can also be influenced by the devices connecting network segments or subnets, particularly in terms of packet size limits and UDP checksum processing. While it may initially appear that there is no network constraint, there may be a device, adaptor, or network segment that is not performing at expected levels.

With this type of investigation, you may already have discovered that your primary constraint is a congested network. If so, you may elect to pursue such alternatives as:

- increasing network bandwidth,
- changing network topology,
- scheduling applications during low system usage, or
- modifying applications to reduce data transfer.

ACTION 2: Monitor network activity

It may be necessary to monitor network activity over a period of time to determine what is causing a performance problem. In addition to whatever monitoring your network administrator may be able to provide, the **netstat -s** command, available on many NFS client platforms, may provide some insight as well. This command reports such statistics as bad checksums, dropped fragments, non-forwardable packets, various timeouts, and socket buffer overflows. In some cases, it may be necessary to use a network analyzer to determine exactly what is being sent over a network segment. While it can be difficult to analyze congested networks, network analyzers occasionally capture evidence that may indicate a problem with the NFS client or server system.

NFS Client System Performance Tuning

SYMPTOM: Excessive NFS retransmissions

A high number of NFS retransmissions can be detected with the **nfsstat -c** command. See section "The Impact of the NFS Protocol on Performance" on page 8 for more information. A general rule of thumb is that the number of retransmissions should not be more than 2% of the total calls. Use traceroute <hostname> command to trace the route that IP packets take to a network host. This helps isolate where the bottleneck may occur.

ACTION 1: Increase network queue length

If the **nfsstat -c** command shows evidence of bad calls in addition to timeouts, or the **netstat -s** indicates that packets or fragments are being dropped, increasing the network queue length may be advisable. Occasionally, a network analyzer will detect that the server has responded to an NFS request *before* the client retransmits the request. This indicates that the network queue on the NFS client system was full and the response was dropped.

ACTION 2: Increase maximum transmission unit

The **netstat -in** command will display the maximum transmission unit (MTU) for each network interface active on the NFS client system. The MTU can be reset with the **ifconfig** command. This command is avaiable on most UNIX NFS clients. For the OS/390 NFS client, MTU can be reset in the TCP/IP profile. Increasing the MTU can improve overall performance and alleviate excessive IP fragmentation. However, with heterogeneous networks the MTU should not be set larger than the smallest acceptable MTU over the entire network route. For instance, when the network consists of a mix of Token Rings and Ethernets, the MTU is typically set at 1492 that is less than the 1500 MTU for an Ethernet and a 2000 MTU for Token Ring. Changing the network topology may allow you to increase the NFS client system MTU.

ACTION 3: Change socket buffer size

Changing the buffer size is another way to tune the NFS client system. The AIX/6000* **no -a** command will display the current settings, in particular the *udp_sendspace* and *udp_recvspace*. The AIX/6000 **no -o udp_sendspace=32768** and **no -o udp_recvspace=32768** commands can be used to reset the *udp_sendspace* and *udp_recvspace*, respectively. For the OS/390 NFS client, the *udpsendbfrsize* and *udprecvbfrsize* in the TCP/IP profile should be set respectively. Recommended values for SUN NFS V3 protocol are 65536. When processing with TCP protocol, it may be necessary to modify the **tcpsendbfrsize** and **tcprecvbfrsize**.

ACTION 4: Increase number of BIODs

If data on an NFS server is typically accessed sequentially, or if there are many users on an NFS client system, it may be advantageous to increase the number of Block I/O Daemons (BIODs). This increases the NFS client processor utilization and should be weighed against other processor requirements on a multiple user system. If access is typically at random offsets within a file or file sizes are very

small, you may even consider decreasing the number of BIODs. It should also be noted that some NFS client implementations honor BIODs differently to provide read or write bias.

ACTION 5: Increase timeout parameters

If you have determined that NFS retransmissions are not caused by dropped NFS responses, you may consider increasing the timeout value. *This should be one of the last alternatives you select since this can have adverse effects.* If information is lost or dropped between the NFS client and the NFS server, increasing the timeout value can make performance worse. In this case the client would wait longer to decide to resend a request. In a heavily used NFS environment, increasing the timeout value also increases the likelihood that internal client or server buffers will become stagnant or stolen to service other network requests. You would probably only want to attempt this if you have strong evidence, probably from a network analyzer, that the NFS server is sending responses *after* the NFS client retransmits requests. The timeout values can be reset with the **mount** command.

NFS Server System Performance Tuning

Regardless of whatever network and NFS client system tuning is done, some level of tuning or resource balancing should be addressed on the OS/390 NFS server system. Typically, these systems have tuning and capacity planning procedures in place along with a high level of experience. Consequently, this section will be limited to a surface level discussion of the resource components to be considered when tuning the NFS server system.

Network Controller Constraints

SYMPTOM: Insufficient network capacity

ACTION 1: Change controller configuration

The performance of OS/390 NFS is dependent on how well the MVS TCP/IP application and network controllers perform. This section will use the IBM 3172 Interconnect Controller, hereafter referred to as 3172, as an example of some items to consider for network controllers.

The 3172 is available in a variety of hardware configurations. The 3172, also available with different memory and processor capability, supports Ethernet, Token Ring, and FDDI network adaptors as well as parallel and ESCON* channel attachment. In addition to variations in 3172 hardware configurations, there are alternatives for the software configuration as well. The 3172 can run the Interconnect Control Program (ICP) or an optional feature supporting TCP/IP Offload.

Running the 3172 with TCP/IP Offload reduces the CPU time consumed by the TCP/IP address space. This reduction in CPU consumption is exchanged for a reduction in data transfer rates from the client system to the server system. The degree of impact is dependent on the amount of memory and processor capability of the 3172. Data transfer rates from the server system to the client system are generally not impacted. Tuning of the 3172 running TCP/IP Offload should be done in conjunction with tuning of the MVS TCP/IP application.

ACTION 2: Tune the network controller

When the 3172 is running in ICP mode, there are two parameters to consider: the **block delay time** and the **maximum response length**. The block delay time is the amount of delay allowed to block network frames for transmission. The recommended value for best performance is 10 milliseconds. The response length is the length of the longest frame to be sent to the MVS system without blocking. The recommended value for the maximum response length is 500 bytes.

ACTION 3: Upgrade the network adapter

The S/390 Open System Adapter 2 (OSA2) is an integrated S/390 hardware feature plugged in place of a channel card. The OSA2 becomes an integral component of the I/O subsystem to enable convenient LAN attachment. The OSA2 is available in a variety of configuration modes. Using OSA2 may eliminate constraints with 3172 controller. Tune OSA2 on MVS TCP/IP if TCP/IP passthru mode is used.

MVS Constraints

SYMPTOMS: High CPU utilization with low aggregate throughput

The following actions should be weighed against any increase in MVS storage they may require:

ACTION 1: Increase MVS TCP/IP send and receive buffers

There are other applications besides OS/390 NFS which are dependent on the 3172 and the MVS TCP/IP application. Some applications have different tuning requirements. The default values for **udpsendbfrsize** (or **tcpsendbfrsize**) and udp **udprcvbrfsize** (or **tcprcvbrfsize**) is 16KB. It may be useful to increase these parameters up to 64KB for better throughput with large files.

ACTION 2: Increase MVS TCP/IP UDP queue

Another area to consider is the MVS TCP/IP UDP queue. As discussed in section "NFS Client System Performance Tuning" on page 13, NFS responses are dropped when network adaptor queues are full. A similar situation occurs when the UDP queue of MVS TCP/IP is full; in this case incoming NFS requests are dropped. The **noudpqueuelimit** keyword in the ASSORTEDPARMS section of the TCP/IP profile data set can be specified to enable the MVS TCP/IP server to accept incoming UDP datagrams. Without specifying this keyword, the default queue length of 30 may not be sufficient.

ACTION 3: Modify NFS and TCP/IP dispatching priorities

It is recommended that the OS/390 NFS procedure have a relative dispatching priority less than that of TCP/IP. With MVS/ESA Version 5 Release 1.0 or later this is especially important since the System Resource Manager (SRM) algorithms have changed. Prior to this release of MVS mean time to wait algorithms resulted in the OS/390 NFS receiving a lower dispatching priority than TCP/IP due to the order in which the address spaces were started. With this newer release of MVS mean time to wait dispatching priorities are adjusted based on increased I/O activity. Since TCP/IP has higher network I/O than OS/390 NFS, the TCP/IP dispatching priority is lowered. Assigning fixed dispatching priorities, with TCP/IP dispatched at a higher relative value than the OS/390 NFS, can alleviate this situation.

ACTION 4: Run TCP/IP Offload on 3172

Running TCP/IP Offload on the 3172 may reduce CPU time in the MVS TCP/IP address space. Smaller buffer sizes typically recommended for TCP/IP Offload may improve MVS storage usage as well. However, transfer rates will be reduced when the capacity of the 3172 has been reached. Transfers from the client system to the server system are particularly prone to reduced transfer rates.

ACTION 5: Select transport protocols

OS/390 NFS supports TCP and UDP as transport protocols for the server, for both NFS Version 2 and Version 3 protocols. UDP is primarily used for its efficiency on high bandwidth, low latency networks, such as LANS. TCP is used for its efficiency on low bandwidth, high latency networks, such as WANS. However, for better throughput performance, choose UDP as a transport protocol.

ACTION 6: Use UDP+ protocol

UDP+ is available only in conjunction with the OSA-2 adapter. It can help reduce transport protocol pathlength, compared to standard UDP protocol.

SYMPTOMS: Constrained throughput with low CPU utilization

ACTION 1: See "Subtasking" on page 32

SYMPTOMS: High DASD utilization and NFS command response time

ACTION 1: Evaluate placement of MVS catalog data sets

The performance of OS/390 NFS will be impacted when serving files located on heavily utilized storage devices or devices behind congested storage controllers. Before deciding whether or not to move data or upgrade storage systems, make sure system catalogs are not on heavily utilized devices. Many of the NFS commands that are executed for conventional MVS data sets involve catalog access. In fact, listing directories, one of the most commonly executed commands on NFS client systems, accounts for a significant portion of the NFS Get Attribute and Lookup commands. Such commands not only cause the MVS catalog to be accessed but might also cause the entire file to be read depending on:

- the OS/390 NFS processing options,
- prior access via OS/390 NFS, and
- whether or not files are DFSMS-managed.

ACTION 2: DFSMS-managed data accessed from the network

Files that are primarily accessed via OS/390 NFS should be DFSMS-managed for improved performance. The OS/390 NFS maintains file attributes for DFSMS-managed data sets when the OS/390 NFS **nofastfilesize** processing option is specified. The **nofastfilesize** processing option provides exact file size determination rather than approximating file size as with the **fastfilesize** processing option. DFSMS-management also provides improvements in terms of better storage utilization and specification of service levels.

ACTION 3: Evaluate placement of data accessed from the network

While deciding whether or not to access DFSMS-managed files via the OS/390 NFS, consider also the placement of such files. Files with critical performance

requirements should be placed on the appropriate devices via the Storage Class parameters. If reallocation is necessary, you might also consider allocating SAM Striped data sets for larger files or an HFS for smaller files. If you elect to maintain data set organization, you may choose to reblock the data set to a more optimal block size, such as half track blocking. You may even determine for some files that the best alternative is to store the files locally on the NFS client system.

Chapter 4. Evaluating OS/390 NFS Performance

This chapter assists both NFS client users and NFS server system administrators in evaluating the performance of OS/390 NFS. Test methods are provided to assist in collecting performance data for evaluation. The appropriateness and usage of these test cases and methods will depend on:

- the workload requirements to be evaluated,
- the level of expertise of the evaluator, and
- the monitoring tools available.

Evaluating Throughput

Throughput refers to the rate at which data is transferred between the NFS client and server systems. Establishing throughput baselines for the Client-Server environments of interest will assist in determining any benefits achieved from tuning. Baselines and measurement techniques are also useful in diagnosing performance problems.

As discussed in "How is Performance Characterized?" on page 5, a set of work activities or workloads should be identified for measurement. For throughput the type of workload selected should reflect the type of NFS read and write access expected by typical users or specific applications. Points to consider are:

- average file size or range of file sizes,
- sequential or random access,
- data set types and organizations to be accessed, and
- requirements for ASCII/EBCDIC translation.

Single Process Throughput

The easiest place to begin evaluating NFS read and write throughput is on a single client and a single process basis. While measuring in a controlled and isolated laboratory situation is advantageous, it may not be practical. Measuring the actual environment of interest can provide more meaningful information as well as establishing a methodology for monitoring performance in the future. Under such uncontrollable situations, it is important to determine range and normal variation in measurements with particular attention to peak and non-peak periods of activity.

Once the NFS environment has been selected for measurement, some simple techniques can be used to initially evaluate NFS read and write throughput. One of the easiest methods is to use commands available on the NFS client system, such as **copy** or **cp**, to generate NFS reads and writes. Copy commands along with **date**, **time**, or **timex** commands can be used to determine the elapsed time for the copy. Throughput can be readily calculated from the elapsed time and the size of the file.

Some additional considerations when measuring NFS throughput are the effects of graphical user interfaces (GUI), any overhead associated with opening files, and I/O to and from local physical storage. Measuring with and without a GUI will show the effect on NFS throughput caused by the GUI. Using a network analyzer can assist with isolating the time spent opening files from the time spent executing NFS reads or writes. Techniques such as copying to **/dev/null** can eliminate local physical I/O for NFS reads. For more complex requirements such as reading and writing at random offsets, you may use an existing application to provide such

access or write a program to execute NFS procedure calls at random offsets within a file. Program generated NFS procedure calls can also be useful when available local physical storage is insufficient for the file sizes to be measured.

Multiple Process Throughput

For NFS client systems with multiple users or applications you may want to determine aggregate throughput on a multiple process basis. If throughput on a single process basis has achieved the network capacity, then multiple process throughput will be limited by the same network capacity as well as any NFS client system limitations. A simple method for evaluating multiple process throughput is to simply execute multiple single process measurements simultaneously.

Multiple Client Throughput

Evaluating multiple client throughput requires the ability to propagate the measurement workload over more than one NFS client system, preferably the same type of system. Such measurements are more complex in that they require controlled execution and collection of results from multiple systems. Depending on your network configuration, remotely executing your single process workload on multiple systems is a relatively simple method to use to initially evaluate multiple client aggregate throughput.

Evaluating NFS Command Response Time

A user is probably more interested in the response time for typical user commands, such as listing a directory, making or removing a directory, and creating or removing a file. However, when such commands are executed for NFS mounted file systems, the NFS command response times become a factor in user command response time. The transparent nature of the NFS protocol allows a user to define a set of commands as a measurement workload with a minimum of effort.

For those desiring more detail on NFS command response time, there are public domain programs as well as industry standard benchmarks. Both of these options require more experience on the part of the user and may have limitations on their usage. Such programs are typically written for the UNIX environment and may not support all NFS client and server systems.

Evaluating CPU Utilization

Evaluating CPU utilization on an MVS system can be quite complex, particularly for a production system. The activity on such systems is already generally monitored and OS/390 NFS is simply one more application to be monitored. High level monitoring can be accomplished from the Display Active (DA) display of the System Display and Search Facility (SDSF). This display provides an overview of the total activity on the system. Of particular importance are the CPU usage and relative dispatching priorities of TCP/IP and OS/390 NFS. Monitoring tools such as the Resource Measurement Facility* (RMF*) Monitor I and Monitor II can provide more detail on the performance characteristics of the MVS system.

Collecting Server Usage Data

You can use the SMF records that OS/390 NFS produces to keep track of how files are accessed and how long each NFS user session lasts. The following SMF records can be produced:

- Record type 42 subtype 7

This record, written when a file times out, provides file usage statistics.

- Record type 42 subtype 8

  This record, written when a client user logs out of the OS/390 NFS, provides user session statistics.

Chapter 5. Tuning the OS/390 NFS Server

Previous chapters discussed performance tuning from a total system perspective. This chapter discusses performance tuning at the OS/390 NFS server level. In particular, this chapter focuses on the OS/390 NFS data set creation, processing, and site attributes.

Data Set Creation Attributes

When creating new MVS data sets, the allocation parameters are the most readily accessible means for the end user to influence the performance of OS/390 NFS server. They are also not well known outside of the MVS user community. This is primarily due to the fact that MVS has a record oriented data structure and most NFS client systems have a byte oriented data structure. This section looks at how a variety of allocation parameters affect performance.

Block Size and Record Length

Block size (**blksize**) specifies the maximum length, in bytes, of a physical block of storage in MVS. If **blksize(0)** is specified, the system will determine the optimal block size based on the maximum record length (**lrecl**) and the physical characteristics of the disk, or approximately half of a physical track. Half track blocking is optimal in that it reduces the amount of wasted storage on the disk.

The **blksize** is also important to the performance of the OS/390 NFS, which performs physical I/O on a block basis. While half track blocking is generally optimal, for OS/390 NFS server you should be aware of the relationship between the block size, the file record length, the network packet size and the NFS client **mount** write and read buffer sizes, **wsize** and **rsize**.

A network packet size will be negotiated between the NFS client and the NFS server when the NFS **mount** is processed. In general, the NFS server will determine the packet size for write operations, and the NFS client will determine the packet size for read operations. For OS/390 NFS, this packet size is 8192 bytes, when processing with SUN NFS V2 protocol. For several NFS client implementations the default packet size is 8192. However, there are some implementations with a 4096 byte maximum packet size. The maximum packet size is 65536 bytes when processing with SUN NFS V3 protocol.

The packet size can also be automatically reset to a smaller value when a constraint is detected. Such a reduction in packet size causes an increase in the number of NFS requests for the server to process the same amount of data. This increases the load on the server and may or may not have been the initial constraint causing the packet size reduction. This can have an adverse effect on OS/390 NFS server performance, particularly when processing write requests.

Regardless of the packet size, the OS/390 NFS server must reassemble the packets, possibly arriving out of sequence, into records. OS/390 NFS server buffers (see “Ordering Out of Sequence Data” on page 29) and orders these packets until a block of data can be written to disk. Write requests, especially those for large files, are processed more efficiently when the likelihood of packets spanning records and blocks is reduced.

When OS/390 NFS server is processing read requests, the physical I/O is again on a block basis. Consequently, read throughput is improved by increasing the block size up to half track blocking.

If files are allocated with a default block size (BLKSIZE=0) on RAMAC 3* DASD, the actual block sizes will probably vary between 24 and 27 KB, depending on the specified record length and record format. Table 1 shows the actual allocation block sizes for physical sequential data sets allocated on a RAMAC 3 DASD with a variety of record lengths and with both fixed and variable length records.

Table 1. Default Block Sizes for RAMAC 3 DASD

Record Format	Record Length (bytes)	Block Size (bytes)
FB	80	27,920
FB	1024	27,648
FB	4096	24,576
FB	8192	24,576
VB	84	27,998
VB	1028	27,648
VB	4100	27,998
VB	8196	27,998

Record Format

Record format (**recfm**) specifies the format and characteristics of the records in the data set. This section will be limited to a discussion of the performance considerations associated with fixed and variable length records.

There seems to be a natural affinity toward allocating MVS data sets with variable length records from the perspective of a byte oriented NFS client operating system. Such allocations enhance the feeling of transparent access and eliminate requirements to determine a fixed record length. However, the placement of incoming packets into records of varying lengths is more complex than placing packets into records with fixed lengths. Consequently, write performance for OS/390 NFS is generally more efficient when MVS data sets are allocated with fixed length records.

Data Set Organization and Data Set Type

Data set organization (**dsorg**) specifies the organization of an MVS data set. The OS/390 NFS data set organization attribute can be physical sequential (**ps**), direct access (**da**), or VSAM. OS/390 NFS server supports three types of VSAM files: key-sequenced (KSDS), entry-sequenced (ESDS), and relative record (RRDS). However, keyed access and relative-number access are not supported by the NFS protocol.

The **dsorg** attribute is ignored for directory-oriented NFS client commands. The data set type (**dsntype**) specifies whether a PDSE or PDS is to be created when the OS/390 NFS receives a **mkdir** command from the NFS client. A PDSE is created when **library** is specified, and a PDS is created when **pds** is specified. You cannot create another PDS (or PDSE) within a PDS (or PDSE).

Another MVS data set type is extended (EXT) which identifies an extended format data set. With OS/390 NFS these DFSMS-managed data sets are allocated based on

specification of a data class (**dataclas**). An appropriate data class must have been defined on the MVS system for an extended format data set to be allocated. One of the values specified when creating an MVS data class is the **volume count** which determines the number of stripes allocated for an extended format data set. If Automatic Class Selection (ACS) Routines have been written to allocate extended format data sets based on such criteria as naming conventions, it would not be necessary to specify a **dataclas** to allocate an extended format data set via the OS/390 NFS server.

The Hierarchical File System (HFS) is another data set type. The HFS must be DFSMS-managed and be mounted within the OS/390 UNIX sub-system. While an HFS cannot be allocated via OS/390 NFS, an HFS can be accessed via the OS/390 NFS server.

Data set organizations and data set types are generally selected based on user and application requirements. However, performance should be considered as one of these requirements. An important factor to consider when evaluating performance is the size of the files to be accessed. For smaller file sizes, we can differentiate between the overhead costs associated with creating or opening a file for output. As the file size increases, this impact becomes less of a factor in the transfer rates.

Processing Attributes

Processing attributes are used to control how files are accessed from the NFS client system. Default values can be specified when the server is started. These defaulted values can be overridden at the NFS **mount** level or at the individual command level. The processing attributes, like the data set creation attributes, can affect the performance of the Network File System. This section will be limited to those processing attributes which have the most influence on performance.

Character Translation

The processing attributes that specify whether or not data conversion occurs between ASCII and EBCDIC formats may have the most influence on the performance of OS/390 NFS. The **binary** attribute, which specifies no data conversion, is the most efficient manner in which to access data via OS/390 NFS. However, data conversion may be a requirement, particularly when data is shared with MVS users or applications. Under such circumstances the **text** processing attribute would be specified instead of the **binary** processing attribute. The actual impact associated with data conversion will depend on the data to be converted.

File Size Determination

The attributes discussed previously have primarily affected data transfer rates. Whether or not to determine the exact file size in bytes has more of an effect on user command response time. This is because the file size in bytes is one of the file attributes obtained by an NFS Get Attribute procedure. The NFS Get Attribute procedure is executed for every user command accessing an NFS mounted file system.

OS/390 NFS provides processing attributes to specify estimated file size determination, **fastfilesize**, or exact file size determination, **nofastfilesize**. The accuracy of the file size with **fastfilesize** processing is best for **binary** processing of data sets with fixed length records. For other situations, or when exact file size is a requirement, the **nofastfilesize** processing attribute should be specified.

When the **nofastfilesize** processing attribute is specified, there are performance factors to consider. The most important factor is that file size in bytes is maintained by OS/390 NFS for DFSMS-managed data sets with many data set organizations.

Figure 4 compares the differences in response time between the OS/390 NFS attribute support for DFSMS-managed data sets and the case when attributes are not cached for non-managed data sets. In this example, all data sets were members of a PDSE data set allocated with a variable block record format. The **ls -l** command was executed with the OS/390 NFS **nofastfilesize** and **binary** processing attributes.

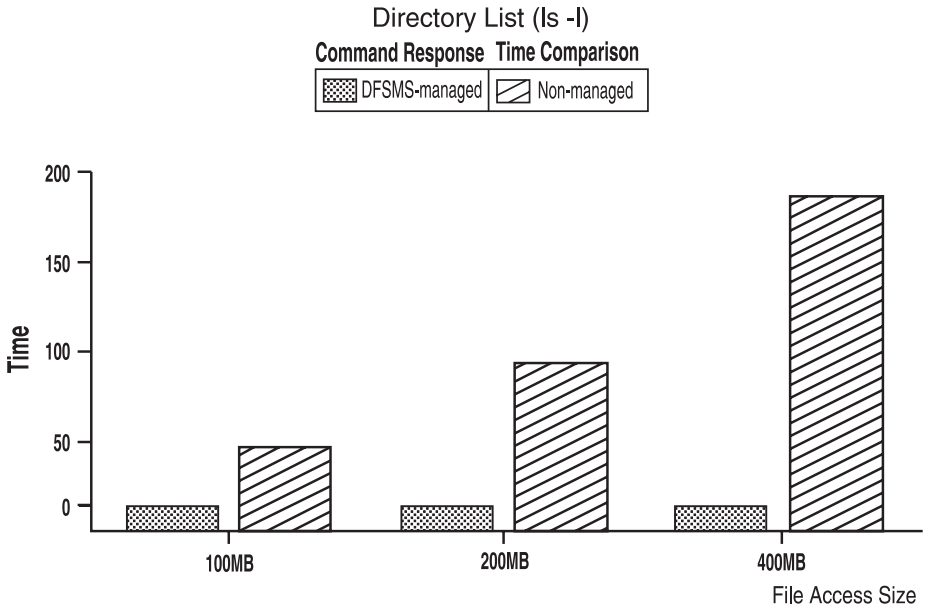


Figure 4. Directory List Comparison Between DFSMS-managed and Non-managed

Data sets processed with the **nofastfilesize** attribute that are not DFSMS-managed will need to be read entirely to determine the exact file size in bytes when initially accessed via OS/390 NFS. OS/390 NFS will cache this information until the data set is modified (see section "Buffer Usage and Caching" on page 28) or the attribute time value has expired. The larger the file is, the greater the impact to command response time is. Figure 4 shows how the response time for a directory list command can be impacted when the file has to be read to determine the exact file size in bytes.

Maintaining the file size attribute by OS/390 NFS also affects read throughput when accessing larger DFSMS-managed data sets with the **nofastfilesize** processing attribute. In the **nofastfilesize** case, reading entire file to determine byte file size is not necessary. The impact on the end-to-end response time for the read is lessened by reducing the response time for the initial access to obtain file attributes.

Data Set Timeout Specification

There are three timeout processing attributes that specify when OS/390 NFS will release a data set following certain NFS operations. The **readtimeout** and **writetimeout** processing attributes specify how long OS/390 NFS will wait after respective read and write operations before releasing a data set. The **attrtimeout** specifies how long a data set will remain allocated after an NFS Get Attribute or

Lookup operation. When accessing data sets that are not DFSMS-managed with the **nofastfilesize** processing attribute, it may be advantageous to increase the **attrtimeout** value so that OS/390 NFS caches data set attributes for a longer period of time. However, all three processing timeout attributes must be within the range of the **maxtimeout** and **mintimeout** site attributes.

Accessing Migrated Files

While migrating less frequently accessed data sets to tape improves MVS storage utilization, retrieving migrated data sets can impact the performance of OS/390 NFS. OS/390 NFS provides three processing attributes to specify how migrated data sets are to be handled by the server: **retrieve(wait)**, **retrieve(nowait)**, and **noretrieve**.

The **retreive(wait)** attribute instructs the server to wait for a migrated data set to be recalled to a direct access storage device. The NFS client user or application process will not receive a response from OS/390 NFS until the data set has been recalled. If the migrated file is relatively large, or migrated to tape, this can cause the NFS client to retransmit the request. The NFS client will retransmit the request a fixed number of times as specified by the **retrans** parameter of the **mount** command. If the recall of a migrated data set takes longer than the product of the **timeo** value on the **mount** command and one plus the **retrans** value for the mount, **timeo** * (**retrans** +1), the initial command will most likely need to be executed again. Using the default values for the AIX/6000 **mount** command of 7 tenths of a second for **timeo** and 3 attempts for **retrans**, a recall would need to take less than 3 seconds. An alternative approach is to use the **retrieve(nowait)** processing attribute.

The **retrieve(nowait)** attribute instructs the server to recall a migrated data set and immediately return a ″device not available″ message to the client without waiting for the recall to complete. With this option, users attempting to access the file can continue to use their session for other activity rather than waiting an indeterminate time for the recall of a file. They can attempt to access the file again after allowing some period of time for the recall to have completed.

If it is critical to the user that a file be recalled before further work can continue, the **retrieve(wait)** processing attribute can be specified more selectively as part of the NFS client user command syntax along with the file name. While doing this provides the user more control over command execution, the user command may still timeout and have to be executed again.

Specifying the **retrieve(wait)** processing attribute as the system default may also impact the availability of OS/390 NFS subtasks to process such requests. The second parameter of the **nfstasks** site attribute determines the number of OS/390 NFS subtasks available to process such recall operations. See “Subtasking” on page 32 for more information on the **nfstasks** site attribute.

Another method of avoiding the unnecessary recall of migrated data sets is to specify the **noretrieve** processing attribute. With the **noretrieve** attribute, the server does not recall a migrated data set and a ″device not available″ message is returned to the client. This processing attribute is particularly useful when listing a directory with unknown contents.

Asynchronous HFS Processing

OS/390 NFS, with SUN NFS V2, protocol provides two processing attributes that only apply to HFS processing. The **sync** attribute specifies that HFS write requests

should be committed to nonvolatile media (for instance, DASD) when received by the server. Some performance improvement can be obtained with the alternative **async** processing attribute. This type of processing causes HFS write requests to be cached at a level within the OS/390 UNIX architecture prior to the physical I/O. For more information on HFS tuning, see "Ordering Out of Sequence Data" on page 29 and "Subtasking" on page 32.

The **async** and **sync** attributes specified in the control file will be ignored when OS/390 NFS server is communicating with SUN NFS V3 protocol. The processing attribute is determined in the client application implementation. Asynchronous write is also recommended for faster throughput.

Site Attributes

The site attributes control OS/390 NFS resources. Tuning of these parameters should be done with caution. They are only specified at the start of the OS/390 NFS server and cannot be modified by client users. This section is limited to those attributes which more directly affect performance.

Buffer Usage and Caching

The **bufhigh** site attribute specifies the maximum size in bytes of allocated buffers before any buffer reclamation is initiated (see Figure 5). When the **bufhigh** limit has been reached, a percentage of the buffers will be reclaimed, and the amount of reclamation is determined by the **percentsteal** attribute. OS/390 NFS uses this buffer area to cache file information, thereby satisfying requests more efficiently.

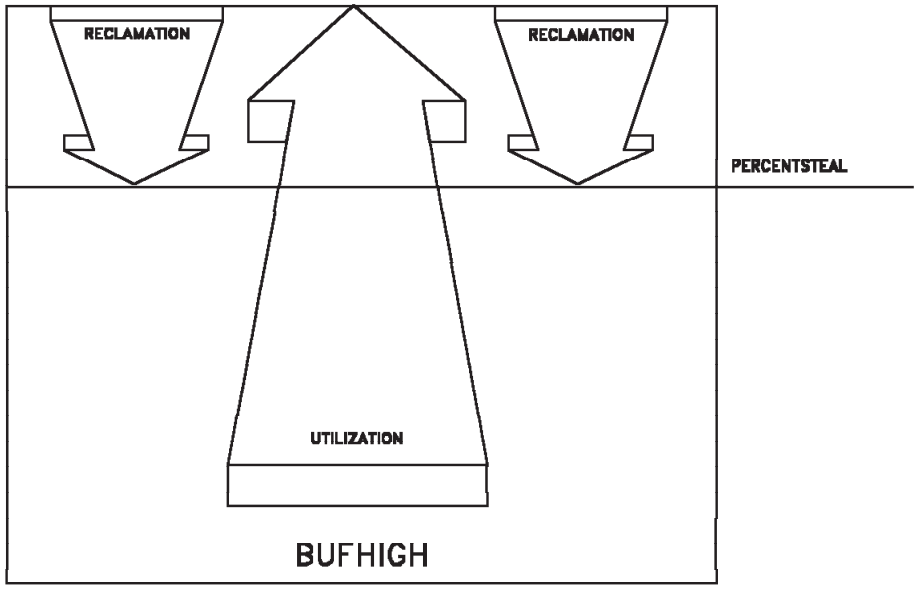


Figure 5. Bufhigh Utilization with Percentsteal

There are two additional site attributes, **readaheadmax** and **maxrdforszleft**, that also affect **bufhigh** usage. The **readaheadmax** attribute specifies the number of

bytes to be read to fill the internal buffers so that additional read requests for that file may be satisfied directly from cache. The **maxrdforszleft** site attribute defines the number of physical block buffers to cache after determining a file's size (see section "File Size Determination" on page 25). This information is cached to satisfy any subsequent read requests for the same file.

Keep in mind also that some information will be cached on a file handle basis, in other words, for every file accessed within the timeout periods. Tuning of the **bufhigh**, **readaheadmax**, and **percentsteal** values should be determined based on:

- number of files to be accessed,
- available region, and
- amount of data to cache per file.

For example, suppose that, on average, 1,000 physical sequential (DSORG=PS) files allocated on 3390 Model 3 DASD are accessed within the same timeout period via the OS/390 NFS server. Further, assume that we would like two blocks of a file cached internally to satisfy read requests. If we assume an average block size of 25 KB (see "The Client-Server Relationship and Terminology" on page 1 for more detail), we would need at least 50 MB of storage for internal buffers. With a 20 percent value for **percentsteal**, a 64 MB value for **bufhigh** would probably be more reasonable. For this example, a reasonable value for the **readaheadmax** attribute is the file block size, or 25 KB. The **readaheadmax** value should not be less than twice the maximum record length of files accessed via the OS/390 NFS server.

Ordering Out of Sequence Data

While the previous section primarily addressed satisfying read requests from cache, this section discusses the caching used by OS/390 NFS to satisfy write requests. OS/390 NFS provides two site attributes for this purpose: **cachewindow** and **logicalcache**. As with all site attributes, **cachewindow** and **logicalcache** can only be modified at server startup.

The **cachewindow** attribute specifies the window size, in terms of number of packets, used in logical I/O to buffer client block writes received out of sequence. Figure 6 on page 30 shows how out of order packets are buffered in a cache window until a block of data can be physically written to storage. Since out of sequence packets are an inevitability, a certain amount of buffering is necessary.

In a congested network environment, the nature of the underlying protocols should be taken into consideration when tuning the **cachewindow** attribute. Under such circumstances, NFS, UDP, and IP requests are more likely to timeout. When this situation is detected within the underlying protocols, the packet size can be reduced automatically. The packet size is generally halved until the server and client systems are balanced. The minimum packet size that can be negotiated is 512 bytes. Accessing the OS/390 NFS server in such an environment can result in the default 8 KB read and write buffers being reduced to 512 bytes, or 16 packets instead of one packet. *Under these circumstances the recommended value for* **cachewindow** *is 16 times the number of BIODs.*

Another factor to consider in tuning the **cachewindow** attribute relates to the relationship between packet size, data set record length, and block size. If the packet size is typically greater than or equal to the data set block size, there is less of a need to buffer the packets. On the other hand, if the packet size is smaller than the block size and/or record length, a larger **cachewindow** value will be required to buffer the packets until a block I/O is executed.

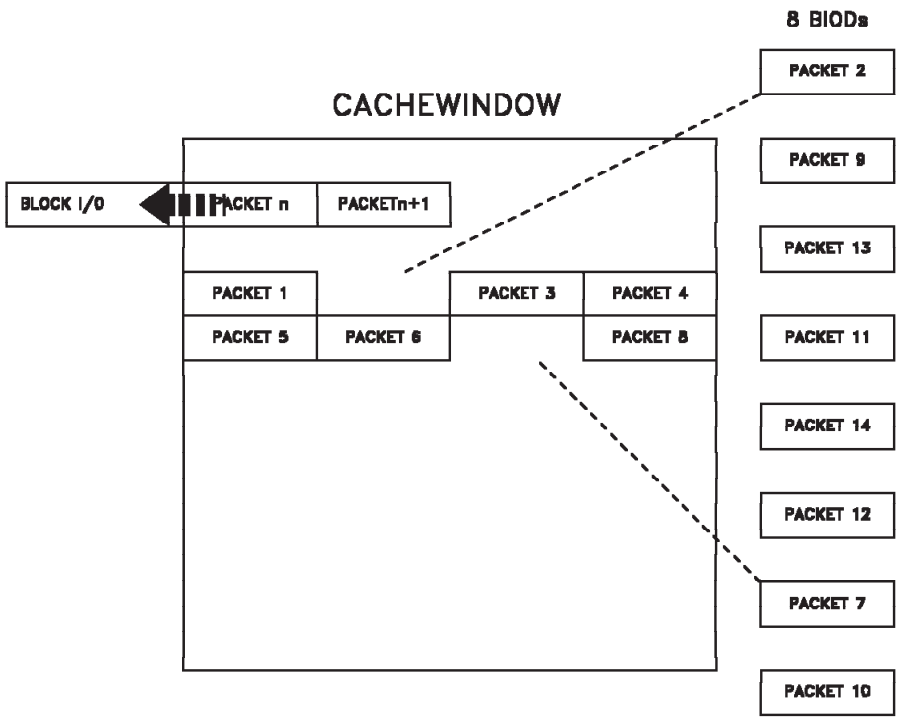


Figure 6. Relationship Between Cachewindow and BIODs

The **logicalcache** site attribute specifies the size (in bytes) of allocated buffers for all the cache windows combined (see Figure 7 on page 31). The number of cache windows in use will depend on the number of file handles accessed within a given timeout period.

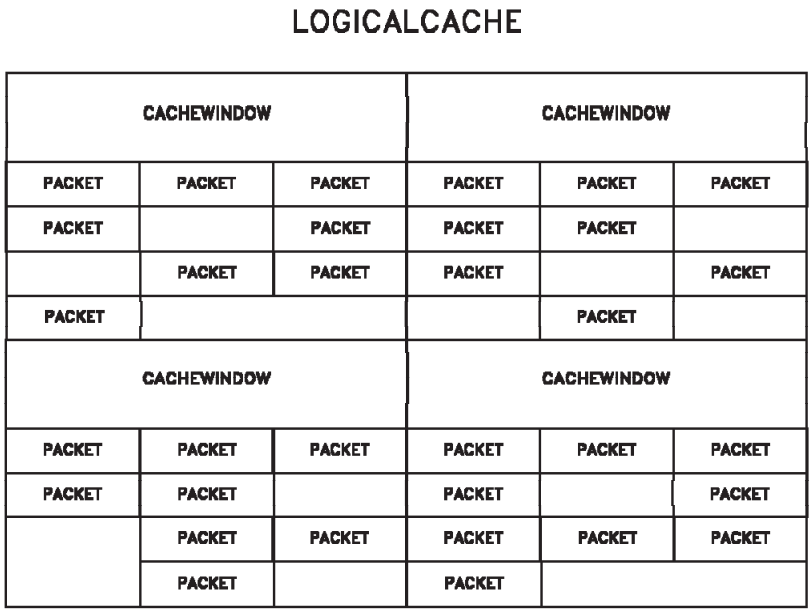


Figure 7. Logicalcache Utilization for Cachewindows

Tuning of the **logicalcache** and **cachewindow** values should be determined based on:

- number of files accessed for write,
- available region,
- typical packet size,
- data set block size, and
- number of client BIODs.

The **cachewindow** and **logicalcache** attributes do not apply to HFS processing.

For example, suppose you have AIX/6000 NFS client workstations on a lightly loaded network, with the default value of six BIODs, writing to physical sequential MVS data sets allocated with 8 KB fixed length records. Further, assume that these files were allocated on 3390 Model 3 DASD with a 24 KB block size and that the network packet size is 8 KB. Then, a value of 12, twice the number of NFS client BIODs, for **cachewindow** means that an out of sequence packet could be internally buffered within a 96 KB or four block range. If a packet is received outside this range, OS/390 NFS must read data from storage in order to write the packet at the correct offset into the data set. OS/390 NFS detects and ignores duplicate requests.

A value of 12 for **cachewindow** and a network packet size of 8 KB in our example results in a 96 KB buffer requirement per client actively writing to MVS data sets via OS/390 NFS. A 5 MB value for **logicalcache** is sufficient to satisfy sustained write requests from 50 NFS client workstations at the same time.

Subtasking

The **nfstasks** (n,m) site attribute specifies the number of server processes to initiate on startup. The valid value range for both *n* and *m* is 1 to 24 (8 is the default for each). The sum of *n* and *m* must be less than or equal to 25).

The absolute and relative value of *n* and *m* should be tuned for the expected system usage. If conventional MVS data sets will be accessed, primarily, then *n* should be relatively high. If OS/390 UNIX files will be accessed, primarily, then *m* should be relatively high. The absolute value of these will influence the amount of system resources consumed (higher values will make more system resources available to process NFS request.

The parameter *n* is the number of processes started to handle asynchronous operations (conventional MVS data set access) and short duration synchronous operations (SAF calls). The parameter *m* is the number of processes started to handle longer duration synchronous operations (OS/390 UNIX file access and migrated MVS data set recall).

Chapter 6. Tuning the OS/390 NFS Client

This chapter discusses performance tuning at the OS/390 NFS client level. Performance of the OS/390 NFS client is affected by the following installation parameters that can be tuned. The default value of each parameter is specified in parentheses.

- attrcaching (Y)
- biod (6)
- bufhigh (32)
- datacaching (Y)
- delaywrite (16)
- dynamicsizeadj (Y)
- readahead (8)

Most of the installation parameters specified in member BPXPRMxx in SYS1.PARMLIB can be overridden over a mount point by using the **mount** command. The exception to this is *biod* and *bufhigh*, which require restarting OS/390 UNIX.

The following are MOUNT parameters that affect performance and can be tuned. The default values are specified in parentheses.

- attrcaching (Y)
- datacaching (Y)
- delaywrite (16)
- readahead (8)
- rsize (8192 for V2) (No default for V3 — system determined)
- wsize (8192 for V2) (No default for V3 — system determined)
- vers (3)

Caching

Caching of file attributes (attrcaching=y) and data (datacaching=y) are recommended for performance. By caching the file data, all subsequent references to the cached data is done locally, avoiding the network overhead. Further authorization checking for subsequent access to the cached data or for other client users is done on the OS/390 NFS client system. With attribute caching, OS/390 NFS client will perform consistency checks to maintain valid cached data.

Bufhigh

The **bufhigh** parameter specifies the maximum size in megabytes of allocated buffers for caching. OS/390 NFS client uses this buffer area to cache file data, thereby satisfying requests efficiently.

When a client system is suffering high CPU utilization and low throughput, increase the bufhigh storage size. Increasing the bufhigh storage size by twice the default value (within available region) may improve system performance. Also, if the client system is processing mostly large files, it is good to increase the bufhigh storage size. Any changes to the bufhigh parameter require restarting OS/390 UNIX.

Biod

The OS/390 NFS client uses Block I/O Daemons (BIOD) to perform asynchronous read or write when datacaching is enabled. The **biod** parameter specifies the number of asynchronous block I/O daemons to use. If there are a lot of users on the NFS client system, it may be advantageous to increase the number of BIODs. The maximum number of BIODs should never exceed two times the number of client system processors. This will avoid excessive TCB switching. Tuning of this parameter should be done with caution. Any change to the **biod** parameter require restarting OS/390 UNIX.

Readahead

With caching enabled, read requests may be satisfied from cache. The **readahead** parameter specifies the maximum number of disk blocks to read ahead. Thus, additional read requests for the file may be satisfied directly from cache. When application is anticipating mostly sequential reads, increase the readahead to the maximum. If application is mostly processing random reads, use the default or a minimal readahead value.

DelayWrite

With caching enabled, write requests may be cached. The **delaywrite** parameter specifies the maximum number of disk blocks to cache. The OS/390 NFS client will issue the WRITE RPC whenever the system load is low or if the cache data are in sequence. Delaywrite increases WRITE performance by overlapping the disk I/O. This performance increase is most apparent with many small writes or with many random writes. When there is a high volume of write requests, set the delaywrite parameter to the maximum value(32). If there is a low volume of write requests, set the delaywrite to the default (16).

Vers

SUN NFS Version 3 protocol support is available on OS/390 V2R6 NFS. It has performance enhancements over SUN NFS Version 2 protocol. The parameter does not have to be specified unless you would like to override the default. The OS/390 NFS client will communicate at the highest protocol level supported by the server, by default. Verify the protocol level for tuning of read or write buffer size (**wsize**, **rsize**). Use **nfsstat −m /mountpoint** command and look for **vers(n)** in the report or use **rpcinfo −p servername** command and look for **100003 3 udp 2049 nfs** in the report.

wsize and rsize

wsize and **rsize** specify the buffer size to use for read and write request. For NFS Version 2 protocol, **rsize** and **wsize** are a multiple of 512 bytes, up to a maximum of 8192 bytes.

For the NFS server with Version 3 protocol, **rsize** and **wsize** are negotiated between the OS/390 NFS client and the NFS server. The maximum buffer size that is supported in OS/390 NFS client system is 32KB. The negotiated read and write buffer size will be the smaller of the buffer size supported by the server and 32KB (min(server_info, 32KB)).

The OS/390 NFS client system will reduce the read or write buffer size when there is constraint on the storage resource.

When using NFS Version 3 protocol the system normally determines the optimal buffer values. However, if the number of retransmissions or the number of time-out from **nfsstat** report is high, reduce the write buffer size (wsize) to 8KB to avoid retransmission of large 32KB data.

Glossary

The following terms are defined as they are used in the OS/390 NFS Library. If you do not find the term you are looking for, see the IBM Software Glossary: http://www.networking.ibm.com/nsg/nsg

This glossary is an ever-evolving document that defines technical terms used in the documentation for many IBM software products.

A

access. To obtain computing services.

access permission. A group of designations that determine who can access a particular AIX or UNIX file and how the user can access the file.

ACS routine. A procedural set of ACS language statements. Based on a set of input variables, the ACS language statements generate the name of a predefined SMS class, or a list of names of predefined storage groups, for an MVS file.

AIX. Advanced interactive executive.

alias. An alternative name for an MVS user catalog, a non-VSAM file, or a member of a partitioned data set (PDS) or PDSE.

alias entry. An entry that relates an alias to the real entryname of a user catalog or non-VSAM data set.

allocation. Generically, the entire process of obtaining a volume and unit of external storage, and setting aside space on that storage for a data set.

ASCII. American National Standard Code for Information Interchange.

atime. The time when the file was last accessed.

automatic class selection (ACS). A mechanism for assigning Storage Management Subsystem classes and storage groups to data sets.

B

block. A string of data elements recorded, processed, or transmitted as a unit. The elements can be characters, words, or physical records.

C

client. (1) A user. (2) A consumer of resources or services. (3) A functional unit that receives shared services from a server. (4) A system that is dependent on a server to provide it with programs or access to programs. (5) On a network, the computer requesting services or data from another computer.

client-server relationship. Any process that provides resources to other processes on a network is a *server*. Any process that employs these resources is a *client*. A machine can run client and server processes at the same time.

connection. An association established between functional units for conveying information.

D

DA. Direct access.

DASD volume. A Direct Access Storage Device space identified by a common label and accessed by a set of related addresses. See also *volume,primary storage, migration level 1, migration level 2.*

DCB. Data control block.

data set. In DFSMS/MVS, the major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. In OS/390 non-UNIX environments, the terms *data set* and *file* are generally equivalent and sometimes are used interchangeably. See also *file*. In OS/390 UNIX environments, the terms *data set* and *file* have quite distinct meanings.

data set organization. The way in which data is arranged within an MVS file (data set) on a mainframe.

direct access file. A type of MVS file for storing data on a random access device that is accessed using a record address.

DOS. Disk operating system.

E

end user. A person in a data processing installation who requires the services provided by the computer system.

EOR. End of record.

ESDS. Entry-sequenced data set.

entry-sequenced data set (ESDS). A VSAM file whose records are loaded without respect to their contents, and whose relative byte addresses (RBAs) cannot change. Records are retrieved and stored by addressed access, and new records are added at the end of the file.

exports data set. An MVS file on the server containing entries for directories that can be exported to Network File System clients. It is used by the server to determine which MVS files and prefixes can be mounted by a client, and to write-protect MVS files on the server.

extended binary-coded decimal interchange code (EBCDIC). A coded character set consisting of 8-bit coded characters.

External Data Representation (XDR). A standard developed by Sun Microsystems, Incorporated for representing data in machine-independent format. XDR is a vendor independent way of representing the data. By using the XDR standard data representation convention, systems do not have to understand and translate every data format that exists on the network; there is only the one convention. Data is translated into XDR format before it is sent over the network and, at the reception point, is translated into the data convention used there. This means that new computer architectures can be integrated into the network without requiring the updating of translation routines. The new architecture simply includes a routine that translates its data format into XDR format and the new member of the network is ready to go.

Using XDR, data can be accessed or exchanged among machines of various hardware and software architectures without any translation or interpretation problems. Word lengths, byte ordering, and floating point representations appear to be the same to all nodes in the network.

F

file. A collection of information treated as a unit. In non-OpenEdition MVS environments, the terms *data set*and *file*are generally equivalent and are sometimes used interchangeably. See also *data set*.

file handle. A file handle is used by the client and server sides of the Network File System to specify a particular file or prefix.

file system. In the AIX operating system, the collection of files and file management structures on a physical or logical mass storage device, such as a diskette or minidisk.

File Transfer Protocol (FTP). A TCP/IP protocol used for transferring files to and from foreign hosts. FTP also provides the capability to access directories.

foreign host. Any host on the network other than the local host.

free space. Space reserved within the control intervals of a key-sequenced data set for inserting new records into the data set in key sequence or for lengthening records already there; also, whole control intervals reserved in a control area for the same purpose.

G

gateway. A functional unit that interconnects two computer networks with the different network architectures. A gateway connects networks or systems of different architectures. A bridge interconnects networks or systems with the same or similar architectures.

GID. See *group number*.

group. (1) With respect to partitioned data sets, a member and the member's aliases that exist in a PDS or PDSE, or in an unloaded PDSE. (2) A collection of users who can share access authorities for protected resources.

group number (GID). A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

H

hierarchical file system (HFS) data set. A data set that contains a POSIX-compliant hierarchical file system, which is a collection of files and directories organized in a hierarchical structure, that can be accessed using the OpenEdition MVS facilities.

host. A computer connected to a network that provides an access method to that network. A host provides end-user services.

I

ICF. Intersystem communications functions.

IDCAMS. Integrated catalog access methods services.

internet. See *internetwork*.

Internet. A specific internetwork that includes ARPANET, MILNET, and NSFnet. These networks use the TCP/IP protocol suite.

Internet Protocol (IP). The TCP/IP layer between the higher-level host-to-host protocol and the local network protocols. IP uses local area network protocols to carry packets in the form of diagrams to the next gateway or destination host.

internetwork. A collection of packet-switched networks that are connected by gateways. They work as a single network.

J

JCL. Job control language.

K

key-sequenced data set (KSDS). A VSAM data set whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the data set in key sequence because of free space allocated in the data set. Relative byte addresses of records can change because of control interval or control area splits.

L

local host. The computer to which a user's terminal is directly connected.

M

management class. A collection of management attributes, defined by the storage administrator, used to control the release of allocated but unused space; to control the retention, migration, and backup of data sets; to control the retention and backup of aggregate groups, and to control the retention, backup, and class transition of objects.

master catalog. A catalog that contains extensive data set and volume information that VSAM requires to locate data sets, to allocate and deallocate storage space, to verify the authorization of a program or operator to gain access to a data set, and accumulate usage statistics for data sets.

migration level 1. DFSMShsm-owned DASD volumes that contain data sets migrated from primary storage volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage* and *migration level 2*.

migration level 2. DFSMShsm-owned tape or DASD volumes that contain data sets migrated from primary storage volumes or from migration level 1 volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage* and *migration level 1*.

mkdir. An AIX, UNIX, OS/2 and DOS command used to make a new directory.

mount. (1) The process of accessing a directory from a disk attached to the machine making the mount request (4.2 mount), or to the remote disk on a network (Network File System mount). (2) An operation that associates a group of files on a server with a directory (mount point) on a client to provide transparent access to the files through that directory. The files must be in a hierarchical arrangement.

mount handle data set. A data set used to store the file handles of mount points.

mount point. A directory established in a workstation or a server local directory that is used during the transparent accessing of a remote file.

mtime. The time of last data modification.

MVS/ESA. An MVS operating system environment that supports ESA/390.

mvslogin. A client command to connect your workstation to MVS.

mvslogout (or mvslogut). A client command to break the connection between your workstation and MVS.

N

NFS. Network File System.

network. (1) An arrangement of nodes and connection branches. (2) A configuration of data processing devices and software connected for information interchange.

O

object storage hierarchy. A hierarchy consisting of objects stored in DB2 table spaces on DASD, on optical or tape volumes that reside in a library, and an optical or tape volumes that reside on a shelf. See also *storage hierarchy*.

OpenEdition MVS. See *OS/390 UNIX System Services*

optical volume. Storage space on an optical disk, identified by a volume label. See also *volume*.

OS/390 UNIX System Services (OS/390 UNIX). The set of functions provided by the SHELL and UTILITIES, kernel, debugger, file system, C/C++ Run-Time Library, Language Environment, and other elements of the OS/390 operating system that allow users to write and run application programs that conform to UNIX standards.

P

partioned data set (PDS). A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partitioned data set extended (PDSE). A system-managed data set that contains an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

PDS directory. A set of records in a partitioned data set (PDS) used to relate member names to their locations on a DASD volume.

permission code. A three-digit octal code, or a nine-letter alphabetic code, indicating the access permissions. The access permissions are read, write, and execute.

permission field. One of the three-character fields within the permissions column of a directory listing indicating the read, write, and run permissions for the file or directory owner, group, and all others.

primary storage. A DASD volume available to users for data allocation. The volumes in primary storage are called primary volumes. See also *storage hierarchy*. Contrast with *migration level 1* and *migration level 2*.

protocol. (1) A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication. (2) A specification for the format and relative timing of information exchanged between communicating parties.

R

relative byte addresses (RBA). The displacement of a data record or a control interval from the beginning of the data set to which it belongs; independent of the manner in which the data set is stored.

relative record data set (RRDS). A VSAM file whose records are loaded into fixed-length or variable-length slots. Each slot has a unique relative record number. Data is placed in a specific slot based on a user-supplied relative record number.

remote procedure call (RPC). RPC is an independent set of functions used for accessing remote nodes on a network. Using RPC network services, applications can be created in much the same way a programmer writes software for a single computer using local procedure calls. The RPC protocols extend the concept of local procedure calls across the network, which means that distributed applications can be developed for transparent execution across a network.

Resource Access Control Facility (RACF). An IBM-licensed program or a base element of OS/390, that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, logging the detected unauthorized attempts to enter the system, and logging the detected accesses to protected resources.

rmdir. An AIX, UNIX, OS/2 and DOS command used to remove (delete) a directory.

root. The user name for the system user with the most authority.

S

sequential file. A type of MVS file that has its records stored and retrieved according to their physical order within the file. It must be on a direct access volume.

server. (1) A functional unit that provides shared services to workstations over a network; for example, a file server, a print server, a mail server. (2) On a network, the computer that contains the data or provides the facilities to be accessed by other computers in the network. (3) A program that handles protocol, queuing, routing, and other tasks necessary for data transfer between devices in a computer system.

showattr. A client command used to display the values of the site, processing, and data set creation attributes.

stale file handle. A file handle is stale when the file handle for a file or prefix is no longer valid.

stateless. A stateless server can function correctly without maintaining any protocol state information about any of its clients. The NFS protocol is intended to be as stateless as possible. This avoids complex crash recovery; a client just resends requests until a response is received. The stateless protocol is chosen to minimize the probability of data losses due to a server crash.

storage hierarchy. An arrangement of storage devices with different speeds and capacities. The levels of the storage hierarchy include main storage (memory, DASD cache), primary storage (DASD containing uncompressed data), migration level 1 (DASD containing data in a space-saving format), and migration level 2 (tape cartridges containing data in a space-saving format). See also *primary storage*, *migration level 1*, *migration level 2*, and *object storage hierarchy*.

Storage Management Subsystem (SMS). A DFSMS/MVS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

T

tape volume. Storage space on a tape, identified by a volume label, which contains data sets or objects and available free space. A tape volume is the recording space on a single tape cartridge or reel. See also*volume*.

Transmission Control Protocol/Internet Protocol (TCP/IP). The two fundamental protocols of the Internet protocol suite. The abbreviation TCP/IP is frequently used to refer to this protocol suite. TCP/IP provides for the reliable transfer of data, while IP transmits the data through the network in the form of datagrams. Users can send mail, transfer files across the network, or execute commands on other systems.

U

user number (UID). In the AIX operating system, a number that uniquely identifies a user to the system. It is the internal number associated with the user ID.

V

Virtual Storage Access Method (VASM). An access method for direct or sequential processing of fixed and variable–length records on direct access storage devices. The records in a VSAM file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the file (entry sequence), or by relative record number.

volume. The storage space on DASD, tape, or optical devices, which is identified by a volume label. See also *DASD volume*, *optical volume*, and *tape volume*.

X

XDR. See *External Data Representation*.

Index

Readers' Comments — We'd Like to Hear from You

OS/390
Network File System
Performance Tuning Guide

Publication No. SC26-7255-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

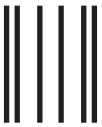
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SC26-7255-00

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Buisness Machines Corporation
RCF Processing Department
G26/M86 050
5600 Cottle Road
SAN JOSE, CA 95193-0001

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

File Number: S370-40
Program Number: 5647–A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC26-7255-00