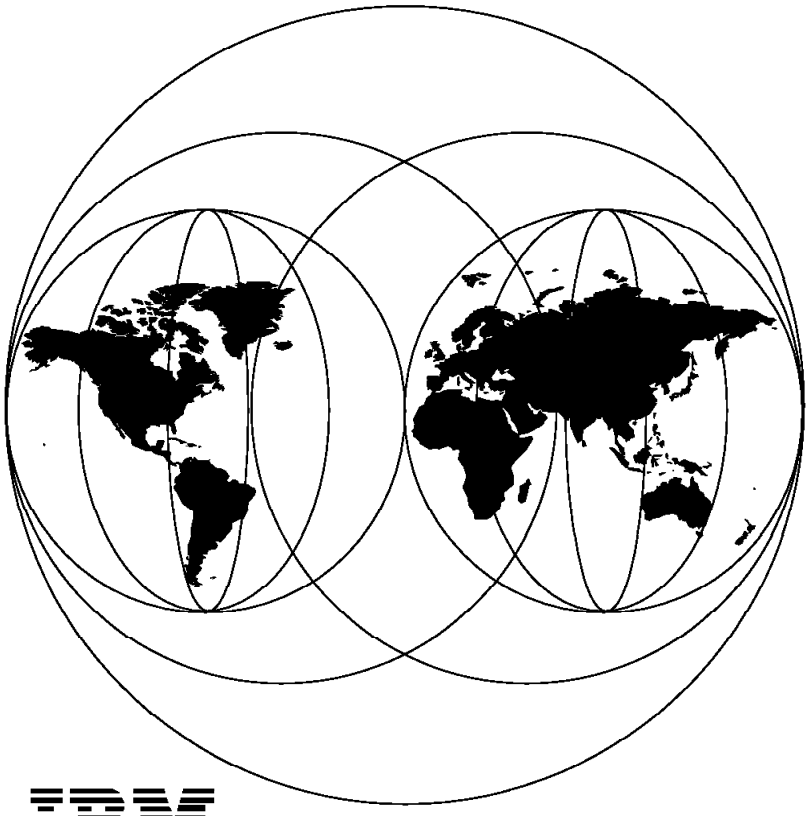
SG24-2158-00

Exploring NFS on AS/400

December 1997



IBM

International Technical Support Organization
Rochester Center

International Technical Support Organization

SG24-2158-00

Exploring NFS on AS/400

December 1997

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix H, “Special Notices” on page 171.

First Edition (December 1997)

This edition applies to the licensed program IBM Operating System/400 (Program 5716-SS1), Version 3 Release 7 Modification 0 and licensed program IBM TCP/IP Connectivity Utility for AS/400 (Program 5716-TC1), Version 3 Release 7 Modification 0.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. JLU Building 107-2
3605 Highway 52N
Rochester, Minnesota 55901-7829

Contents

Preface

This redbook explores the Network File System on the AS/400 system. The OS/400 Network File System (NFS) Support, which was introduced with V3R7 as part of TCP/IP Utility, provides a new system function for the AS/400 system that allows you to construct a distributed network system where all users can access the data they need.

This redbook will help you use the features of the NFS to share data across systems. It explains the security considerations to be aware of while using NFS. It then goes on to explore the National Language Support features offered by NFS. Further on, it describes the enhancements in the NFS when compared to the File Server Support/400, which was made available for versions of OS/400 earlier than V3R7. This book also includes the steps involved to migrate from FSS/400 to NFS. The final chapter in this redbook covers a few common problems that you might come across while using the NFS and how to solve them.

The intended audiences for this redbook are system administrators building up and maintaining a distributed network using NFS and AS/400 customers or programmers working with the Network File System. It is assumed that the reader has basic working experience with the AS/400 system.

How This Redbook is Organized

This redbook is organized as follows:

- Chapter 1, “Integrated File System Overview”

  This chapter provides an overview of the integrated file system and explains the various concepts involved in this. It also gives you the characteristics of individual file systems.

- Chapter 2, “Network File System Overview”

  This chapter gives you an introduction to network file system. It gives you details of AS/400 system as an NFS server and AS/400 system as an NFS client. Also this chapter gives you details of the lab environment used for the residency which produced this redbook.

- Chapter 3, “Operating and Using NFS”

  This chapter explains with example the various operations possible with NFS. These include starting the server, ending it, exporting a file system and mounting a file system. It also gives you details of the /etc files, which contain configuration information for NFS. File locking and recovery are also dealt with in this chapter.

- Chapter 4, “NFS Security”

  This chapter deals in detail with the security concepts of NFS. This chapter explains the UIDs and GIDs and various authority considerations for the various actions like changing ownership. Also given is the security checking flowchart and examples for various scenarios.

- Chapter 5, “National Language Support”

This chapter explains the national language support provided by NFS. It explains how to use code pages and how to create your own conversion tables.

- Chapter 6, “Migrating from FSS/400 to NFS”

  This chapter explains the differences in NFS compared to FSS/400. The steps involved in migrating from FSS/400 to NFS are also described.

- Chapter 7, “Common Problems Encountered in NFS”

  The common problems involved while starting an NFS Server are explained. Also problems relating to security and commands are listed using examples.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Rochester Center.

Suehiro Sakai is an Advisory International Technical Support Specialist for the AS/400 system at the International Technical Support Organization, Rochester Center. He writes extensively and teaches IBM classes worldwide in all areas of AS/400 communications. Before joining the ITSO a year ago, he worked in AS/400 Brand, Japan as an AS/400 Solution Specialist.

Thomas Barlen is a Software Specialist in Germany working in the AS/400 Software Support Center. He has been at IBM for 15 years and working on AS/400 systems for 9 years. His areas of expertise include AS/400 network connectivity and communications as well as LAN and WAN network design and implementation.

Divya Rajagopal is a Software Engineer in India. She has one year of experience in the field of application development on AS/400 systems. She has worked at IBM for one year. Her areas of expertise include AS/400 communications and Client Server Application development on AS/400 systems.

Thanks to the following people for their invaluable contributions to this project:

Ray Bills
IBM Rochester Development Lab
Dan Brossoit
IBM Rochester Development Lab
Frank Zhang
IBM Rochester Development Lab
Carl Pecinovsky
IBM Rochester Development Lab

Lois Douglas
ITSO Rochester Center

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 183 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

  For Internet users `http://www.redbooks.ibm.com`
  For IBM Intranet users `http://w3.itso.ibm.com`

- Send us a note at the following address:

  `redbook@vnet.ibm.com`

Chapter 1. Integrated File System Overview

This chapter provides an overview of the integrated file system (IFS). The first section of this chapter gives an introduction to the integrated file system and lists the various features of the integrated file system and its advantages. The second section of this chapter attempts to describe the various integrated file system concepts such as stream files, file systems, directories, and links. Then it goes on to discuss the various integrated file system commands, menus, and displays. Finally, this chapter covers the characteristics of individual file systems such as the Root, QOpenSys, QSYS.LIB, QDLS, QLANSrv, QOPT, QFileSvr.400, UDFS, NFS, and the QNetWare File Systems.

1.1 Introduction

The integrated file system is a part of OS/400 that supports stream input/output and storage management similar to the personal computer and UNIX operating system, while providing an integrating structure over all information stored in the AS/400 system.

The key features of the integrated file system are:

- Support for storing information in stream files that can contain long continuous strings of data.
- A hierarchical directory structure that allows objects to be organized and accessed by specifying the path through the directories to the object.
- A common interface that allows users and applications to access not only the stream files but also database files, documents, and other objects stored in the AS/400 system.
- A common view of all the stream files stored locally on the AS/400 system. Stream files can also be stored on a LAN server, a Novell NetWare server, another AS/400 system, or a Network File System (NFS) server.

The following benefits are provided by the integrated file system:

- Provides fast access to OS/400 data.
- Allows more efficient handling of the important types of stream data such as images, audio, and video.
- Provides a file system and directory base for supporting UNIX based open system standards such as POSIX.
- Allows file support with unique capabilities (such as record oriented database files, UNIX based stream files, and file serving) to be handled as separate file systems, while allowing them to be managed through a common interface.
- Allows PC users to take better advantage of their graphical user interface.
- Provides continuity of object names and associated object information across national languages.

1.2 Integrated File System Concepts

In this section, we summarize the various integrated file system concepts and terms such as the stream files, file systems, hierarchical directory structure, path names, and links.

1.2.1 Stream Files

The integrated file system provides support for storing and operating information in the form of stream files. A stream file is simply a file containing a continuous stream of data. Documents stored in the AS/400 folders are stream files. PC files and files in the UNIX systems also belong to the category of stream files. To better understand the concept of stream files, you can compare them with the AS/400 database files. A database file is record oriented and has predefined subdivisions consisting of one or more fields that have specific characteristics such as length and data type.

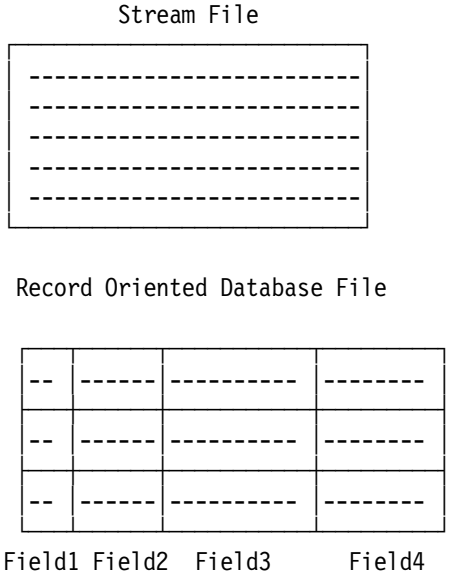


Figure 1. Comparison of a Stream File and Record Oriented File. This figure helps you to understand the concept of stream files by comparing it with the record-oriented structure of AS/400 database files.

Stream files are particularly well suited for storing strings of data such as the text of a document, images, audio, and video. An integrated file system stream file is a system object that has an object type of *STMF.

1.2.2 File Systems

A file system provides the support that allows users and applications to access specific segments of storage that are organized as logical units. These logical units are files, directories, libraries, and objects. Each file system has a set of logical structures and rules for interacting with information in storage. These structures and rules may be different from one file system to another. In fact, from the perspective of structures and rules, the OS/400 support for accessing database files and various other object types through libraries can be thought of as a file system. Similarly, the OS/400 support for accessing documents (which are actually stream files) through the folders structure may be thought of as a separate file system.

The integrated file system does indeed treat the library support and folders support as separate file systems. Other types of file management support that have differing capabilities are also treated as separate file systems. The individual file systems are explained in a later section.

The users and application programmers can interact with any of the file systems through a common IFS interface. This interface is optimized for input/output of stream data in contrast to the record input/output provided through the data management interfaces. A set of user interfaces (commands, menus, and displays) and application program interfaces (APIs) are provided for interacting with the file systems through this common interface.

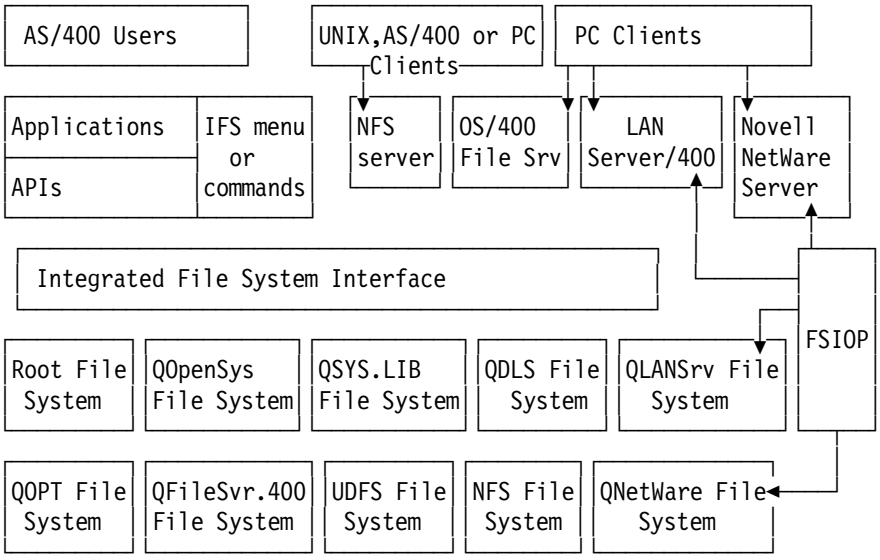


Figure 2. File Systems, File Servers, and IFS Interface

The OS/400 file server also uses a part of the IFS application interface. This file server provides file serving capabilities equivalent to shared folders, but allows PC clients to access information in any of the AS/400 file systems. The PC clients use their own user interfaces to give seamless access to the AS/400 file systems.

PC clients using the LAN Server requestor or Novell NetWare requestor interact with the Integrated PC Server, formerly known as the File Server I/O Processor (FSIOP), directly rather than the IFS interface. The Integrated PC Server provides even higher rates of access to its files.

1.2.3 Hierarchical Directory Structure

A directory is a special object that is used to locate objects by names specified by users. Each directory contains a list of objects that are attached to it. That list may include other directories. The integrated file system provides a hierarchical directory structure that allows users and application programs to access all objects in the AS/400 system. You might think of this directory structure as an inverse tree where the roots are at the top and the branches below. The branches represent directories in the directory hierarchy. These directory branches have subordinate branches that are called sub-directories.

Attached to the various directory and sub-directory branches are objects such as files. An object is located by specifying a path through the directories to the sub-directory to which the object is attached. Objects attached to a particular directory are sometimes described as being “in” that directory. A particular directory branch along with all of its subordinate branches (sub-directories) and all of the objects attached to those branches is referred to as a sub-tree.

Each file system is a major sub-tree in the integrated file system directory structure. In the library file system (QSYS.LIB sub-tree), a library is handled the same way as a sub-directory. Objects in a library are handled the same as objects in a sub-directory. Because database files contain objects (database file members), they are handled the same as sub-directories rather than objects. In the document library services file system (QDLS sub-tree), folders are handled the same as sub-directories and documents in folders are handled the same as objects in a sub-directory. Because of differences in file systems, the operations you can perform in one sub-tree of the directory hierarchy may not work in another sub-tree.

The integrated file system directory support is similar to the directory support provided by the DOS and OS/2 file systems. In addition, it provides features typical of UNIX systems such as the ability to store a file only once but access it through multiple paths using links.

From the viewpoint of a PC user who is connected to the AS/400 system through Client Access, the integrated file system is just another disk drive containing directories and objects. The highest level, the “root” (/) level, represents the entire AS/400 system. The AS/400 system appears as a drive letter that is similar to the A (diskette) drive and the C (disk) drive. Each directory at the next level below the root level appears as a directory in the “AS/400 drive.” This level includes the directories representing the file systems (such as QSYS.LIB and QDLS) and any first level sub-directories of the “root” (/) file system (such as home and tmp).

1.2.4 Path Names

Path name is a part of directory support. The support of path name and other related functions such as wild card support are almost identical to that of PCs and UNIX systems. A path name tells the system how to locate an object. The path name is expressed as a sequence of directory names followed by the name of the object. Individual directories and the object name are separated by a slash(/) character.

There are two ways of indicating a path name:

- An absolute path name begins at the highest level or root directory (which is identified by the / character). Let’s consider an example for this.

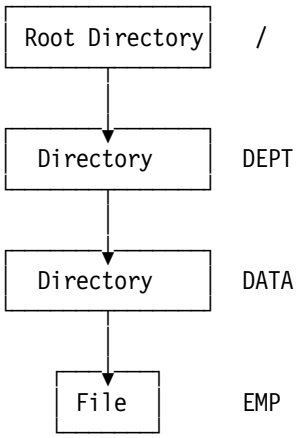


Figure 3. The Components of a Path Name

The absolute path name to the file EMP is /DEPT/DATA/EMP. The absolute path name is also known as the full path name.

- If the path name does not begin with the / character, the system assumes that the path begins at the user's current directory. This type of path name is called a relative path name. For example, for the preceding directory structure, if the current directory is DEPT, then the relative path name for EMP file is DATA/EMP. Notice that the path name does not include the name of the current directory. The first item in the name is the directory or object at the next level below the current directory.

1.2.5 Links

A link is a named connection between a directory and an object. A user or program can tell the system where to find an object by specifying the name of a link to the object. A link can be used as a path name or as part of a path name. There are two types of links: hard link and symbolic link.

1.2.5.1 Hard Link

A hard link, sometimes called just a link, cannot exist unless it is linked to an actual object. When an object is created in a directory, the first hard link is established between the directory and the object. Users and application programs can add other hard links. Each hard link is indicated by a separate directory entry in the directory. Links from the same directory cannot have the same name, but links from different directories can have the same name. There can be only one hard link from a directory to another directory. Hard links can be removed without affecting the existence of an object as long as there is at least one remaining hard link to that object. An object cannot be opened after the last hard link is removed. Also a hard link cannot cross file systems.

1.2.5.2 Symbolic Link

A symbolic link, which is called a soft link, is a path name contained in a file. When the system encounters a symbolic link, it follows the path name provided by the symbolic link and then continues on any remaining path that follows the symbolic link. Since the object pointed to by a symbolic link is resolved only when the link is used, a symbolic link can point to a non-existent object. Also a symbolic link can cross file systems.

1.3 IFS Menus, Displays, and Commands

This section introduces the IFS user interfaces such as menus, commands, and displays that are used to work with the stream files, directories, and other objects supported by the integrated file system.

1.3.1 IFS Menus and Displays

You can perform operations on files and other objects in the integrated file system by using a set of menus and displays provided by the system. The integrated file system menus can be requested through an option on the files, libraries, and folders menu.

- On any command line, enter:

  GO FILESYS

- Press the Enter key.

The following display is shown.

```
FILESYS              Integrated File System
                                                    System: ASSYS01
Select one of the following:

     1. Directory commands
     2. Object commands
     3. Security commands









Selection or command
 ===>

 F3=Exit   F4=Prompt   F9=Retrieve  F12=Cancel  F13=Information Assistant
 F16=AS/400 Main menu
 (C) COPYRIGHT IBM CORP. 1980, 1996.
```

To see a menu of mounted file system commands, at the command prompt type:

```
GO CMDMFS
```

Press the Enter key. The following display is shown.

```
CMDMFS              Mounted File System Commands

Select one of the following:

  Commands
     1. Add Mounted FS                                  ADDMFS
     2. Display Mounted FS Information                  DSPMFSINF
     3. Add Mounted FS                                  MOUNT
     4. Remove Mounted FS                               RMVMFS
     5. Display Mounted FS Information                  STATFS
     6. Remove Mounted FS                               UNMOUNT









Selection or command
===>

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F16=Major menu
(C) COPYRIGHT IBM CORP. 1980, 1996.
```

To see a menu of Network File System (NFS) commands, at the command prompt, type:

```
GO CMDNFS
```

Press the Enter key. The following display is shown.

```
CMDNFS              Network File System Commands

Select one of the following:

  Commands
     1. Change NFS Export                                CHGNFSEXP
     2. End NFS Server                                   ENDNFSSVR
     3. Change NFS Export                                EXPORTFS
     4. Start NFS Server                                 STRNFSSVR

  Related Command Menus
     5. Mounted File System Commands                     CMDMFS
     6. User-Defined FS Commands                         CMDUDFS




Selection or command
===>

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F16=Major menu
(C) COPYRIGHT IBM CORP. 1980, 1996.
```

To see a menu of user-defined file systems commands, at command prompt, type:

GO CMDUDFS

Press the Enter key. The following display is shown.

```
CMDUDFS             User-Defined FS Commands

Select one of the following:

  Commands
     1. Create User-Defined FS                          CRTUDFS
     2. Delete User-Defined FS                          DLTUDFS
     3. Display User-Defined FS                         DSPUDFS

  Related Command Menus
     4. Mounted File System Commands                    CMDMFS
     5. Network File System Commands                    CMDNFS




 Selection or command
===>

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F16=Major menu
(C) COPYRIGHT IBM CORP. 1980, 1996.
```

From the integrated file system menus, you can request displays on which you can do the following operations:

- Create and remove a directory.

- Display and change the name of the current directory.
- Add, display, change, and remove object links.
- Copy, move, and rename objects.
- Check out and check in objects.
- Save (back up) and restore objects.
- Display and change object owners and user authorities.
- Copy data between stream files and database file members.
- Create, delete, and display the status of user-defined file systems.
- Export file systems from a server.
- Mount and dismount file systems on a client.

A point to note is that some file systems do not support all of these operations.

1.3.2 IFS Commands

All the operations that you can do through the IFS menus and displays can also be done by using control language (CL) commands. These commands can operate on files and other objects in any file system that is accessible through the IFS interface.

1.3.2.1 Directory Commands

1. CHGCURDIR or CD or CHDIR command:
   - Syntax: CHGCURDIR DIR(dir1)
   - Description: Changes the current directory to dir1.
2. CRTDIR or MKDIR or MD command:
   - Syntax: CRTDIR DIR(dir1)
   - Description: Creates directory dir1.
3. DSPCURDIR command:
   - Syntax: DSPCURDIR
   - Description: Shows the name of the current directory on the display.
4. RMVDIR, RD or RMDIR command:
   - Syntax: RMVDIR DIRECTORY(dir1)
   - Description: Removes or deletes the directory dir1.
5. RTVCURDIR command:
   - Syntax: RTVCURDIR RTNDIR(NAME) DIRNAMLEN(LEN)
   - Description: Retrieves the name of the current directory into the variable NAME and length of the directory name into variable LEN. This command is used inside CL programs.

1.3.2.2 Link Commands

1. ADDLNK command:
   - Syntax: ADDLNK OBJ(obj1) NEWLNK('/obj2')
   - Description: Adds a hard or soft link between a directory and an object.
2. DSPLNK command:
   - Syntax: DSPLNK OBJ('/qdls')
   - Description: Shows a list of objects in a directory and provides options to display additional information about those objects.
3. RMVLNK command:
   - Syntax: RMVLNK OBJLNK(obj1)
   - Description: Removes a link to an object.
4. WRKLNK command:
   - Syntax: WRKLNK
   - Description: Shows a list of objects in a directory and provides a Program Development Manager (PDM) like interface for performing actions on objects.

1.3.2.3 Other Commands

1. CHGAUD command:
   - Syntax: CHGAUD OBJ('/qibm')
   - Description: Turns auditing on or off for an object.
2. CHGAUT command:
   - Syntax: CHGAUT OBJ('/qibm/file1') USER(NFSTEST) DTAAUT(*RWX) OBJAUT(*NONE)
   - Description: Gives specific authority for an object to a user or group of users. For more details, refer to Chapter 4, "NFS Security" on page 77.
3. CHGOWN command:
   - Syntax: CHGOWN OBJ('/qibm/file1') NEWOWN(NEWNFS)
   - Description: Transfers object ownership from one user to another user. For more details, refer to Chapter 4, "NFS Security" on page 77.
4. CHGPGP command:
   - Syntax: CHGPGP OBJ('/qibm/file1') NEWPGP(GROUP1)
   - Description: Changes the primary group from one group profile to another group profile. For more details, refer to Chapter 4, "NFS Security" on page 77.
5. CHKIN command:
   - Syntax: CHKIN OBJ('/qibm/file1')
   - Description: Checks in a previously checked out object.
6. CHKOUT command:
   - Syntax: CHKOUT OBJ('/qibm/file1')
   - Description: Checks out an object. Object can be read or copied but not changed.

7. CPY command:
   - Syntax: CPY OBJ('/qibm/file1') TOOBJ(file2)
   - Description: Copies an object or a group of objects.
8. CPYFRMSTMF command:
   - Syntax: CPYFRMSTMF FROMSTMF('STMF.TXT') TOMBR('/QSYS.LIB/MYLIB.LIB/MYFILE.FILE/MYMBR.MBR')
   - Description: This command copies the data contained in stream file STMF.TXT in the current working directory to database file member /QSYS.LIB/MYLIB.LIB/MYFILE.FILE/MYMBR.MBR. For this command, please note that the file MYFILE should either be a source physical file or it should be a physical file with only one field.
9. CPYTOSTMF command:
   - Syntax: CPYTOSTMFF FROMSTMF('STMF.TXT') FROMMBR('/QSYS.LIB/MYLIB.LIB/MYFILE.FILE/MYMBR.MBR') TOSTMF('STMF.TXT') For this command, please note that the file MYFILE should either be a source physical file or it should be a physical file with only one field.
   - Description: This command copies the data contained in database file member/QSYS.LIB/MYLIB.LIB/MYFILE.FILE/MYMBR.MBR to a stream file named STMF.TXT in the current working directory.
10. DSPAUT command:
    - Syntax: DSPAUT OBJ(/QSYS.LIB/MYLIB.LIB/MYFILE.FILE/MYMBR.MBR)
    - Description: Shows a list of authorized users of an object and their authorities for that object.
11. MOV command:
    - Syntax: MOV OBJ(obj1) TOOBJ(newobj)
    - Description: Moves an object to a different directory.
12. RNM command:
    - Syntax: RNM OBJ(obj1) NEWOBJ(obj2)
    - Description: Renames an object.
13. RST command:
    - Syntax: RST DEV('/QSYS.LIB/TAP01.DEVD') OBJ('/QSYS.LIB/MYLIB.LIB/*.FILE')
    - Description: This command restores all files in the library MYLIB from the tape device named TAP01.
14. SAV command:
    - Syntax: SAV DEV('/QSYS.LIB/MYLIB.LIB/MYSAVF.FILE') OBJ(MYDIR)
    - Description: This command saves the directory MYDIR to a save file named MYSAVF.
15. WRKAUT command:
    - Syntax: .WRKAUT
    - Description: Shows a list of users and their authorities and provides options for adding, changing, or removing a user's authority.

16. WRKOBJOWN command:
    - Syntax: WRKOBJOWN
    - Description: Shows a list of objects owned by a user profile and gives options for performing actions on the objects.
17. WRKOBJPGP command:
    - Syntax: WRKOBJPGP PGP(GRP01)
    - Description: Shows a list of objects controlled by a primary group and gives options for performing actions on the objects.

1.4 Characteristics of Individual File Systems

- Root File System

  The root file system (/) is designed to take full advantage of the stream file support and hierarchical directory structure of the IFS. It has characteristics of the DOS and OS/2 file system. The file names are not case sensitive here.

- QOpenSys File System

  The open systems file system is designed to be compatible with UNIX based open system standards such as POSIX and XPG. Similar to the root file system, it takes advantage of the stream file and directory support provided by the IFS. It supports case sensitive object names.

- QSYS.LIB File System QSYS.LIB is the library file system that supports the AS/400 library structure and provides access to the database files and all of the other AS/400 system object types that are managed by the library support.

- QDLS File System

  QDLS is the document library services file system. It supports the folders structure and provides access to documents and folders.

- QLANSrv File System

  The LAN server file system provides access to the same directories and files that are accessed through the LAN Server/400 licensed program. It allows users of the OS/400 file server and AS/400 applications to use the same data as LAN Server/400 clients.

- QOPT File System

  QOPT is the optical file system. It provides access to stream data stored on optical media.

- QFileSvr.400 File System

  The OS/400 file server file system provides access to other file systems residing on remote AS/400 systems.

- User Defined File System

  This file system resides on the Auxiliary Storage Pool (ASP) of the user's choice. The user creates and manages this file system.

- Network File System

  This file system provides the user with access to data and objects that are stored on a remote NFS server. Network file systems can be exported from

an NFS server and then dynamically mounted by NFS clients. We will be dealing with this file system in the rest of the redbook.

- QNetWare File System

  This file system provides access to local or remote data and objects that are stored on a server that runs Novell NetWare 3.12 or 4.10. A user can dynamically mount NetWare file systems over existing local file systems.

Chapter 2. Network File System Overview

This chapter provides an overview of the Network File System (NFS). The first section of this chapter gives an introduction to NFS with a brief history and attempts to describe its features as a file system. In the next section, we look at the NFS Client/Server model. This section describes the server and client side daemons and client side caches. In the next section, we give you the lab setup at IBM.

2.1 NFS Concepts

This section gives the reader an introduction to the Network File System (NFS). Then it goes on to give a brief history of NFS and later discusses the features of NFS as a file system.

2.1.1 Introduction

OS/400 Network File System support introduces a new system function for the AS/400 system that aids users and administrators who work with network applications and file systems. You can use the Network File System (NFS) to construct a distributed network system where all users can access the data they need. Furthermore, the NFS provides a method of transmitting data in a client/server relationship.

The NFS makes remote objects stored in file systems appear to be local, as if they reside in the local host. With NFS, all the systems in a network can share a single set of files. This eliminates the need for duplicate file copies on every network system. NFS gives users and administrators the ability to distribute data across a network by exporting local file systems from a local server for access by remote clients and mounting remote server file systems over local client directories. This allows AS/400 client systems to work with file systems that have been exported from a remote server. Refer to Chapter 3, “Operating and Using NFS” on page 25 for more details on exporting and mounting.

2.1.2 History

This section gives you an insight into the development history of NFS. Sun Microsystems, Inc. released NFS in 1984. Sun introduced NFS Version 2 in 1985. In 1989, the Request For Comments (RFC) standard 1094 describing NFS Version 2 was published. X/Open published a compatible version that is a standard for NFS in 1992. Sun published the NFS Version 3 protocol in 1993.

Sun developed NFS in a UNIX environment and, therefore, many UNIX concepts were integrated into the final protocol. Yet, the NFS protocol remains platform independent. Today, almost all UNIX platforms use NFS, as do many PCs, mainframes, and workstations. Most implementations of NFS are of Version 2, although several vendors are already offering products that combine Version 3 and Version 2. The basis for the AS/400 implementation of the Network File System is Version 2.

2.1.3 NFS as a File System

The Network File System provides “transparent” access to remote files. This means that local client files and files that are accessed from a remote server operate and function similarly and are indistinguishable. An efficient NFS network also gives the right people access to the right amount of data at the right times. Files and directories can be made available to clients by exporting from the server and mounting on clients through a pervasive NFS client/server relationship. An NFS client can also (at the same time) function as an NFS server just as an NFS server can function as a client. For more details on export and mount operations, refer to Chapter 3, “Operating and Using NFS” on page 25.

NFS incorporates the Remote Procedure Call (RPC) for client/server communication. RPC is a high-end network protocol that encompasses many simpler protocols such as Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

2.2 NFS Client/Server Model

To clearly understand how the Network File System (NFS) works on the AS/400 system, we explain the communication relationship between a server and various clients. The client/server model involves a local host (the client) that makes a procedure call that is usually processed on a different remote network system (the server). To the client, the procedure appears to be a local one even though another system processes the request. In some cases, however, a single computer can act as both an NFS client and an NFS server.

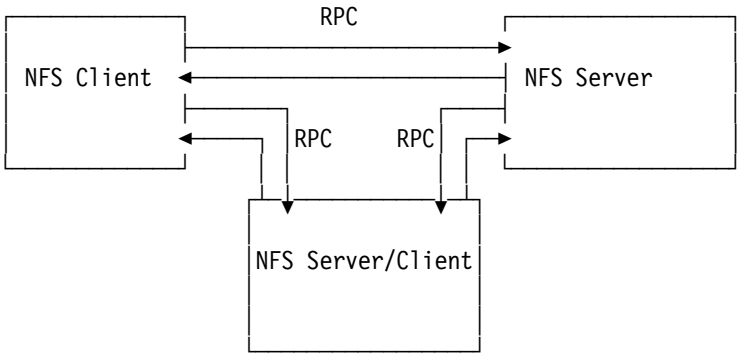


Figure 4. NFS Client/Server Model

RPC is the mechanism for establishing a client/server relationship within NFS. RPC bundles up the arguments intended for a procedure call into a packet of data called a network datagram. The NFS client creates an RPC session with an NFS server by connecting to the proper server for the job and transmitting the datagram to that server. The arguments are then unpacked and decoded on the server. The operation is processed by the server and a return message is sent back to the client. On the client, this reply is transformed into a return value for NFS. The user’s application is re-entered as if the process had taken place on a local level.

2.2.1 NFS Client/Server Design and Process Layout

The logical layout of the Network File System on the client and server involves numerous daemons, caches, and the NFS protocol breakdown. A daemon is a process that performs continuous or system-wide functions such as network control. NFS uses many different types of daemons to complete user requests. A cache is a type of high-speed buffer storage that contains frequently accessed instructions and data. Caches are used to reduce the access time for this information. Caching is the act of writing data to a cache.

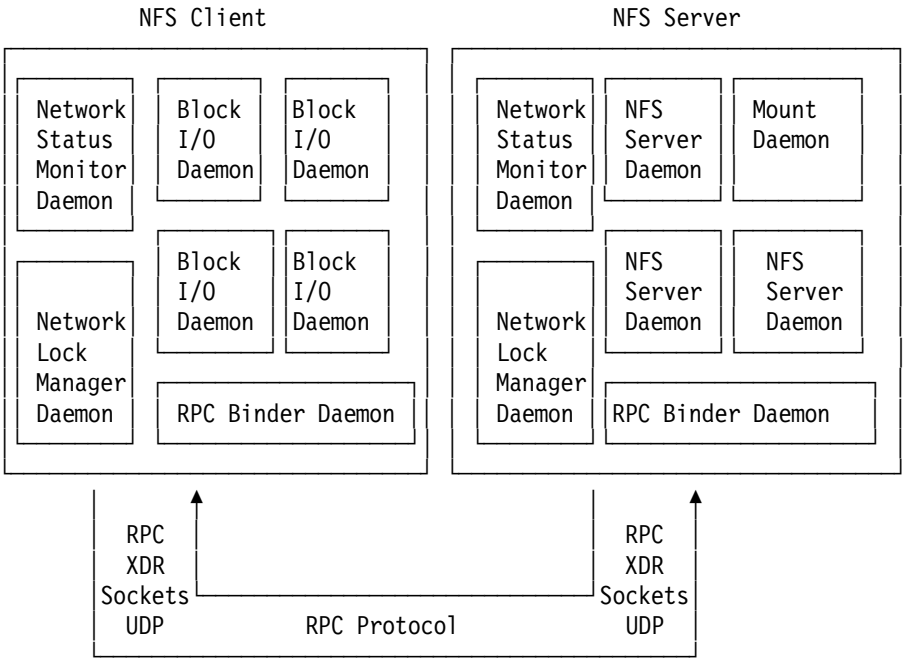


Figure 5. NFS Client/Server Protocol

Local processes that are known as daemons are required on both the client and the server. These daemons process both local and remote requests and handle client/server communication. Both the NFS client and server have a set of daemons that carry out user tasks. In addition, the NFS client also has data caches that store specific types of data locally on the client.

Simple low-end protocols make up a high-end complex protocol such as NFS. For an NFS client command to connect with the server, it must first use the Remote Procedure Call (RPC) protocol. The request is encoded into External Data Representation (XDR) and then sent to the server using a socket. The simple User Datagram Packet (UDP) protocol actually communicates between client and server. Some aspects of NFS use the Transmission Control Protocol (TCP) as the base communication protocol. The typical NFS flow includes the following steps:

1. The server waits for requests from one or more clients.
2. The client sends a request to the server and waits for a response.
3. When a request arrives, the server calls a dispatch routine.
4. The dispatch routine performs the requested service and returns with the results of the request. The dispatch routine can also call a sub-routine to

handle the specific request. Sometimes the sub-routine returns results to the client by itself, and other times it reports back to the dispatch routine.

5. The server sends those results back to the client.
6. The client then de-blocks.

2.2.2 AS/400 System as an NFS Server

The NFS server is composed of many separate entities that work together to process remote calls and local requests. These are:

- NFS server daemons:

  These daemons handle access requests for local files from remote clients. Multiple instances of particular daemons can operate simultaneously.

- Export command:

  This command allows a user to make local directories accessible to remote clients.

- /etc/exports file:

  This file contains the local directory names that the NFS server exports automatically when starting up. The administrator creates and maintains this file, which is read by the export command. Refer to Chapter 3, “Operating and Using NFS” on page 25 for more details.

- Export table:

  This table contains all the file systems that are currently exported from the server. The export command builds the /etc/exports file into the export table. Users can dynamically update the export table with the export command. This table is not directly accessible by the user; it is internal to the system.

2.2.2.1 NFS Server Side Daemons

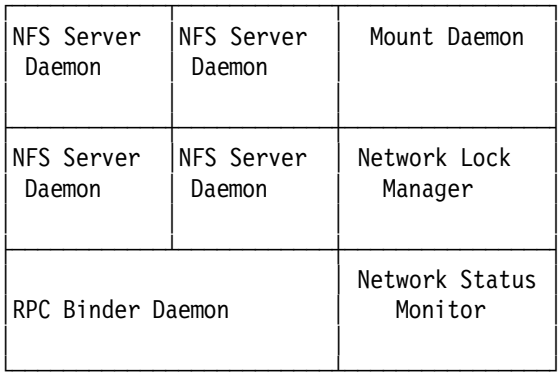


Figure 6. The NFS Server

NFS is similar to other RPC-based services in its use of server-side daemons to process incoming requests. NFS may also use multiple copies of some daemons to improve overall performance and efficiency.

We discuss the various daemons in the following list:

- RPC Binder Daemon (RPCD):

This daemon is analogous to the port mapper daemon that many implementations of NFS use in UNIX. Clients determine the port of a specified RPC service by using the RPC Binder Daemon. Local services register themselves with the local RPC binder daemon (port mapper) when initializing.

- NFS Server Daemons (NFSD):

  The most pressing need for NFS server daemons centers around the need for multi-threading NFS RPC requests. Running daemons in user-level processes allows the server to have multiple independent threads of processes. In this way, the server can handle several NFS requests at once. As a daemon completes the processing of a request, the daemon returns to the end of a line of daemons that wait for new requests. Multiple instances of this daemon can perform tasks simultaneously.

- Mount Daemon (MNTD):

  Each NFS server system runs a mount daemon that listens to requests from client systems. This daemon acts on mount and unmount requests from clients. If the mount daemon receives a client mount request, the daemon checks the export table. The mount daemon compares it with the mount request to see if the client is allowed to perform the mount. If the mount is allowed, the mount daemon sends an opaque data structure (the file handle to the requesting client). This structure uniquely describes the mounting point that is requested by the client. This enables the client to represent the root of the mounted file system when making future requests.

- Network Status Monitor Daemon (NSMD):

  The Network Status Monitor (NSM) is a stateful NFS service that provides applications with information about the status of network hosts. The Network Lock Manager (NLM) daemon heavily uses the NSM to track hosts that have established advisory byte-range locks as well as hosts that maintain such locks. There is a single NSM server per host. It keeps track of the state of clients and notifies any interested party when this state changes (usually after recovery from a crash). The NSM daemon keeps a notify list that contains information on hosts to be informed after a state change. After a local change of state, the NSM notifies each host in the notify list of the new state of the local NSM. When the NSM receives a state change notification from another host, it notifies the local network lock manager daemon of the state change.

- Network Lock Manager Daemon (NLMD)

  The Network Lock Manager (NLM) daemon is another stateful service that provides advisory byte-range locking for NFS files. The NLM maintains state across requests, and makes use of the Network Status Monitor daemon (NSM) that maintains state across crashes. The NLM supports two types of byte-range locks:

  – Monitored locks: These are reliable and helpful in the event of system failure. When an NLM server crashes and recovers, all the locks it had maintained are reinstated without client intervention. Likewise, NLM servers release all old locks when a client crashes and recovers. A Network Status Manager (NSM) must be functioning on both the client and the server to create monitored locks.

  – Unmonitored locks: These locks require explicit action to be released after a crash and re-established after start up. This is an alternative to

monitoring locks, which requires the NSM on both the client and the server systems.

2.2.3 AS/400 System as an NFS Client

Several entities work together to communicate with the server and local jobs on the NFS client. These processes are:

- RPC Binder Daemon: This daemon communicates with the local and remote daemons using the RPC protocol. Client requests for remote file systems are transmitted to the server in this fashion.
- Network Status Monitor and Network Lock Manager: These two daemons are not mandatory on the client. Many client applications, however, establish byte-range locks on parts of remote files on behalf of the client without notifying the user. For this reason, we recommend that the NSM and NLM daemons exist on both the NFS client and server.
- Block I/O daemon: This daemon manages the data caches and is, therefore, stateful in operation. It performs caching and assists in routing client-side NFS requests to the remote NFS server. Multiple instances of this daemon can perform tasks simultaneously.
- Data and attribute caches: These two caches enhance NFS performance by storing information on the client-side to prevent a client/server interaction. The attribute cache stores file and directory attribute information locally on the client while the data cache stores frequently used data on the client.
- Mount and unmount commands: Users can mount and unmount a file system in the client name space with these commands. These are general tools, used not only in NFS, but also to dynamically mount and unmount other local file systems. Refer to Chapter 3, “Operating and Using NFS” on page 25 for more details.

2.2.3.1 NFS Client Side Daemons

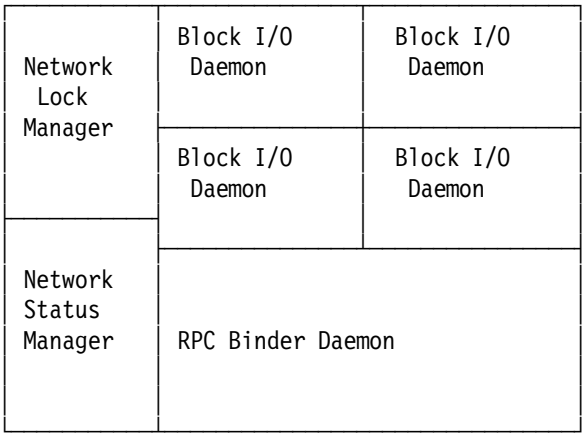


Figure 7. The NFS Client

Besides the RPC Daemon, the NFS client has only one daemon to process requests and to transfer data from and to the remote server, the block I/O daemon. NFS differs from typical client/server models in that processes on NFS clients make some RPC calls themselves, independently of the client block I/O daemon. An NFS client can optionally use both a Network Lock Manager (NLM) and a Network Status Monitor (NSM) locally, but these daemons are not required for standard operation. We recommend that you use both the NLM and NSM on your client because user applications often establish byte-range locks without the knowledge of the user.

The block I/O daemon handles requests from the client for remote files or operations on the server. Running only on NFS clients or servers that are also clients, this daemon manages the data and attribute caches for the user. The block I/O daemon is stateful and routes client application requests either to the caches or on to the NFS server. All data cached by the block I/O daemon is updated at regular intervals that a user can specify. Users can start multiple daemons to perform different operations simultaneously.

2.2.3.2 NFS Client Side Caches

Caching file data or attributes gives administrators a way of tuning NFS performance. The caching of information allows you to delay writes or to read ahead. Client-side caching in NFS reduces the number of RPC requests sent to the server. The NFS client can cache data, which can be read out of local memory instead of from a remote disk. The caching scheme available for use depends on the file system being accessed. Some caching schemes are prohibited because they cannot guarantee the integrity and consistency of data that multiple clients simultaneously change and update. The standard NFS cache policies ensure that performance is acceptable while also preventing the introduction of state into the client/server communication relationship. There are two types of client caches: the directory and file attribute cache and the data cache.

Directory and File Attribute Cache: Not all file system operations use the data in files and directories. Many operations get or set the attributes of the file or directory such as its length, owner, and modification time. Because these

attribute-only operations are frequent and do not affect the data in a file or directory, they are prime candidates for using cached information.

The client-side file and directory cache store file attributes so that every operation that gets or sets attributes does not have to go through the connection to the NFS server. When a file's attributes are read, they remain valid on the client for some minimum period of time. This time period can be set using the *acregmin* option on the mount command. If the client modifies the file (therefore updating its attributes), that change is made to the local copy of the attributes and the cache validity period is extended for another minimum time period. The attributes of a file remain static for a maximum period and then they are flushed from the cache and written back to the server if they have been modified. This time period can be set with the *acregmax* option on the mount command. To force a refresh of remote attributes when opening a file, do not use the *nocto* option on the mount command. Specifying the *noac* option suppresses all local caching of attributes, negating the *acregmin*, *acregmax*, *acdirmin*, and the *acdirmax* options on the mount command. Refer to Chapter 3, "Operating and Using NFS" on page 25 for more details.

The same mechanism is used for directory attributes. The minimum and maximum time period for directory attribute flushing from the cache is set by the *acdirmin* and *acdirmax* options on the mount command.

Attribute caching allows a client to make multiple changes to a file or directory without having to constantly get and set attributes on the server. Intermediate attributes are cached, and the sum total of all updates is later written to the server when the maximum attribute cache period expires. Frequently accessed files and directories have their attributes cached locally on the client so that some NFS requests can be performed without having to make an RPC call. By preventing this type of client/server interaction, caching attributes improves the performance of NFS.

Data Cache: The data cache is similar to the directory and file attribute cache in that it stores frequently used information locally on the client. The data cache, however, stores data that is frequently or likely to be used instead of file or directory attributes. The data cache provides data in cases where the client must access the server to retrieve information that has already been read. This operation improves the performance of NFS.

Whenever a user makes a request on a remote object, a request is sent to the server. If the request is to read a small amount of data (for example, one byte (B)), the server returns four kilobytes (KB) of data. This "extra" data is stored in the client caches because, presumably, it will soon be read by the client.

When users access the same data frequently over a given period of time, the client can cache this information to prevent a client/server interaction. This caching also applies to users who use data in one "area" of a file frequently. This is called locality and involves not only the primary data that is retrieved from the server, but also a larger block of data around it. When a user requests data frequently from one area, the entire block of data is retrieved and then cached. There is a high probability that the user will soon want to access this surrounding data. Because this information is already cached locally on the client, the performance of NFS is improved.

Client Timeout: If the client does not have a cache loaded, all requests go to the server. This takes extra time to process each client operation. With the mount command, users can specify a timeout value for re-sending the command. The client cannot distinguish between a slow server and a server that is not operational, so it will retry the command. The default timeout value is two seconds. If the server does not respond in this time, the client continues to retry the command, which overloads the server with client requests. The solution to this difficulty is to increase the timeout value on the mount command.

2.3 Lab Environment for this Residency

The test network for the residency is shown in the following figure.

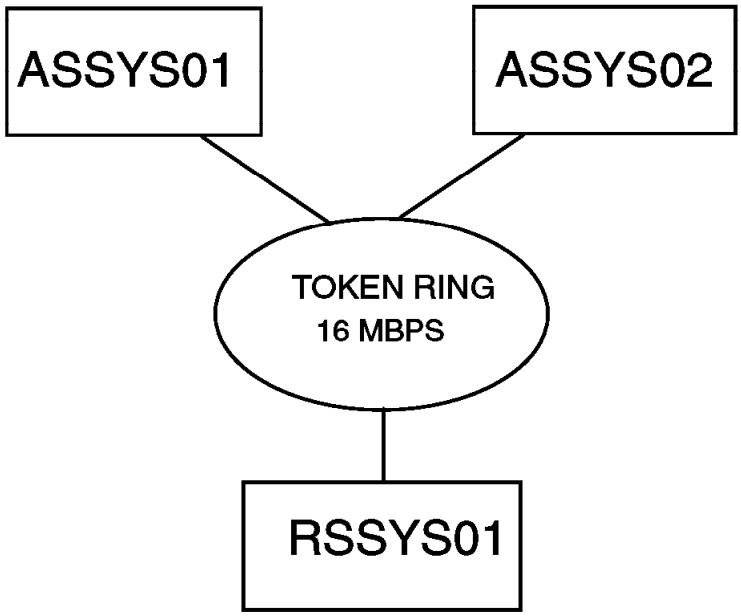


Figure 8. Test Network for Residency

The following descriptions are of the various systems shown in the preceding figure.

- AS/400 systems:
  - ASSYS01:
    - System Model: 436
    - OS/400 Version: V3R7
    - PTF level: C7252370
    - Additional PTFs: SF43263, SF41672, SF41686
  - ASSYS02:
    - System Model: 510
    - OS/400 Version: V3R7
    - PTF level: C7252370
    - Additional PTFs: SF43263, SF41672, SF41686
- RS/6000 System:
  - RSSYS01:
    - AIX Version: 4.1

Attention

If you are using a system with OS/400 version V4R1, please make sure that you have applied the following PTFs:

- SF41921
- SF41928

Chapter 3. Operating and Using NFS

This chapter contains information about the various operations performed for functioning of the NFS. First, we cover the start up and shut down of the NFS server. This section also lists the prerequisites for using NFS and the authorities you need to use the various features of NFS. Next we describe the server exporting of file systems using the EXPORTFS command. Then we describe the client mounting of the exported file systems. The various option parameters in the EXPORTFS and MOUNT command are explained in detail. Later we also cover the */etc* files, which are actually stream files that contain configuration information for the NFS.

3.1 Starting Up the NFS Server

This section discusses the proper startup scenario for the NFS server and client. Also, this section explains the Start NFS Server (STRNFSSVR) command in detail.

3.1.1 Typical Startup Procedure

The following startup procedure is typical for the NFS server:

1. The user starts the RPC binder (port mapper) daemon (QNFSRPCD). This daemon then waits on a known port (#111) for local RPC requests to register a service. This daemon also waits for remote RPC requests to query a local service.
2. The user calls the EXPORTFS command, which creates a list of exported directories in the export table from information contained in the /etc/exports file. For more information on these files, refer to Section 3.5.2, “Using the /etc/exports File” on page 54.
3. The user starts the NFS server daemon (QNFSNFSD) or daemons. It registers to the local RPC binder daemon, which knows on which port the NFS server waits for requests (the standard is #2049). All server daemons use this same port. The NFS server daemons then wait on the port for RPC requests from NFS clients to access local files.
4. The user starts the mount daemon (QNFSMNTD). This daemon registers to the local RPC binder daemon. It then waits on the assigned port for RPC requests from NFS clients to mount local file systems.
5. The user starts the NSM daemon (QNFSNSMD). It registers to the local RPC binder daemon. It then waits on the assigned port for RPC requests to monitor systems.
6. The user starts the NLM daemon (QNFSNLMD). It registers to the local RPC binder daemon. It then waits on the assigned port for RPC requests to manage locks.

If you specify *ALL for the SERVER parameter on the Start Network File System Server (STRNFSSVR) command, all of the daemons start automatically in the correct order.

Use the following steps for a typical NFS client startup:

1. The user starts the RPC binder (port mapper) daemon if it is not already operational. On a given system, a single port mapper is used for both client and server.
2. The user starts the block I/O daemon (QNFSBIOD) or daemons. This daemon controls the caching of data and attributes that have been transmitted from the server.
3. The user starts the NSM daemon if it is not already operational. On a given system, a single NSM operates both the client and server.
4. The user starts the NLM daemon if it is not already operational. On a given system, a single NLM is used for both the client and server.

3.1.2 STRNFSSVR Command

The Start Network File System Server (STRNFSSVR) command starts one or all of the Network File System (NFS) server daemons. Use the SERVER(*ALL) option, which starts the daemons in the following order as well as calling the EXPORTFS command. Use the following recommended order for starting the Network File System.

.

1. The Remote Procedure Call (RPC) binder daemon
2. The block I/O (BIO) daemon
3. Call the EXPORTFS command.
4. The server (SVR) daemon
5. The mount (MNT) daemon
6. The network status monitor (NSM) daemon
7. The network lock manager (NLM) daemon

If you attempt to start a daemon or daemons that are already running, they do not cause the command to fail. The command continues to start other daemons you have requested to start. The command issues a diagnostic message CPDA1BA if the daemon is already running. For best results, end NFS daemons before attempting the STRNFSSVR command.

To display NFS server daemons that are running, use the Work with Active Jobs (WRKACTJOB) command and look in the subsystem QSYSWRK for the existence of the following jobs:

- QNFSRPCD, the RPC Binder Daemon (RPCD)
- QNFSNFSD, the NFS Server Daemon (NFSD, there may be multiple entries for this daemon.)
- QNFSMNTD, the Mount Daemon (MNTD)
- QNFSNSMD, the Network Status Monitor Daemon (NSMD)
- QNFSNLMD, the Network Lock Manager Daemon (NLMD)
- QNFSBIOD, the NFS Client Block I/O Daemon

The following display shows the QSYSWRK subsystem when you use the Work with Active Jobs (WRKACTJOB) command.

```
                  Work with Active Jobs                  ASSYS01
                                                      09/30/97  11:29:47
CPU %:     .0     Elapsed time:   00:00:00     Active jobs:   153

Type options, press Enter.
  2=Change   3=Hold   4=End   5=Work with  6=Release  7=Display message
  8=Work with spooled files   13=Disconnect ...

Opt  Subsystem/Job  User        Type  CPU %  Function        Status
     QSYSWRK        QSYS        SBS      .0                 DEQW
       NTSIPCS      QSYS        BCH      .0  PGM-QFPAMONB    TIMW
       QAPPCTCP     QSYS        BCH      .0  PGM-QZPAIJOB    TIMW
       QCQEPMON     QSVMSS      BCH      .0  PGM-QCQEPMON    MSGW
       QCQRCVDS     QSVMSS      BCH      .0  PGM-QCQAPDRM    MSGW
       QECS         QSVSM       BCH      .0  PGM-QNSECSJB    DEQW
       QMSF         QMSF        BCH      .0                 DEQW
       QNETWARE     QSYS        BCH      .0  PGM-QFPANTWJ    DEQW
       QNFSBIOD     DIVYA       BCH      .0  DLY-1000        DLYW
                                                                 More..
Parameters or command
===>
F3=Exit    F5=Refresh  F10=Restart statistics F11=Display elapsed data
F12=Cancel F23=More options   F24=More keys
```

Use the Page Down key to scroll to the following display.

```
                  Work with Active Jobs                  ASSYS01
                                                      09/30/97  11:29:47
CPU %:     .0     Elapsed time:   00:00:00     Active jobs:   153

Type options, press Enter.
  2=Change   3=Hold   4=End   5=Work with  6=Release  7=Display message
  8=Work with spooled files   13=Disconnect ...

Opt  Subsystem/Job  User        Type  CPU %  Function        Status
       QNFSMNTD     DIVYA       BCH      .0                 SELW
       QNFSNFSD     DIVYA       BCH      .0                 TIMW
       QNFSNLMD     DIVYA       BCH      .0                 SELW
       QNFSNSMD     DIVYA       BCH      .0                 SELW
       QNFSRPCD     DIVYA       BCH      .0                 SELW
       QNPSERVD     QUSER       BCH      .0                 SELW
       QNSCRMON     QSVSM       BCH      .0  PGM-QNSCRMON    DEQW
       QPRFSYNCH    QSYS        BCH      .0  PGM-QFPAPRFJ    DEQW
       QQQTEMP1     QPGMR       BCH      .0  PGM-QQQTEMP1    DEQW
                                                                 More..
Parameters or command
===>
F3=Exit    F5=Refresh  F10=Restart statistics F11=Display elapsed data
F12=Cancel F23=More options   F24=More keys
```

Notice that there are different status values listed. The status of the first NFSD not in use is TIMW and all other NFSDs are listed as MTXW.

The Start NFS Server (STRNFSSVR) command has the following parameters:

- Server daemon:

This is a required parameter. The various options for this parameter are *ALL, *RPC, *BIO, *SVR, *MNT, *NSM, and *NLM. This suggests that you can start all of the daemons or any of them individually.

- Number of server daemons:

  This parameter specifies the number of NFS server (*SVR) daemon jobs you want to have running. Additional daemons are started if the number you specify on this parameter is greater than the number of server daemons already running on the system. This parameter can only be used if SERVER(*SVR) or *ALL is specified. For this parameter, you can have any integer value between one and 20. For best performance, you are advised to start more than one server daemon so that multiple RPC requests can be routed to different server daemons. This avoids a long RPC request from tying up the NFS server. The default value for this parameter is one.

- Number of block I/O daemons:

  This parameter specifies the number of NFS block I/O (*BIO) daemon jobs you want to have running. Additional daemons are started if the number you specify on this parameter is greater than the number of block I/O daemons already running on the system. This parameter can only be used if SERVER(*BIO) is specified. For this parameter, you can have any integer value between one and 20. The default value for this parameter is one.

- Timeout for start of daemon:

  This parameter specifies the number of seconds to wait for all daemons to successfully start. If all daemons have not started within the timeout value, the command will fail. For this parameter, you can specify any value between 1 and 3600 seconds. The default value for this parameter is 30 seconds.

The STRNFSSVR command is shown in the following display:

```
                  Start NFS Server (STRNFSSVR)

Type choices, press Enter.

Server daemon  . . . . . . . . . > *ALL        *ALL, *RPC, *BIO, *SVR...
Number of server daemons . . . .   1          1-20 server daemons
Number of block I/O daemons  . .   1          1-20 server daemons
Timeout for start of daemon  . .   30         1-3600 seconds

                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

3.1.3 Prerequisites for Using the STRNFSSVR Command

1. To use the STRNFSSVR command, you must first have TCP/IP operating on the AS/400 system.
2. The user must have *IOSYSCFG special authority to use this command.
3. The user must be enrolled in the system distribution directory. To enroll in the system distribution directory, use the Add Directory Entry (ADDDIRE) command.

```
                    Add Directory Entry (ADDDIRE)

Type choices, press Enter.

User identifier:
  User ID  . . . . . . . . . . > DIVYA         Character value
  Address  . . . . . . . . . . > ASSYS01       Character value
User description . . . . . . . > 'Test User Profile'

User profile . . . . . . . . . > TESTUSER      Name, *NONE
System name:
  System name  . . . . . . . .   *LCL          Character value, *LCL,
  System group . . . . . . . .                 Character value
Network user ID  . . . . . . .   *USRID

Last name  . . . . . . . . . .   *NONE

First name . . . . . . . . . .   *NONE
Middle name  . . . . . . . . .   *NONE
Preferred name . . . . . . . .   *NONE
                                                                  More...
F3=Exit   F4=Prompt   F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display       F24=More keys
```

3.2 Shutdown of the NFS Server

This section discusses the proper shutdown scenario for the NFS server and client. Also, this section explains the End NFS Server (ENDNFSSVR) command in detail.

3.2.1 Typical Shutdown Procedure

Shutting down an NFS server properly allows for all jobs to finish and all requests to complete. In general, the order of actions required for the server to shut down are the exact opposite of actions required for the server to start up:

1. The user ends the NLM daemon (QNFSNLMD).
2. The user ends the NSM daemon (QNFSNSMD). All locks that are held on local files by remote client applications are disengaged.
3. The user ends the mount daemon (QNFSMNTD). All remote client mounts of local file systems are disengaged.
4. The user ends the NFS server daemon (QNFSNFSD) or daemons.
5. The user ends the RPC binder (port mapper) daemon (QNFSRPCD).

If you specify *ALL for the SERVER parameter of the End Network File System Server (ENDNFSSVR) command, all of the daemons are automatically ended in the correct order.

The order of client shutdown processes is generally the opposite from which the user starts the processes.

1. The user ends the NLM daemon if it exists on the client. A single server NLM can operate both the client and server.
2. The user ends the NSM daemon if it exists on the client. A single server NSM can operate both the client and server.
3. The user ends the block I/O daemon (QNFSBIOD) or daemons.
4. The RPC binder (port mapper) daemon is ended.

One of the main considerations during shutdown is the TCP/UDP timeout conflict. When ending the NFS server, the socket port closes. If the NFS server is immediately re-started, the server may not be able to connect to the socket port. The underlying TCP/IP support on the AS/400 system renders this port unavailable for a short period. If you wait for a short period before re-starting the NFS server, it connects to the socket port as usual.

3.2.2 ENDNFSSVR Command

The End Network File System Server (ENDNFSSVR) command ends one or all of the Network File System (NFS) server daemons. Use SERVER(*ALL), which ends the daemons in the following order. This order is the recommended order for ending the Network File System daemons.

1. The network lock manager (NLM) daemon
2. The network status monitor (NSM) daemon
3. The mount (MNT) daemon
4. The server (SVR) daemon
5. The block I/O (BIO) daemon
6. The Remote Procedure Call (RPC) binder daemon

If you choose to end just one daemon, be sure you understand the appropriate order for ending NFS daemons and the possible consequences of ending daemons in an order other than that previously specified. If you attempt to end a daemon that is not running, it does not cause the command to fail, and it continues to end other daemons you have requested to end.

To display NFS client daemons, you can use the Work with Active Jobs (WRKACTJOB) command and look in the subsystem QSYSWRK for the existence of the following jobs:

- QNFSRPCD, the RPC Binder Daemon (RPCD)
- QNFSMNTD, the Mount Daemon (MNTD)
- QNFSNSMD, the Network Status Monitor Daemon (NSMD)
- QNFSNLMD, the Network Lock Manager Daemon (NLMD)
- QNFSBIOD, the Block I/O Daemon (BIOD, there may be multiple entries for this daemon).

The following display shows the various daemons.

```
                    Work with Active Jobs                   ASSYS01
                                                         09/30/97  12:28:57
CPU %:     .0     Elapsed time:   00:00:00     Active jobs:   153

Type options, press Enter.
  2=Change   3=Hold   4=End   5=Work with  6=Release  7=Display message
  8=Work with spooled files   13=Disconnect ...

Opt  Subsystem/Job  User        Type  CPU %  Function        Status
       QNFSMNTD     DIVYA       BCH      .0                  SELW
       QNFSNFSD     DIVYA       BCH      .0                  TIMW
       QNFSNLMD     DIVYA       BCH      .0                  SELW
       QNFSNSMD     DIVYA       BCH      .0                  SELW
       QNFSRPCD     DIVYA       BCH      .0                  SELW
       QNPSERVD     QUSER       BCH      .0                  SELW
       QNSCRMON     QSVSM       BCH      .0  PGM-QNSCRMON     DEQW
       QPRFSYNCH    QSYS        BCH      .0  PGM-QFPAPRFJ     DEQW
       QQQTEMP1     QPGMR       BCH      .0  PGM-QQQTEMP1     DEQW
                                                                  More..
Parameters or command
===>
F3=Exit    F5=Refresh  F10=Restart statistics F11=Display elapsed data
F12=Cancel F23=More options   F24=More keys
```

The End NFS Server (ENDNFSSVR) command has the following parameters:

- Server daemon:

  This is a required parameter and it specifies the daemons to be ended using the ENDNFSSVR command. If you specify this parameter as *ALL, all of the daemons are ended in the proper order.

- Timeout for end of daemon:

  This parameter specifies the number of seconds to wait for each daemon to end successfully. If a daemon has not ended within the specified timeout value, the command will fail. You can specify any integer value between 1 and 3600 for this parameter.

The ENDNFSSVR command is shown in the following display:

```
                    End NFS Server (ENDNFSSVR)

Type choices, press Enter.

Server daemon  . . . . . .   *all        *ALL, *RPC, *BIO, *SVR...
Timeout for end of daemon  . . .   30          1-3600 seconds

                                                                  Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

3.2.3 Prerequisites for Using the ENDNFSSVR Command

The user must have *IOSYSCFG special authority to use this command.

3.3 Exporting File Systems

This section covers the export function within NFS. There is information on how to export file systems, which includes all parameters and the restrictions that exist. On a couple of examples, you see the impact of the various parameters that can be specified.

3.3.1 Introduction to Exporting File Systems

The Network File System (NFS) as a server supports sharing file systems that include directories, data, or programs. The share function within NFS is called export. The OS/400 supports a typical generic command, which is the Change NFS Export (CHGNFSEXP) command as well as the command that is well known in the AIX or UNIX world, the Export File System (EXPORTFS) command. Both commands can be called from the command line and have exactly the same parameters.

In this publication, we only explain the EXPORTFS command even though the CHGNFSEXP command can also be used.

Exporting a file system means that you give access for objects located on one system (NFS server) to other systems (NFS clients) within an Internet Protocol (IP) network. NFS also provides security functions to control the access for different clients. There are several parameters that allow or deny client access to the exported directories.

Within the EXPORTFS command, you can also specify parameters that allow you to connect clients using different languages or code pages to the NFS server. This means you can export one file system to NFS clients using different code

pages. For detailed information about the National Language Support (NLS), refer to Chapter 5, “National Language Support” on page 117.

Once you have exported a file system, the NFS client has to access this data. This is done by the MOUNT or Add Mounted File System (ADDMFS) command. For further information on mounting file systems, refer to Section 3.4, “Mounting File Systems” on page 43.

3.3.2 The Export File System (EXPORTFS) Command

The EXPORTFS or CHGNFSEXP command lets you export a file system or a part of it. This command has different parameters; some of them are mandatory, others not.

Before you export a directory, make sure you know what you really want to export. For instance, you have a directory */account* with subdirectories for several departments. Imagine that you export this directory; then any client user from any department has access to all subdirectories of the */account* directory. Is that really what you want? In this case, export the directory */account/dept1* to all clients within department 1, then the appropriate directory for all clients within department 2, and so on. This gives access only to those users that are working within the appropriate department.

In any case, before exporting directories, create a plan and consider the real needs.

The following display shows the EXPORTFS command:

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . . . OPTIONS        *DFT_____________________
 _____________________________________________________________________
 _____________________________________________________________________
 _____________________________________________________
 Directory  . . . . . . . . . . . DIR            /home/jesse_____________
 _________________

                            Additional Parameters

 Host options:                     HOSTOPT
   Host name  . . . . . . . . . .                *DFT______
   Data file code page  . . . . .                __________
   Path name code page  . . . . .                __________
   Force synchronous write  . . .                __________
                           + for more values

                                                                      Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

The following list contains the explanation of each parameter that can be specified within the EXPORTFS command.

OPTIONS

This parameter consists of several values called flags.

***DTF**

The **DFT* value represents the *-A* flag.

-A When this flag is specified, no other parameter within the EXPORTFS command has to be filled out. The export function looks into the */etc/exports* file and exports all the entries defined within the file. For more information about the */etc* files, refer to Section 3.5, “Using the /etc Files” on page 51.

-I When the *-I* flag is specified, the definitions made within the */etc/exports* file are ignored. Instead, it uses the export information given within the *DIR* parameter as well as the additional export options that may specified within the *-O* flag.

-U This flag indicates that the specified directory tree entered in the *DIR* parameter is to be unexported. This means the exported directory is made unavailable to NFS clients. If the *-A* flag is also specified, all entries from the internal system export table are unexported. The internal system export table contains the currently exported directories specified within a single EXPORTFS command and the entries exported from the */etc/exports/* file.

-O The *-O* flag specifies the export characteristics for the directory tree to be exported. The options list following the -O flag list consists of keywords separated by commas. Some keywords are followed by an equal sign (=) and a value (or list of values separated by colons (:)). The options list may contain spaces. If a keyword is not specified, the default value for that option is used. The *-O* flag is only valid when the *-I* flag is specified. If options are required and not specified, the following options are the default options:

- *RW=:* All host names have read-write access to the directory tree.
- *ANON=UID:* This is associated with the profile QNFSANON.
- Requests to set bits in the mode other than the permission bits are allowed.
- Root access is not allowed for any hosts.
- All clients are allowed to mount the directory.

The following list contains the available options and their description:

RO Specifies the protection for the exported directory tree. If *RO* is specified, the directory tree is exported allowing read-only access to the directory and its contents. If it is not specified, read-write access is allowed to the directory and its contents.

RW= or RW=HOSTNAME1

Specifies the client host name or host names or netgroups that are allowed read-write access to the exported directory and its contents. For host names not specified here, the directory and its contents are exported allowing read-only access only.

In any case, even if you specify the *rw=* option for a client, the client user has to have the appropriate permission to perform write operations to the exported directory or object. If neither *RO* or *RW* is specified, then *RW=* is assumed, and all NFS clients have read-write access to the exported directory.

ANON=UID

If a client request comes in from an unknown user, this UID is used as the effective userid. The default value for this option is the UID associated with the user profile QNFSANON. If you do not want to allow any requests from unknown or anonymous users, use the value *ANON=-1*.

Note: Root users or NFS server users with *ALLOBJ special authority are considered unknown, unless the client system is specified on the ROOT= parameter.

NOSUID

This option specifies that any attempt by the client to enable bits other than the permission bits is ignored. If this option is not specified, attempts to set bits other than the permission bits are carried out. These bits are usually used by AIX or UNIX clients. They are stored within each object authority if the file system supports storing SUID bits. AS/400 NFS clients are not using these bits. They can be compared with a kind of authority adoption.

ROOT=HOSTNAME1

Specifies the NFS client host name or host names for which root access is allowed to the exported directory tree. If root access is allowed for a client host, an incoming UID of 0 is mapped to the user profile QSECOFR. Incoming requests from client users for which the user profile with the corresponding UID on the NFS server has the *ALLOBJ special authority are mapped to the correct user profile. If root access is not allowed for a client host, an incoming UID of 0 and incoming requests from users with *ALLOBJ special authority are mapped to the user profile specified in the ANON= parameter. Within this option you can specify host names or netgroups. If this option is not specified, no root access is granted to client hosts.

ACCESS=CLIENT1

This option specifies the client or clients that are allowed to mount the exported directory tree. A client can be a host name or a netgroup. If no clients are specified, all clients are allowed to mount the directory tree.

DIR Specifies the path name of the existing directory to be exported or unexported. This directory cannot be a subdirectory or a parent of an already exported directory (unless it is in a different file system). This parameter is not allowed when the *-A* flag is specified on the *OPTIONS* parameter.

HOSTOPT

The *HOSTOPT* parameter has four elements that specify additional information about the NFS clients that a directory tree is being exported to. If you do not specify the *HOSTOPT* parameter for a host name you are exporting the directory tree to, the defaults for each of the elements of the *HOSTOPT* parameter are assumed for that host.

The possible values are:

***DFT**

**DFT* specifies that the default values for the elements of the *HOSTOPT* parameter are used for all clients to which the directory

tree or directory trees are being exported. The data file code page is **BINARY*, the path name code page is **ASCII*, and Force synchronous write is **SYNC*.

Element 1 - Host name

The name of the host or a netgroup for which you are specifying additional options. This host should be specified in the *OPTIONS -O* list as a host that has access to the exported directory tree. Specify either a single host name that is an alias for an address of a single host or a netgroup name to be associated with these options. You can assign names to an Internet address with the Work with TCP/IP host table entries option on the Configure TCP/IP menu (CFGTCP command). Also, a remote name server can be used to map remote system names to Internet addresses.

Element 2 - Data file code page

The data file code page is used for data of the files sent to and received from the specified host name or group of hosts (netgroup name). For any hosts not specified on a *HOSTOPT* parameter, the default code page (binary, no conversion) is used. The code page may be one of the following types:

- **BINARY*

  Uses the default network data file code page (binary, no conversion).

- **ASCII*

  The ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used. For further information about the selection of equivalent ASCII code pages, refer to the publication *AS/400 International Application Development V3*, SC41-4603.

- **JOBCCSID*

  The code page obtained from the default job CCSID associated with the current job is used.

- *data file code page*

  A code page for data files. Values range from 1 to 32767. Conversion tables are not supported yet.

Element 3 - Path name code page

The path name code page is used for the path name components of the files or directories sent to and received from the specified host name or netgroup name. For any hosts not specified on a *HOSTOPT* parameter, the default path name code page (ASCII) is used. The code page may be one of the following types:

- **ASCII*

  The ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used. For further information about the selection of equivalent ASCII code pages, refer to the publication *AS/400 International Application Development V3*, SC41-4603.

- **JOBCCSID*

The code page obtained from the default job CCSID associated with the current job is used.

- *path name code page*

  A code page for path name components of files or directories. Specify a value in range from 1 to 32767.

Element 4 - Force synchronous write

Specifies whether write requests are handled synchronously or asynchronously for this host name or group of hosts (netgroup name). The default value of **SYNC* means that the data is written to the disk immediately. **ASYNC* does not guarantee that the data is written to the disk immediately and can be used to improve server performance.

- **SYNC*

  Write requests are performed synchronously.

- **ASYNC*

  Write requests are performed asynchronously.

Note: The Network File System (NFS) protocol Version 2 uses traditionally synchronous writes.

3.3.3 Using the EXPORTFS Command

When using the EXPORTFS command, you can specify parameters in various ways. Different parameter combinations may cause results that you do not expect.

You cannot export the same directory twice. Every time you perform the EXPORTFS command, the previous export is overwritten. So when you want to change one criteria, for instance, you have already exported a directory and specified the access parameter and now want to change the User ID (UID) for unknown client requests, you have to specify the entire parameter string again (in this case, the *access=* and the *anon=* option). Otherwise, the new export just contains the *anon=* option and the *access=* information is lost.

You cannot export all file systems that exist on the AS/400 system. The following list shows which file systems are supported to be exported:

- "Root" (/)
- QOpenSys
- QDLS
- QSYS.LIB
- QOPT
- UDFS

Usually the EXPORTFS command can only be used by a user that has **ALLOBJ* and **IOSYSCFG* special authority. If you want to authorize other users to use the command, you must give them **IOSYSCFG* special authority and private authority for the EXPORTFS command and the CHGNFSEXP command.

Attention

If you grant the permissions previously mentioned and you still receive error messages that the user does not have the proper authority, ensure you have installed the PTF level mentioned in Section 2.3, “Lab Environment for this Residency” on page 23.

If you want to make an already exported directory unavailable, you must use the *-U* export option. To unexport the directories that are currently exported, specify the *-U* and the *-A* flag. The message that appears after the unexport is shown in the following. display.

```
                    Additional Message Information

Message ID . . . . . :  CPCA1C6      Severity . . . . . : 00
Message type . . . . . :  Completion
Date sent  . . . . . :  10/01/97     Time sent  . . . . . : 11:27:31

Message . . . . :  1 entries exported, 0 entries not exported.
Cause . . . . . :  See job log (DSPJOBLOG command) or press F10 (Display
  the messages in job log) for additional information.









                                                                    Bottom
Press Enter to continue.

F3=Exit   F6=Print   F9=Display message details
F10=Display messages in job log  F12=Cancel  F21=Select assistance level
```

The message shown on the previous display is misleading. Even though the message text states that the entry is exported, it is actually unexported. The message just means that the EXPORTFS command completed successfully.

When you have clients that use code pages other than the code page used on the NFS server, you can specify a code page for data or path names for each client. For clients not defined within the *HOSTOPT* parameter, the default values are used. These are:

- **BINARY* for the Data File Code Page
- **ASCII* for the Path Name Code Page

The following examples show you how to use the EXPORTFS command and what the result is.

Example 1:

The first example shows the easiest way to export a directory.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . > '-i'


Directory  . . . . . > '/tools'









                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display        F24=More keys
```

The following list gives you some information about what the previous command does.

Flags and Options used
The *-I* export flag exports the directory specified within the *DIR* parameter. It ignores the definitions made within the */etc/exports* file. There are no more flags specified, which means that default options are used.

Exported directory
The */tools* directory is exported.

Clients access to the exported directory
Because default options are used, access is permitted to all NFS clients throughout the network.

Write Authority to the exported directory
Using the default export options means that all clients who mount the directory have write permission to it (only if the data permissions specified within IFS authority are sufficient). For changing authorities within IFS objects, refer to Chapter 4, “NFS Security” on page 77.

Root access
Since there is no *root=* option specified, no client user with the UID of 0 or a client user for which the corresponding user profile on the NFS server has the **ALLOBJ* special authority is mapped to the corresponding user profile. Instead, these users are mapped to the anonymous user profile QNFSANON.

Anonymous access
Because there is no *anaon=* option specified, all client requests from unknown users are mapped to the QNFSANON user profile.

Special bits
There is no *nosuid* options specified. This means client requests to change the “Set UID” and “Set GID” bits are accepted.

HOSTOPT parameter

The *HOSTOPT* is not specified; therefore, the defaults are used (**BINARY* for the data file code page, **ASCII* for the path name code page, and **SYNC* for the force synchronous write element) for all NFS clients.

Example 2:

The following display shows the EXPORTFS command with other parameters specified.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . > '-i -o access=sys01:sys02,rw=sys01'


Directory  . . . . . > '/tools'

                                                                  Bottom
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display        F24=More keys
```

The following list gives you some information about what the previous command does.

Flags and Options used

The *-I* export flag exports the directory specified within the *DIR* parameter. It ignores the definitions made within the */etc/exports* file. The *-O* flag is also specified, which means options follow.

Exported directory

The */tools* directory is exported.

Clients access to the exported directory

Because the *access=* option is specified, only client *sys01* and *sys02* are allowed to mount the exported directory.

Write Authority to the exported directory

There is one client specified within the *rw=* option. In this case, only users from client *sys01* have the permission to perform write operations within the exported directory and all users from system *sys02* have read only (RO) access to the exported directory.

Root access

Since there is no *root=* option specified, no client user with the UID of 0 or a client user for which the corresponding user profile on the NFS server has the **ALLOBJ* special authority is mapped to the corresponding user profile. Instead, these users are mapped to the anonymous user profile QNFSANON.

Anonymous access

Because there is no *anaon=* option specified, all client requests from unknown users are mapped to the QNFSANON user profile.

Special bits

There is no *nosuid* options specified. This means client requests to change the “Set UID” and “Set GID” bits are accepted.

HOSTOPT parameter

The *HOSTOPT* is not specified; therefore, the defaults are used (**BINARY* for the data file code page, **ASCII* for the path name code page, and **SYNC* for the force synchronous write element) for all NFS clients.

Example 3:

The following display shows the EXPORTFS command with other parameters specified.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . > '-i -o access=sys01:sys02,anon=-1,root=sys02'


Directory  . . . . . > '/pgms'


                           Additional Parameters

 Host options:
   Host name  . . . . . .   sys02     Character value, *DFT
   Data file code page  . . . . .   *binary   1-32767, *BINARY, *ASCII
   Path name code page  . . . . .   037       1-32767, *ASCII, *JOBCCSID
   Force synchronous write  . . .   *sync     *SYNC, *ASYNC
               + for more values

                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display        F24=More keys
```

The following list gives you some information about what the previous command does.

Flags and Options used

The *-I* export flag exports the directory specified within the *DIR* parameter. It ignores the definitions made within the */etc/exports* file. The *-O* flag is also specified, which means that options follow.

Exported directory

The */pgms* directory is exported.

Clients access to the exported directory

Because the *access=* option is specified, only client *sys01* and *sys02* are allowed to mount the exported directory.

Write Authority to the exported directory

Since there is no *rw=* option specified, both clients have the permission to perform write operations within the exported directory.

Root access

The *root=* option is specified for the *sys02* client. This means for the *sys02* client, the client user with the UID of 0 or a user for which the corresponding user profile on the NFS server has the **ALLOBJ* special authority is mapped to the corresponding user profile. Users meeting the same criteria from the NFS client *sys01* are not allowed to work with the exported directory at all. The reason is that anonymous access is not allowed.

Anonymous access

The option *anaon=-1* causes all client requests from anonymous users (see also the *root=* option) to be rejected. This means no anonymous access is allowed.

Special bits

There is no *nosuid* options specified. This means client requests to change the “Set UID” and “Set GID” bits are accepted.

HOSTOPT parameter

The *HOSTOPT* is specified for the NFS client *sys02*. For this client, the path name code page of 037 is used. For the client *sys01*, the defaults are used (**BINARY* for the data file code page, **ASCII* for the path name code page, and **SYNC* for the force synchronous write element).

Example 4:

The following display shows the EXPORTFS command with other parameters specified.

```
                     Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . > '-u'


Directory  . . . . . > '/tools'

                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display        F24=More keys
```

The following list gives you some information about what the previous command does.

Flags and Options used

The *-u* export flag unexports the directory specified within the *DIR* parameter.

Note: For unexporting all entries currently exported, specify the *-u -a* flags.

Exported directory
The */tools* directory is unexported (this means made unavailable).

Example 5:

The following display shows the EXPORTFS command with other parameters specified.

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-a'


 Directory  . . . . . >

                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
 F13=How to use this display        F24=More keys
```

The following list gives you some information about what the previous command does.

Flags and Options used
The *-a* export flag exports all the directory specified within the */etc/exports* stream file.

Exported directory
The *DIR* parameter remains empty.

3.4 Mounting File Systems

This section covers the mount function within NFS. After giving an introduction to mounting file systems, we cover the mount command in detail with special emphasis on the various parameters and options you can specify for these parameters.

3.4.1 Introduction to Mounting

After exporting a file system, the next major step in setting up a transparent relationship between client and server is mounting the file system. The mount command places the remote file system over a local directory on an NFS client. Mounting allows clients to actually make use of the various file systems that the server has exported. Clients can use the mount command to map an exported file system over all or just part of a local file system. This action occurs so

seamlessly that local applications probably do not distinguish between file systems mounted from a remote server and file systems existing locally. Multiple clients can mount and work with a single or multiple file systems at the same time. Once file systems have been exported from a remote server, clients can then mount these accessible file systems and make them a part of their local namespace. Clients can dynamically mount and unmount all or part of exported server file systems. For more details on the export function, refer to Section 3.3, “Exporting File Systems” on page 32.

Given the proper authority, an NFS client can mount any file system or part of a file system that has been exported from an NFS server. Mounting is the local client action of selecting an exported directory from a remote server and making it accessible to the integrated file system namespace of the local client. When file systems are mounted on the client, they “cover up” any file system, directories, or objects that exist beneath the mount point. This means that mounted file systems also cause any file systems, directories, or objects that exist locally downstream from the mount point to become inaccessible. This aspect of mounting causes whatever local data beneath the newly mounted file system to become invisible to the user. They do not appear in the integrated file system namespace. These covered file systems are inaccessible until the mounted file system is unmounted. The client can dynamically unmount any file system that has previously been mounted. This action allows for uncovering a file system, directory, or object that has been mounted over. It also breaks the connection with the server (and, therefore, access) for that particular mounted file system.

Users can mount three different types of file systems on the AS/400 system:

- Network File Systems
- User-Defined File Systems
- Novell NetWare file systems

3.4.2 Mount Points

Mount points mark the area of the local client and remote server namespaces where users have mounted exported file systems. Mount points show where the file system has been mounted from on the server and show where it is mounted to on the client. It is possible to mount an NFS file system over all or part of another client file system. This is possible because the directories used as mount points appear the same no matter where they actually reside.

To the client, NFS file systems appear to be and function as normal, local file systems. Users can mount network file systems over the following AS/400 client file systems:

- Root (though not over the root directory itself)
- QOpenSys
- NFS
- UDFS

When a client mounts an exported file system, the newly mounted file system covers up whatever is beneath it. This is true for mounting remote file systems over local directories as well as mounting remote file systems over previously-mounted remote file systems. Any file system that is covered up in such a manner is inaccessible until all of the file systems on top are unmounted. For example, in the following figure, we find the view of the mount point before and after mounting. Below the mount point RSMOUNT, we see a directory

clientdir1, but once the mount is over, we find that this directory is no longer accessible.

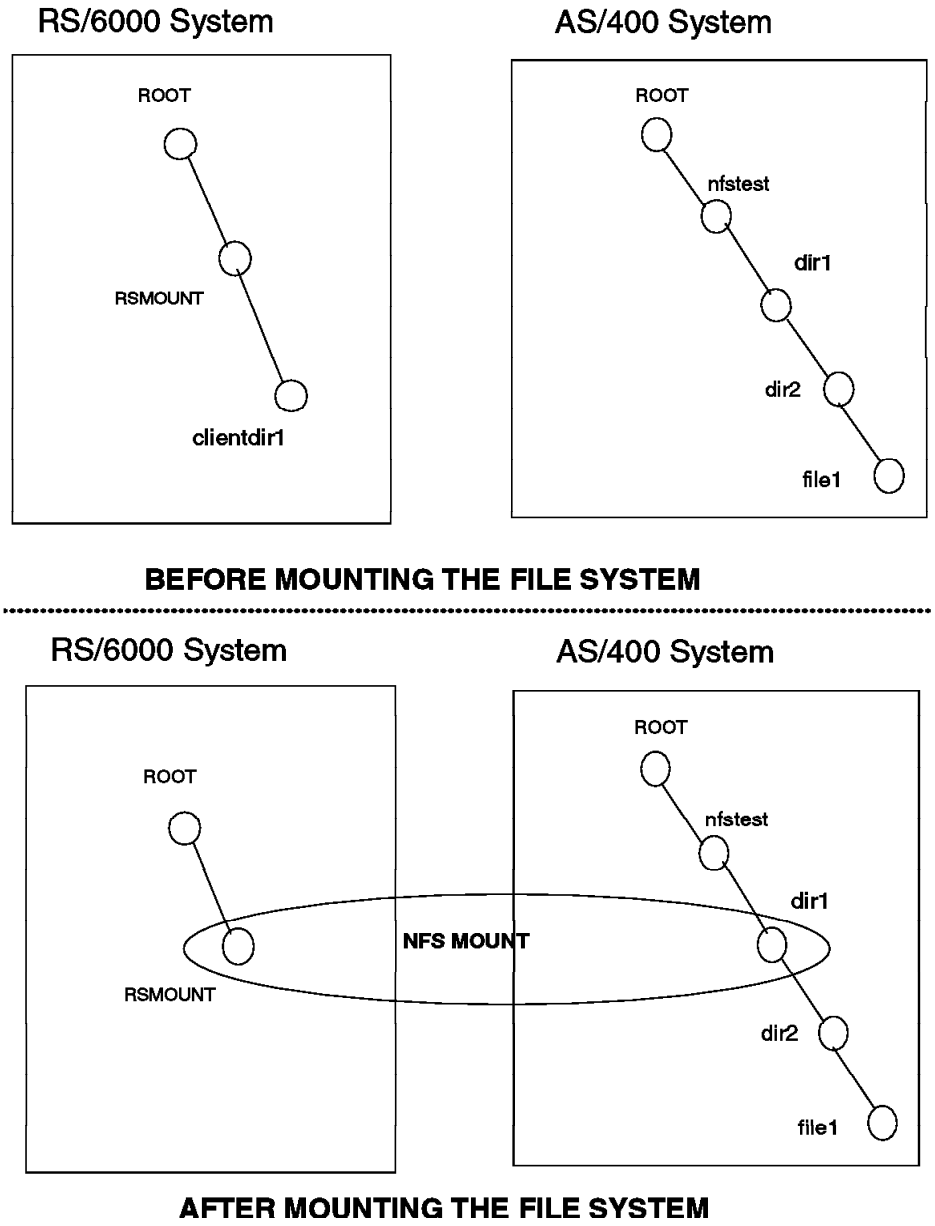


Figure 9. Mount Point Before and After the MOUNT Command

We notice that the mounted file system completely covers up everything on the local directory tree that is downstream from the mount point. The mount point in the preceding example is /root/rsmount.

3.4.3 The MOUNT Command

The MOUNT command makes the objects in a file system accessible to the integrated file system name space. The following file systems can be mounted:

- A user-defined file system (*UDFS) on the local system
- A remote file system accessed through a local Network File System client (*NFS)

- A local or remote NetWare file system (*NETWARE)

The directory that is the destination for the mount must exist. After completing the mount, the contents of this directory are ″covered up″ and rendered inaccessible to the integrated file system namespace. Users can issue this command by using the alternative Add Mounted File System (ADDMFS) command.

The MOUNT command has the following parameters:

- Type of file system:

  This is a required parameter. This parameter specifies the type of the file system to be mounted. The possible values for this parameter are:

  – *NFS: This value for the parameter indicates that the file system to be mounted is a Network File System. If this value is specified for the TYPE parameter, the MFS parameter should be hostname (pathname where hostname may either be the name of the system or its IP address).

  – *UDFS: This value specifies the user-defined file system.

  – *NetWare: This value specifies the NetWare file system.

- File system to mount (MFS):

  This is a required parameter. This parameter specifies the path name of the file system to be mounted. It can be the path to a local Block Special File (*BLKSF), a remote NFS path name, or the path of a NetWare file system. The MFS format should match the value specified in the TYPE parameter. For NFS, the parameter should be specified as hostname (pathname where the hostname may either be the name of the system or the IP address of the system).

- Directory to mount over (MNTOVRDIR):

  This is a required parameter. In this parameter, you have to specify the path name of the existing directory over which the file system has to be mounted. This directory gets covered by the mounted file system.

- Mount options (OPTIONS):

  This parameter holds the options for the mount command. This is a list of keywords (discussed later) separated by commas. If a keyword is not specified, the default value for that option is used. The various options for this parameter are discussed:

  – *DFT: This is the default value for the options parameter. The values of the various keywords for this option for NFS are given below *′rw, suid, rsize=8096, wsize=8096, timeo=20, retrans=5, acregmin=30, acregmax=60, acdirmin=30, acdirmax=60, hard′*

    Now we discuss the various keywords in detail.

  The various keywords in the options list are:

  – *rw|ro*

    This option specifies the protection for the mounted file system. Either ro (read-only) or rw (read-write) may be specified. If neither is specified, rw is assumed. If the ro option is specified, we get a CPFA0A1 error when we try to create objects within the mounted directory.

  – *suid|nosuid*

For mounting a Network File System, if suid is specified, then setuid execution is allowed. This means that bits other than the permission bits may be set. It nosuid is specified or if nothing is specified for this option, then setuid execution is not allowed.

– *hard|soft*

This option specifies if the NFS mount is hard or soft mounted. Hard mounted means that all operations on them are retried until they are acknowledged by the server. Soft mounted means that a timeout error is returned if a remote operation fails the number of times specified on the *retrans* parameter. If neither is specified, hard mount is assumed. If this parameter is specified as hard, the retrans parameter is ignored.

– *rsize=n*

This option allows you to specify the size of the read buffer in bytes. This read buffer is used for data transfer between the NFS client and the remote NFS server on an NFS read request. For this option, you can specify an integer value in the range of 512 to 8096. If rsize is not specified, the default value of 8096 is assumed. For good performance, we recommend that the read buffer should be a multiple of the application buffer size.

– *wsize=n*

This option allows you to specify the size of the write buffer in bytes. This write buffer is used for data transfer between the NFS client and the remote NFS server on an NFS write request. For this option, you can specify an integer value in the range of 512 to 8096. If wsize is not specified, the default value of 8096 is assumed. For good performance, we recommend that the write buffer should be a multiple of the application buffer size.

– *timeo=n*

This option allows you to specify the amount of time, in tenths of seconds, to wait for the client to respond on each try. The allowed range is 0 to 10000. If timeo is not specified, the default value of 20 tenths of a second (2 seconds) is assumed. The value of this parameter is decisive in the performance of your client. If you specify a small value for this, you flood your server with the same request. So it is advised that you specify a fairly large but reasonable value for the timeo parameter.

– *retry=n*

This option allows you to specify the number of times to retry the mount operation. The allowed range is 0 to 10000. If retry is not specified, the default value of five times is assumed. This value is used only for the initial mount operation.

– *retrans=n*

This option allows you to specify the number of times to retry the transmission to the client. The allowed range is 0 to 10. If retrans is not specified, the default value of five re-transmission attempts is assumed.

– *acregmin=n*

This option allows you to specify the minimum number of seconds to hold locally stored file attributes after file updates. The allowed range is 1 to 3600. If acregmin is not specified, the default value of 30 seconds is assumed.

– *acregmax=n*

This option allows you to specify the maximum number of seconds to hold locally stored file attributes after file updates. The allowed range is 1 to 2 000 000 000. If acregmax is not specified, the default value of 60 seconds is assumed. For best performance, we suggest that you increase the acregmin and acregmax values.

– *acdirmin=n*

This option allows you to specify the minimum number of seconds to hold locally stored directory attributes after a directory update. The allowed range is 1 to 3600. If acdirmin is not specified, the default value of 30 seconds is assumed.

– *acdirmax=n*

This option allows you to specify the maximum number of seconds to hold locally stored directory attributes after a directory update. The allowed range is 1 to 3600. If acregmax is not specified, the default value of 60 seconds is assumed.

– *nocto*

This option allows you to specify whether to force the refresh of remote attributes when opening a file. If this keyword is present, attributes are not refreshed from the server when opening a file, and changes are not sent to the server on the last close. If nocto is not present, the default value of refresh is assumed.

– *noac*

This option allows you to specify whether to suppress local storage of attributes and names. If this keyword is present, local storage of attributes and names is suppressed. If noac is not present, the default value of no suppression is assumed. If noac is specified, you do not use the attribute cache. For good performance, we suggest that you use attribute caches. This means that you should try to avoid using noac for best performance. If you specify noac, the values specified for acregmin, acregmax, acdirmin, and acdirmax may be specified but these values are not used.

• Code page parameters:

There are two code page parameters. The first code page parameter specifies which code page should be used for data files on the remote system. The second code page specifies which code page should be used for path names on the remote system.

– Data file code page parameter:

The possible values for this parameter are:

- *BINARY

This value for the parameter implies that no conversion is to be used.

- *ASCII

This value for the parameter uses the ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job for conversion.

- *JOBCCSID

The code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used for conversion purposes.

- data-code-page

  A code page that is explicitly specified is to be used for data files on the remote system. Please take care to specify a code page that has the same number of bytes per character as the original data. Single-byte or double-byte code pages are supported, but mixed code pages are not supported.

– Path name code page parameter:

  The possible values for this parameter are:

  - *ASCII

    This value for the parameter uses the ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job for conversion.

  - *JOBCCSID

    The code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used for conversion purposes

  - path-name-code-page

    A code page is explicitly specified for use. Any AS/400 code page is supported on this parameter. While using the AS/400 system as an NFS server and client in a multilingual environment, specify an EBCDIC code page for the path name on both the server and the client. The AS/400 system uses EBCDIC encoding so there is no need to convert to ASCII and then back to EBCDIC.

3.4.4 Examples for using the MOUNT Command

The MOUNT command is shown in the following display:

```
                    Add Mounted FS (MOUNT)

 Type choices, press Enter.

 Type of file system  . . . . > *NFS          *NFS, *UDFS, *NETWARE
 File system to mount . . . . > 'ASSYS01:/QSYS.LIB/DIR1.LIB'

 Directory to mount over. . . > '/DIVTEST'

 Mount options  . . . . .    'rw,suid,retry=5,rsize=8096,wsize=8096,timeo
=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


 Code page:
   Data file code page  . . . . .   *BINARY    1-32767,*ASCII,*JOBCCSID..
   Path name code page  . . . . .   *ASCII     1-32767,*ASCII,*JOBCCSID..




                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

The following examples show the MOUNT command using the various options:

- Mount command with default options:

  ```
  MOUNT TYPE(*NFS) MFS('ASSYS01:/QSYS.LIB/DIR1.LIB') MNTOVRDIR('/DIVTEST')
  ```

- Mount command with the following options: Read only, Hard mount, Number of retries=2, Timeout=1 second, No. of re-transmissions=2. All other options take their default values.

  ```
  MOUNT TYPE(*NFS) MFS('ASSYS01:/QSYS.LIB/DIR1.LIB') MNTOVRDIR('/DIVTEST')
  OPTIONS('ro,retry=2,timeo=10,retrans=2,hard')
  ```

- Mount command with the following options: Read Write, No attribute caching. All other options assume default values.

  ```
  MOUNT TYPE(*NFS) MFS('ASSYS01:/QSYS.LIB/DIR1.LIB') MNTOVRDIR('/DIVTEST')
  OPTIONS('rw,noac')
  ```

- Mount command with default options and using code page 273 for the path name.

  ```
  MOUNT TYPE(*NFS) MFS('ASSYS01:/QSYS.LIB/DIR1.LIB') MNTOVRDIR('/DIVTEST')
  CODEPAGE(*BINARY 273)
  ```

3.4.5 Prerequisites for using the MOUNT Command

- You must have *IOSYSCFG special authority to use this command.
- If you are mounting a user-defined file system or a Network File System, you need *R (read) authority to the file system being mounted.
- If you are mounting a NetWare file system, you need *X (execute) authority to the file system being mounted.
- You must have *W (write) authority to the directory being mounted over.

3.5 Using the /etc Files

This section explores the different files stored within the /etc directory. These files are used to keep different information used during the startup phase of NFS or during normal operation.

These files are platform independent; this means the same file names are used on AIX or other platforms as well. But some of them do not support all of the parameters that are used on other platforms. Instead, this information is stored in other places within OS/400.

3.5.1 Editing Files within the /etc Directory

The files stored within the */etc* directory are stream files. They can be edited in many different ways. We just mention four of them:

- Edit using the Edit File (EDTF) command.
- Edit using the Data File Utility (DFU).
- Edit using a PC based editor.
- Edit using a UNIX editor through NFS.

3.5.1.1 Editing Stream Files using Edit File (EDTF) Command

The Edit File (EDTF) command allows you to edit stream files or database files. By default, this command is not part of the base operating system. Instead, it is provided by a PTF. The PTF in V3R7 is SF38832. Refer to Appendix D, "Installation of the Edit File (EDTF) command" on page 161 for further information on how to install the PTF.

Once you have installed the PTF, you can edit the desired file by specifying the absolute or relative path name. For example:

Absolute Path Name

```
EDTF STMF('/etc/exports')
```

This command edits the stream file *exports* within the directory */etc*.

```
EDTF FILE(PGMLIB/ACCOUNT) MBR(ACC1)
```

This command edits the file member *ACC1* within the physical file PGMLIB/ACCOUNT.

Relative Path Name

```
EDTF STMF('exports')
```

This command edits the file *exports* within the current directory.

The difference between the absolute and the relative path name is the "/" at the beginning of the path name. The "/" as the first character means this is an absolute path name. Absolute path names start every time from the root directory. Relative path names start from the current directory.

The following EDTF display is shown.

```
 Edit File: /etc/exports
 Record . :    1 of  154 by  10                Column: 1 of 70  by 126
 Control  :

CMD....+....1....+....2....+....3....+....4....+....5....+....6....+..
    ******************* Beginning of data ********************
   ######################################################################
   #  /etc/exports                                                      #
   #                                                                    #
   #  This file contains the paths of the directory trees that you wish #
   #  to export to the rest of the world via the NFS/400 server.        #
   #  There are several restrictions on the format of the entries.      #
   #  They are:                                                         #
   #                                                                    #
   #  1. All entries are ended by the end-of-line character.            #
   #                                                                    #
   #  2. The continuation character '\' can be used to extend one       #
   #     entry across multiple lines when it is the last character      #
   #     in the line.                                                   #
   #                                                                    #
   #  3. All path names must be absolute, beginning at the root '/'.    #
   #                                                                    #

 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

All of the appropriate line commands are shown when you press the F1 (Help) Function Key.

3.5.1.2 Editing Stream Files using the Data File Utility

The second way to edit a file is using the Data File Utility (DFU). DFU does not support editing stream files; instead, you must perform a couple more steps to edit a stream file. Assume you want to edit the stream file *netgroup*. Use the following steps for editing the stream file */etc/netgroup*:

1. Create a temporary physical file:

   Because DFU does not support editing stream files, you need a physical file that stores the data from the stream file. Use the following command to create such a file:

   ```
   CRTPF FILE(QGPL/NETGROUP) RCDLEN(132)
   ```

2. Copy the stream file into the physical file:

   After the physical file is created, copy the stream file into the physical file using the following command:

   ```
   CPYFRMSTMF FROMSTMF('/etc/netgroup')
     TOMBR('/qsys.lib/qgpl.lib/netgroup.file/netgroup.mbr') MBROPT(*REPLACE)
   ```

3. Edit the physical file using DFU:

   Use the Work with Members Using PDM (WRKMBRPDM) command to list your physical file. Move the cursor in front of the file name *NETGROUP*. Use option 18 to start a DFU program for this file. The following display is shown:

```
                    Work with Members Using PDM                 ASSYS01

File  . . . . . .   NETGROUP
  Library . . . .     QGPL                Position to  . . . . .

Type options, press Enter.
 3=Copy    4=Delete         5=Display   7=Rename   8=Display description
 9=Save    13=Change text   18=Change using DFU    25=Find string ...

Opt  Member      Date        Text
18   NETGROUP    09/29/97    ____________________________________________




Parameters or command
===>                                                              Bottom
F3=Exit         F4=Prompt             F5=Refresh           F6=Create
F9=Retrieve     F10=Command entry     F23=More options     F24=More keys
```

After DFU has started, the following display is shown:

```
WORK WITH DATA IN A FILE                      Mode . . . . :   CHANGE
Format . . . . :   NETGROUP                   File . . . . :   NETGROUP

*RECNBR: _______









F3=Exit                F5=Refresh              F6=Select format
F9=Insert              F10=Entry               F11=Change
                                 (C) COPYRIGHT IBM CORP. 1980, 1996   +
```

Now scroll forward to display the file and perform your changes. Each record represents one row of the stream file. If you want to add a record, press the F10 Function Key. Refer to the online help text for further information on how to use DFU.

After you finish your desired changes, press the F3 Function Key to save your changes.

4. Copy the physical file back to the stream file:

   When all your changes are made and the physical file is saved, copy the physical file back to the stream file using the following command:

   ```
   CPYTOSTMF FROMMBR('/qsys.lib/qgpl.lib/netgroup.file/netgroup.mbr')
      TOSTMF('/etc/netgroup') STMFOPT(*REPLACE)
   ```

3.5.1.3 Editing Stream Files using a PC-Based Editor

The third method to edit the *netgroup* file is described briefly. If you want to use a PC-based editor, you need access to the */etc* directory on the AS/400 system. This is accomplished by using one of the following ways:

- Using Client Access/400
- Using the File Transfer Protocol (FTP)

If you are using Client Access/400, you have to have direct access to the */etc* directory. Therefore, your PC-based editor edits the file on the AS/400 system.

Using FTP to edit a stream file, the file first has to be transferred to the PC. Then use your editor of choice and perform your desired changes. Once you have finished your work, use FTP again to transfer the file back into the */etc* directory.

3.5.1.4 Editing Stream Files using a UNIX Editor through NFS

Another way to edit stream files on the AS/400 system is using a UNIX editor through NFS. In this case, you have to perform the following steps:

1. Export the */etc* directory on the NFS server using the EXPORTFS command.
2. Mount the */etc/* directory to your UNIX system using the MOUNT command.
3. Change into the */etc* directory using the Change Directory (CD) command.
4. Edit the desired file using the VI editor or any other editor of your choice.

3.5.2 Using the /etc/exports File

The *exports* file is a stream file used to store information about file systems or directories to be exported. It is located within the IFS directory */etc*. For each directory to be exported, you need an entry in the *exports* file.

You can specify all parameters from the Export File System (EXPORTFS) command within an entry in the *exports* file. Of course, they need to be specified in a predefined order.

To export the entries listed in the *exports* file, use the following command:

```
EXPORTFS OPTIONS('-A')
```

The following pages contain a few examples that show how to specify an entry in the *exports* file using the information from the EXPORTFS command.

Example 1:

Assume you use the EXPORTFS command to export a directory. The following display shows the appropriate parameter:

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-I -O RO,ROOT=RSSYS01'



 Directory  . . . . . > '/thomas'

                                                                    Bottom
F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
F13=How to use this display        F24=More keys
```

Specifying the previous parameter allows only read access to the exported directory. Client root users from the host *RSSYS01* are mapped to the corresponding user on the NFS server.

The appropriate entry within the *exports* file is shown in the following display:

```
 Edit File: exports
 Record . :   139 of   151 by  10                Column:  1 of 70 by 126
 Control  :

CMD....+....1....+....2....+....3....+....4....+....5....+....6....+....
   #  #HOSTOPT HostName=wheatley,         \                            #
   #  DataFileCodePage=367                                             #
   #                                                                   #
   #  A user exports /qsys.lib/jon.lib to the netgroup dept-OA5.  A    #
   #  codepage is specified for data files.  Because the directory     #
   #  exists in the QSYS.LIB file system, NFS will use the open() API  #
   #  with the O_TEXTDATA and O_CODEPAGE options to work with data     #
   #  files.  Note that wheatley will use these options only if it is  #
   #  defined as a member of the netgroup dept-OA5.                    #
   #                                                                   #
   #                                                                   #
   ####################################################################
   /thomas ro,root=rssys01
    ****************** End of data *************************




 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

Example 2:

Assume you use the EXPORTFS command to export a directory. The following display shows the appropriate parameter:

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-i -o anon=600'



 Directory  . . . . . > '/tools'









                                                                    more..
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

The following display shows the additional parameters specified on the EXPORTFS command.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.


                          Additional Parameters

Host options:
  Host name  . . . . . > ASSYS01     Character value, *DFT
  Data file code page  . . . . . > *BINARY     1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 037         1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC       *SYNC, *ASYNC

  Host name  . . . . . > RSSYS02     Character value
  Data file code page  . . . . . > *BINARY     1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 819         1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *ASYNC      *SYNC, *ASYNC
              + for more values

                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

The previous EXPORTFS command exports the directory */tools* using different settings for the two NFS clients. Anonymous access is mapped to the user profile that has the UID of 600 assigned to it.

The appropriate entry within the *exports* file is shown in the following display:

```
 Edit File: exports
 Record . :    14 of    23 by 509          Column:    1 of   70 by  70
 Control  :

CMD....+....1....+....2....+....3....+....4....+....5....+....6....+....
   #                                                                    #
   #                                                                    #
   #                                                                    #
   ######################################################################
   /tools  anon=600                                \
   #HOSTOPT HostName=assys01,                      \
            PathNameCodePage=037:                  \
            HostName=rssys02,                      \
            PathNameCodePage=819,                  \
            NoWaitForWrites
    ****************** End of data **************************




F2=Save F3=Save/Exit F10/11=Left/Right F12=Cancel F16=Find F17=Chg
F15=Copy
```

Example 3:

Assume you use the EXPORTFS command to export a directory. The following display shows the appropriate parameter:

```
                     Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-i -o rw=rssys01:assys87,anon=600,
   root=rssys01,access=rssys01:assys02:assys87'


 Directory  . . . . . > '/tools'









                                                                    more..
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

The following display shows the additional parameters specified on the EXPORTFS command.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.


                           Additional Parameters

Host options:
  Host name  . . . . . . . . . . > ASSYS02     Character value, *DFT
  Data file code page  . . . . . > *BINARY     1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 037         1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC       *SYNC, *ASYNC

  Host name  . . . . . . . . . . > ASSYS87     Character value
  Data file code page  . . . . . > *BINARY     1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 273         1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC       *SYNC, *ASYNC
              + for more values

                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

Within this example, the EXPORTFS command exports the directory *tools*. The following list shows the meaning of each parameter and export option:

rw=rssys01:assys87
: From all clients that mount the exported directory, just the users from the clients *rssys01 and assys87* have write access to the directory.

anon=600
: All anonymous client requests are mapped to the AS/400 NFS server user profile that has the UID of 600 assigned to it.

root=rssys01
: By default, users with UID 0 or users whose matching user profiles on the AS/400 server have the special authority *ALLOBJ are mapped to anonymous users. To map to the proper user profiles, you can use this option to name a remote system. In this example, the remote system named *rssys01* can have the proper mapping.

access=rssys01:assys02:assys87
: Only the clients mentioned in this parameter are allowed to mount the directory. All other client requests are denied.

The appropriate entry in the *exports* file is shown in the following display:

```
 Edit File: exports
 Record . :    14 of    23 by 509          Column:    1 of   70 by  70
 Control  :

CMD....+....1....+....2....+....3....+....4....+....5....+....6....+....
   #                                                                     #
   #                                                                     #
   #                                                                     #
   ######################################################################
   /tools rw=rssys01:assys87,anon=600,           \
   root=rssys01,access=rssys01:assys02:assys87   \
   #HOSTOPT HostName=assys02,                    \
            PathNameCodePage=037:                \
            HostName=assys87,                    \
            PathNameCodePage=273




F2=Save F3=Save/Exit F10/11=Left/Right F12=Cancel F16=Find F17=Chg
F15=Copy
```

As you see from the previous examples, the export options in the *exports* file are specified in the same way as on the EXPORTFS command. The main difference are the values specified in the HOSTOPT parameter. Usually all the text behind the "#" character is treated as a comment. The HOSTOPT parameter is the only exception. In this parameter, you may have more than one client defined. In this case, each client entry is separated by a colon (:). Values that are not changed within this parameter must not be specified in the *exports* file.

The "#" used to specify the HOSTOPT parameter is useful when sending the *exports* file to non-AS/400 systems. Then this file is still useable.

The following description shown the four possible values within the HOSTOPT parameter:

HostName
The name of the remote NFS client for which you are specifying additional export options.

PathNameCodePage
The path name code page is used for the path name components of the files exported to and mounted on the specified NFS client or netgroup.

DataFileCodePage
The data file code page is used for data of the files exported to and mounted on the specified NFS clients or netgroups.

NoWaitForWrites
This option specifies whether write requests are handled synchronously or asynchronously for the NFS client or netgroup name. The absence of this option means that data is written to disk immediately (synchronously).

For more information about the export parameter and options, refer to Section 3.3, "Exporting File Systems" on page 32. For a complete description of the

layout and the rules within the *exports* file, refer to Appendix A, “Layout and Rules of the /etc/exports File” on page 153.

3.5.3 Using the /etc/netgroup File

The EXPORTFS command has parameters that allows you to specify more than one client for a specific type of criteria. This means if you want to specify the *PathNameCodePage* of 819 for 20 clients on the EXPORTFS command, you do not have to type 20 entries with the same contents; instead, use a netgroup.

A netgroup consists of a netgroup name and the list of host names assigned to it. A host name can exist in more than one netgroup. A netgroup can also contain other netgroups. This means you can inherit other groups within one group.

Assume you want to export a directory and the access should be restricted to one department that has 12 clients installed. The following example shows the EXPORTFS command:

```
EXPORTFS OPTIONS('-i -o access=dpt1cl1:dpt1cl2:dpt1cl3:dpt1cl4:dpt1cl5:
dpt1cl6:dpt1cl7:dpt1cl8:dpt1cl9:dpt1cl10:dpt1cl11:dpt1cl12')
DIR('/dpt1pgms')
```

As you see, the parameter string is pretty long. Typing errors easily happen. Imagine you want to give write permission to just six clients on your list. In this case, you have to add the *RW* option and specify the six clients. The parameter string becomes longer and longer and may not fit into the command line. Therefore, it is more convenient to create groups that contain the appropriate clients.

In the following example, you see an efficient way of using netgroups. Assume you have Department 1 with 12 clients. Client 1 to Client 7 has write permission to a exported directory. Client 8 to Client 12 has read permission only. In addition, Client 3 and Client 6 have a UID for which the corresponding user profile on the AS/400 system has *ALLOBJ special authority.

Under normal circumstances, the EXPORTFS command is specified as:

```
EXPORTFS OPTIONS('-i -o access=dpt1cl1:dpt1cl2:dpt1cl3:dpt1cl4:dpt1cl5:
dpt1cl6:dpt1cl7:dpt1cl8:dpt1cl9:dpt1cl10:dpt1cl11:dpt1cl12,root=dpt1cl3:
dpt1cl6,rw=dpt1cl1:dpt1cl2:dpt1cl3:dpt1cl4:dpt1cl5:dpt1cl6:dpt1cl7')
DIR('/dpt1pgm')
```

In the following display, the appropriate netgroup setup is shown:

```
 Edit File: netgroup
 Record . :     56 of     70 by   9          Column:    1 of   71 by  74
 Control  :

CMD...+....1....+....2....+....3....+....4....+....5....+....6....+....
  #                                                                     #
  #  8. Imbedded hexadecimal characters are not allowed.  This         #
  #     includes characters below ASCII x020.                          #
  #                                                                     #
  #  9. All entries are NOT case-sensitive.                             #
  #                                                                     #
  #  10. Triplets (x,y,z) (a,b,c) may be separated by spaces            #
  #      or commas only.                                                #
  #                                                                     #
  ######################################################################
  dpt1rt   (dpt1cl3,,),(dpt1cl6,,)
  dpt1rw   (dpt1rt,,),(dpt1cl1,,),(dpt1cl2,,),(dpt1cl4,,),(dpt1cl5,,),  \
  (dpt1cl7,,)
  dpt1all  (dpt1rw,,),(dpt1cl8,,),(dpt1cl9,,),(dpt1cl10,,),(dpt1cl11,,),\
  (dpt1cl12,,)
 ****************** End of data *************************

F2=Save F3=Save/Exit F10/11=Left/Right F12=Cancel F16=Find F17=Chg
F15=Copy
```

After the three groups have been created, the following EXPORTFS command gives the same access and permissions to the exported directory as the command used without netgroups.

```
EXPORTFS OPTIONS('-i -o access=dpt1all,root=dpt1rt,rw=dpt1rw')
DIR('/dpt1pgm')
```

As you see from the previous example, it is more efficient and it saves more typing work to define netgroups.

Attention

All host names specified within the *netgroup* file have to exist within the AS/400 system host table or within a Domain Name Server (DNS). Otherwise, the EXPORTFS command will fail. If you do not know the proper name, use IP addresses instead.

For a complete description of the rules and the structure of the *netgroup* file, refer to Appendix B, “Layout and Rules of the /etc/netgroup File” on page 157.

3.5.4 Other /etc Files

There are two more stream files within the */etc* directory. Both of them should never be edited by the user. The files are:

/etc/pmap File

The *pmap* file is used by the RPC binder (port mapper) daemon. Information for all daemons is stored in this file, which is used to register daemons upon recovery of an RPC binder daemon crash.

/etc/statd File

This file is used by the server Network Status Monitor (NSM) daemon. Information about any clients that hold byte-range locks on open server

files are stored in this file. When a user starts the NSM daemon after a crash, the NSM daemon contacts all the clients listed in this file so that the clients can release any locks they hold on the server.

3.6 Scenarios using the EXPORTFS and MOUNT Command

Exporting and mounting file systems require various definitions to be made. Within this section, there are several scenarios that describe the export function and the appropriate mount operation.

In every scenario, there is an explanation about the parameters that are used and the impact of them.

3.6.1 Scenario 1 - Using One AS/400 NFS Client

Within this scenario, there is an AS/400 NFS server that exports a subdirectory of the “Root” file system. The appropriate NFS client is also an AS/400 system. Assume that the data within the exported subdirectory is sensitive and you want to ensure that only known users have access to this data.

Figure 10 shows the environment used for this scenario.

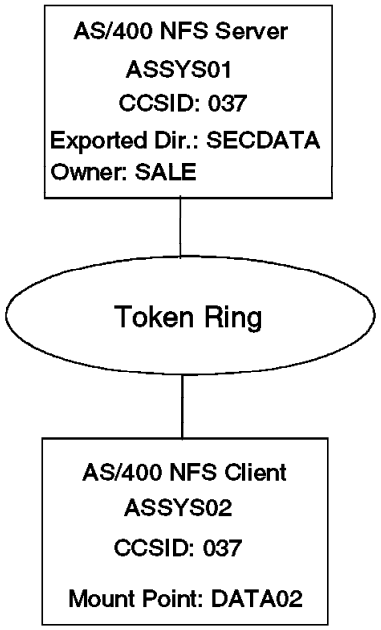


Figure 10. Scenario 1 - AS/400 NFS Server and AS/400 NFS Client

The information about the existing user profiles on the NFS Server as well as the NFS Client is listed in the following table.

Table 1 (Page 1 of 2). User Profile Information for Scenario 1

Existing User Profiles on the AS/400 NFS Server

Table 1 (Page 2 of 2). User Profile Information for Scenario 1		
User Profile	**UID**	**Spec. Authority**
SALE	500	*NONE
LISA	650	*NONE
SAM	960	*ALLOBJ
JOHN	330	*NONE
Existing User Profiles on the AS/400 NFS Client		
User Profile	**UID**	
LISA	650	
SAM	960	
TOM	730	
MARY	441	

The EXPORTFS command used to export the directory *secdata* is shown on the following display.

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-i -o access=assys02,anon=-1,root=assys02'


 Directory  . . . . . > '/secdata'









                                                                      Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
 F13=How to use this display        F24=More keys
```

The following display shows the appropriate MOUNT command to mount the exported directory to the mount point *data02*

```
                    Add Mounted FS (MOUNT)

 Type choices, press Enter.

 Type of file system  . . . . . > *NFS       *NFS, *UDFS, *NETWARE
 File system to mount . . . . . > 'assys01:/secdata'

 Directory to mount over  . . . . > '/data02'

 Mount options  . . . . . .   'rw,suid,retry=5,rsize=8096,wsize=8096,timeo
=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


 Code page:
   Data file code page  . . . . . > *BINARY    1-32767, *ASCII, *JOBCCSID
   Path name code page  . . . . . > *ASCII     1-32767, *ASCII, *JOBCCSID




                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

The directory *secdata* from the AS/400 NFS server is mounted over the mount point *data02* on the AS/400 NFS client. User *LISA* and *SAM* have access to the exported directory even though the corresponding user profile of user *SAM* on the NFS server has **ALLOBJ* special authority. This is because the *root=assys02* export option is specified on the EXPORTFS command. Users *TOM* and *MARY* have no access to the directory because there is no user profile on the NFS server that has a matching UID and anonymous access is not allowed (*anon=-1* option). Since the *access=assys02* export option is specified, all requests from clients other than the *assys02* client are rejected.

There is no need to specify any language parameter because both systems (the NFS client and the NFS server) use the same CCSID.

The *type of file system*, the *file system to mount*, and the *directory to mount over* parameters are specified within the MOUNT command. All other parameters contain their default values.

3.6.2 Scenario 2 - Using an AS/400 System and an AIX NFS Client

Within this scenario, there is an AS/400 NFS server that exports a subdirectory of the “Root” file system. There are two NFS clients - one AS/400 client and one AIX client. Assume that only ordinary users should have access to the exported directory. You, as the owner, allow both clients to mount the directory with “write” permission, but the administrator from client *assys02* wants to give only “read” access for its users.

Figure 11 on page 65 shows the environment used for this scenario.

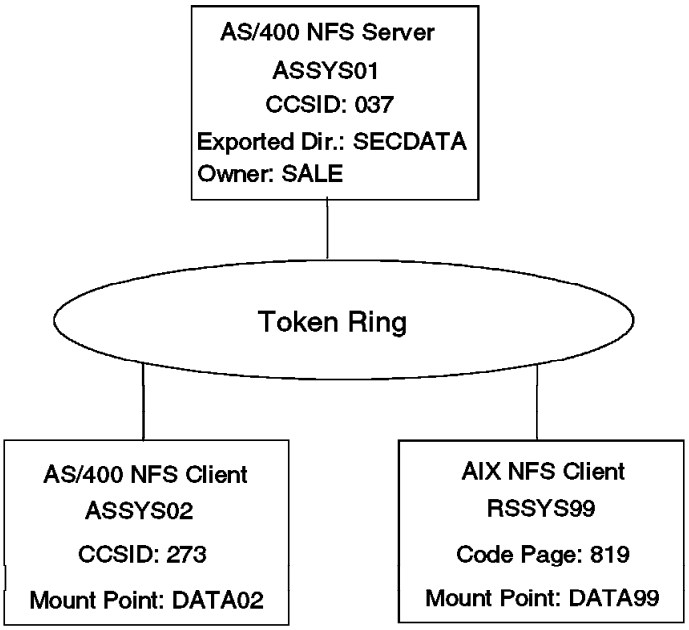


Figure 11. Scenario 2 - AS/400 NFS Server and an AS/400 and AIX NFS Client

The information about the existing user profiles on the NFS server as well as the NFS clients is listed in the following table.

Table 2. User Profile Information for Scenario 2

Existing User Profiles on the AS/400 NFS Server		
User Profile	**UID**	**Spec. Authority**
SALE	500	*NONE
MARION	650	*NONE
THOMAS	111	*ALLOBJ
JOHN	330	*NONE
PAT	551	*NONE
Existing User Profiles on the AS/400 NFS Client		
User Profile	**UID**	
MARION	650	
THOMAS	111	
DAN	788	
MARY	441	
Existing Users on the AIX NFS Client		
User Name	**UID**	
JOHN	330	
PAT	551	
BILL	1445	
STEVE	6511	

The EXPORTFS command used to export the directory *secdata* is shown in the following display.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . . . > '-i -o access=assys02:rssys99'



Directory  . . . . . . . . . . . > '/secdata'

                                                                      more..
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display        F24=More keys
```

The language dependent parameter is shown on the second display of the EXPORTFS command.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.


                          Additional Parameters

Host options:
  Host name  . . . . . . . . . . > ASSYS02    Character value, *DFT
  Data file code page  . . . . . > 273        1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 273        1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC      *SYNC, *ASYNC

  Host name  . . . . . . . . . . > RSSYS99    Character value
  Data file code page  . . . . . > 819        1-32767, *BINARY, *ASCII
  Path name code page  . . . . . > 819        1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC      *SYNC, *ASYNC
              + for more values

                                                                      Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

The following display shows the appropriate MOUNT command to mount the exported directory to the mount point *data02* on the AS/400 client.

```
                    Add Mounted FS (MOUNT)

 Type choices, press Enter.

 Type of file system  . . . . . > *NFS       *NFS, *UDFS, *NETWARE
 File system to mount . . . . . > 'assys01:/secdata'

 Directory to mount over  . . . . > '/data02'

 Mount options  . . . . . > 'ro,suid,retry=5,rsize=8096,wsize=8096,timeo
=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


 Code page:
   Data file code page  . . . . . > 273        1-32767, *ASCII, *JOBCCSID
   Path name code page  . . . . . > 273        1-32767, *ASCII, *JOBCCSID




                                                                      Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

The MOUNT command that is specified on the AIX NFS client is shown in the following display.

```
#
#
#
#
#
# mount assys01:/secdata /data99
# cd /data99
# ls
data1
# ls -l
total 1
-rwx---r-x   1 901      system       725 Oct 02 02:46 data1
#
```

- Mounting on the AS/400 NFS client:

  There is no *anon=* export option defined; therefore, all client users have access to the exported directory. Client user THOMAS has **ALLOBJ* special authority on the NFS server but is also able to access the directory because the user is mapped to the default anonymous user (QNFSANON). Since the administrator has specified the *ro* mount option, there are no write operations allowed, even though the directory is exported with write permission for the clients. (If nothing is specified on the export options, write access is allowed.)

Since the AS/400 NFS client system uses a code page different than the NFS server, the proper code pages for data and path names have to be specified on the EXPORTFS command and the MOUNT command.

- Mounting on the AIX NFS client:

  All users from the AIX NFS client have write access to the exported directory. Even if some users with their corresponding UIDs do not exist on the NFS server, they are allowed to access the directory. These users are mapped to the default anonymous user on the NFS server.

  Within AIX, there is no possibility to specify any language or code page related parameter during the mount process. Therefore, the proper code page is specified within the *HOSTOPT* parameter on the EXPORTFS command.

The *data file code page* is also defined because it is assumed that there is data within the exported directory. Leave the default value of **BINARY* in the *data file code page* parameter in two situations:

1. When using the same code pages on both the NFS server and the NFS client
2. When programs or other data are stored within the directory that should not be converted

3.6.3 Scenario 3 - Using AS/400 and AIX NFS Clients with Netgroups

This scenario covers a more complex environment using the following key functions of NFS:

- Code page support
- Various export flags with its options
- The /etc/exports file
- The /etc/netgroup file

Assume that a directory *pgms* is exported on the AS/400 NFS server. There are different NFS clients but some of them use the same settings and, therefore, can be grouped together. The following groups apply:

AS4GRP The clients *ASSYS02* up to *ASSYS12* belong to this group.

AIXGRP1 The clients *RSSYS01* up to *RSSYS15* belong to this group.

AIXGRP2 There are two AIX NFS clients that are put together into one group. These are the *RSSYS75* and the *RSSYS99* clients.

NFSCL Contains all NFS clients.

Only the clients shown in Figure 12 on page 69 are allowed to mount the exported directory. Anonymous client requests are mapped to the *PGMDFT* user profile. *Root* access is only granted to the clients belonging to the *AIXGRP2* group. We also take the different code pages into account. All of the AIX NFS clients have only read permission to the exported directory.

Figure 12 on page 69 shows the environment used for this scenario.

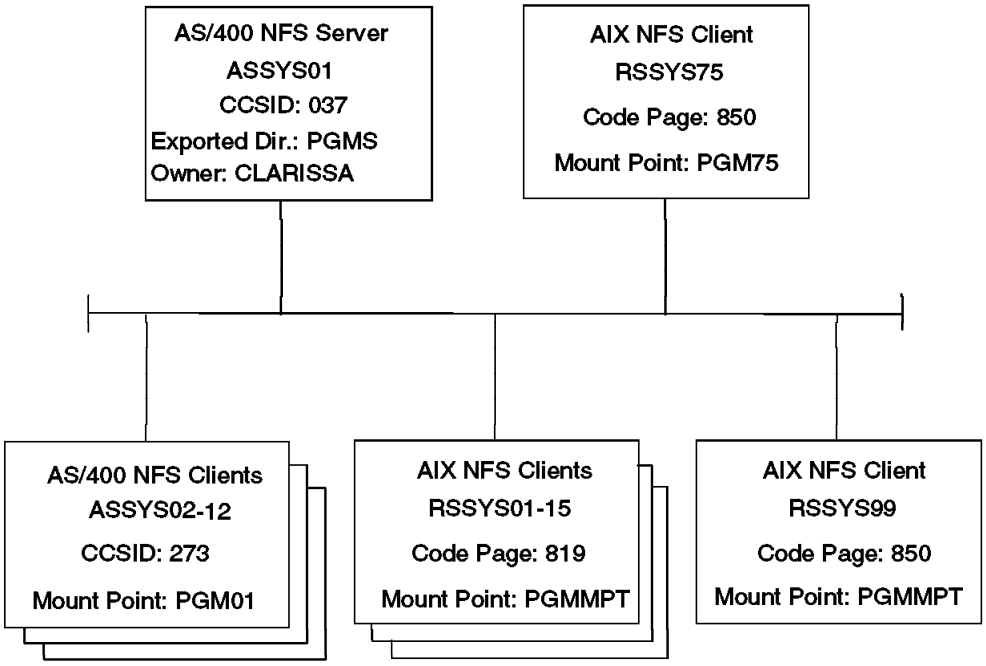


Figure 12. Scenario 3 - AS/400 NFS Server and AS/400 System and AIX NFS Clients

The information about the existing user profiles on the NFS Server as well as the NFS Clients is listed in the following table.

Table 3 (Page 1 of 2). User Profile Information for Scenario 3

Existing User Profiles on the AS/400 NFS Server		
User Profile	**UID**	**Spec. Authority**
CLARISSA	521	*NONE
MARION	650	*NONE
THOMAS	111	*ALLOBJ
RICARDA	335	*ALLOBJ
RAY	555	*ALLOBJ
KIM	5411	*NONE
PGMDFT	777	*NONE
Existing User Profiles on the AS/400 NFS Client *ASSYS09*		
User Profile	**UID**	
MARION	650	
THOMAS	111	
DAN	788	
Existing User Profiles on the AS/400 NFS Client *ASSYS03*		
User Profile	**UID**	
CLARISSA	521	
RICARDA	135	
JIM	78887	

Table 3 (Page 2 of 2). User Profile Information for Scenario 3	
Existing Users on the AIX NFS Client *RSSYS75*	
User Name	**UID**
RICARDA	335
PAT	551
Existing Users on the AIX NFS Client *RSSYS12*	
User Name	**UID**
RAY	555
HIRO	6323
Note: All other NFS clients have user profiles as well, but they are not relevant to this scenario.	

The following display shows the group definitions within the */etc/netgroup/* file.

```
 Edit File: netgroup
 Record . :   65 of   74 by  10                Column:  1 of  70  by 126
 Control  :

CMD ....+....1....+....2....+....3....+....4....+....5....+....6....+....
    ####################################################################
    as4grp   (assys02,,),(assys03,,),(assys04,,),(assys05,,),      \
    (assys06,,),(assys07,,),(assys08,,),(assys09,,),(assys10,,),   \
    (assys11,,),(assys12,,)
    aixgrp1  (rssys01,,),(rssys02,,),(rssys03,,),(rssys04,,),      \
    (rssys05,,),(rssys06,,),(rssys07,,),(rssys08,,),(rssys09,,),   \
    (rssys10,,),(rssys11,,),(rssys12,,),(rssys13,,),(rssys14,,),   \
    (rssys15,,)
    aixgrp2  (rssys75,,),(rssys99,,)
    nfscl    (as4grp,,),(aixgrp1,,),(aixgrp2,,)
     ****************** End of data **************************









 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

The netgroup definitions are shown on the previous display. As you can see, the *nfscl* netgroup contains netgroup names rather than client names. In this case, the three major client groups are defined within this netgroup.

Note: Be careful when you create netgroups. All client names have to exist in either the host table within the CFGTCP menu or within a Domain Name Server (DNS); otherwise, your export operation fails, even if you do not use a netgroup name on your EXPORTFS command.

All export definitions are made within the */etc/exports* file as shown in the following display:

```
 Edit File: exports
 Record . :  140 of   157 by  10                Column:  1 of  70 by 126
 Control  :

CMD ....+....1....+....2....+....3....+....4....+....5....+....6....+....
    #  A user exports /qsys.lib/jon.lib to the netgroup dept-OA5.  A
    #  codepage is specified for data files.  Because the directory
    #  exists in the QSYS.LIB file system, NFS will use the open() API
    #  with the O_TEXTDATA and O_CODEPAGE options to work with data
    #  files.  Note that wheatley will use these options only if it is
    #  defined as a member of the netgroup dept-OA5.
    #
    #
    ###################################################################
    /pgms  access=nfscl,rw=as4grp,root=aixgrp2,anon=777  \
    #HOSTOPT HostName=as4grp,                            \
             PathNameCodePage=273:                      \
             HostName=aixgrp1,                          \
             PathNameCodePage=819:                      \
             HostName=aixgrp2,                          \
             PathNameCodePage=850
     ****************** End of data *************************

 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

The previous display shows the export definition made within the */etc/exports* file. The following settings apply:

/pgms This is the directory to be exported.

access=nfscl
: Only clients specified on this export option are allowed to mount the exported directory. In this case, the value represents a netgroup. The netgroup *nfscl* contains all NFS clients mentioned in Figure 12 on page 69.

rw=as4grp
: Only clients specified on the *rw* export option have the permission to perform write operations within the exported directory. In this case, all AS/400 NFS clients have permission.

root=aixgrp2
: Client requests from ROOT users or client users for which the corresponding user profile with the matching UID on the NFS server has the **ALLOBJ* special authority are mapped to the proper user on the NFS server. All other clients that meet this criteria are mapped to the anonymous user.

anon=777
: All anonymous client requests; this means that requests where the client UID does not exist on the NFS server are mapped to the user profile to which this UID is assigned. For further information about anonymous access, refer to Chapter 4, “NFS Security” on page 77.

HostName and PathNameCodePage
: For every client group, there is an entry that contains the code page used for path names. The *DataFileCodePage* parameter is not defined because programs are stored within the exported directory. This means for programs, you need no conversion.

The EXPORTFS command used to export the directory *pgms* is shown on the following display.

```
                    Change NFS Export (EXPORTFS)

 Type choices, press Enter.

 NFS export options . . . . . > '-a'



 Directory  . . . . . >

                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
 F13=How to use this display        F24=More keys
```

Mounting the directory on the AS/400 NFS clients

The following display shows the MOUNT command used on any AS/400 NFS client within the *as4grp* netgroup.

```
                    Add Mounted FS (MOUNT)

 Type choices, press Enter.

 Type of file system  . . . . . > *NFS       *NFS, *UDFS, *NETWARE
 File system to mount . . . . . > 'assys01:/pgms'

 Directory to mount over  . . . . > '/pgm01'

 Mount options  . . . . . .  'rw,suid,retry=5,rsize=8096,wsize=8096,timeo
=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


 Code page:
   Data file code page  . . . . .   *BINARY    1-32767, *ASCII, *JOBCCSID
   Path name code page  . . . . .   273        1-32767, *ASCII, *JOBCCSID


                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

Mounting the directory on the AIX NFS clients

The following example shows the MOUNT command used on AIX NFS clients:

```
mount assys01:/pgms  /pgmmpt
```

For the AIX client *RSSYS75*:

```
mount assys01:/pgms  /pgm75
```

How users from different client are treated on the NFS server

- Users on the *ASSYS09* NFS client:

  Client user MARION is mapped to the corresponding user on the server because the UID exists on both the server and the client.

  Client user THOMAS is mapped to the anonymous user PGMDFT because Root access is not allowed for all AS/400 clients and user THOMAS has **ALLOBJ* special authority on the server.

  Client user DAN is also mapped to the PGMDFT user because its client UID does not exist on the server.

- Users on the *ASSYS03* NFS client:

  Client user CLARISSA is mapped to the corresponding user on the server because the UID exists on both the server and the client.

  Client user RICARDA is mapped to the anonymous user PGMDFT even if the user profile RICARDA exists on both the server and the client but the UIDs of both user profiles do not match.

  Client user JIM is also mapped to the PGMDFT user because its client UID does not exist on the server.

- Users on the *RSSYS75* NFS client:

  Client user RICARDA is mapped to the corresponding user on the server even if the corresponding user profile on the server has **ALLOBJ* special authority but in this case, Root access is allowed for this client.

  Client user PAT is mapped to the PGMDFT user because its client UID does not exist on the server.

- Users on the *RSSYS12* NFS client:

  Client user RAY is mapped to the anonymous user PGMDFT because Root access is not allowed for this client and user RAY has **ALLOBJ* special authority on the server.

  Client user HIRO is mapped to the PGMDFT user because its client UID does not exist on the server.

3.7 Locks and Recovery

Clients can introduce locks onto mounted server file systems. Locks give a user shared or exclusive access to a file or part of a file. When a user locks a file, any process requiring access to the file must wait for the lock to be released before processing can continue.

There are two kinds of locks that clients can establish on all or part of a file: exclusive and shared. When a client obtains an exclusive lock, no other processes can obtain a lock on that portion of the file. When a client obtains a shared lock, other processes can obtain a shared lock to the same portion of the file.

3.7.1 Importance of Locking

A user or application can use byte-range locks through NFS to improve NFS performance. Because the NFS protocol is stateless, a given client may not be aware of changes made on the server (due to client caches). Locking ensures that this problem does not occur during critical times when the server updates files.

3.7.2 Locking through NFS

An application at the client controlled by the user can start a lock request against a remote file that is mounted with NFS. The client then sends the operation to its local network lock manager Daemon through an RPC request. The client-side NLM daemon forwards the request to the corresponding server-side NLM through RPC.

Users can call the Perform File Control Command (fcntl()) API to lock parts of files and specify lock parameters. A job can use fcntl() to lock out other jobs from a part of a file. By locking out other jobs, the job can read or write to that part of the file without interference from others. File locking can ensure data integrity when several jobs have a file accessed concurrently. The syntax for fcntl() is given:

```
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
int fcntl(int fildes, int cmd, arg);
```

The various parameters are explained:

fildes The descriptor on which the control command is to be performed.

cmd The command to be performed.

arg This depends on the cmd parameter. For file locking the relevant values are:

- **_GETLK** Obtains locking information for a file. fcntl() returns 0 if it successfully obtains the locking information.
- **F_SETLK** Sets or clears a file segment lock. fcntl() returns 0 if it successfully clears the lock.
- **F_SETLKW** Sets or clears a file segment lock; however, if a shared or exclusive lock is blocked by other locks, fcntl() waits until the request can be satisfied.

Once you use the fcntl() API with the Set Lock command, the server NLM daemon then performs the lock operation against the corresponding file. The server NLM daemon generates a response that the client NLM daemon receives. The client NLM daemon generates a response for the NFS client with the results of the lock request. The NFS client then returns the result to the user through the application.

3.7.3 Recovery after Abnormal System Failure

The statelessness of the Network File System does not integrate well with the statefulness of file locks. Problems can occur in releasing locks if either the client or server fails while their respective daemons hold a lock or locks. If a client with a granted lock request should happen to fail, a specific set of operations occur at startup time to recover the locks:

1. When the user restarts the NSM daemon on a system, the daemon sends a change of state RPC to other NSM daemons in the network. This message is transmitted only to the other NSM daemons that the failed system is aware of in the network.
2. After receiving a change of state from a remote system, the NSM uses RPC to communicate with the local NLM. The NSM informs the NLM that a lock was held at the time of system failure.
3. If the failed system is a server, then after recovery, the local server NLM attempts to re-establish file locks that client processes might have held. The information about which remote clients held locks on the server is stored in the local file /etc/statd. Users should not edit or change this file in any way. Improperly changing this file may affect recovery from a server failure.

When the local system is a server and the client never restarts, an infinite backlog of requests builds up for the locked file. This file is never released from its lock. This can be averted with the Release Integrated File System Locks (RLSIFSLCK) command.

3.7.3.1 Release IFS Locks (RLSIFSLCK) command

The Release Integrated File System Locks (RLSIFSLCK) command can be used to release all Network File System (NFS) byte-range locks. These locks might be held by a specified NFS client or on a particular specified object. This command should only be used to free resources that cannot be freed using normal means. When locks are released with this command, a successful return code is sent to the locking application.

The various parameters on this command are:

- Remote location: This parameter specifies the host name or Internet address of a remote system whose NFS-related locks on local files are to be released.
- Object: This parameter specifies the path name of an object on which all byte-range locks are to be released. This releases all locks on that object, regardless of the type of lock or the type of process that holds it.

3.7.3.2 Examples of RLSIFSLCK Command

The following display shows the RLSIFSLCK command:

```
                Release File System Locks (RLSIFSLCK)

Type choices, press Enter.

Remote location  . . . . . .   assys02



Object . . . . . .



                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

This command releases all locks held by the remote system ASSYS02. The following display gives the command to release all byte-range locks held on the object /divtest/file1/file2.

```
                Release File System Locks (RLSIFSLCK)

Type choices, press Enter.

Remote location  . . . . . .

Object . . . . . .  /divtest/file1/file2.









                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

Chapter 4. NFS Security

This chapter discusses the security concepts within the Network File System (NFS) and how the NFS security integrates with OS/400 security. After a brief overview of NFS security, this chapter goes on to explain the security checking algorithm that is used with the NFS. It then discusses the special security considerations of the NFS that deal mainly with the user and group identifications and public authority. This chapter also gives you a scenario to get a better understanding of how the NFS security works. It is not the intent of this publication to explain all the parameters in commands used in the examples in this chapter. For additional information on these commands, refer to the following publications.

- *OS/400 Security APIs V3R7*, SC41-4872
- *Integrated File System Introduction*, SC41-4711
- *OS/400 Network File System Support*, SC41-4714
- *AS/400 CL Reference*, SC41-4722
- *OS/400 Security - Reference V3R7*, SC41-4302
- *AIX Version 4 Commands Reference*, SBOF-1851
- *OS/400 UNIX-Type APIs V3R7*, SC41-4875

4.1 Overview

NFS is a protocol that uses user identifications (UIDs), group identifications (GIDs), and public authority to perform authority checking between clients and servers. UIDs are made up of a number that represents a user on a particular system. The UID does not include a user's name or any other information about the user at all. UIDs are simply a way for the system to recognize users and activate their user profile for use with authorization checking. The GID operates in the same fashion as a UID as a way of identifying the user's access on a given system. It acts as a general authority for a group of users or machines. GIDs are not relative to UIDs and do not correspond with UIDs. A user profile may or may not have a GID associated with it.

4.2 UIDs, GIDs, and Permissions

This section describes where to find the UIDs and GIDs within OS/400. It also tells you how to change the access permissions to objects.

4.2.1 User Identification (UID)

UIDs are specified in user profiles. Each UID has to be unique. We have a few IBM supplied user profiles that have pre-assigned UIDs. For example, the QSECOFR user profile has a UID of 0; the QSYS user profile has a UID of 101 .

4.2.1.1 Automatic Generation of UIDs

Creating a user profile automatically assigns a unique UID to the new user profile by default as shown in the following display.

```
                    Create User Profile (CRTUSRPRF)

Type choices, press Enter.

User profile . . . . . . . . . . USRPRF       > NEWUSER
User password  . . . . . . . . . PASSWORD       *NONE
Set password to expired  . . . . PWDEXP         *NO
Status . . . . . . . . . . . . . STATUS         *ENABLED
User class . . . . . . . . . . . USRCLS         *USER
Assistance level . . . . . . . . ASTLVL         *SYSVAL
Current library  . . . . . . . . CURLIB         *CRTDFT
Initial program to call  . . . . INLPGM         *NONE
  Library  . . . . . . . . . . .
Initial menu . . . . . . . . . . INLMNU         MAIN
  Library  . . . . . . . . . . .                  *LIBL
Limit capabilities . . . . . . . LMTCPB         *NO
Text 'description' . . . . . . . TEXT         > 'Network User 1'



                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display     F24=More keys
```

Use the Create User Profile (CRTUSRPRF) command.
Use Function key F4 to Prompt the user.

```
                    Create User Profile (CRTUSRPRF)

Type choices, press Enter.

Sort sequence  . . . . . . . . . SRTSEQ         *SYSVAL
  Library  . . . . . . . . . . .
Language ID  . . . . . . . . . . LANGID         *SYSVAL
Country ID . . . . . . . . . . . CNTRYID        *SYSVAL
Coded character set ID . . . . . CCSID          *SYSVAL
Locale job attributes  . . . . . SETJOBATR      *SYSVAL
                          + for more values
Locale . . . . . . . . . . . . . LOCALE         *SYSVAL

User options . . . . . . . . . . USROPT         *NONE
                          + for more values
User ID number . . . . . . . . . UID          > *GEN
Group ID number  . . . . . . . . GID            *NONE
Home directory . . . . . . . . . HOMEDIR        *USRPRF

Authority  . . . . . . . . . . . AUT            *EXCLUDE

                                                                    Bottom
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
```

Use Function Key F10 to get the Additional Parameters List. Scroll down to the last display. The value *GEN for the keyword UID automatically generates the UID for the user profile.

4.2.1.2 Manually Assigning UIDs

You can also assign your own UID when creating or changing a user profile. While assigning the UID, you have to ensure that the UID has not already been assigned to another user profile. The following displays show how to assign a UID manually.

```
                    Create User Profile (CRTUSRPRF)

 Type choices, press Enter.

 User profile . . . . . . . . . . USRPRF       > NEWUSER
 User password  . . . . . . . . . PASSWORD       *NONE
 Set password to expired  . . . . PWDEXP         *NO
 Status . . . . . . . . . . . . . STATUS         *ENABLED
 User class . . . . . . . . . . . USRCLS         *USER
 Assistance level . . . . . . . . ASTLVL         *SYSVAL
 Current library  . . . . . . . . CURLIB         *CRTDFT
 Initial program to call  . . . . INLPGM         *NONE
   Library  . . . . . . . . . . .
 Initial menu . . . . . . . . . . INLMNU         MAIN
   Library  . . . . . . . . . . .                  *LIBL
 Limit capabilities . . . . . . . LMTCPB         *NO
 Text 'description' . . . . . . . TEXT         > 'Network User 1'



                                                                      Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
 F13=How to use this display     F24=More keys
```

Use the Create User Profile (CRTUSRPRF) command.
Use Function key F4 to Prompt the user.

```
                    Create User Profile (CRTUSRPRF)

 Type choices, press Enter.

 Sort sequence  . . . . . . . . . SRTSEQ         *SYSVAL
   Library  . . . . . . . . . . .
 Language ID  . . . . . . . . . . LANGID         *SYSVAL
 Country ID . . . . . . . . . . . CNTRYID        *SYSVAL
 Coded character set ID . . . . . CCSID          *SYSVAL
 Locale job attributes  . . . . . SETJOBATR      *SYSVAL
                           + for more values
 Locale . . . . . . . . . . . . . LOCALE         *SYSVAL

 User options . . . . . . . . . . USROPT         *NONE
                           + for more values
 User ID number . . . . . . . . . UID          > 304
 Group ID number  . . . . . . . . GID            *NONE
 Home directory . . . . . . . . . HOMEDIR        *USRPRF

 Authority  . . . . . . . . . . . AUT            *EXCLUDE

                                                                      Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

Use Function Key F10 to get the Additional Parameters List. Scroll down to the last display. The value specified in the UID keyword (304) is assigned to the user profile.

Assigning a UID that has already been assigned to another user profile gives you the following error message.

```
                    Additional Message Information

Message ID . . . . . : CPF22CE    Severity . . . . . : 40
Message type . . . . . : Diagnostic
Date sent  . . . . . : 09/05/97   Time sent  . . . . . : 13:29:50

Message . . . . : The UID value 304 is used by another user profile.
Cause . . . . . : The specified user profile was not created or changed
 because the UID value specified was previously assigned to another user
 profile.
Recovery  . . . : Change the UID value and try the request again.









                                                                    Bottom
Press Enter to continue.

F3=Exit  F6=Print  F9=Display message details
F10=Display messages in job log  F12=Cancel  F21=Select assistance level
```

4.2.2 Group Identification (GID)

GIDs are defined in a totally different way on AS/400 systems as compared to UNIX or AIX systems. Within the AIX environment, you can distinguish between users and groups. This means, you have different ways to define a user or a group. A UID is assigned to a user and a GID is assigned to a group.

On the AS/400 system, there is no ability to create groups. Instead, groups are also user profiles, but with an assigned GID. By default, a new user profile has no GID assigned to it. If you want to define a group, you have to assign a GID to a user profile. At this moment, the user profile becomes a group profile. The GID has to be unique. This means you cannot specify the same GID as a GID on another user profile. To make a user profile a member of a group, you have to specify the name of the group user profile in the group profile (GRPPRF) parameter within the member user profile. Figure 13 on page 81 shows the relationship between the group user profile and the member user profile parameter.

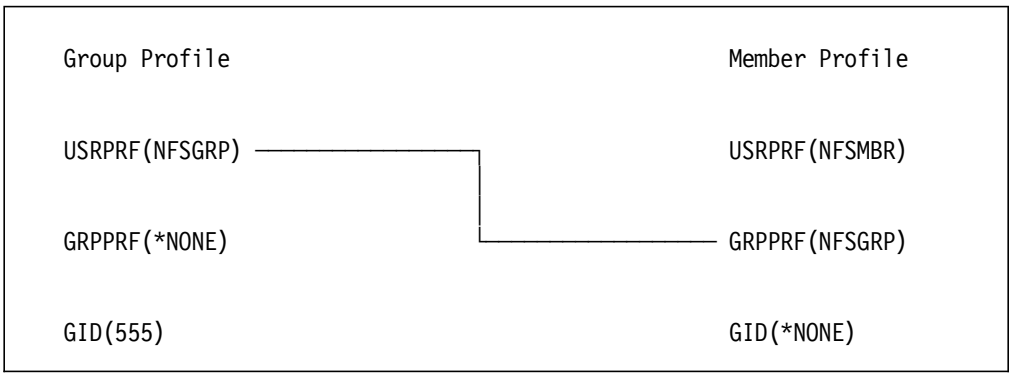


Figure 13. Group/Member Parameter. This figure shows the relationship between group and member profile parameters.

There are also supplemental GIDs that can act as a third, fourth, or nth method of access authority to an object. GIDs are assigned to users based on which groups the user belongs to. AIX servers and clients can use supplemental GIDs to determine the authority of a user. The Network File System on AS/400 systems ignores supplemental GIDs, focusing only on the UID and GID of a user. Another difference compared to UNIX and AIX is that the current AS/400 implementation of GID does not allow to specify GID value of 0. This leads to the security checking implications. GRPPRF(*NONE) has a special meaning to the security checking. The AS/400 system deals with GRPPRF(*NONE) differently depending on which side (server or client) the AS/400 system is. For the AS/400 system as a server, objects created by users with these values (that is, GRPPRF(*NONE)) appear as if they have a primary group owner of GID of 0. For the AS/400 system as a client, a user with GRPPRF(*NONE) is not a member of any groups. Then, this user is ignored by any servers for GID permission, even to the objects with GRPPRF(*NONE) owner because this user does not have GID of 0, and the objects have the primary group of GID of 0.

4.2.3 Changing UIDs or GIDs for User Profiles Owning Objects

When you change the UID or GID of an existing user profile, you can run into problems if the user profile owns objects within the Integrated File System (IFS) except the QSYS.LIB and QDLS file system. The error message you get is shown in Figure 14 on page 82. You do not have to change the ownership of all objects owned by the user profile. There is a security application programming interface (API) available for changing the UIDs or GIDs of user profiles that own objects within the IFS. The name of the API is QSYCHGID API.

```
                    Additional Message Information

Message ID . . . . . : CPF22DC     Severity . . . . . : 40
Message type . . . . . : Diagnostic
Date sent  . . . . . : 09/08/97    Time sent  . . . . . : 10:54:27

Message . . . . :   Not allowed to change UID of the user profile.
Cause . . . . . :   The UID of a user profile may not be changed when
  the profile is the owner of an object in a directory.
Recovery  . . . :   Change the value of the UID parameter to *SAME and
  try the command again.









                                                                  Bottom
Press Enter to continue.

F3=Exit  F6=Print  F9=Display message details
F10=Display messages in job log  F12=Cancel  F21=Select assistance level
```

Figure 14. Error Changing UID when User Owns Objects. This figure shows that the error occurs when changing a UID and the user profile owns objects.

While using the QSYCHGID API, specify the parameters in the correct syntax. The API has four parameters.

User Profile Specify the user profile as a 10-character field.

UID Specify the UID as an 8-digit hexadecimal number. The special value XFFFFFFFF keeps the UID unchanged.

GID Specify the GID as an 8-digit hexadecimal number. The special value XFFFFFFFF keeps the GID unchanged.

Error Code Specify the error code as a numeric field. Calling this API in an interactive mode does not give you a result so you can specify a value of 0 to indicate that no error code structure is being supplied.

We give you a few examples of the QSYCHGID API in the following list:

- Consider user SMITH with UID 500 and no GID. We have to change the UID of this user to 600 and keep the GID as *NONE.

  ```
  CALL PGM(QSYCHGID) PARM('SMITH' X'00000258' X'FFFFFFFF' 0)
  ```

- We have to change the GID of user BAKER from *NONE to 20850, leaving the UID unchanged.

  ```
  CALL PGM(QSYCHGID) PARM('BAKER' X'FFFFFFFF' X'00005172' 0)
  ```

- We now change the UID for user JOHN to 1500 and the GID to 7600.

  ```
  CALL PGM(QSYCHGID) PARM('JOHN' X'000005DC' X'00001DB0' 0)
  ```

Attention

You have to convert the UID and GID from decimal to hexadecimal when you specify them in the parameter list.

For more details about this API, refer to Appendix E, “CHOWN() API Description” on page 163.

4.2.4 Access Permissions

The NFS is based on the Integrated File System (IFS) and hence adopts its way of security implementation from the IFS.

An experienced AS/400 system user can see that the permissions used within IFS and the OS/400 object permissions are not identical. IFS uses the access permission concept that is also used in the UNIX/AIX environment.

The following table shows the mapping of OS/400 security with IFS/AIX security.

Table 4. Authority Mapping Table between IFS and OS/400 Authority

IFS Data Authority or Mask Bits	Data Permissions						OS/400 Authority
	OBJOPR	Read	Add	Update	Delete	Execute	
*R	X	X					User defined
*W	X		X	X	X		User defined
*X	X					X	User defined
*RW	X	X	X	X	X		User defined
*RX	X	X				X	*USE
*WX	X		X	X	X	X	User defined
*RWX	X	X	X	X	X	X	*CHANGE / *ALL (1)
*EXCLUDE							*EXCLUDE

Note: (1) *ALL Object Authority is only applicable to objects within the QSYS.LIB file system and QDLS file system.

IFS authority and OS/400 authority both refer to data and object rights. The only difference is that Object Operational (OBJOPR) right belongs to data rights in the IFS (Figure 15 on page 84) and to the object right in the OS/400 authority (Figure 16 on page 85).

```
                    Work with Authority

Object . . . . . :   /qsys.lib/thomas.lib
Owner  . . . . . :   THOMASB
Primary group  . . . . . :   NFSGRP4
Authorization list . . . . . :   *NONE


Type options, press Enter.
  1=Add user   2=Change user authority   4=Remove user

                   Data     --Object Authorities--
Opt  User        Authority  Exist  Mgt  Alter  Ref

     *PUBLIC     *RX
     THOMASB     *RWX         X     X     X     X
     NFSGRP4     *RWX

                                                                  Bottom
Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh      F9=Retrieve
F11=Display detail data authorities   F12=Cancel   F24=More keys
(C) COPYRIGHT IBM CORP. 1980, 1996.
```

```
                    Work with Authority

Object . . . . . :   /qsys.lib/thomas.lib
Owner  . . . . . :   THOMASB
Primary group  . . . . . :   NFSGRP4
Authorization list . . . . . :   *NONE


Type options, press Enter.
  1=Add user   2=Change user authority   4=Remove user

                   Data     -------------Data Authorities-------------
Opt  User        Authority  Objopr  Read  Add  Update  Delete  Execute

     *PUBLIC     *RX          X      X                            X
     THOMASB     *RWX         X      X     X     X       X        X
     NFSGRP4     *RWX         X      X     X     X       X        X

                                                                  Bottom
Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve
F11=Display object authorities     F12=Cancel   F24=More keys
```

Figure 15. IFS Authority Display. This figure shows the Work with Authority (WRKAUT) display.

The following figure shows the object and data authorities displayed when using the Edit Object Authority (EDTOBJAUT) command. The object for which the authority is displayed refers to the object from Figure 15 on page 84.

```
                    Edit Object Authority

Object . . . . . :   THOMAS        Owner  . . . . . :   THOMASB
  Library  . . . . :     QSYS        Primary group  . . . :   NFSGRP4
Object type  . . . . :   *LIB

Type changes to current authorities, press Enter.

  Object secured by authorization list  . . . . . .   *NONE

                       Object    ----------Object-----------
User        Group      Authority  Opr  Mgt  Exist  Alter  Ref
THOMASB                *ALL        X    X     X      X     X
NFSGRP4                *CHANGE     X
*PUBLIC                *USE        X

                                                              Bottom
F3=Exit  F5=Refresh  F6=Add new users  F10=Grant with reference object
F11=Display data authorities   F12=Cancel   F17=Top   F18=Bottom
```

```
                    Edit Object Authority

Object . . . . . :   THOMAS        Owner  . . . . . :   THOMASB
  Library  . . . . :     QSYS        Primary group  . . . :   NFSGRP4
Object type  . . . . :   *LIB

Type changes to current authorities, press Enter.

  Object secured by authorization list  . . . . . .   *NONE

                       Object    ---------------Data---------------
User        Group      Authority  Read  Add  Update  Delete  Execute
THOMASB                *ALL        X     X     X       X        X
NFSGRP4                *CHANGE     X     X     X       X        X
*PUBLIC                *USE        X                            X

                                                              Bottom
F3=Exit  F5=Refresh  F6=Add new users F10=Grant with reference object
F11=Nondisplay detail   F12=Cancel         F17=Top   F18=Bottom
```

Figure 16. OS/400 Authority Display. This figure shows the Edit Object Authority (EDTOBJAUT) display.

4.2.5 Changing IFS Authorities

The AS/400 system provides a common interface to change authorities for all file systems. It gives you the ability to change data and object authorities. In the following figures, we illustrate the steps involved in changing the object and data authorities. Data Authorities are also known as mask bits.

```
MAIN                  AS/400 Main Menu
                                                        System:   ASSYS01
 Select one of the following:

      1. User tasks
      2. Office tasks
      3. General system tasks
      4. Files, libraries, and folders
      5. Programming
      6. Communications
      7. Define or change the system
      8. Problem handling
      9. Display a menu
     10. Information Assistant options
     11. Client Access/400 tasks

     90. Sign off

 Selection or command
 ===> WRKLNK

 F3=Exit F4=Prompt F9=Retrieve F12=Cancel F13=Information Assistant
 F23=Set initial menu
 (C) COPYRIGHT IBM CORP. 1980, 1996.
```

Type the Work with Link (WRKLNK) command and press Enter.

```
                     Work with Object Links

 Directory  . . . . :   /

 Type options, press Enter.
   3=Copy   4=Remove   5=Next level   7=Rename   8=Display attributes
   11=Change current directory ...

 Opt   Object link            Type     Attribute    Text
  5    nfs2                   DIR
       os400                  DIR
       pubhome3.htm           STMF
       qserver                DIR
       temp                   DIR
       tmp                    DIR
       usertest               DIR
       wblserv.cfg            STMF
       welcome.html           STMF
                                                              More...
 Parameters or command
 ===>
 F3=Exit F4=Prompt F5=Refresh F9=Retrieve F12=Cancel F17=Position to
 F22=Display entire field           F23=More options
```

Select Option 5 to go to the next level.

```
                    Work with Object Links

Directory  . . . . :   /nfs2

Type options, press Enter.
  3=Copy   4=Remove   5=Next level   7=Rename   8=Display attributes
  11=Change current directory ...

Opt   Object link            Type     Attribute    Text
9     data                   DIR
      programs               DIR









                                                                    Bottom
Parameters or command
===>
F3=Exit F4=Prompt F5=Refresh F9=Retrieve F12=Cancel F17=Position to
F22=Display entire field           F23=More options
```

Select Option 9 (Work with Authority (WRKAUT)).

```
                    Work with Authority

Object . . . . . . . . . . . . :   /nfs2/data
Owner  . . . . . . . . . . . . :   NFS4
Primary group  . . . . . . . . :   *NONE
Authorization list . . . . . . :   *NONE


Type options, press Enter.
  1=Add user   2=Change user authority   4=Remove user

                   Data     --Object Authorities--
Opt  User        Authority  Exist  Mgt  Alter  Ref

     *PUBLIC     *RWX         X     X     X     X
 2   NFS4        *RWX         X     X     X     X
     QSYS        *RWX         X     X     X     X

                                                                  Bottom
Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh      F9=Retrieve
F11=Display detail data authorities   F12=Cancel   F24=More keys
```

On this display, you are allowed to perform the following actions:

- Add a user.
- Change user authority.
- Remove a user.

We take Option 2 to change the user authority.

```
                    Change Authority (CHGAUT)

Type choices, press Enter.

Object . . . . . > '/nfs2/data'  Path name
User . . . . . > NFS4          Name, *PUBLIC, *NTWIRF
New data authorities . . . . . > *RX           *SAME, *NONE, *RWX, *RX..
New object authorities . . . . > *OBJMGT       *SAME, *NONE, *ALL...
                               >
                               > *OBJALTER
                               >
Authorization list . . . . . .                 Name, *NONE









                                                                      Bottom
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
```

In this example, we remove the *W data authority and the *OBJEXIST and *OBJREF object authorities. Then press Enter twice.

```
                    Work with Authority

Object . . . . . :   /nfs2/data
Owner  . . . . . :   NFS4
Primary group  . . . . . :   *NONE
Authorization list . . . . . :   *NONE


Type options, press Enter.
  1=Add user   2=Change user authority   4=Remove user

                   Data     --Object Authorities--
Opt  User        Authority  Exist  Mgt  Alter  Ref

     *PUBLIC     *RWX         X     X     X     X
     NFS4        *RX                X     X
     QSYS        *RWX         X     X     X     X

                                                                      Bottom
Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh      F9=Retrieve
F11=Display detail data authorities   F12=Cancel   F24=More keys
```

Press F5 to refresh the display. Now you see the modified authorities on your display.

4.2.6 Changing the Owner of an Object

With the command usually used to change the owner of an object within OS/400 (Change Object Owner (CHGOBJOWN)), you cannot change the ownership of an object within an IFS file system except the QSYS.LIB file system. Instead, you have to use the Change Owner (CHGOWN) command.

```
                    Change Owner (CHGOWN)

Type choices, press Enter.

Object . . . . . .   /nfs2/pgm/account/v1

New owner  . . . . . .   THB           Name
Revoke current authority .   *YES          *NO, *YES
Symbolic link  . . . . . .   *NO           *NO, *YES

                                                                  Bottom
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
```

Within this command, you can specify the object name as a relative or absolute name.

- Relative Object Name:

  `CHGOWN OBJ('v1') NEWOWN(THB)`

  In this case, the object is being searched within the current directory. Therefore, pay attention to which directory you are currently in. Otherwise, you may get error messages that the object cannot be found, even if it exists.

- Absolute Object Name:

  `CHGOWN OBJ('/nfs2/pgm/account/v1') NEWOWN(THB)`

  When specifying the absolute object path, you have to start from the root directory. The absolute object path starts with a slash (/) as the first character.

When using the CHGOWN command on the AS/400 NFS system, you just have to have the proper authority to the user profile that becomes the new owner.

Pay special attention when changing the ownership on the AS/400 NFS server from a NFS client. Section 4.2.6.1, “Changing the Owner of an Object from a Client” on page 90 gives you more information about which dependencies exist when changing the owner on the server from a client.

4.2.6.1 Changing the Owner of an Object from a Client

This section contains information you need to be aware of when changing the owner of an object on the AS/400 NFS server from an NFS client.

Changing from an AS/400 Client: When you change the owner of an object on the server, you must have the appropriate permission to the user profile on the client and on the server, which should become the new owner. Usually this task is done by a user profile with *SECOFR special authority. This means when using default options on the Export File System (EXPORTFS) command, users with *ALLOBJ special authority are mapped to the QNFSANON user. In this case, if the QNFSANON user does not have the proper access authority to the intended new owner user profile, the change owner (CHGOWN) command is rejected. Refer to Section 4.4, “Special Considerations on Exported File Systems” on page 102 for further details on UID mapping. In case you do not have *ALLOBJ special authority on the server, you must be the owner of the object to change the ownership.

The following example shows the minimum permissions to be set when changing the ownership.

Example:

Assume user profile JIMB on the NFS server currently owns directory */etc/invoice* and user JIMA on the NFS client wants to assign user profile TOMB on the server to be the new owner of this directory. The following table shows the settings that apply to change the owner.

Current Owner			
AS/400 NFS Client		AS/400 NFS Server	
User Profile Name	JIMA	User Profile Name	JIMB
UID	2500	UID	2500
Private Authority to the user profile that becomes the new owner	*USE	Private Authority to the user profile that becomes the new owner	*CHANGE
New Owner			
AS/400 NFS Client		AS/400 NFS Server	
User Profile Name	TOMA	User Profile Name	TOMB
UID	3500	UID	3500
Note: The authorities shown in this table are the minimum private authorities to be defined to change the ownership.			

As shown in the preceding example, both the current and the new owner user profile must exist on the client and on the server with the matching UID. The current owner must have private authority to the new owner’s user profile.

The following command, which is specified on the client system, changes the ownership of the directory on the server system. The server directory */etc/invoice* is mounted to the */mpt* directory on the client.

```
CHGOWN OBJ('/mpt') USER(TOMA)
```

The easiest way to change the owner is by using user profiles with *SECOFR authority and matching UIDs on the client and the server. But then you have to specify the *ROOT=* parameter on the export options so the client user is mapped to the appropriate user profile on the server.

Changing from an AIX Client: When changing the owner of an object on the server from an AIX client, you just have to have the proper authority on the server to the new owner user profile. You do not need an appropriate user with a matching UID on the client as well as no special authorities exist on the AIX client to perform the Change Owner (CHOWN) command. When not having the *ALLOBJ special authority on the server, you must be the owner of the object for which you want to assign a new owner. Once you have changed the ownership, you cannot change it again for this object. In this case, the server user profile needs *ALLOBJ special authority again to perform changes to the ownership. The following example shows you the important user settings when changing the ownership from an AIX client.

Example:

Assume user profile DAISY on the NFS server currently owns directory */tools/files* and user KIRK on the NFS client wants to assign user profile WILLI on the server to be the new owner of this directory. The following table shows the settings that apply to change the owner.

Current Owner			
AIX NFS Client		AS/400 NFS Server	
User Profile Name	KIRK	User Profile Name	DAISY
UID	120	UID	120
Private Authority to the user profile that becomes the new owner	No user required	Private Authority to the user profile that becomes the new owner	*CHANGE
New Owner			
AIX NFS Client		AS/400 NFS Server	
User Profile Name	No user required	User Profile Name	WILLI
UID	No UID required	UID	244
Note: The authorities shown in this table are the minimum private authorities to be defined to change the ownership.			

Based on the preceding example, the following command, which is specified on the client system, changes the ownership of the directory on the server system. The server directory */tools/files* is mounted to the */mpt* directory on the client.

```
chown 244 /mpt
```

4.2.7 Changing the Primary Group

Within IFS, each object may or may not have a primary group assigned to it. The primary group is usually adopted from the parent directory when a new object is created within this directory. By default, no primary group is assigned when creating an object under the root directory. In this case, the primary group name value is *NONE. To change the primary group, use the Change Primary Group (CHGPGP) command.

```
                    Change Primary Group (CHGPGP)

Type choices, press Enter.

Object . . . . . . . . . . . . . > '/nfs2/pgm/account'

New primary group  . . . . . . . > ACCGRP1       Name, *NONE
New data authorities . . . . . . > *OLDPGP       *OLDPGP, *PRIVATE, *RWX
New object authorities . . . . .   *NONE         *NONE, *ALL, *OBJEXIST.
              + for more values
Revoke current authority . . . .   *YES          *NO, *YES
Symbolic link  . . . . . . . . .   *NO           *NO, *YES









                                                                    Bottom
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
```

While changing the primary group with the CHGPGP command, you can also assign permissions to that group. These are:

Data Authorities

Authorities in any combination of Read (R), Write (W) or Execute (X)

Object Authorities

The object authorities are shown in the following list:

*ALL
*OBJEXIST
*OBJALTER
*OBJMGT
*OBJREF
*NONE

Attention

You can also change permissions to the group *NONE. If you want to remove an existing primary group, specify *NONE within the *New primary group* parameter.

4.2.8 Authority on AIX Client

Once you have mounted the directory from an AS/400 NFS server to your AIX Client, you can change the data authorities for objects within the mounted directory or its sub-directories. In any case, you need the appropriate permission to change authorities. On the AIX operating system, there are three different authority types. These are:

- Owner authority
- Group authority
- Public authority

Use the List (ls) command with option -l (long) to display the authority attributes on the AIX system.

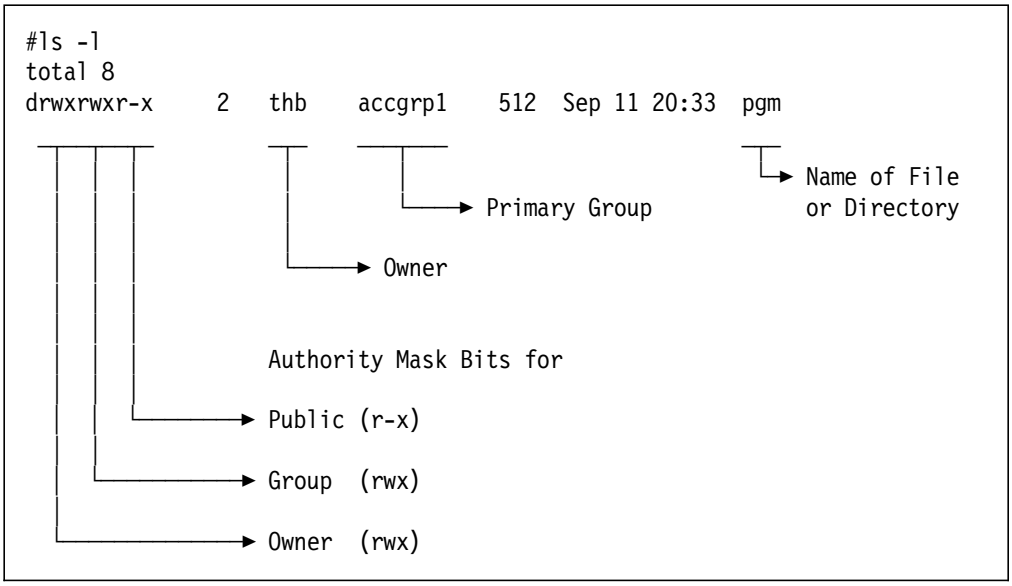


Figure 17. Authority Mask Bits using the ls Command

The following list explains the information given in Figure 17:

Authority Bits or Mask Bits

The rwx values within the mask bits refer to the read, write, and execute permissions. The "-" at any of these positions means that you do not have that particular permission.

Example:

r-x

means you have read and execute permission, but the write permission is denied.

--x

means you have execute permission but no read or write permission.

Owner Authority

These are the permissions given to the owner of a file or directory.

Group Authority

These are permissions given to the primary group of the file or directory.

Public Authority

These are permissions given to the public, which includes the users who are not the owners or members of the primary group of the file or

directory. On the AS/400 system, the PUBLIC authority is listed under the Work with Authority (WRKAUT) command as the *PUBLIC user.

Owner

This gives you the UID or name of the user who owns the file or directory. The name displayed at this position is not the name of the owner on the server, but is the name of the user with the corresponding UID at the client. Finding a UID at this position means that the UID of the owner on the server does not exist on the client. In AIX/UNIX, users are defined in the */etc/passwd*file.

Group

This gives you the GID or name of the primary group of the file or directory. The name displayed at this position is not the name of the group on the server, but is the name of the group with the corresponding GID at the client. Finding a GID at this position means that the GID of the group on the server does not exist on the client. In AIX/UNIX, groups are defined in the */etc/group*file.

File or Directory

This gives you the name of the file or directory.

Examples:

```
drw-r-x--- 2  501 nfsgrp 512 Sep 09  08:42 pgmdir
```

Owner Authority	Read, Write
Group Authority	Read, Execute
Public Authority	None
Owner	501 (UID of 501 does not exist on the client)
Primary Group	nfsgrp (GID exists on both client and server)
File or Directory name	pgmdir

```
drwx---r-- 2  nfstest 212   512 Sep 09  08:42 accfile
```

Owner Authority	Read, Write, Execute
Group Authority	None
Public Authority	Read
Owner	nfstest (UID exists on both, client and server)
Primary Group	212 (GID of 212 does not exists on the client)
File or Directory name	accfile

Attention

By default, the primary group SYSTEM has a GID of 0 on the AIX system. Even though the current AS/400 implementation of GID cannot have a group with GID value of 0, the AS/400 system assigns a GID of 0 for all objects that do not have a specified primary group. (That is, the owners of the objects have GRPPRF(*NONE) and GID(*NONE) defined in their user profiles.) This means when an AIX NFS client displays a group name of SYSTEM to the directories or files, the object on the AS/400 NFS server has no primary group defined.

4.2.8.1 Changing Authorities from an AIX NFS Client

The Change Mode (CHMOD) command allows you to perform the authority changes from the AIX Client.

The syntax of the CHMOD command is given in the following example:

```
chmod  PermissionCode { File ... | Directory ... }
```

The chmod command permits you to use octal notation for the mode. The numeric mode is the sum of one or more of the following values:

400	Permits read by owner.
200	Permits write by owner.
100	Permits execute or search by owner.
040	Permits read by group.
020	Permits write by group.
010	Permits execute or search by group.
004	Permits read by others.
002	Permits write by others.
001	Permits execute or search by others.

Examples:

```
chmod  755  data
```

Owner	read, write and execute
Primary Group	read and execute
Public	read and execute

```
chmod  761  data
```

Owner	read, write and execute
Primary Group	read and write
Public	execute

The user can also change permissions by specifying the authorizations symbolically. Refer to the publication *AIX Version 4 Commands Reference*, SBOF-1851, for more details on the CHMOD command.

Attention

You can only change data authorities on the AS/400 NFS server from the AIX client. AS/400 object authorities cannot be changed from any NFS client.

4.2.9 Changing Authorities from an AS/400 NFS Client

You cannot use the Work with Authority (WRKAUT) command as mentioned in Section 4.2.5, “Changing IFS Authorities” on page 86 to change the authorities from an AS/400 NFS client even if the command prompt gives you the ability to do that. Instead, you have to use the chmod() API. The inter-operability between different operating system platforms is provided only with the POSIX compliant APIs. The OS/400 commands (also known as generic commands) are not always POSIX compliant. Therefore, we recommend using APIs to ensure POSIX compliance. For more details of the Change File Authorizations- chmod() API, refer to the publication *OS/400 UNIX-Type APIs V3R7*, SC41-4875.

4.3 Creating Objects from NFS Clients

There are differences in the authorities granted to objects when they are created from different NFS clients. In this section, we use examples to cover the creation of objects from an AIX NFS client and an AS/400 NFS client. The NFS server is the AS/400 system in both cases.

4.3.1 Creating Objects from an AIX NFS Client

We want to create a subdirectory *PGMSUB* beneath the parent directory *PGM*.

Example 1:

The following setup is assumed for the example.

AIX Client		**AS/400 Server**		
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory		
User Name	SMITH6	Owner User Profile	SMITH4	
UID	300	Owner UID	300	
Primary Group Name	DEPT5	Primary Group Name	DEPTRS6K	
GID	666	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rwx	*OBJMGT
		*PUBLIC Authority	rwx	*NONE

The authority settings of the *PGMSUB* subdirectory created by the user SMITH6 are:

Owner User Profile	SMITH4	
Owner UID	300	
Primary Group Name	DEPTRS6K	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	*NONE	*OBJMGT
*PUBLIC Authority	*NONE	*NONE

In this example, the UID of the client user exists on the server. So the owner of the new subdirectory is the user with the matching UID (in this case, SMITH4). Further, you see that the object authorities are inherited from the parent directory. From an AIX NFS client, the data authorities (by default) are always set to the values shown in the table.

Example 2:

The following setup is assumed for the example.

AIX Client		AS/400 Server		
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory		
User Name	BAKER	Owner User Profile	SMITH4	
UID	519	Owner UID	300	
Primary Group Name	PGMR6	Primary Group Name	DEPTRS6K	
GID	304	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rwx	*ALL
		*PUBLIC Authority	rwx	*OBJREF

The authority settings of the *PGMSUB* subdirectory created by the user BAKER are:

Owner User Profile	QNFSANON	
Owner UID	4294770710	
Primary Group Name	DEPTRS6K	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	*NONE	*ALL
*PUBLIC Authority	*NONE	*OBJREF

In this example, the UID of the client user does not exist on the server. So the owner of the new subdirectory is the QNFSANON user, which is the default owner for objects created by anonymous client users. The object and data authorities are treated in a similar way as in Example 1. Since the AS/400 system does not care about group memberships of the client user when assigning the primary group, the primary group on the server is also inherited from the parent directory.

Attention

When exporting a directory from the AS/400 system, by default, the root user of an AIX NFS client with UID 0 is mapped to the QNFSANON user on the AS/400 NFS server. This default mapping can be changed by specifying the ROOT parameter within the export options of the Export File System (EXPORTFS) command.

Example 3:

The following setup is assumed for this example.

AIX Client		AS/400 Server		
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory		
User Name	MARION	Owner User Profile	SMITH4	
UID	400	Owner UID	300	
Primary Group Name	DEPT5	Primary Group Name	DEPTRS6K	
GID	666	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rwx	*OBJMGT
		*PUBLIC Authority	rwx	*NONE

Note: The UID of 400 exists on the AS/400 system for the user profile ANDY.

The authority settings of the *PGMSUB* subdirectory created by the user MARION are:

Owner User Profile	ANDY	
Owner UID	400	
Primary Group Name	DEPTRS6K	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	*NONE	*OBJMGT
*PUBLIC Authority	*NONE	*NONE

In this example, the UID of the client user exists on the server. The owner of the new subdirectory is the user with the matching UID (in this case, ANDY). NFS security is only checked through the UID and not the user profile name. In this example, client user MARION created a directory on a server but on the NFS server, the UID corresponds to a user profile ANDY. Therefore, ANDY became the owner of the new directory.

The object and data authorities are treated in a similar way as in the previous examples.

4.3.2 Creating Objects from an AS/400 NFS Client

We want to create a subdirectory *PGMSUB* beneath the parent directory *PGM*.

Example 1:

The following setup is assumed for this example.

AS/400 Client		AS/400 Server	
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory	
User Name	THOMAS	Owner User Profile	KRIS

AS/400 Client		AS/400 Server		
UID	300	Owner UID	300	
Primary Group Name	DEPT77	Primary Group Name	DEPTAS4	
GID	666	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rx	*OBJMGT, *OBJEXIST
		*PUBLIC Authority	rwx	*OBJREF

The authority settings of the *PGMSUB* subdirectory created by the user THOMAS are:

Owner User Profile	KRIS	
Owner UID	300	
Primary Group Name	DEPTAS4	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	rx	*OBJMGT, *OBJEXIST
*PUBLIC Authority	rwx	*OBJREF

In this example, the UID of the client user exists on the server. The owner of the new subdirectory is the user with the matching UID. The object and data authorities are both inherited from the parent directory.

Example 2:

The following setup is assumed for this example.

AS/400 Client		AS/400 Server		
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory		
User Name	FRANK	Owner User Profile	PETER	
UID	5249	Owner UID	300	
Primary Group Name	DEPT77	Primary Group Name	DEPTAS4	
GID	111	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rx	*OBJMGT, *OBJEXIST
		*PUBLIC Authority	rwx	*OBJREF

Note: The UID of 5249 and GID of 111 do not exist on the AS/400 NFS server.

The authority settings of the *PGMSUB* subdirectory created by the user FRANK are:

Owner User Profile	QNFSANON	
Owner UID	4294770710	
Primary Group Name	DEPTAS4 On the client, you see the value *NONAME* corresponding to this.	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	rx	*OBJMGT, *OBJEXIST
*PUBLIC Authority	rwx	*OBJREF

In this example, the UID of the client user does not exist on the server. The owner of the new subdirectory is the QNFSANON user, which is the default owner for objects created by anonymous client users. Even if the GID of the client users group does not exist on the server, the primary group is inherited from the parent directory. On the client, the primary group name is NONAME. The object and data authorities are also both inherited from the parent directory.

Example 3:

The following setup is assumed for this example.

AS/400 Client		AS/400 Server		
Mount Point: ASPGM		Exported Directory: PGM Authority Settings for the exported directory		
User Name	STEPHEN	Owner User Profile	MIKE	
UID	445	Owner UID	445	
Primary Group Name	DEPT77	Primary Group Name	DEPTAS4	
GID	666	GID	666	
			Data Authority	Object Authority
		Owner Authority	rwx	*ALL
		Group Authority	rx	*OBJMGT, *OBJEXIST
		*PUBLIC Authority	rwx	*OBJREF

The authority settings of the *PGMSUB* subdirectory created by the user STEPHEN are:

Owner User Profile	QNFSANON	
Owner UID	4294770710	
Primary Group Name	DEPTAS4	
GID	666	
	Data Authority	Object Authority
Owner Authority	rwx	*ALL
Group Authority	rx	*OBJMGT, *OBJEXIST
*PUBLIC Authority	rwx	*OBJREF

In this example, the UID of the client user exists on the server, but the user profile MIKE corresponding to this UID (445) has *ALLOBJ special authority. The owner of the new subdirectory is the QNFSANON user. The user profile with *ALLOBJ special authority is, by default, mapped to the QNFSANON user. Instead, if you want user MIKE as the owner, you have to specify the system that MIKE signs onto in the ROOT parameter within the export options of the Export File System (EXPORTFS) command. The object and data authorities are both inherited from the parent directory.

4.4 Special Considerations on Exported File Systems

There are special security considerations when exporting the file system using NFS.

To secure the AS/400 system as much as possible, NFS has special parameter values when exporting a file system or parts of it. By default, NFS does not allow client users to use the QSECOFR profile or other user profiles with *ALLOBJ special authority on the AS/400 NFS server. There are also other parameters that maintain the different access levels to the AS/400 system.

UID is 0

The UID of 0 on the AIX system refers to the ROOT user. On the AS/400 system, it refers to the QSECOFR user profile. To allow client users with UID of 0 to access the AS/400 server through the QSECOFR user profile, specify the *ROOT=* parameter. On the *ROOT=* parameter, you can specify HOSTNAMES for which users with a UID of 0 are to be mapped to the QSECOFR user profile. Otherwise, these users (by default) are mapped to the QNFSANON user profile unless the *ANON=* parameter is specified.

USER with *ALLOBJ special authority

Users on the server with the *ALLOBJ special authority are also treated the same way as client users with UID of 0. This means that if the UID of the client user matches a UID on the server with the user profile having *ALLOBJ special authority, you are mapped to the QNFSANON user by default. In this case, you have to use the *ROOT=* parameter again to map to the user profile with the *ALLOBJ special authority.

Anonymous user access

Client users who do not have a matching UID on the AS/400 NFS server are (by default) mapped to the QNFSANON user. This user also gets the ownership of objects created by the client user. The *ANON=* parameter within the export options of the Export File System (EXPORTFS) command allows you to specify a different UID of the user profile that gets mapped on anonymous access, or to deny anonymous access with the value of “-1.”

Access parameter

To allow only specific hosts to access the exported file system, you can specify one or more host names in the *ACCESS=* parameter. Only those hosts that are specified in the *ACCESS=* parameter can access the exported directory on the server.

General read and write permissions

When exporting a file system or a part of it (by default), you get read/write access to the exported file system. Instead, you can specify read only access using *RO*. In this case, no client has write access to the exported file system regardless of the authorities specified on the server. You can also limit the read/write access to one or more hosts by specifying the *RW=* parameter with the appropriate host names. If you use the *RW=* parameter, all HOSTS not listed on the *RW=* list, get *RO* access.

Deleting objects

When an NFS client user intends to delete objects within an exported file system on the AS/400 NFS server, the user needs at least the *WX Mask Bits and the *OBJEXIST object authority to that object. Usually the *OBJEXIST object authority is adopted from the parent directory when creating an object. But if a user has removed this *OBJEXIST object authority from the parent directory, it is, therefore, not given to the objects

created within the directory. Ensure that you have set the proper Mask Bits and Object Authorities.

Refer to Section 3.3, “Exporting File Systems” on page 32 for detailed information about export options.

4.5 Security Checking Algorithm

On every attempt of a client user to perform an action within a mounted file system, the AS/400 NFS server has a special security checking procedure to ensure the proper access rights for the incoming client users. This procedure is illustrated in Figure 18 on page 104.

Later on we give you a couple of examples that show (on given scenarios) the flow through the flowchart.

Attention when Deleting Objects

Remember to specify the *OBJEXIST object authority in addition to the Mask Bits that are required to delete an object. Even if you have the *RWX Mask Bits set for a specific object, you cannot delete it until you have *OBJEXIST object authority.

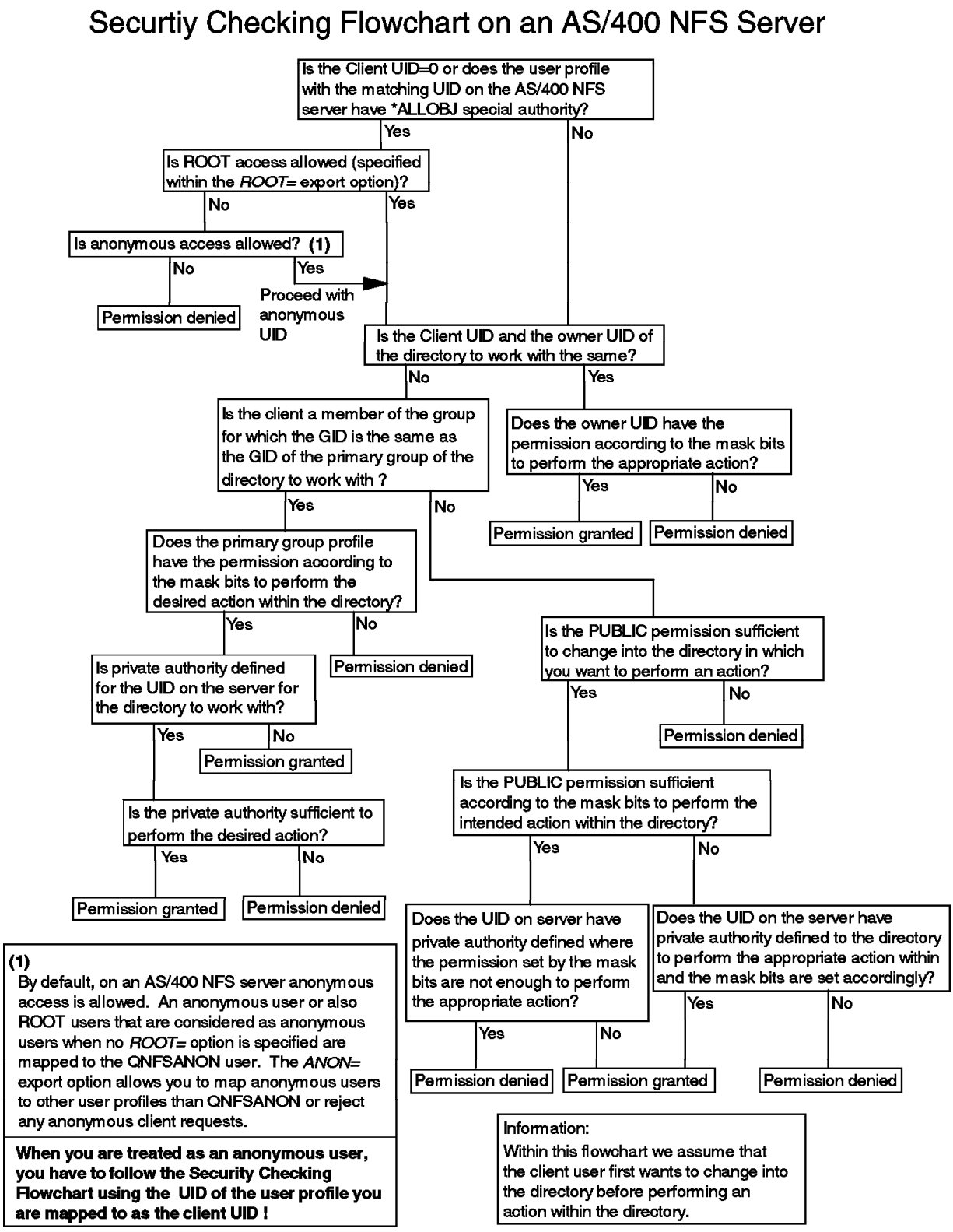


Figure 18. Security Checking Flowchart. This flowchart shows the Security Checking Algorithm used by NFS to check permissions for client requests.

Example 1:

Consider the following situation:

User MARY from the NFS client has mounted the exported directory */pgm/invoice* from the AS/400 system ASSYS01 to mount point */mptsys01*. Mary wants to change from the root directory into the *invoice* directory with the command:

```
CD '/mptsys01'
```

Then Mary tries to create a subdirectory *april* under the parent directory */mptsys01/pgm/invoice* using the following command.

```
MD 'april'
```

The following table shows the given settings.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory	/pgm/invoice			
Owner	MARY41			
Primary Group	PGMRS1			
	Mask Bits			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
MARY41	X	X	X	X
PGMRS1	X	-	X	X
*PUBLIC	-	-	-	X
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
MARY41	500	*NONE		*NONE
PGMRS1	43254	1200		*NONE
QSECOFR	0	*NONE	X	*NONE
Export Options				
-i				
AS/400 NFS Client				
Systemname	ASSYS02			
Mount point	/mptsys01			
Client User Profile	MARY			
Client UID	500			
Member of Group				
GID				
Note: Export Options: -i means, defaults are used.				

The Security Checking Flowchart for this example follows:

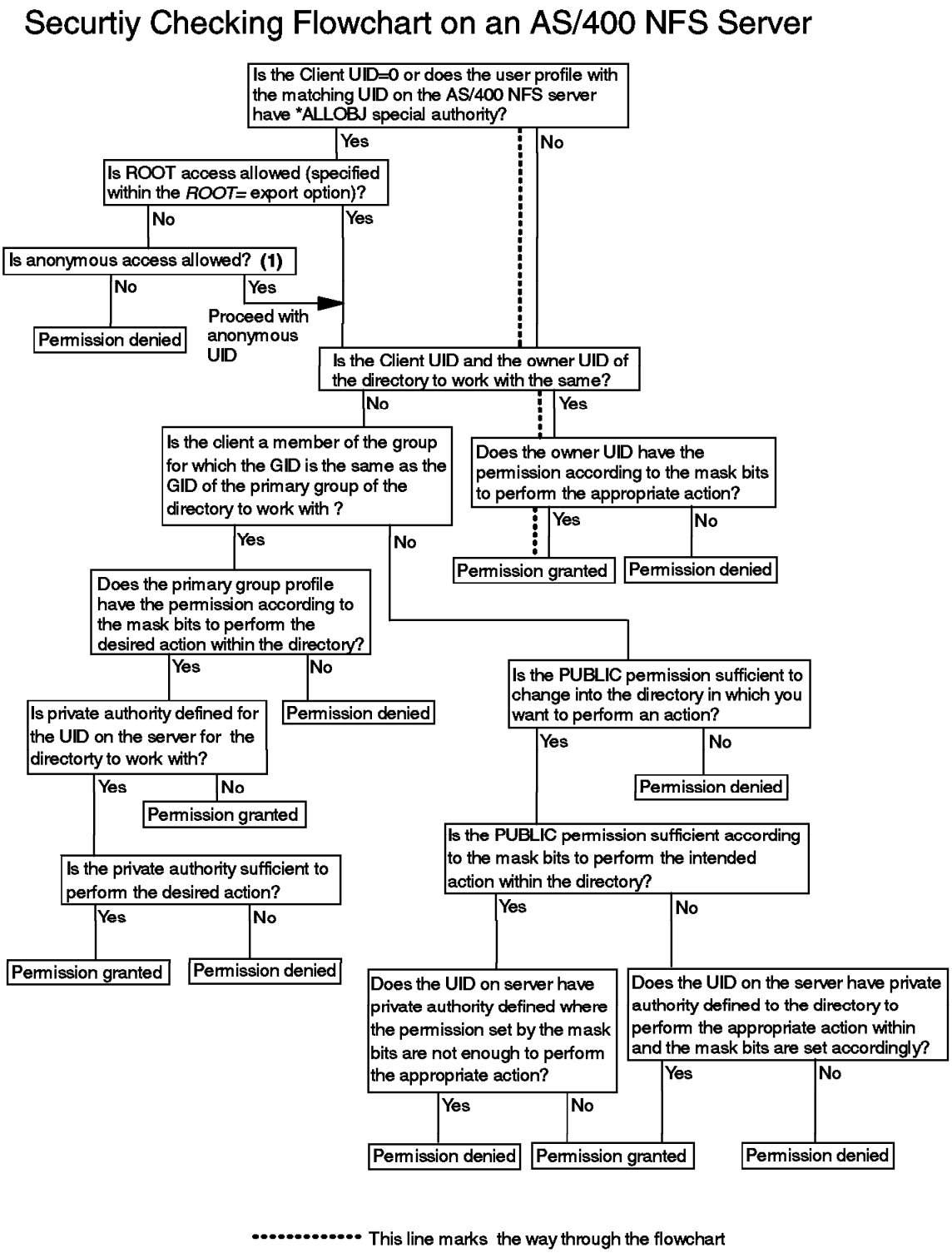


Figure 19. Security Checking Flowchart for Example 1

Example 2:

Consider the following situation:

User MARV from the NFS client has mounted the exported directory */pgm/account* from the AS/400 system ASSYS01 to mount point */mptsys01*. MARV wants to change from the root directory into the *account* directory with the command:

```
CD '/mptsys01'
```

Then Marv tries to delete file *empl* within the current directory.

```
DEL 'empl'
```

The appropriate AS/400 system generic command is the Remove Link (RMVLNK) command. The following table shows the given settings for this example.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory	/pgm/account			
Filename	empl			
Owner	AC10			
Primary Group	*NONE			
	Mask Bits for parent directory (X) / file (Y)			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
AC10	X / Y	X / Y	X / Y	X / Y
*PUBLIC	X / Y	X / -	X / Y	X / Y
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
AC10	555	*NONE		*NONE
QSECOFR	0	*NONE	X	*NONE
Export Options				
-i				
AS/400 NFS Client				
Systemname	ASSYS02			
Mount point	/mptsys01			
Client User Profile	MARV			
Client UID	600			
Member of Group	*NONE			
GID	*NONE			
Note: Export Options: -i means, defaults are used. There is no user profile with UID 600 in the server. GID(*NONE) for client is different from for server's. See 4.2.2, "Group Identification (GID)" on page 80 for deail.				

The Security Checking Flowchart for this example follows:

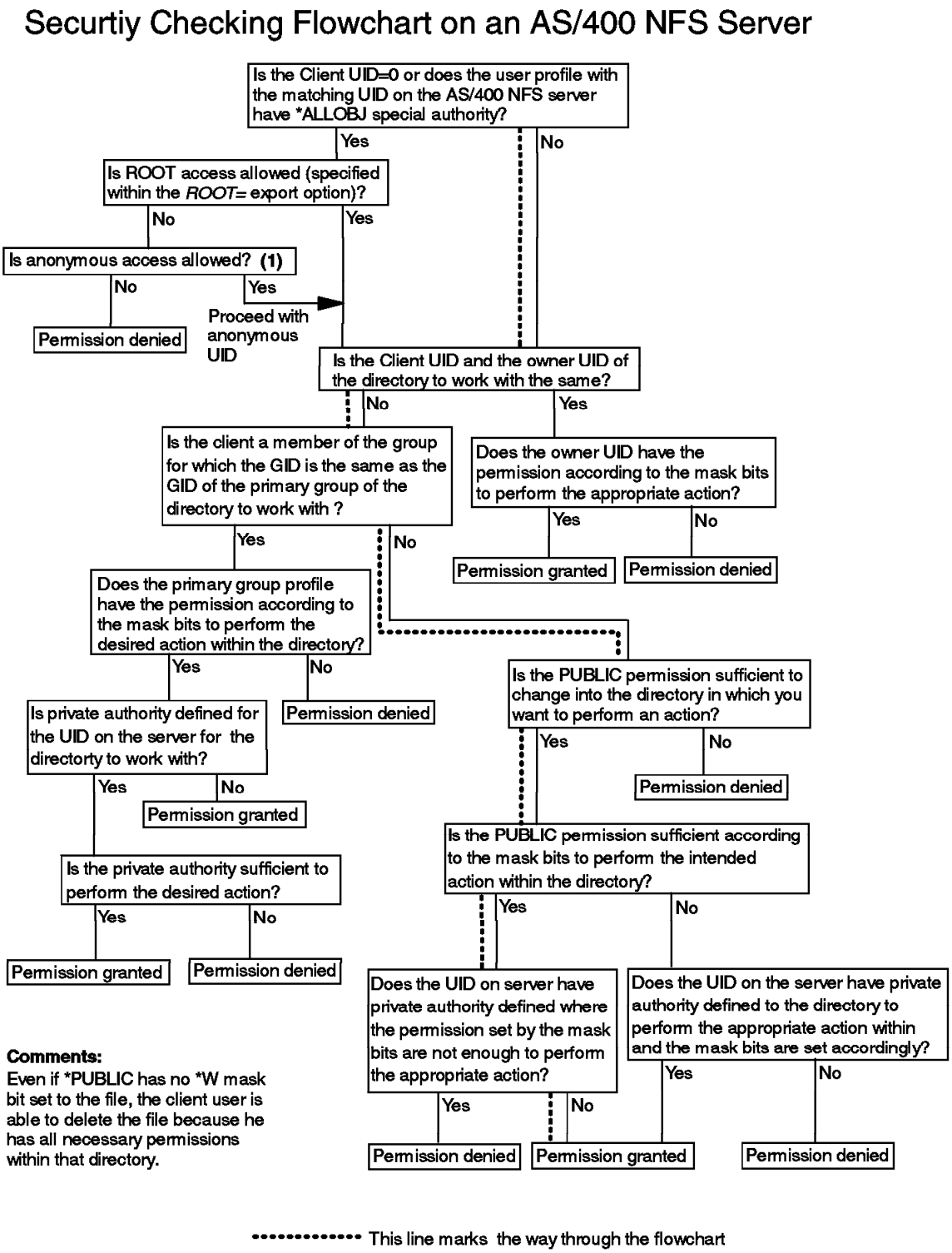


Figure 20. Security Checking Flowchart for Example 2

Example 3:

Consider the following situation:

User THB from the NFS client has mounted the exported directory */pgm/account* from the AS/400 system ASSYS01 to mount point */mptsys01*. THB wants to change from the root directory into the *account* directory with the command:

```
CD '/mptsys01'
```

Now THB tries to delete the file *sale* within the current directory.

```
DEL 'sale'
```

The appropriate AS/400 system generic command is the Remove Link (RMVLNK) command. The following table shows the given settings for this example.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory	/pgm/account			
Filename	sale			
Owner	THBSERV			
Primary Group	*NONE			
	Mask Bits for parent directory (X) / file (Y)			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
THBSERV	X / Y	X / Y	X / Y	X / Y
*PUBLIC	X / -	- / -	X / -	X / Y
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
THBSERV	755	*NONE	X	*NONE
QSECOFR	0	*NONE	X	*NONE
QNFSANON	4294770710	*NONE		*NONE
Export Options				
-i				
AS/400 NFS Client				
Systemname	ASSYS02			
Mount point	/mptsys01			
Client User Profile	THB			
Client UID	755			
Member of Group	*NONE			
GID	*NONE			
Note: Export Options: -i means, defaults used. GID(*NONE) for client is different from for server's. See 4.2.2, "Group Identification (GID)" on page 80 for deail.				

The Security Checking Flowchart for this example follows:

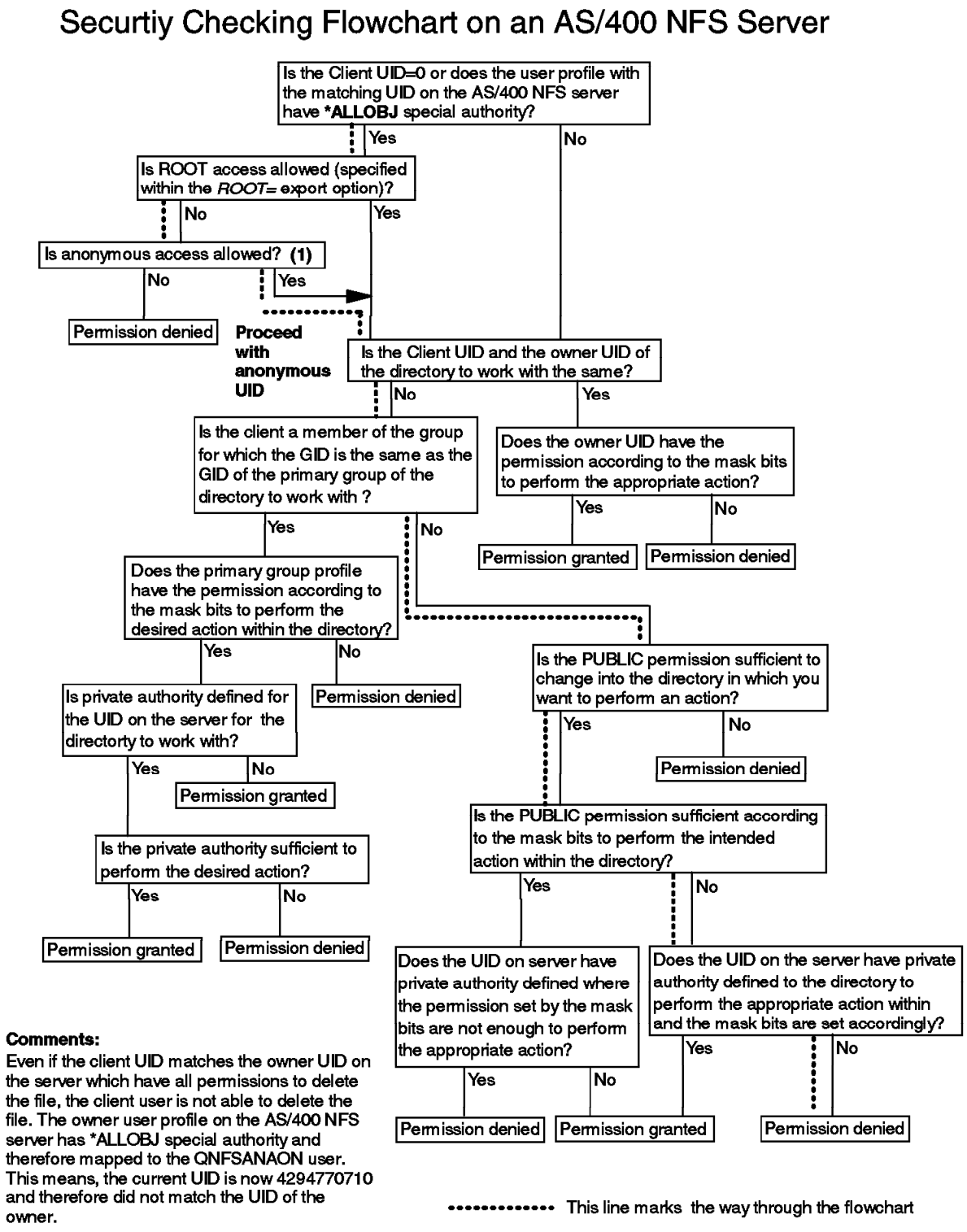


Figure 21. Security Checking Flowchart for Example 3

Example 4:

Consider the following:

User STEVE from the NFS client has mounted the exported directory */data/misc* from the AS/400 system ASSYS01 to mount point */mptsys01*. Steve wants to change from the root directory into the *misc* directory with the command:

```
CD '/mptsys01'
```

Now the user tries to create a directory *tool* within the current directory.

```
MD 'tool'
```

The following table shows the given settings for this example.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory	/data/misc			
Owner	STEVE2			
Primary Group	*NONE			
	Mask Bits			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
STEVE2	X	X	X	X
*PUBLIC	X	-	X	X
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
STEVE2	800	*NONE	X	*NONE
QSECOFR	0	*NONE	X	*NONE
QNFSANON	4294770710	*NONE		*NONE
Export Options				
-i -o ROOT=ASSYS02				
AS/400 NFS Client				
Systemname	ASSYS02			
Mount point	/mptsys01			
Client User Profile	STEVE			
Client UID	800			
Member of Group				
GID				
Note: Export Options: -i -o ROOT=ASSYS02 means, root access is allowed for clients from host ASSYS02.				

The Security Checking Flowchart for this example follows:

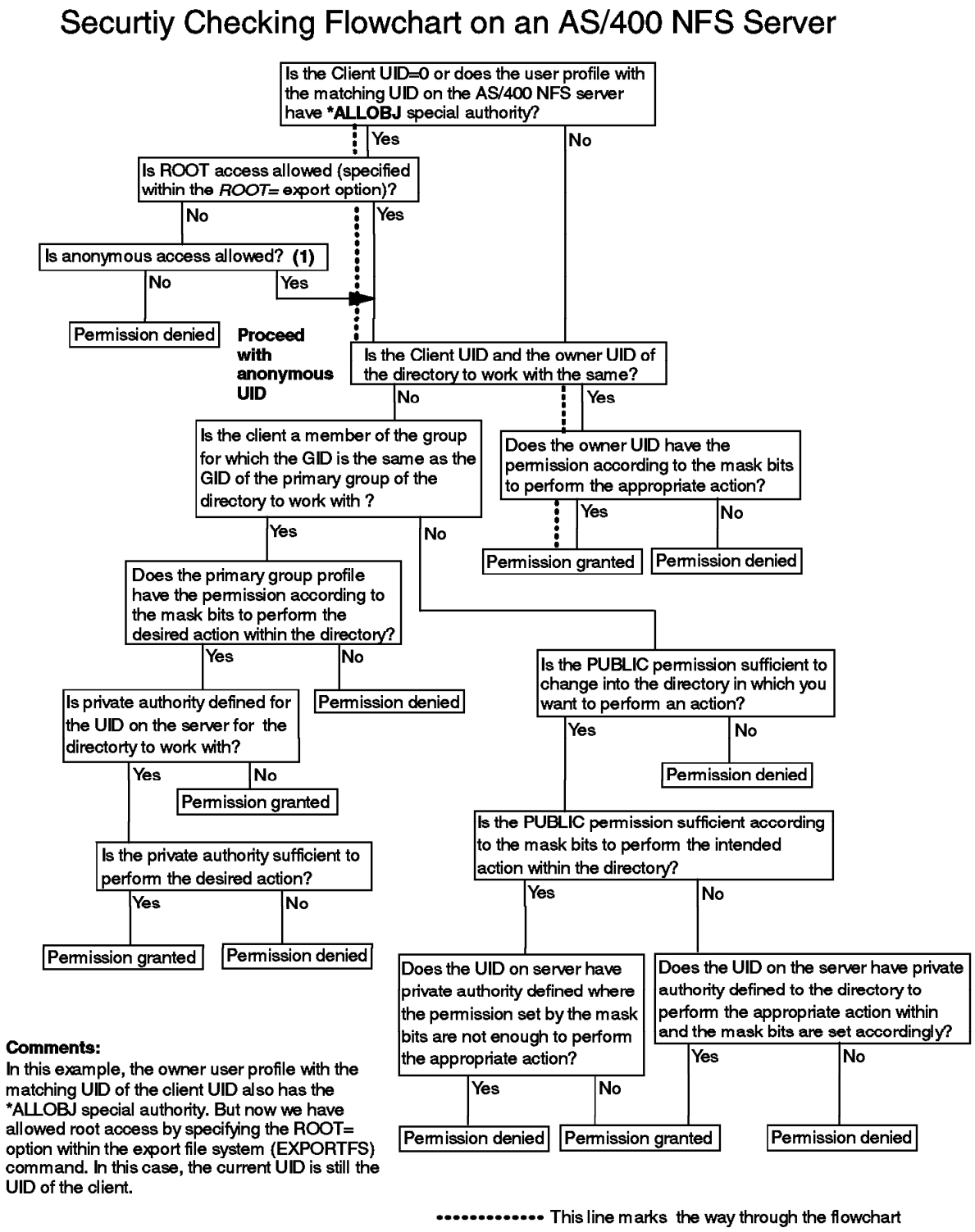


Figure 22. Security Checking Flowchart for Example 4

Example 5:

Consider the following situation:

User JILL from the NFS client has mounted the exported directory */pgm/c* from the AS/400 system ASSYS01 to mount point */mptsys01*. Jill wants to change from the root directory into the *c* directory with the command:

```
CD '/mptsys01'
```

Now Jill tries to create a directory *calc* within the current directory.

```
mkdir calc
```

The following table shows the given settings for this example.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory	/pgm/c			
Owner	DAVE			
Primary Group	NFSGRP4			
	Mask Bits			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
DAVE	X	X	X	X
NFSGRP4	X	X	X	X
*PUBLIC	-	-	-	X
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
DAVE	555	*NONE		*NONE
QSECOFR	0	*NONE	X	*NONE
NFSGRP4	3245	600		*NONE
JIM4	255	*NONE		NFSGRP4
Export Options				
-i				
AIX NFS Client				
Systemname	RSSYS05			
Mount point	/mptsys01			
Client User	JILL			
Client UID	255			
Member of Group	NFSGRP6			
GID	600			
Note: Export Options: -i means, defaults used.				

The Security Checking Flowchart for this example follows:

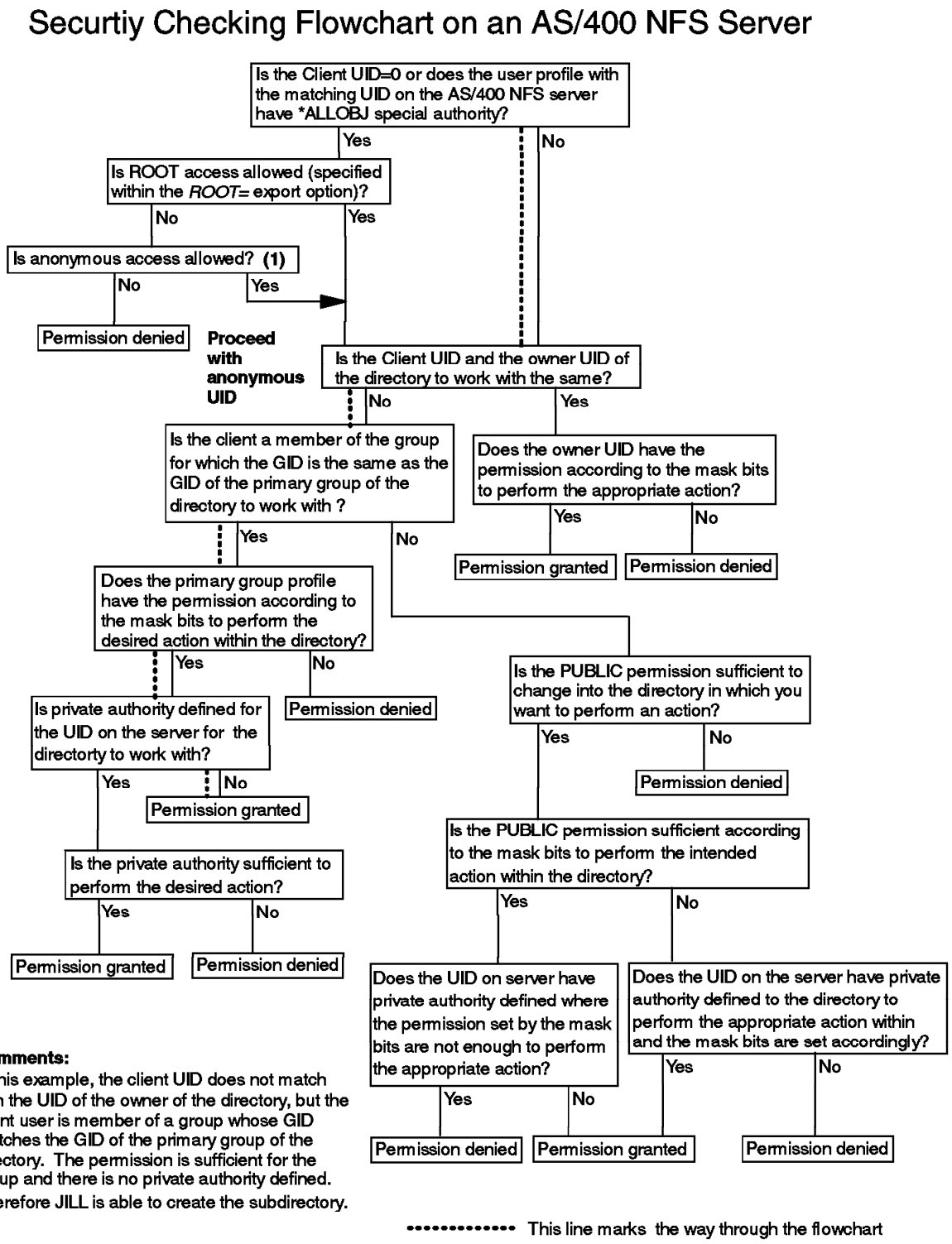


Figure 23. Security Checking Flowchart for Example 5

Example 6:

Consider the following situation:

User MIKE from the NFS client has mounted the exported directory */data/anni* from the AS/400 system ASSYS01 to mount point */mptsys01*. Mike wants to delete the file *expense* out of the directory */data/anni* with the command:

```
del /mptsys01/expense
```

Note: Please note that the user does not change into the directory. The following table shows the given settings for this example.

AS/400 NFS Server				
Systemname	ASSYS01			
Authority Information				
Parent Directory (X)	/data/anni			
Filename (Y)	expense			
Owner	BILL			
Primary Group	*NONE			
	Mask Bits for parent directory (X) / file (Y)			**Object authority**
User Profile	**R**	**W**	**X**	***OBJEXIST**
BILL	X / Y	X / Y	X / Y	X / Y
MIKE4	X / Y	X / Y	X / Y	X / -
*PUBLIC	X / Y	- / -	X / Y	X / Y
User Profile Information				
User Profile	**UID**	**GID**	***ALLOBJ**	**Group Profile**
BILL	123	*NONE		NFSGRP4
QSECOFR	0	*NONE	X	*NONE
NFSGRP4	3245	600		*NONE
MIKE4	255	*NONE		NFSGRP4
Export Options				
-i				
AIX NFS Client				
Systemname	RSSYS05			
Mount point	/mptsys01			
Client User	MIKE			
Client UID	255			
Member of Group	NFSGRP6			
GID	600			
Note: Export Options: -i means, defaults used.				

The Security Checking Flowchart for this example follows:

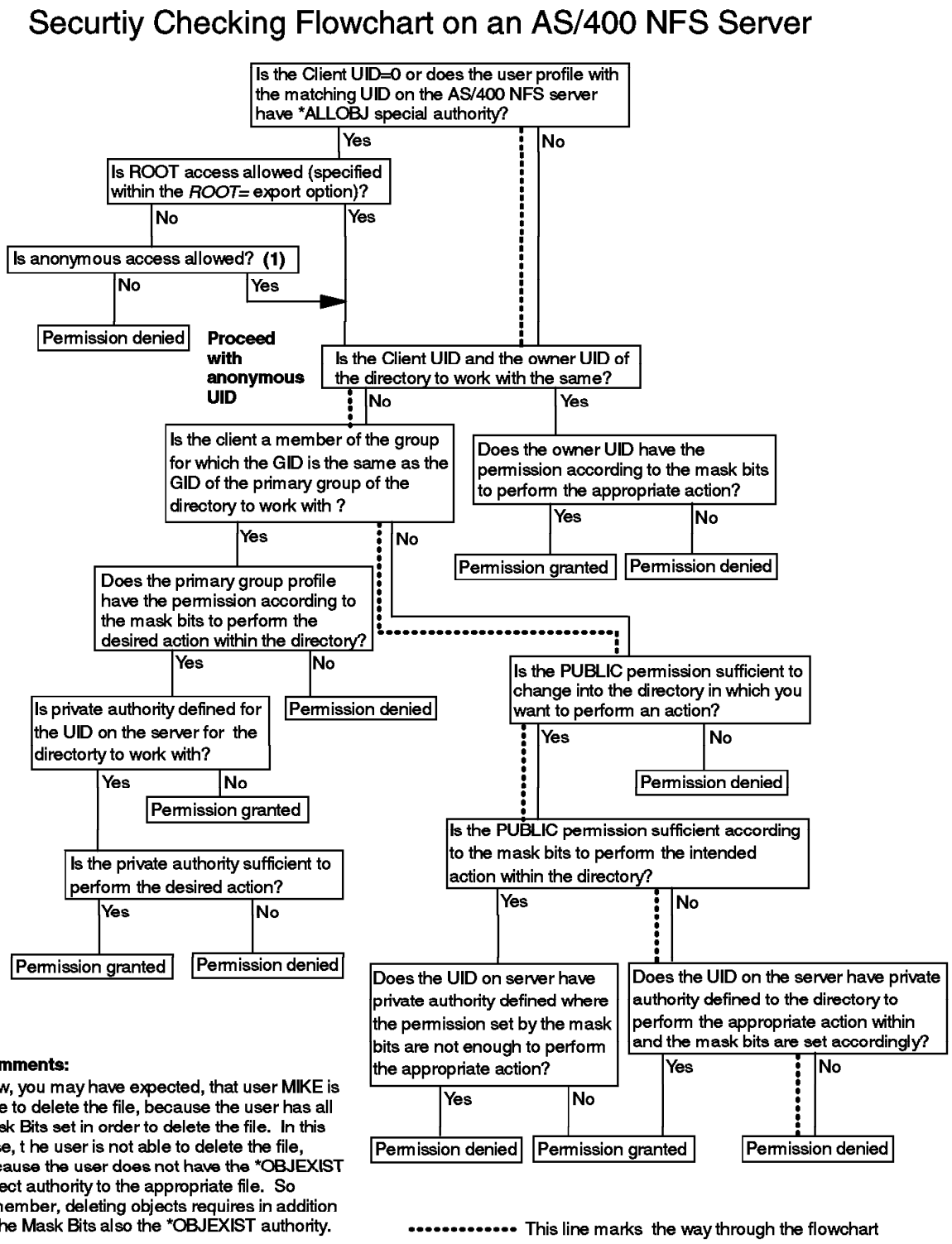


Figure 24. Security Checking Flowchart for Example 6

Chapter 5. National Language Support

This chapter discusses the National Language Support (NLS) within the Network File System Support (NFS) on the AS/400 system. This includes the AS/400 system as the NFS server and the client as well as an AIX system as an NFS client. For further details on the information given in this chapter, refer to the following publications.

- *AS/400 National Language Support V3R7*, SC41-4101
- *OS/400 National Language Support APIs V3R7*, SC41-4863
- *International Application Development*, SC41-4603
- *Character Data Representation Architecture* , SC09-2190

5.1 Introduction

The AS/400 system supports many national languages. You can work on an AS/400 system in the language of your choice. The AS/400 system also ensures that the data you send to and receive from the system appears in the form and order you expect.

All AS/400 systems use a common set of program code, regardless of which language you use on the system. For example, the program code on a U.S. English AS/400 system and the program code on a Spanish AS/400 system are identical. Different sets of textual data are used for different languages.

All characters used in one language are put into a kind of table called a code page. Within a code page, every character has an assigned binary value. When exchanging data between systems, these binary values are sent to or received from the partner system. Once these binary values (also known as a code point for a given character) are received on a system, the code page on this system is then used to translate the code point back to the character. Between systems using the same set of characters and the same code page, the translation usually works without problems.

OS/400 uses Extended Binary Coded Decimal Interchange Code (EBCDIC) encoding for storing the representation of a character into a code page. Each character is stored as an 8-bit binary (hexadecimal) value, which is a single byte. With this encoding, systems are referred to as single-byte character set (SBCS) systems. OS/400 also supports several other encoding schemes such as double-byte character set (DBCS), mixed-byte character set (MBCS), and so on

Other operating systems use different encoding. For example, DOS or AIX uses ASCII.

In a single byte, there are 256 possible values. Some of the values are reserved for control purposes, leaving 190 values to represent characters.

A code page is displayed as a matrix of rows and columns that assigns characters to special hexadecimal code points. The code page may vary from country to country. Each code page is given a unique number that is used as an identifier.

Appendix C, “NLS Code Page Example” on page 159 illustrates the U.S. English code page 37. Please refer to the *International Application Development*, SC41-4603, for additional illustrations of code pages supported on AS/400 systems.

NFS on the AS/400 system offers the full national language support to communicate with systems using code pages different from its own.

5.2 Code Pages within NFS

Because NFS is supported on different systems on multiple platforms, there are various places within OS/400 to specify when and how a translation takes place. The AS/400 system as an NFS server or client allows you to specify code pages for:

- Data files
- Path names

When using different code pages on the NFS client and the NFS server, you have to specify the correct path name code page or data file code page within the Add Mounted File System (MOUNT) command. If your system does not support the use of different code pages on mounted file systems, you can specify the data file and path name code page on the Export File System (EXPORTFS) command for each client.

5.2.1 Mounting File Systems

Under OS/400, the MOUNT command has two parameters related to NLS. You have to specify the code page that is used when processing path name information or data from the server.

For example, you store PC programs on your NFS server within an IFS directory called *PGM*. Now, you want to call a program, *Übung*, that has one German Umlaut (Ü) as a part of the file name. This means the program or data itself is in binary but the file name is in EBCDIC. So you have to distinguish between data and file names.

The following display shows the appropriate MOUNT command.

```
                    Add Mounted FS (MOUNT)

Type choices, press Enter.

Type of file system  . . . . . > *NFS          *NFS, *UDFS, *NETWARE
File system to mount . . . . . > 'assys01:/pgm'

Directory to mount over  . . . > '/mptsys01'

Mount options  . . . . . .   'rw,suid,retry=5,rsize=8096,wsize=8096
timeo=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


Code page:
  Data file code page  . . . .   *BINARY      1-32767, *ASCII, *JOBCCSID
  Path name code page  . . . .   273          1-32767, *ASCII, *JOBCCSID




                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

As shown on the preceding display, the code page parameter for data files is set to **BINARY* so data is not converted. For the path name code page parameter, we specified the value *273* (EBCDIC Austria, German).

In this example, we assume that the directory *PGM* is exported with the appropriate code page. Otherwise, the path name is not shown correctly. Within NFS, you have to be aware of matching export and mount code page parameters.

When leaving the default values in the code page parameter, the OS/400 uses the **BINARY* value as the data file code page and the **ASCII* value for the path name code page. The **ASCII* value uses the ASCII equivalent of the coded character set identifier (CCSID) associated with the job of the user performing the MOUNT command. This means if the NFS server uses also the **ASCII* value when exporting the directory and the default CCSID is different, you may run into problems.

We recommend (in a multilingual environment) that you always specify a code page value within the path name code page as well as in the data file code page when you do not use the **BINARY* value.

Attention

When using the AS/400 system as an NFS server and a client in a multilingual environment, specify an EBCDIC code page for the path name on both sides, the server and the client. The AS/400 system uses EBCDIC encoding and, therefore, there is no need to convert first to ASCII and back to EBCDIC.

5.2.2 Exporting File Systems

When exporting a file system or a part of it, the NFS server has to know which clients are connected to it. To ensure that all national characters are transmitted and displayed in the correct way and order on all clients regardless of which country they are located in, you have to specify the correct code page. Thinking of the entire data flow from the AS/400 NFS server down to the AS/400 NFS client and the attached workstations, the code page definitions must be consistent. The following display shows the NLS related parameter when exporting a file system.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . . . OPTIONS      > '-i'



Directory  . . . . . . . . . . . DIR          > '/pgm'


                           Additional Parameters

Host options:                     HOSTOPT
  Host name  . . . . . . . . . .              > ASSYS02
  Data file code page  . . . . .              > *BINARY
  Path name code page  . . . . .              > 037
  Force synchronous write  . . .                *SYNC

                          + for more values

                                                                      Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

When you use the same CCSIDs on the NFS server and the client, leave the **DFT* value within the Host name parameter. Then the export file system (EXPORTFS) command exports the desired directory with a data file code page of **BINARY* and the path name code page of **ASCII*.

But when you have clients that use other CCSIDs or other operating systems with different languages installed, you have to enter a separate entry within the HOSTOPT parameter for each host or a group of hosts.

It is always important to use the same code page on the EXPORTFS command on the NFS server and the MOUNT command on the NFS client. Using different code pages may cause the MOUNT command to fail with a message that the object could not be found on the client.

For example, you have a directory */pgms* with a subdirectory *greeting*. The following display shows the contents of the directory.

```
                    Work with Object Links

Directory  . . . . :   /pgms/greeting

Type options, press Enter.
  3=Copy   4=Remove   5=Next level   7=Rename   8=Display attributes
  11=Change current directory ...

Opt   Object link            Type     Attribute    Text
      solong                 STMF
      Grüße                  STMF
      sincer                 STMF









Parameters or command
===>                                                              Bottom
F3=Exit  F4=Prompt  F5=Refresh  F9=Retrieve  F12=Cancel  F17=Position to
F22=Display entire field           F23=More options
```

Now you export the directory using the following parameter.

```
                    Change NFS Export (EXPORTFS)

Type choices, press Enter.

NFS export options . . . . . > '-i'



Directory  . . . . . > '/pgms'


                           Additional Parameters

Host options:
  Host name  . . . . . .   ASSYSO2    Character value, *DFT
  Data file code page  . . . . .   *BINARY    1-32767, *BINARY, *ASCII
  Path name code page  . . . . .   273        1-32767, *ASCII, *JOBCCSID
  Force synchronous write  . . .   *SYNC      *SYNC, *ASYNC
              + for more values

                                                                  Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

On the NFS client, you mount the directory using the following command.

```
                    Add Mounted FS (MOUNT)

 Type choices, press Enter.

 Type of file system  . . . . . > *NFS      *NFS, *UDFS, *NETWARE
 File system to mount . . . . . > 'assys02:/pgms'

 Directory to mount over  . . . . > '/msys02'

 Mount options  . . . . . '  'rw,suid,retry=5,rsize=8096,wsize=809
=20,retrans=5,acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'


 Code page:
   Data file code page  . . . . . > *BINARY   1-32767, *ASCII, *JOBCCSID
   Path name code page  . . . . . > 37        1-32767, *ASCII, *JOBCCSID




 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

After you mount the directory, you get the following result.

```
                    Work with Object Links

 Directory  . . . . :   /msys02/pgms/greeting

 Type options, press Enter.
   3=Copy   4=Remove   5=Next level   7=Rename   8=Display attributes
   11=Change current directory ...

 Opt   Object link            Type     Attribute    Text
       solong                 STMF
       Gr}~e                  STMF
       sincer                 STMF









 Parameters or command
 ===>                                                          Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F9=Retrieve  F12=Cancel  F17=Position to
 F22=Display entire field          F23=More options
```

As you see from this example, the result is different than we expected. All German Umlauts are displayed wrong. When using different code pages on the server and the client, sometimes a directory or its entries is not shown at all.

In this example, we use the German code page (273) on the AS/400 NFS server and on the AS/400 NFS client, we use the U.S. code page (037). The following table shows you an abbreviated version of the two code pages.

Table 5. Portion of Some Characters within Code Page 273 and 037

Country	Code Page	Code Point							
		C7	D0	A1	85	C0	43	E0	FC
EBCDIC Germany/Austria	273	G	ü	ß	e	ä	{	Ö	]
EBCDIC U.S./Canada	037	G	}	~	e	{	ä		Ü

Most of the commonly-used characters are on the same code point location on both code pages. Therefore, most of the time, you may not have problems and sometimes you run into problems when not using the proper code page settings.

When exporting a file system on an AS/400 NFS server specifying the default HOSTOPT parameter, you can just specify the corresponding code page on the MOUNT command on the AS/400 NFS client. In this case, you need the same default job CCSID when mounting the file system.

5.2.3 NLS using an AIX Client

If you are using an AIX NFS client, you cannot specify a code page within the MOUNT command. Therefore, you have to specify the appropriate code page on the EXPORTFS command on the AS/400 NFS server. Usually on AIX systems, the code page ISO8859-1 is used, which refers to the code page number 819 (ISO 8859, Part 1 Latin Alphabet No. 1). So when you export a directory to an AIX NFS client, specify code page 819 for the path name code page. To avoid specifying each AIX NFS client on the EXPORTFS command, you can create groups of clients. For more details on how to use and create net groups, refer to Section 3.5.3, “Using the /etc/netgroup File” on page 60.

5.3 Character Conversion through the Network

In this section, we give you information about the flow from a workstation attached to the AS/400 NFS server through the AS/400 NFS client down to the workstation attached to it.

Assume that a workstation PC in Germany uses Personal Communications (PCOM) 5250 emulation. The 5250 session is configured to use code page 273 (EBCDIC Germany/Austria). The user is connected to the AS/400 NFS server and has a default job CCSID of 273. The user creates a file with the name *Grüße* within the directory *pgms*. This directory is exported with the path name code page 850 (IBM PC Multilingual) to the NFS client. The client uses the same ASCII code page 850 when mounting the directory. The workstation attached to the AS/400 NFS client also uses PCOM with a 5250 session. The code page on the emulation session is 037 (EBCDIC U.S.A./Canada), which the user has as its default job CCSID.

The following figure shows the way through the network with all of the code pages that are used:

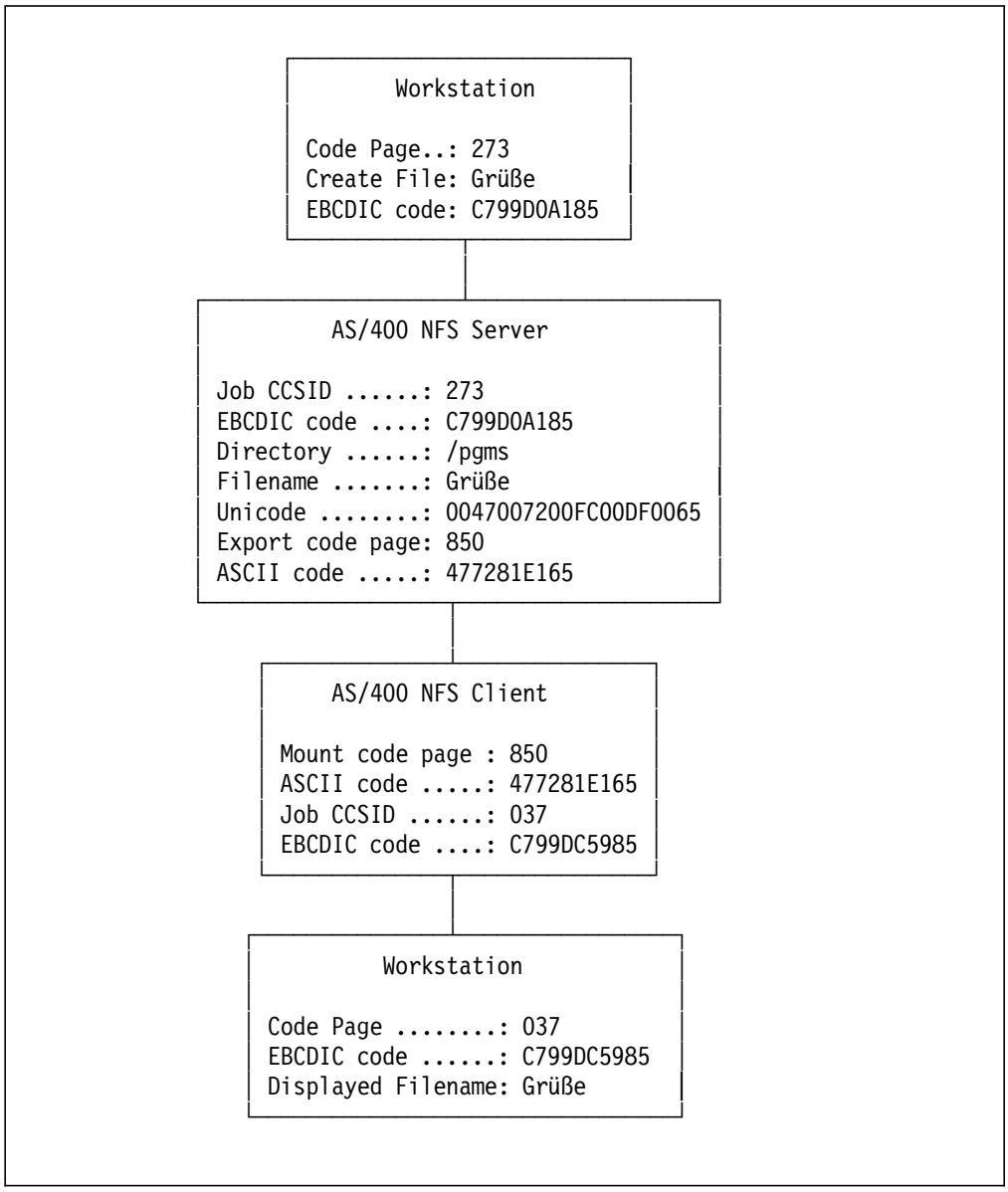


Figure 25. Flow through a Character Conversion

In the preceding figure, a user creates the file *Grüße* using the code page 273. On the AS/400 NFS server, the user is signed on using the default job CCSID 273. The filename is then stored as Unicode. The AS/400 system stores names not as EBCDIC values using standard code pages. Instead, it uses the Unicode page UCS2 Level 1 that is a subset of the ISO 10646 standard. This code page uses double bytes to encode each character.

The filename is translated into ASCII using code page 850. The AS/400 NFS client then converts the filename from ASCII into EBCDIC using a translation table Q850337037. Now the workstation PC attached to the NFS client shows the file name correctly.

For more information about code pages, refer to *AS/400 International Application Development V3R7*, SC41-4603.

5.4 Creating and Using Your Own Conversion Tables

In the previous sections, you have seen that many code pages may be involved when connecting different systems with different languages. The definition (which code page is used when exporting or mounting file systems within NFS) is done by code page numbers. But these code pages contain only code points (binary values) pointing to a character within this code page.

To map one character from one code page to the same character on another code page, the AS/400 system uses conversion tables. Sometimes conversion tables are also known as translation tables, but that is not correct.

There are no characters within a conversion table. Instead, this table contains mappings from a code point of one code page to the corresponding code point of another code page.

The MOUNT command and EXPORTFS command on the AS/400 system only support the definition of code pages and not conversion tables as used in other Internet Protocol (IP) applications such as TELNET or FTP. Therefore, the conversion uses only the system provided code pages and the appropriate conversion tables.

But sometimes you may need to convert one character to another specific character. For instance, you want to copy a stream file located in a mounted directory into a database file using the Copy to Stream File (CPYTOSTMF) command. While you are copying the data into the database file, you can convert some characters using your own conversion table.

Because you usually do not find examples on how to create conversion tables, we decided to put an example into this chapter. Most of the time, you can use an existing conversion table as a model.

Up to release V3R7, there is no command to retrieve the source of a conversion table.

Instead of displaying an existing conversion table and typing the information into a new source member to create your own conversion table, you can use the DMPOBJ command to get an almost usable output that you can copy and paste into a source member. The following description shows the tasks to be performed to create a conversion table.

Note: To follow this example, use an emulation session with 24x132 columns that supports the cut-and-paste function.

In our example, dump the Q850337037 conversion table using the following command.

```
DMPOBJ OBJ(Q850337037) OBJTYPE(*TBL)
```

The result is a spooled file. List your spooled files with the command:

```
WRKJOB
```

Select Option 4 to get the list of spooled files created during your session. Choose Option 5 beside the spooled file with the name *QPSRVDMP*. The following display is shown:

```
                    Display Spooled File
 File  . . . . . :   QPSRVDMP
 Control . . . . .
 Find  . . . . . .
 *...+....1....+....2....+....3....+....4....+....5....+....6....+....7..
 5716SS1 V3R7M0  961108                      AS/400 DUMP           003809
 DMPOBJ PARAMETERS
 OBJ- Q850337037                      CONTEXT- *LIBL
 OBJTYPE- *TBL
 OBJECT TYPE-           SPACE                                          *
 NAME-        Q850337037                      TYPE-          19   SUBTYPE
 LIBRARY-     QUSRSYS                         TYPE-          04   SUBTYPE
 CREATION-    06/14/96  15:43:12              SIZE-          0000001000
 OWNER-       QPGMR                           TYPE-          08   SUBTYPE
 ATTRIBUTES-          0800                    ADDRESS-       0CA0AE807D 0
 SPACE ATTRIBUTES-
   000000   00FFFF00 00000060 1906D8F8 F5F0F3F3   F7F0F3F7 40404040 40404
   000020   40404040 40404040 A0000000 00000000   00000F00 40000000 00000
   000040   00000000 00000000 24A6259C C9000400   00000000 00000000 00000
 SPACE-
   000000   00010203 372D2E2F 1605250B 0C0D0E0F   10111213 3C3D3226 18191
   000020   405A7F7B 5B6C5D7D 4D5D5C4E 6B604B61   F0F1F2F3 F4F5F6F7 F8F97
   000040   7CC1C2C3 C4C5C6C7 C8C9D1D2 D3D4D5D6   D7D8D9E2 E3E4E5E6 E7E8E
   000060   79818283 84858687 88899192 93949596   979899A2 A3A4A5A6 A7A8A

 F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

Scroll down with the “+5” control command to get to the part called *SPACE* from offset 000000 until 0000E0 onto the display.

```
                     Display Spooled File
 File  . . . . . :   QPSRVDMP
 Control . . . . .   +5
 Find  . . . . . .
 *...+....1....+....2....+....3....+....4....+....5....+....6....+....7..
 NAME-        Q850337037                      TYPE-          19   SUBTYPE
 LIBRARY-     QUSRSYS                         TYPE-          04   SUBTYPE
 CREATION-    06/14/96  15:43:12              SIZE-          0000001000
 OWNER-       QPGMR                           TYPE-          08   SUBTYPE
 ATTRIBUTES-          0800                    ADDRESS-       0CA0AE807D 0
SPACE ATTRIBUTES-
  000000   00FFFF00 00000060 1906D8F8 F5F0F3F3   F7F0F3F7 40404040 40404
  000020   40404040 40404040 A0000000 00000000   00000F00 40000000 00000
  000040   00000000 00000000 24A6259C C9000400   00000000 00000000 00000
SPACE-
  000000  *00010203 372D2E2F 1605250B 0C0D0E0F   10111213 3C3D3226 18191
  000020  *405A7F7B 5B6C5D7D 4D5D5C4E 6B604B61   F0F1F2F3 F4F5F6F7 F8F97
  000040  *7CC1C2C3 C4C5C6C7 C8C9D1D2 D3D4D5D6   D7D8D9E2 E3E4E5E6 E7E8E
  000060  *79818283 84858687 88899192 93949596   979899A2 A3A4A5A6 A7A8A
  000080  *68DC5142 43444748 52535457 56586367   719C9ECB CCCDDBDD DFECF
  0000A0  *4555CEDE 49699A9B ABAF5FB8 B7AA8A8B   2B2C0921 28656264 B4383
  0000C0  *22172906 202A4666 1A350839 36303A9F   8CAC7273 740A7576 77231
  0000C0  *22172906 202A4666 1A350839 36303A9F   8CAC7273 740A7576 77231
  0000E0  *EE59EBED CFEFA08E AEFEFBFD 8DADBCBE   CA8F1BB9 B6B5E19D 90BDB

 F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

In this example, the preceding display just holds 80 characters and, therefore, you do not see the entire contents shown on your 24x132 display.

Use the cut-and-paste function of your emulation to copy the area marked with "*." Ensure that you copy all eight columns, each represented by four bytes. This means you have to mark 32 bytes per row for eight rows.

Now create a source member for the new conversion table using the Program Development Manager (PDM). Create the member with the following parameters.

```
                    Start Source Entry Utility (STRSEU)

 Type choices, press Enter.

 Source file  . . . . . . . . . . > QTBLSRC       Name, *PRV
   Library  . . . . . . . . . . .>   THOMAS      Name, *LIBL, *CURLIB,
 Source member  . . . . . . . . .  Newtable      Name, *PRV, *SELECT
 Source type  . . . . . . . . . .  TBL           Name, *SAME, BAS, BASP
 Text 'description' . . . . . . .  This is my changed conversion table

                                                                    Bottom
 F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
 F24=More keys
```

The following display is shown:

```
 Columns . . . :    1  80                                      Edit
 SEU==>
 FMT **  ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+
        *************** Beginning of data ********************************
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
,,,,,,,
        ***************** End of data *********************************

 F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F10=Cursor   F11=Toggle
 F16=Repeat find       F17=Repeat change          F24=More keys
Member NEWTABLE added to file THOMAS/QTBLSRC.
```

Move the cursor to the first position within the EDIT display and paste the table from the DMPOBJ spooled file into it.

```
 Columns . . . :    1  80                                        Edit
 SEU==>
 FMT **  ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+
        ********** Beginning of data ********************************
'''''' 00010203 372D2E2F 1605250B 0C0D0E0F   10111213 3C3D3226 18191C27
'''''' 405A7F7B 5B6C5D7D 4D5D5C4E 6B604B61   F0F1F2F3 F4F5F6F7 F8F97A5E
'''''' 7CC1C2C3 C4C5C6C7 C8C9D1D2 D3D4D5D6   D7D8D9E2 E3E4E5E6 E7E8E9BA
'''''' 79818283 84858687 88899192 93949596   979899A2 A3A4A5A6 A7A8A9C0
'''''' 68DC5142 43444748 52535457 56586367   719C9ECB CCCDDBDD DFECFC70
'''''' 4555CEDE 49699A9B ABAF5FB8 B7AA8A8B   2B2C0921 28656264 B4383134
'''''' 22172906 202A4666 1A350839 36303A9F   8CAC7273 740A7576 77231514
''''''  EE59EBED CFEFA08E AEFEFBFD 8DADBCBE   CA8F1BB9 B6B5E19D 90BDB3DA
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
        ********** End of data **********************************

 F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F10=Cursor   F11=Toggle
 F16=Repeat find       F17=Repeat change          F24=More keys
Member NEWTABLE added to file THOMAS/QTBLSRC.
```

Note: On your display, you see eight columns instead of the seven shown on the previous display.

Remove all SPACES out of the table. The following display is shown:

```
 Columns . . . :    1  80                                        Edit
 SEU==>
 FMT **  ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+
        ********** Beginning of data ********************************
'''''' 00010203372D2E2F1605250B0C0D0E0F101112133C3D322618191C273F1D1E1F
'''''' 405A7F7B5B6C5D7D4D5D5C4E6B604B61F0F1F2F3F4F5F6F7F8F97A5E4C7E6E6F
'''''' 7CC1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9BAE0BBBC6D
'''''' 798182838485868788899192939495969798 99A2A3A4A5A6A7A8A9C04FD0A1FF
'''''' 68DC5142434447485253545756586367719C9ECBCCCDDBDDDFECFC70B180BF07
'''''' 4555CEDE49699A9BABAF5FB8B7AA8A8B2B2C092128656264B4383134334AB224
'''''' 22172906202A46661A35083936303A9F8CAC7273740A757677231514046A783B
''''''  EE59EBEDCFEFA08EAEFEFBFD8DADBCBECA8F1BB9B6B5E19D90BDB3DAFAEA3E41
'''''' 0000404040404040404040404040404040404040404040404040404040404040
'''''' 4040404040404040404040404040404040404040404040404040404040404040
''''''
''''''
''''''
''''''
''''''
''''''
''''''
        ********** End of data **********************************

 F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F10=Cursor   F11=Toggle
 F16=Repeat find       F17=Repeat change          F24=More keys
Member NEWTABLE added to file THOMAS/QTBLSRC.
```

Now you have the source of the Q850337037 conversion table. It consists of eight rows by 64 columns, which is 256 bytes.

You have to interpret a conversion table in a similar way as a code page. The table has 32 bytes per row. The first row starts with 00 to 1F, the second 20 to 3F, the third 40 to 5F, and so on.

The following figure shows part of the table just created on the previous display with some explanations.

```
00  02  04  06  08  0A  0C  0E  10  12  14  16  18  1A
  01  03  05  07  09  0B  0D  0F  11  13  15  17  19   ...
000102033 72D2E2F1605250B0C0D0E0F101112133C3D322618191C2

20  22  24  26  28  2A  2C  2E  30  32  34  36  38  3A
  21  23  25  27  29  2B  2D  2F  31  33  35  37  39   ... continuous
405A7F7B5B6C507D4D5D5C4E6B604B61F0F1F2F3F4F5F6F7F8F97A5

40  42  44  46  48  4A  4C  4E  50  52  54  56  58  5A
  41  43  45  47  49  4B  4D  4F  51  53  55  57  59   ...
7CC1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9B

60  62  64  66  68  6A  6C  6E  70  72  74  76  78  7A
  61  63  65  67  69  6B  6D  6F  71  73  75  77  79   ...
7981828384858687888991929394959679899A2A3A4A5A6A7A8A9C
68DC51424344474852535457565863677 19C9ECBCCCDDBDDDFECFC7
4555CEDE49699A9BABAF5FB8B7AA8A8B2B2C092128656264B438313
22172906202A46661A35083936303A9F8CAC7273740A75767723151
EE59EBEDCFEFA08EAEFEFBFD8DADBCBECA8F1BB9B6B5E19D90BDB3D
000040404040404040404040404040404040404040404040404040
404040404040404040404040404040404040
```

Figure 26. Part of the Conversion Table used in This Example

To better understand the structure of a conversion table, there is an extract of the code page 850 and 037 in the following table.

Table 6. Extract of Some Characters within Code Page 850 and 037

Country	Code Page	Character / Code Point's							
		T	h	e	SP	W	a	l	l
ASCII Multilingual	850	54	68	65	20	57	61	6C	6C
EBCDIC U.S.A./Canada	037	E3	88	85	40	E6	81	93	93
Note: SP = SPACE									

In the previous table, you find the words “The Wall.” Together with this table and the information from Figure 26, you can explore how to modify a conversion table.

Because the data passed to the system is in ASCII using code page 850, you have to use the code point of each character to locate the appropriate code point for the corresponding EBCDIC code page 037.

Example 1:

The “T” has a code point of “54” within code page 850. Go into the code page to the third row and locate the value given on position 54. This is the appropriate code point within the EBCDIC code page 037. In this case, the code point is E3.

Example 2:

The “a” has a code point of “61” within the code page 850. Go into the code page to the forth row and locate the value given on position 61. This is the appropriate code point within the EBCDIC code page 037. In this case, the code point is 81.

Now using the code point “81,” you find the character “a” in the code page 037.

Assume you want all lower case characters to be converted into upper case characters. You have to go into code page 037 and locate the code points for the appropriate upper case characters. For example, upper case “A” has the code point “C1” and upper case “L” has the code point “D3.”

Edit the source of your new conversion table and locate code point 61 and code point 6C. Change the values within these locations to the new values. The following display shows the changed table entries.

```
 Columns . . . :    1  80                                        Edit
 SEU==>
 FMT **  ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+
        *************** Beginning of data ********************************
’’’’’’’ 00010203372D2E2F1605250B0C0D0E0F101112133C3D322618191C273F1D1E1F
’’’’’’’ 405A7F7B5B6C507D4D5D5C4E6B604B61F0F1F2F3F4F5F6F7F8F97A5E4C7E6E6F
’’’’’’’ 7CC1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9BAE0BBB06D
          **                    **
’’’’’’’*79C1828384858687888991920394959697989 9A2A3A4A5A6A7A8A9C04FD0A1FF
’’’’’’’ 68DC5142434447485253545756586367719C9ECBCCCDDBDDDFECFC70B180BF07
’’’’’’’ 4555CEDE49699A9BABAF5FB8B7AA8A8B2B2C092128656264B4383134334AB224
’’’’’’’ 22172906202A46661A35083936303A9F8CAC7273740A757677231514046A783B
’’’’’’’  EE59EBEDCFEFA08EAEFEFBFD8DADBCBECA8F1BB9B6B5E19D90BDB3DAFAEA3E41
’’’’’’’ 0000404040404040404040404040404040404040404040404040404040404040
’’’’’’’ 4040404040404040404040404040404040404040404040404040404040404040
’’’’’’’

        ****************** End of data *********************************

 F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F10=Cursor   F11=Toggle
 F16=Repeat find       F17=Repeat change          F24=More keys
Member NEWTABLE added to file THOMAS/QTBLSRC.
```

In the previous display, the changed values are marked with an “*.” Save the source member and select Option 14 to create a conversion table as shown in the following display. The command used within option 14 is the Create Table (CRTTBL) command.

```
                    Work with Members Using PDM                ASSYS02

File  . . . . . .   QTBLSRC
  Library . . . .     THOMAS              Position to  . . . . .

Type options, press Enter.
 2=Edit         3=Copy  4=Delete 5=Display       6=Print     7=Rename
 8=Display description  9=Save  13=Change text  14=Compile  15=Create mo

Opt  Member      Type        Text
14   NEWTABLE    TBL         This is my new conversion table









Parameters or command
===>
F3=Exit         F4=Prompt            F5=Refresh           F6=Create
F9=Retrieve     F10=Command entry    F23=More options     F24=More keys
```

Using your new conversion table, you get the following result when passing the words “The Wall” to your conversion table.

```
The WALL
```

With the Work with Tables (WRKTBL) command, you can display your conversion table. The following display shows the conversion table at the location of the changed values.

```
                    Display Conversion Table

Table:   NEWTABLE       Library:   THOMAS

 Hex    Hex                Hex    Hex                Hex    Hex
Input  Output             Input  Output             Input  Output
 5A      E9                69      89                78      A7
 5B      BA                6A      91                79      A8
 5C      E0                6B      92                7A      A9
 5D      BB               *6C      D3                7B      C0
 5E      B0                6D      94                7C      4F
 5F      6D                6E      95                7D      D0
 60      79                6F      96                7E      A1
*61      C1                70      97                7F      FF
 62      82                71      98                80      68
 63      83                72      99                81      DC
 64      84                73      A2                82      51
 65      85                74      A3                83      42
 66      86                75      A4                84      43
 67      87                76      A5                85      44
 68      88                77      A6                86      47
                                                               More...
F3=Exit   F12=Cancel   F17=Position to
```

We marked the changes with an asterisk (*). For further details about conversion, refer to the publication *AS/400 National Language Support V3R7*, SC41-4101.

Chapter 6. Migrating from FSS/400 to NFS

This chapter discusses the migration steps from File Server Support/400 (FSS/400) to the Network File System Support (NFS) on V3R7. It starts with a brief overview about FSS/400. Then it covers the differences between FSS/400 and NFS. Finally you find some information about the migration steps. The following list shows publications that give you more information about FSS/400 and NFS.

- *TCP/IP File Server Support for OS/400 Installation and User's Guide*, SC41-0125
- *OS/400 Network File System Support V3*, SC41-4714

6.1 Overview

FSS/400 was first announced for V2R2 and is the former product of NFS that supports a subset of the functions available with NFS. It has only the NFS server support implemented. FSS/400 allows only the export of the QSYS.LIB or QDLS file system even if you use it on an OS/400 Version 3 system that also supports additional file systems.

Since NFS was announced as a base operating system function of V3R7, the AS/400 system supports the NFS server as well as the NFS client.

Users changing their release to V3R7 have to perform a couple of migration steps to use the configuration defined within a previous release. However, there is no migration tool that automatically performs the migration from FSS/400 to NFS. The user has to do the migration manually.

The following sections help you to understand the differences between FSS/400 and NFS.

6.2 Difference between FSS/400 and NFS V3R7

Since FSS/400 is a product that uses the authorization checking used on AIX/UNIX platforms, it has to integrate the user identification (UID) and group identification (GID) concept into an operating system that, by default, does not support this function. This is done by a mapping table called Authorized User Table. This table allows the user to assign UIDs to an AS/400 system user profile. GIDs are not supported within FSS/400.

Within OS/400 V3R1, UIDs and GIDs are integrated into the user profile as parameters. They are:

- User ID number (UID)
- Group ID number (GID)

FSS/400 as well as NFS on OS/400 V3R7 has default user profiles used for anonymous or root access. The root access user is the Q7FSOWN user profile within FSS/400 and the QSECOFR user profile within V3R7. The QSECOFR user profile has a UID of 0.

The default user for anonymous access on FSS/400 is the Q7FSUSER user profile. In OS/400 V3R7, it is the QNFSANON user profile that has a UID of 4294770710.

The second major difference is the export function within the NFS server. FSS/400 allows you to only export the QSYS.LIB and QDLS file system. For exporting a file system, FSS/400 uses an Export Table that has five parameters. These are:

Path This parameter specifies the name of the directory you want to export. The NFS client is able to mount this directory or any subdirectory of it.

Client This parameter specifies which client system is allowed to mount the specified directory. A client system is specified by its IP host name that resides in the TCP/IP host table or a name server.

Write This parameter allows you to specify if the client has the permission to perform write operations within the exported directory or just read operations.

Root By setting this parameter to *YES, you allow a client UID of 0 (ROOT) to access the exported directory. This client user is then assigned to the user profile within the Authorized User Table with the UID of *ROOT.

User ID Here you specify the UID to be used when an undefined user mounts the exported file system. This user is also known as an anonymous user.

Exporting a file system or directory within OS/400 V3R7 is done by the export file system (EXPORTFS) command or by adding export entries into the /etc/exports file.

The EXPORTFS command offers you more flexibility when exporting directories. It allows you to specify an export option parameter string that is used to control client access in different ways. There is also a parameter that lets you specify different code pages for each client or a group of clients. For details on the EXPORTFS command, refer to Section 3.3, “Exporting File Systems” on page 32.

6.3 Performing the Migration

As written in the overview of this chapter, you know that there is no tool that automatically does the migration from FSS/400 to NFS on OS/400 V3R7. Instead, you have to perform these steps manually. This section is divided into two subsections, which are:

- Migrating the Security
- Migrating the Exported File Systems

Follow these sections to complete the migration.

6.3.1 Migrating the Security

The migration of the Authorized User Table from FSS/400 is not an easy task. Usually, you assign a UID unique to a user throughout the entire network.

For instance, you have an employee with the name Jesse Smith. You assign a UID of 731 to this employee, which is unique throughout your network. On every system independent of the platform, Jesse Smith is known under the UID of 731. Also on the AS/400 system using FSS/400, this user has this UID. Now you want to change the release to V3R7. What happens during the release change is that every user profile gets a UID assigned to it. These UIDs are generated automatically. You cannot prevent the AS/400 system from performing this task.

After the release change, every user has a UID but probably not the UID that was assigned earlier within the authorized user table.

So you have to change the UIDs after the release change manually. The following list shows the migration steps.

1. Print the authorized user table:

   First you need the user profile/UID mapping information to perform the UID assignment to the V3R7 user profiles. Use the Work with FSS/400 Users (WRKFSSUSR) command to work with the authorized user table. The following display is shown:

```
                      Work with Authorized Users

Type options, press Enter.
  1=Add   2=Change   4=Remove

         User            User
Opt      Profile          ID

         QUSER            305
         Q7FSOWN         *ROOT
         Q7FSUSER          -2
         SMITH            761
         THOMAS           303









                                                                    Bottom
Parameter for option 2 or command
===>

F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F12=Cancel
```

   If you have just a few users registered within the table, make a hardcopy of it. When you have defined more users, you can print the contents of the physical file where this information is stored using the following command:

   ```
   CPYF FROMFILE(QUSRSYS/QA7FSUSRPF) TOFILE(QSYSPRT)
   ```

   The previous command creates a spooled file containing all user profiles with their associated UIDs. If the spooled file does not contain all of your defined users, try to print the file again using the library QSYS2924 for the file QA7FSUSRPF.

   Keep this information for later use.

2. Assign UIDs from FSS/400 to user profiles on OS/400 V3R7:

Now you have to change the UID parameter within each user profile you want to assign a specific UID. Use the Change User Profile (CHGUSRPRF) command to assign the UID.

```
CHGUSRPRF USRPRF(SMITH) UID(761)
```

For further information about UIDs and how to assign UIDs to a user profile, refer to Chapter 4, “NFS Security” on page 77.

If you get the following message, the UID you want to assign is already assigned to another user.

```
                        Additional Message Information

Message ID . . . . . : CPF22CE     Severity . . . . . : 40
Message type . . . . . : Diagnostic
Date sent  . . . . . : 09/25/97    Time sent  . . . . . : 11:52:33

Message . . . . : The UID value 761 is used by another user profile.
Cause . . . . . : The specified user profile was not created or changed
 because the UID value specified was previously assigned to another user
 profile.
Recovery  . . . : Change the UID value and try the request again.









                                                                    Bottom
Press Enter to continue.

F3=Exit   F6=Print   F9=Display message details
F10=Display messages in job log  F12=Cancel  F21=Select assistance level
```

In this case, you have to find out which user profile currently has this UID assigned to it by using the Display User Profile (DSPUSRPRF) command to display every user profile, or perform a DSPUSRPRF *ALL to an output file and run a query on it to get a list containing the user profiles and their associated UIDs.

Once you find the appropriate user profile, assign another UID to this user profile and try to change your desired user profile again.

The second error message that might appear comes up when the user profile you want to change owns IFS objects. On the following display, you see this message.

```
                    Additional Message Information

Message ID . . . . . : CPF22DC     Severity . . . . . : 40
Message type . . . . . : Diagnostic
Date sent  . . . . . : 09/25/97    Time sent  . . . . . : 13:11:12

Message . . . . : Not allowed to change UID of the user profile.
Cause . . . . . : The UID of a user profile may not be changed when the
  profile is the owner of an object in a directory.
Recovery  . . . : Change the value of the UID parameter to *SAME and try
  the command again.









                                                                    Bottom
Press Enter to continue.

F3=Exit   F6=Print   F9=Display message details
F10=Display messages in job log  F12=Cancel  F21=Select assistance level
```

If you get the previous message, use the QSYCHGID application programming interface (API) to change the UID. This API allows you to change UIDs and GIDs regardless if the user profile owns IFS objects or not.

For further information about the QSYCHGID API, refer to Section 4.2.3, “Changing UIDs or GIDs for User Profiles Owning Objects” on page 81.

After you have assigned all UIDs used within FSS/400 to the user profiles within OS/400 V3R7, you have finished the migration of the authorized user table.

6.3.2 Migrating the Exported File Systems

As well as the migration of UIDs, the migration of the exported file systems requires manual intervention. To migrate the export table from FSS/400 to the /etc/exports file within OS/400 V3R7, perform the following steps.

1. Collect information about the exported file systems in FSS/400:

   The Work with Export Table (WRKFSSEXT) command shows you the current configuration of exported files systems. The following display shows you the export table.

```
                    Work with Export Table

Type options, press Enter.
  1=Add   2=Change   4=Remove   5=Display

                                                                    User
Opt  Path                                  Client        Write  Root   ID

     /QDLS/DOCS                            *ALL          *NO    *NO     37
     /QDLS/QRUMBA                          *ALL          *NO    *NO     -2
     /QSYS.LIB/QGPL.LIB                    ASSYS02       *YES   *YES    -2
     /QSYS.LIB/QUSRSYS.LIB                 RSSYS01       *NO    *YES    33
     /QSYS.LIB/THOMAS.LIB/LNG.FILE         ASSYS87       *YES   *NO     -2









                                                                    Bottom
F3=Exit  F5=Refresh  F11=Display additional path information  F12=Cancel
```

The data from the export table is stored in three different files. These are:

QA7FSCLIPF This file contains the client host names specified for the exported directories.

QA7FSEXTPF The parameters for *Write, Root and User ID* are stored in this file as well as references to the other two files mentioned here.

QA7FSPTHPF The exported path names are stored in this file. The reference to the path names is specified in the QA7FSEXTPF file.

Because the data is split into three files, it is easier to print every entry within the export table using the WRKFSSEXT command and make hard copies.

Keep this information for the next migration step.

2. Add directories to the /etc/exports file:

   Once you have the information about the exported directories from FSS/400, add this information to the /etc/exports stream file. In this section, we use the Edit File (EDTF) command to edit the *exports* file.

   Note: By default, the EDTF command is not part of the base operating system in V3R7. This command is provided by a PTF. The PTF number for V3R7 is SF38832. Refer to Appendix D, “Installation of the Edit File (EDTF) command” on page 161 for further information about the installation of this PTF and the support of the EDTF command.

   You can also use Client Access/400 to access the /etc/exports file and perform the changes.

   To go into the Edit display, enter the following command:

   ```
   EDTF '/etc/exports'
   ```

   The following display is shown:

```
 Edit File: /etc/exports
 Record . :       1 of     150 by  10          Column: 1 of 70 by 126
 Control  :

CMD ....+....1....+....2....+....3....+....4....+....5....+....6....+....
  ****************** Beginning of data ********************
#########################################################################
#  /etc/exports                                                         #
#                                                                       #
#  This file contains the paths of the directory trees that you wish #
#  to export to the rest of the world via the NFS/400 server.        #
#  There are several restrictions on the format of the entries.      #
#  They are:                                                            #
#                                                                       #
#  1. All entries are ended by the end-of-line character.               #
#                                                                       #
#  2. The continuation character '\' can be used to extend one       #
#     entry across multiple lines when it is the last character      #
#     in the line.                                                      #
#                                                                       #

 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

Scroll to the end and add one entry for each directory to be exported. The following examples show how to specify an exported directory from FSS/400 within the /etc/exports file.

Example 1:

Export definitions in FSS/400:

Path	/QSYS.LIB/THOMAS.LIB/LNG.FILE
Client	ASSYS87
Write	*YES
Root	*NO
User ID	-2

Export definitions within the /etc/exportfs file:

The following example shows you the appropriate entry within the /etc/exports file for the migrated directory:

```
/QSYS.LIB/THOMAS.LIB/LNG.FILE access=assys87
```

By default, all clients that have access to the exported directory have Write authority. Therefore, Write authority must not be specified. In V3R7, all anonymous requests are mapped to the QNFSANON user so you do not need to specify this parameter as well. With the *access* option, you define the client host name that has the permission to mount the exported directory.

Example 2:

Export definitions in FSS/400:

Path	/QSYS.LIB/QUSRSYS.LIB
Client	RSSYS01
Write	*NO
Root	*YES
User ID	33

Export definitions within the /etc/exportfs file:

The following example shows the appropriate entry within the /etc/exports file for the migrated directory:

```
/QSYS.LIB/QUSRSYS.LIB access=rssys01,RO,root=rssys01,anon=33
```

In this example, the *RO* value represents the Write parameter in FSS/400 and means that only read access is permitted. Within the *root* option, specify the client host name or host names where root users are mapped to the QSECOFR user profile on the AS/400 system.

Example 3:

Export definitions in FSS/400:

Path	/QDLS/DOCS
Client	*ALL
Write	*NO
Root	*NO
User ID	37

Export definitions within the /etc/exportfs file:

The following example shows you the appropriate entry within the /etc/exports file for the migrated directory:

```
/QDLS/DOCS RO,anon=33
```

By not specifying the *access* option, all clients can mount the exported directory. In this example, you just have to specify the *RO* option to allow only read operations and the *anon* option to map anonymous user requests to the user profile with the assigned UID of 37.

The following display shows you the /etc/exports file with all definitions made in the previous examples.

```
 Edit File: /etc/exports
 Record . :     140 of     153 by  10         Column: 1 of 70  by  126
 Control  :

CMD ....+....1....+....2....+....3....+....4....+....5....+....6....+....
   #  DataFileCodePage=367                                                #
   #                                                                      #
   #  A user exports /qsys.lib/jon.lib to the netgroup dept-OA5.  A       #
   #  codepage is specified for data files.  Because the directory        #
   #  exists in the QSYS.LIB file system, NFS will use the open() API     #
   #  with the O_TEXTDATA and O_CODEPAGE options to work with data        #
   #  files.  Note that wheatley will use these options only if it is     #
   #  defined as a member of the netgroup dept-OA5.                       #
   #                                                                      #
   #                                                                      #
   ########################################################################
   /QSYS.LIB/THOMAS.LIB/LNG.FILE  access=assys87
   /QSYS.LIB/QUSRSYS.LIB  access=rssys01,RO,root=rssys01,anon=33
   /QDLS/DOCS  RO,anon=33
    ****************** End of data *************************

 F2=Save  F3=Save/Exit  F10=Left  F11=Right  F12=Cancel  F16=Find
 F17=Change  F15=Services
```

This concludes the migration from FSS/400 to NFS. The /etc/exports file gives you a couple of more options and parameters you can specify for the directories to be exported. For more information about the /etc/exports file and its structure, refer to Section 3.5.2, “Using the /etc/exports File” on page 54.

Chapter 7. Common Problems Encountered in NFS

This chapter gives you a set of problems that are usually encountered while using NFS. The problems encountered are mainly in the areas of NFS operations and NFS security. We first discuss the scenario for each of the problems, the cause of the problem, and later the solution for each of the problems.

7.1 Typical Problems in Startup of NFS Server

1. Problems with authorities required:

   For starting the NFS server on the AS/400 system, the command is:

   ```
   STRNFSSVR SERVER(*ALL)
   ```

   The NFS manual for OS/400 states that you need only *IOSYSCFG authority to run this command successfully. So we use a profile that belongs to *SECADM user class that has *IOSYSCFG authority. The following display is shown.

```
MAIN                AS/400 Main Menu
                                                    System:   ASSYS01
Select one of the following:

     1. User tasks
     2. Office tasks
     3. General system tasks
     4. Files, libraries, and folders
     5. Programming
     6. Communications
     7. Define or change the system
     8. Problem handling
     9. Display a menu
    10. Information Assistant options
    11. Client Access/400 tasks

    90. Sign off

Selection or command
===> strnfssvr *all

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F13=Information Assista
F23=Set initial menu
Not authorized to command STRNFSSVR in library *LIBL.
```

   The reason for this error is that with only *IOSYSCFG authority, you do not have authority to the command itself. Use the Edit Object Authority (EDTOBJAUT) command to change the permission of the user that starts the NFS server as *USE.

```
                          Edit Object Authority

Object . . . . . :   STRNFSSVR       Owner  . . . . . :   QSYS
  Library  . . . . :     QSYS         Primary group  . . :   *NONE
Object type  . . . :   *CMD

Type changes to current authorities, press Enter.

   Object secured by authorization list  . . . . . .   *NONE

                         Object
 User        Group       Authority
 QSYS                    *ALL
 *PUBLIC                 *EXCLUDE




 F3=Exit   F5=Refresh   F6=Add new users  F10=Grant with reference object
 F11=Display detail object authorities   F12=Cancel F17=Top   F18=Bottom
 (C) COPYRIGHT IBM CORP. 1980, 1996.
```

Try the same command again:

```
STRNFSSVR SERVER(*ALL)
```

Now we encounter another problem as shown in the following display.

```
                     Command Entry                ASSYS01
                                                  Request level:4
 Previous commands and messages:
    > strnfssvr *all
      File system error occurred.









 Type command, press Enter.
 ===> strnfssvr *all

  F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
  F11=Display full   F12=Cancel  F13=Information Assistant  F24=More Keys
```

Select option F10 to get the detailed message. The following information is shown.

```
                  Display All Messages
                                                      System:   ASSYS01
Job . . :   QPADEV0007    User . . :   TESTUSER   Number . . . :003857

5 > strnfssvr *all
    User not enrolled in system distribution directory.
    File system error occurred.
5 > dspjoblog









                                                              Bottom
Press Enter to continue.

F3=Exit   F5=Refresh   F12=Cancel   F17=Top   F18=Bottom
```

The problem is that the user has not been enrolled in the system distribution directory. Add the user in the system distribution directory using the Add Directory Entry (ADDDIRE) command.

```
                    Add Directory Entry (ADDDIRE)

Type choices, press Enter.

User identifier:
  User ID  . . . . . . . . . . . > TESTUSER      Character value
  Address  . . . . . . . . . . . > ASSYS01       Character value
User description . . . . . . . . > 'Test User Profile'

User profile . . . . . . . . . . > TESTUSER      Name, *NONE
System name:
  System name  . . . . . . . . .   *LCL          Character value, *LCL,
  System group . . . . . . . . .                 Character value
Network user ID  . . . . . . . .   *USRID

Last name  . . . . . . . . . . .   *NONE

First name . . . . . . . . . . .   *NONE
Middle name  . . . . . . . . . .   *NONE
Preferred name . . . . . . . . .   *NONE
                                                                  More...
F3=Exit   F4=Prompt   F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display       F24=More keys
```

Retry the command:

```
STRNFSSVR SERVER(*ALL)
```

```
                    Command Entry                 ASSYS01
                                                  Request level:4
Previous commands and messages:
   > strnfssvr *all
     Start NFS server command completed successfully.









Type command, press Enter.
===>

 F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
 F11=Display full   F12=Cancel  F13=Information Assistant  F24=More Keys
```

Now you should be able to start the servers. If you are unable to do so, and if you get some authority problems, check the PTFs applied to your system and make sure that you have all the necessary PTFs applied. For the required PTFs, refer to Section 2.3, “Lab Environment for this Residency” on page 23.

2. Problems due to improper configuration:

   One of the reasons for the *strnfssvr *all* command to fail is that TCP/IP is not configured properly. Since NFS uses the system configuration to determine the local system name, there is a problem if a name cannot be determined. There needs to be an entry in the hosts table for both the “long” and the “short” versions of the system name. The long version can be found by using option 12 on the CFGTCP menu shown in the following figure.

```
CFGTCP              Configure TCP/IP
                                                 System:  ASSYS01
Select one of the following:

     1. Work with TCP/IP interfaces
     2. Work with TCP/IP routes
     3. Change TCP/IP attributes
     4. Work with TCP/IP port restrictions
     5. Work with TCP/IP remote system information

    10. Work with TCP/IP host table entries
    11. Merge TCP/IP host table
    12. Change local domain and host names
    13. Change remote name server

    20. Configure TCP/IP applications
    21. Configure related tables
    22. Configure point-to-point TCP/IP

Selection or command
===>

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel
```

Using Option 12 on this menu gives you the following display.

```
              Change Local Domain and Host Names
                                                      System:  ASSYS01
Type choices, press Enter.

  Local domain name . . .   nfsland.usa.com


  Local host name . . . .   assys01

                                                                Bottom
F3=Exit   F12=Cancel
```

The “LOCAL HOST NAME” is considered the short name. The “LOCAL DOMAIN NAME” appended to the “LOCAL HOST NAME” with a dot (.) becomes the long name. Both the short and the long names need to be in the local system′s host table (option 10 in CFGTCP) unless they are using a domain name server (DNS). You can determine that they are using a domain name server by checking option 13 of CFGTCP. If there are DNS addresses there, they are using a DNS. The following display is shown when you take option 13.

```
                  Change Remote Name Server
                                                 System: ASSYS01
Type choices, press Enter.

  Server address  . . . .   8.2.103.56            Internet address
                            8.2.103.55

  Server port . . . . . .   53                    1-65535
  Server protocol . . . .   *UDP                  *UDP, *TCP
  Retries . . . . . . . .   2                     1-99
  Retry interval  . . . .   2                     1-99 (seconds)
  Searched first  . . . .   *REMOTE              *REMOTE, *LOCAL

                                                              Bottom
F3=Exit   F12=Cancel
```

If you are using DNS, the command may fail if the DNS is slow or if the server is down. This is the case where you do not have any local entries for the system in option 10 of CFGTCP, but you have valid entries on some remote DNS. If that remote DNS is slow, the command *STRNFSSVR *ALL* will not finish in the default time allotment (30 seconds). One solution is to increase the timeout value to several minutes, and another is to add the entries into the local hosts table and change option 13 to search the local table first.

7.2 Typical Problems with NFS Security

1. Creating objects:

   When we try creating objects within a mounted directory from an AS/400 client, we get a CPFA0A1 error. To find out more about this error, use the Display Message Description (DSPMSGD) command. Then you get the following display.

```
                    Display Formatted Message Text
                                                        System:   ASSYS01
Message ID . . . . . . . . . :   CPFA0A1
Message file . . . . . . . . :   QCPFMSG
  Library  . . . . . . . . . :     QSYS

Message . . . . :   An input/output error occurred.
Cause . . . . . :   Possible problems:
    --Media are full
    --Hardware failure
    --Communications failure.
Recovery  . . : If the problem continues start problem analysis (ANZPRB
  command).









                                                              Bottom
Press Enter to continue.

F3=Exit   F11=Display unformatted message text   F12=Cancel
```

   The reason for this error is that you have specified **ro** (read only) option in the NFS export options of the Export File System (EXPORTFS) command. To avoid this error, change the export options to delete the **ro** option and try again. This time you should be able to create objects. In the preceding problem, if the client is an RS/6000 system, we get the following error on the display.

```
mkdir: 0653-358 Cannot create dir1.
dir1: There is an input or output error.
```

2. Changing permissions or ownership from an AIX client:

   When you try to change permissions or ownership of an object within a mounted directory from an AIX client using the *chown* or *chmod* command, you get an error as follows:

```
chown: dir1: There is an input or output error.
chmod: dir1: There is an input or output error.
```

   This error is because you are not the current owner of that object.

3. Problems with GID = 0:

   The following figure shows you the scenario that this problem discusses.

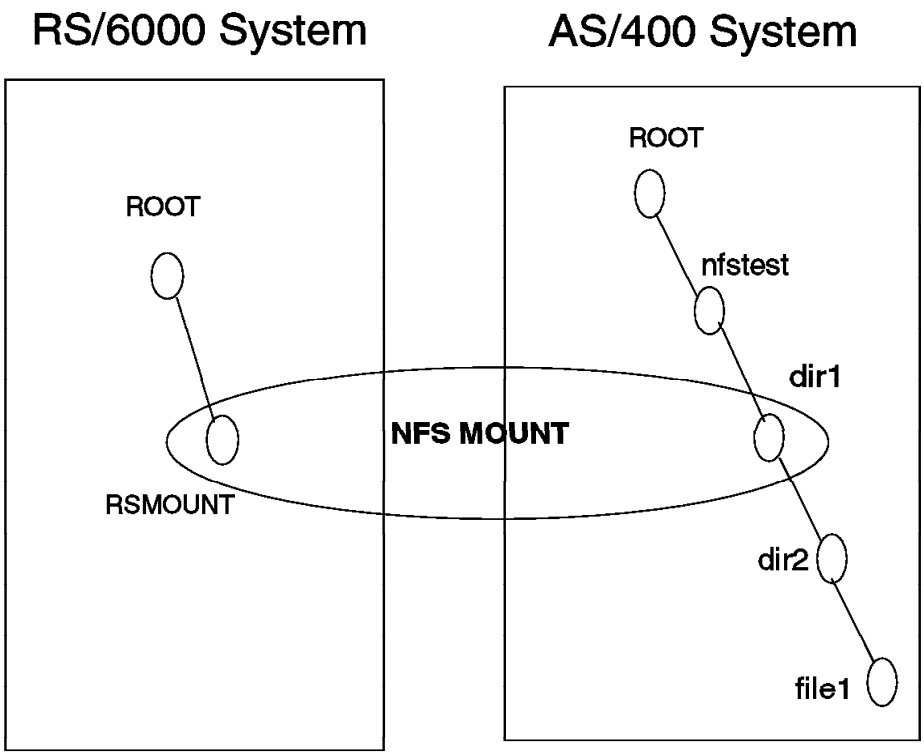


Figure 27. The NFS Mount for the Following Example

Let's assume the user profile on the AS/400 system has the following attributes:

User Name	DIVYA
User ID(UID)	306

The user DIVYA does not belong to any group and does not have any special authorities.

The user profile on the RS/6000 has the following attributes:

User Name	PANKAJ
User ID(UID)	306
Group	SYSTEM

The object authorities for the object *dir2* are now listed.

- The owner of this object is QTCP and it has *RWX authority and *ALL object authority.
- The *PUBLIC has * EXCLUDE authority.
- The primary group is *NONE.

Now, on the RS/6000 client, we change to the directory RSMOUNT and execute the commands in the following order:

```
>cd /RSMOUNT
>cd dir2
```

This command gives an error as follows:

```
ksh: dir2  :Permission denied.
```

Take a look at the contents of dir1 and the authorities of the owner, group, and public for the object dir2.

```
>ls -l
```

The result is:

```
drwx------   3 4294770705   system     45568 Sep 04 11:33 dir2
```

Notice that the public does not have any authority. On the AS/400 system, change the data authority of *PUBLIC to *RWX. Try again to change into directory *dir2*:

```
>cd dir2
```

But again the same error repeats. On the AS/400 server, add user DIVYA with *RWX permissions and then retry the command. But again we get the same error. Now add user QNFSANON with *RWX and repeat the command:

```
>cd dir2
```

The error still persists. Do a list command to recheck the authorities:

```
>ls -l
```

The result is:

```
drwx---rwx   3 4294770705   system     45568 Sep 04 11:38 dir2
```

Since the public has rwx permissions, we expect that this should work. The problem we overlooked in this case was the Group Identifications (GIDs). We did not expect problems with GID because the primary group of the object *dir1* is *NONE. A primary group of *NONE refers to a GID of 0. And the *ls* command shows that the group *system* does not have any data authorities. To check the GID on the RS/6000 system, use the following commands:

```
>cd /etc
>cat group
```

The output now shows the following values:

```
system:!:0: user2,user1,PANKAJ,root
staff:!:1: sakai,nfs3,nfstest1,netinst,aconn
bin:!:2: root,bin
sys:!:3: t1,root,bin,sys
adm:!:4: bin,adm
uucp:!:5: nuucp,uucp
mail:!:6:
security:!:7: root
cron:!:8: root
printq:!:9:
audit:!:10: root
```

```
ecs:!:28:
nobody:!:4294967294: nobody,lpd
usr:!:100: guest
aconn:!:11: aconn
```

This shows that the group *system* has a GID of 0 and the user PANKAJ is a member of that group. Refer to Section 4.5, “Security Checking Algorithm” on page 103. To solve this problem, we have to change the group permissions of the group *NONE on the AS/400 system to give *RWX authorities. The AS/400 display is shown in the following figure.

```
                    Change Primary Group (CHGPGP)

Type choices, press Enter.

Object . . . . . . . . . . . . . > '/nfstest/dir1/dir2'

New primary group  . . . . . . . > *NONE         Name, *NONE
New data authorities . . . . . . > *RWX          *OLDPGP, *PRIVATE, *RWX
New object authorities . . . . .   *NONE         *NONE, *ALL, *OBJEXIST.
               + for more values
Revoke current authority . . . .   *YES          *NO, *YES
Symbolic link  . . . . . . . . .   *NO           *NO, *YES









                                                                    Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

After this, try the commands again from the RS/6000 system:

```
>cd /RSMOUNT
>ls -l
```

The display shows the following values:

```
drwxrwxrwx   3 4294770705   system     45568 Sep 04 11:38 dir2
```

Try the command to change into directory dir2:

```
>cd dir2
>ls
```

The display shows the following message:

```
file1
```

Now we can access directory dir2.

4. Creating files in QSYS.LIB file system:

   Sometimes when you try to create files in the QSYS.LIB file system, you get a “EACCES” error condition. This error condition means that the server denies permission to the object. The process or job made an attempt to access an object in a way that the object access permissions forbid. The process or job does not have access to the specified file, directory, component, or path. If you access a remote file through NFS, the client does not reflect any file permissions at the client until local data updates occur. Also, this problem can occur because you are trying to create a member in a file that you do not own. It can also happen if you are the owner but you have *ALLOBJ special authorities. This causes your user profile to get swapped to the QNFSANON user profile who is not the owner of the file.

7.3 Problems with Commands

The cp UNIX command sometimes gives error messages such as “socket operation not supported” when copying to the QSYS.LIB file system. This happens when the cp command tries to change the owner or group owner of the member that they just copied to be the same as the source object (this is the AIX semantic). Since a member cannot have a different owner from its parent directory, this attempt to change owners will fail and the error message is sent. Note that the cp command actually completes successfully; it is just that you get these extra error messages.

Appendix A. Layout and Rules of the /etc/exports File

This appendix shows the contents of the default *exports* file for your convenience. Because this file is in the IFS, SEU does not work against this file. To edit and see the contents, you need to use the EDTF command, which is not in the base OS/400. This command is described later in the Appendix D, "Installation of the Edit File (EDTF) command" on page 161. This file explains the structure and the rules of export entries.

```
/etc/exports

This file contains the paths of the directory trees that you want
to export to the rest of the world through the NFS/400 server.
There are several restrictions on the format of the entries.
They are:

1. All entries are ended by the end-of-line character.

2. The continuation character '\' can be used to extend one
   entry across multiple lines when it is the last character
   in the line.

3. All path names must be absolute, beginning at the root '/'.

4. The first character must be a slash '/'.

5. There can be no spaces in the path name.

6. Elements in the path name must be separated by a slash '/'.

7. Each path element cannot be longer than 255 characters.

8. No hexadecimal characters allowed.

9. No control-characters are allowed (this includes any
   characters below ASCII 0x020.

10. No tabs or line feeds can be used in the path name.

11. All characters following the pound sign '#' are considered
    comments until the end of the line.  The only exception
    to this rule is the HOSTOPT parameter, which uses the '#'
    character as a starting point for each HOSTOPT entry.

12. After the path, if the first non-blank character is a dash,
    it is ignored.

Formatting the #HOSTOPT parameter:

You can provide additional information for each export entry
related to the remote NFS clients who access your entry.
For each remote NFS client (or netgroup), you can specify
the following values:

/path_name #HOSTOPT HostName=Client1,
                    PathNameCodePage=NNN,
                    DataFileCodePage=NNN,
```

```
                    NoWaitForWrites

HostName - The name of the remote NFS Client for which you are
           specifying additional export options.
           Specify this client by name in one of the lists as
           a host that has access to your export entry.  You can
           also use a netgroup name for this parameter.  You
           can list multiple clients and netgroups for every
           entry, each with its own specific export options.

PathNameCodePage - The path name code page is used for the path
           name components of the files exported to and mounted
           on the specified NFS client or netgroup.  For any
           clients or netgroups not specified on the HostName
           parameter, the defauly network (ASCII) is used.

DataFileCodePage - The data file code page is used for data of
           the files exported to and mounted on the specified
           NFS clients or netgroups.  For any hosts not specified
           on the HostName parameter, the default network data
           code page (binary, no conversion) is used.
           If the entry being exported resolves to an object
           within the QSYS.LIB file system and the network data
           file code page is specified, then data files are
           opened by OS/400 Network File System Support.  NFS
           will use the open() API with the O_TEXTDATA and
           O_CODEPAGE options.
           NOTE:  See the System API Reference, SC41-4801,
           for more details on the open() API and the O_TEXTDATA
           and O_CODEPAGE options.

NoWaitForWrites - This option specifies whether write requests
           are handled synchronously or asynchrounously for the
           NFS client or netgroup name.  The absence of this
           option means that data will be written to disk
           immediately (synchronously).  If this flag is used,
           then the data is not guaranteed to be written to
           disk immediately (asynchronously) and can be used to
           improve performance.

Additional Format Rules for the HOSTOPT:

1. The options must start with #HOSTOPT.  There must only be
   one #HOSTOPT although you could have many host names.

2. Additional host names and their options must be separated
   by a colon.

3. Options are case-insensitive.

4. Options that are not specified are processed as the
   defaults previously mentioned.


EXAMPLES OF FORMATTING /etc/exports WITH #HOSTOPT PARAMETER:

Example 1: Exporting to a host and specifying all options

/home/joe access=sammy          \
```

```
#HOSTOPT HostName=sammy,        \
         PathNameCodePage=367,  \
         DataFileCodePage=850,  \
         NoWaitForWrites
```

A user exports /home/joe to host sammy, specifying code pages for both path names and data files. Data written is not guaranteed to be written immediately to disk (asynchronously). Note the use of the continuation character '\' to break the long entry into more readable lines.

Example 2: Exporting with options to multiple clients and netgroups

```
/home/selfe access=sammy:rainfall:dept-OA5   \
#HOSTOPT HostName=sammy,                     \
         PathNameCodePage=367:               \
         hostname=rainfall,                  \
         NoWaitForWrites
```

A user exports /home/selfe to host sammy, rainfall, and the netgroup dept-OA5. For the host sammy, a code page for path names has been specified. For the host rainfall, data is not guaranteed to be written immediately to disk (asynchronously).

Example 3: Exporting a directory within QSYS.LIB with options

```
/qsys.lib/jon.lib access=dept-OA5   \
#HOSTOPT HostName=wheatley,         \
DataFileCodePage=367
```

A user exports /qsys.lib/jon.lib to the netgroup dept-OA5. A code page is specified for data files. Because the directory exists in the QSYS.LIB file system, NFS will use the open() API with the O_TEXTDATA and O_CODEPAGE options to work with data files. Note that wheatley will use these options only if it is defined as a member of the netgroup dept-OA5.

Appendix B. Layout and Rules of the /etc/netgroup File

This appendix contains the default *netgroup* file that explains the structure and the rules of netgroup entries.

```
/etc/netgroup

This file contains definitions for netgroups.
A netgroup is a mechanism used to associate an entire list with
one name.  For example:

department-OA5 (larrys-box,,),(safe-house,,)
my-hosts   (rainfall,,),(sammy,,),(department-OA5,,)

The preceding example defines the netgroup my-hosts, which
is a list of the following:  rainfall, sammy, larrys-box,
and safe-house.
Note that larrys-box and safe-house were included because
department-OA5 is itself a netgroup.

Traditionally, netgroups are defined as:

netgroup-name    (host-name,user-name,domain-name)

host-name     The name of any host from the /etc/hosts file.
              The AS/400 system does not support the /etc/hosts
              file as a separate integrated file system file.
              The /etc/hosts file is built into the AS/400
              TCP/IP support.

user-name     The name of any user from the /etc/passwd file.
              The AS/400 system does not support the /etc/passwd
              file concept.

domain-name   The name of any domain.

On the AS/400 system, the following rules apply to netgroups:

1. All entries are ended by the end of a line.

2. The continuation character '\' can be used to extend one
   entry across multiple lines when it is the last character
   in the line.

3. All characters following the pound sign (#) are considered
   comments until the end of the line.

4. In the triple (host-name,user-name,domain-name) only the
   host-name is recognized.  Any value supplied for user-name
   or domain-name is ignored.

5. Commas (,) must be used to separate elements in a triplet.
   All syntax elements must be present in each triplet (,,).

6. A blank host-name is not considered as a wild-card.
   There is no wild-card support.

7. Blanks are allowed anywhere except within a netgroup-name
```

or a host-name.

8. Imbedded hexadecimal characters are not allowed. This includes characters below ASCII x020.

9. All entries are **not** case-sensitive.

10. Triplets (x,y,z) (a,b,c) may be separated by spaces or commas only.

Appendix C. NLS Code Page Example

This example shows the EBCDIC code page 37. The hexadecimal (binary) value is found by the first digit combined with the second digit.

For example, the dollar ($) sign is located in hexadecimal '5B'.

HEX DIGITS 1ST → 2ND ↓	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	(SP) SP010000	& SM030000	- SP100000	ø LO610000	Ø LO620000	° SM190000	µ SM170000	^ SD150000	{ SM110000	} SM140000	\ SM070000	0 ND100000
-1	(RSP) SP300000	é LE110000	/ SP120000	É LE120000	a LA010000	j LJ010000	~ SD190000	£ SC020000	A LA020000	J LJ020000	÷ SA060000	1 ND010000
-2	â LA150000	ê LE150000	Â LA160000	Ê LE160000	b LB010000	k LK010000	s LS010000	¥ SC050000	B LB020000	K LK020000	S LS020000	2 ND020000
-3	ä LA170000	ë LE170000	Ä LA180000	Ë LE180000	c LC010000	l LL010000	t LT010000	• SD630000	C LC020000	L LL020000	T LT020000	3 ND030000
-4	à LA130000	è LE130000	À LA140000	È LE140000	d LD010000	m LM010000	u LU010000	© SM520000	D LD020000	M LM020000	U LU020000	4 ND040000
-5	á LA110000	í LI110000	Á LA120000	Í LI120000	e LE010000	n LN010000	v LV010000	§ SM240000	E LE020000	N LN020000	V LV020000	5 ND050000
-6	ã LA190000	î LI150000	Ã LA200000	Î LI160000	f LF010000	o LO010000	w LW010000	¶ SM250000	F LF020000	O LO020000	W LW020000	6 ND060000
-7	å LA270000	ï LI170000	Å LA280000	Ï LI180000	g LG010000	p LP010000	x LX010000	¼ NF040000	G LG020000	P LP020000	X LX020000	7 ND070000
-8	ç LC410000	ì LI130000	Ç LC420000	Ì LI140000	h LH010000	q LQ010000	y LY010000	½ NF010000	H LH020000	Q LQ020000	Y LY020000	8 ND080000
-9	ñ LN190000	ß LS610000	Ñ LN200000	` SD130000	i LI010000	r LR010000	z LZ010000	¾ NF050000	I LI020000	R LR020000	Z LZ020000	9 ND090000
-A	¢ SC040000	! SP020000	¦ SM650000	: SP130000	« SP170000	ª SM210000	¡ SP030000	[SM060000	(SHY) SP320000	1 ND011000	2 ND021000	3 ND031000
-B	. SP110000	$ SC030000	, SP080000	# SM010000	» SP180000	º SM200000	¿ SP160000	] SM080000	ô LO150000	û LU150000	Ô LO160000	Û LU160000
-C	< SA030000	* SM040000	% SM020000	@ SM050000	ð LD630000	æ LA510000	Ð LD620000	¯ SM150000	ö LO170000	ü LU170000	Ö LO180000	Ü LU180000
-D	(SP060000	) SP070000	_ SP090000	' SP050000	ý LY110000	¸ SD410000	Ý LY120000	¨ SD170000	ò LO130000	ù LU130000	Ò LO140000	Ù LU140000
-E	+ SA010000	; SP140000	> SA050000	= SA040000	þ LT630000	Æ LA520000	Þ LT640000	´ SD110000	ó LO110000	ú LU110000	Ó LO120000	Ú LU120000
-F	| SM130000	¬ SM660000	? SP150000	" SP040000	± SA020000	¤ SC010000	® SM530000	× SA070000	õ LO190000	ÿ LY170000	Õ LO200000	(EO)

Code Page 00037

Figure 28. Code Page 37

Appendix D. Installation of the Edit File (EDTF) command

This appendix provides you with the necessary information to install PTF SF38832 on a V3R7 OS/400 release.

This PTF will add the following commands to your system:

Edit File (EDTF)
This commands allows you to edit stream files or database files.

Display Stream File (DSPSTMF)
The DSPSTMF lets you display the contents of stream files.

SQL Utilities (SQLUTIL)
It is an interactive way to start SQL requests.

1. First order the PTF SF38832 through the Send PTF Order (SNDPTFORD) command or call your local IBM software support to get the PTF on your preferred media.
2. Once you have received the PTF, load and apply it using the following commands:

   ```
   LODPTF LICPGM(5716SS1) DEV(*SERVICE) SELECT(SF38832)
   ```

   The value *SERVICE for the DEV parameter loads the PTF when you order it through SNDPTFORD and received it through the Electronic Customer Support (ECS) line. Otherwise, use the device name of your tape drive or optical device.

   ```
   APYPTF LICPGM(5716SS1) SELECT(SF38832)
   ```

   The Apply PTF (APYPTF) command applies the PTF. As a result, you get a save file with the name QGPTOOLS that is saved within the QGPL library.
3. Now restore the appropriate commands and associated objects out of the save file with the following command:

   ```
   RSTOBJ OBJ(*ALL) SAVLIB(QGPTOOLS) DEV(*SAVF) SAVF(QGPL/QGPTOOLS)
          RSTLIB(xyz)
   ```

   Specify the library you want to restore the new commands to with the RSTLIB parameter.
4. If not already done, add the library containing the new commands to your library list using the command:

   ```
   ADDLIBLE LIB(xyz)
   ```

   The value “xyz” represents the library containing the objects restored within the previous step.
5. Now you can use the commands provided by the PTF.

Attention

Because the installation instructions may change in the future, refer to the PTF cover letter to ensure using the proper instructions to create the commands.

Appendix E. CHOWN() API Description

The chown() function changes the owner and group of a file. If the named file is a symbolic link, chown() resolves the symbolic link. The permissions of the previous owner or primary group to the object are revoked. If the file is checked out by another user (someone other than the user profile of the current job), chown() fails. The syntax for the chown() function is in the following example:

```
#include <unistd.h>
int chown(const char *path, uid_t owner, gid_t group);
```

The parameters for this command are explained in the following list:

1. Path: This is a pointer to the null-terminated path name of the file whose owner and group are being changed.
2. Owner: The user ID (UID) of the new owner of the file.
3. Group: The group ID (GID) of the new group for the file.

The following sample program changes the owner and group of an object:

```
/* Include all header files*/
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/types.h>

 void main( int argc, char **argv )
  {
       int x;
       char fn[100]; /*For the File name*/
       uid_t xxx;          /*For the UID*/
       gid_t yyy;          /*For the GID*/
       int fildes;
      struct stat info;

      strcpy( fn, argv[1] );
      sscanf(argv[2],"%d",&xxx);
      sscanf(argv[3],"%d",&yyy);
         x = chown(fn, xxx, yyy); /*The chown function*/
         unlink(fn);
    }
```

The program accepts three parameters. The first one is the path to the object for which you have to change ownership. The second parameter is the UID of the new owner. The next one is the GID for the new owner. The three parameters are to be passed to the program as strings. Within the C program, the sscanf() functions convert it into decimals and pass this on to the chown function. A sample call statement for this program is shown:

```
CALL PGM(CHOWN) PARM('QSYS.LIB/DLIB.LIB/CUSTOMER.FILE' '931' '0)
```

In the preceding statement, we are changing the ownership of the file CUSTOMER in library DLIB to the user profile with UID as 931.

For more information on the chown() API, refer to *AS/400 System API Reference V3R7*, SC41-4801-01.

Appendix F. PC NFS Client Considerations

This appendix describes the considerations when you try to mount file systems on your PC clients. This book discusses the NFS functions of the AS/400 system as the server and client. Since NFS is widely used in UNIX platforms, we use the AIX system as a client to the AS/400 system.

There may be users who want to use PCs as clients to the AS/400 system. Those users may need to address special considerations for its use.

Please note, it is not the purpose of this appendix to describe fully how to configure or how to use them.

F.1 Mounting a File System from PC Client

NFS is a UNIX based function and needs basic UNIX skills to utilize it. To conceal UNIX complexity, and being consistent with other interfaces similar to the PC, PC based NFS client software has different functions.

One of them is user authentication. Since you do not have to input a user name and its password to use a PC, it may not be appropriate to let the PC user mount file systems with only the UID supplied. UNIX users and AS/400 users, on the contrary, were already authenticated in a sense, when they became users on the systems. Then, PC based NFS client software has an authentication process together with an NFS server. This process is based on the user name and its associated password, which are provided by the PC user.

The process is similar to the following steps:

1. First, the PC client sends the user name and password.
2. The server identifies its UID and GID from the user name.
3. The client goes to the mount process with the UID.

To accommodate this authentication, some PC based NFS client software needs a daemon running on an NFS server to check authorization. This daemon is called ″pcnfsd″. The NFS client software expects this daemon running on the NFS server, then sends an authorization request to it when a user on PC client tries to mount a file.

UNIX based NFS servers (including AIX) and PC based NFS servers have this daemon implemented. If these NFS clients try to mount a file system exported from the AS/400 system as an NFS server, the mount operation will fail because the AS/400 system does not have this daemon (pcnfsd) implemented. A typical error message displayed on a PC window looks similar to the following figure.

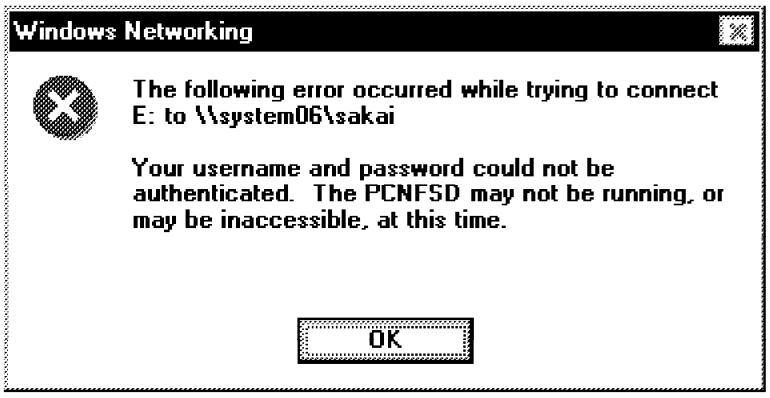


Figure 29. Mount Error Message

One exception so far that we know of is the OS/2 NFS Client Kit. This client software does not need the ″pcnfsd″ daemon on a server. If the server does not have the daemon, it bypasses the security check and simply goes ahead to ask UID, GID. If the server is running the ″pcnfsd″ daemon, OS/2 asks for a user name and its password for the authentication.

F.1.1 Work Around

If you have an NFS server other than an AS/400 system in your network, you can use this server as a security (authentication) server. After you have this security server authorize your access to the AS/400 system, you can mount a file system being exported from the AS/400 system.

We have tested two products of NFS client software in this redbook project. Both products run under Windows95. They are:

- InterDrive V2.1 by FTP Software, Inc.
- NFS Maestro V6.0 by Hummingbird Communications LTD.

For each NFS client software product, we have used an AIX system as the security server.

Note

The ITSO does not intend to recommend this software as the client software to AS/400 server. Also, the ITSO does not guarantee these products work with the AS/400 server.

F.1.2 Defining a Security Server

This section shows how you can configure the security server in each product.

F.1.2.1 Example 1

InterDrive has the security server setting dialog box. This dialog box can be invoked from the Network icon in the Windows95 control panel. When you click the Network icon, it brings the Network Properties window. It contains the entry of the InterDrive Client property setting. One of the tabs is to set up the security server for NFS mounting. Figure 30 on page 167 shows the security tab to define the server for the authentication purpose.

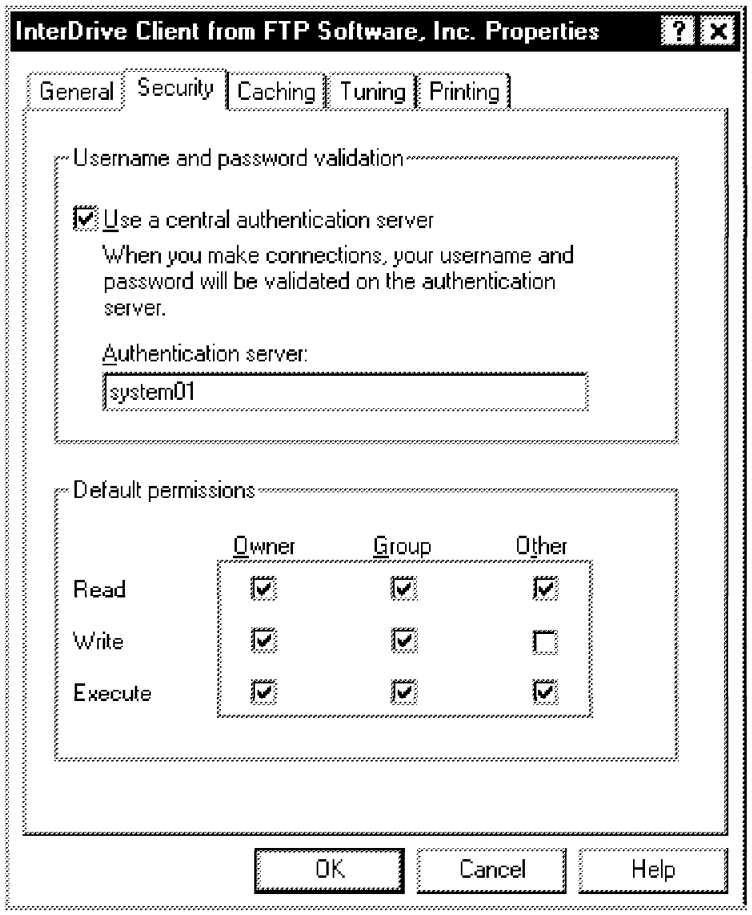


Figure 30. InterDrive Security Set Tab

When you uncheck the box ″Use a central authentication server″, this product assumes that the NFS server has the daemon. In this example, we configured the NFS client to have an AIX system named ″system01″ as the authentication server for the AS/400 system. Because system01 has to authenticate the access, it must have the PC user defined. Otherwise, the mount operation will fail.

F.1.2.2 Example 2

Hummingbird′s Maestro has a similar security setting dialog box. This setting can be also invoked from the Windows95 control panel. The same as the InterDrive security setting, the Network Properties has its entry for Hummingbird′s Maestro. It is one of the client services in the Network properties, and NFS Maestro properties has the entry for the authentication server.

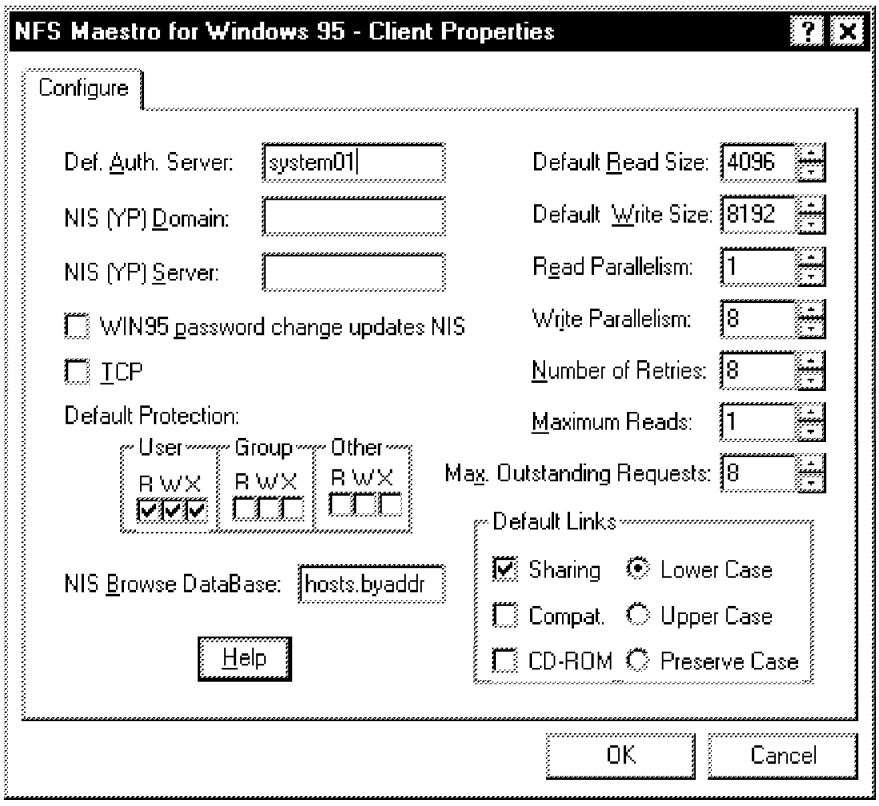


Figure 31. NFS Maestro Security Set Tab

The same as Example 1, we configured the NFS client to have the AIX system ″system01″ as the AS/400 authentication server in the input field ″Def. Auth Server″. A blank in this field means that the NFS server has the deamon.

Appendix G. Displaying Exported Files

When you are on an NFS client, you may wonder what file systems are exported from the AS/400 system. The AIX system, for example, has a shell command, *showmount*, to display all exported files on a specific server.

This command looks similar to the following example:

```
showmount -e system01
```

where system01 is the server name you want to know what file systems are exported from.

The output of this *showmount* command looks similar to Figure 32.

```
# showmount -e system06
export list for system06:
/sakai              (everyone)
/csky               (everyone)
/QSYS.LIB/CSKY.LIB (everyone)
```

Figure 32. Showmount Output Example on AIX Window

To respond to this command, the server needs to have a function that is called the ″mountproc-export″ sub-function. The AS/400 system does not have this sub-function as a base NFS support.

There are AS/400 PTFs available to respond to this request from a client. If you want to see file systems exported from the AS/400 system, you need to apply the PTF. The PTF number is:

- SF45634 for V3R7
- SF45631 for V4R1

This PTF gives the AS/400 system the capability to respond from a client, but does not provide a user interface to evoke this function on the AS/400 system. In other words, the AS/400 system provides the server function for this, not the client function. This means that you need another system to display the exported file systems from the AS/400 system. AIX and some PC based clients can be used for the showmount interface. The PTFs do not return information about which clients can mount the exported file systems. Therefore, it looks as though *everyone* can mount the file systems.

Appendix H. Special Notices

This publication is intended to help system administrators and network coordinators to implement and use the AS/400 Network File System (NFS) support. The information in this publication is not intended as the specification of any programming interfaces that are provided by the AS/400 Network File System Support. See the PUBLICATIONS section of the IBM Programming Announcement for AS/400 Network File System Support for more information about what publications are considered to be product documentation.

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 IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

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

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of

including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX®
AS/400®
BookMaster®
Client Access
IBM®
Operating System/400®
OS/400®
400®
Application System/400®
BookManager®
C/400®
Client Access/400
Information Assistant
OS/2®
RS/6000

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

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

Other company, product, and service names may be trademarks or service marks of others.

Appendix I. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

I.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 175.

- *Speak the Right Language with your AS/400 system*, SG24-2154

I.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

I.3 Other Publications

These publications are also relevant as further information sources:

- *AIX Version 4 Command Reference*, SBOF-1851
- *AS/400 CL Reference*, SC41-4722
- *AS/400 National Language Support V3R7*, SC41-4101
- *Character Data Representation Architecture*, SC09-2190
- *Integrated File System Introduction*, SC41-4711
- *International Application Development*, SC41-4603
- *OS/400 Security - Reference V3R7*, SC41-4302
- *OS/400 Network File System Support*, SC41-4714

  (The following manuals are available only through the online library. If you need the hardcopy of the following manuals, please order *System API Reference*, SC41-4801.)
- *OS/400 National Language Support APIs V3R7*, SC41-4863
- *OS/400 Security APIs V3R7*, SC41-4872
- *OS/400 UNIX-Type APIs V3R7*, SC41-4875

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

  For a list of product area specialists in the ITSO: type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Web Site on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of Abbreviations

ASCII	American National Standard Code for Information Interchange
APA	all points addressable
CCSID	Coded Character Set Identifier
CHRID	Character Identifier
DBCS	Double Byte Character Set
DNS	Domain Name Services
EBCDIC	Extended Binary Coded Decimal Interchange Code
FTP	File Transfer Protocol
GID	Group Identification
IBM	International Business Machines Corporation
IFS	Integrated File System
IP	Internet Protocol
ITSO	International Technical Support Organization
MBCS	Mixed Byte Character Set
NFS	Network File System
NLS	National Language Support
OS/400	Operating System/400
PROFS	Professional Office System
SBCS	Single Byte Character Set
TCP/IP	Transmission Control Protocol / Internet Protocol
UDP/IP	User Datagram Protocol / Internet Protocol
UID	User Identification

Index

Special Characters

A

B

C

D

E

F

G

H

P

Q

R

S

T

U

W

X

ITSO Redbook Evaluation

Exploring NFS on AS/400
SG24-2158-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@vnet.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

Printed in U.S.A.