z/OS

Distributed File Service Customization

SC24-5916-00

z/OS

Distributed File Service Customization

SC24-5916-00

Note!

Before using this information and the product it supports, be sure to read the general information under Appendix E, “Notices” on page 167.

First Edition (March 2001)

This edition, SC24-5916-00, applies to Version 1 Release 1 of z/OS Distributed File Service (program number 5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

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

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

International Business Machines Corporation
Information Development, Dept. G60
1701 North Street
Endicott, NY, 13760-5553
United States of America

FAX (United States & Canada): 1+607+752-2327
FAX (Other Countries):
Your International Access Code +1+607+752-2327

IBMLink™ (United States customers only): GDLVME(PUBRCF)
Internet e-mail: pubrcf@vnet.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/

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

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

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

The following statements are provided by the Open Software Foundation.

Contents

Figures

Tables

About This Book

This book helps system and network administrators configure z/OS Distributed File Service. This book is used after the successful installation of the z/OS release. Installation is described in the *z/OS Program Directory* and the *ServerPac: Installing Your Order* documentation.

The z/OS Distributed File Service includes a DFS function that is based on the OSF DCE Distributed File Service component. The OSF DCE Distributed File Service includes distributed file system capabilities based on the OSF DCE RPC protocols and security.

The z/OS Distributed File Service also includes an SMB File/Print server support. This support is based on the X/Open SMB Version 2 specification and the IETF RFCs on Netbois over IP (RFC1001 and RFC1002). For more information on the SMB File/Print Server support refer to the *z/OS Distributed File Service SMB Administration* book.

The following terms and their meanings are used throughout this book, unless otherwise noted:

- DFS/SMB refers to the z/OS Distributed File Service element.
- DFS refers to the Distributed File Service support for DCE.
- SMB refers to the Distributed File Service SMB File/Print Server support.
- DCE refers to the DCE environment provided by the IBM z/OS DCE base services element.

Who Should Use This Book

This book is intended for system administrators who understand the basic concepts of the Distributed Computing Environment (DCE). A knowledge of TCP/IP communications also helps administrators to use this book more effectively.

How to Use This Book

This book provides overview information on configuring and deconfiguring the Distributed File Service. It takes you through the steps you perform to prepare for configuration by running the DFS Server Configuration Program (**DFSCONF**) from TSO to configure or deconfigure DFS.

Note: The chapters in this book describe procedures that should be performed in the order that they appear.

In addition, there are appendices:

- Appendix A, "Example DFS Server Configuration Log File (dfsconf.log)" on page 117, shows the contents of a log file created after a typical configuration on an DFS/SMB host system using the DFS/SMB Server Configuration Program (**DFSCONF**).
- Appendix B, "Directories and Files" on page 123, lists the important files and subdirectories shipped as part of the DFS/SMB product. A listing of symbolic links created during the installation process is also provided.
- Appendix C, "DFS/SMB Environment Variables" on page 129, lists and describes some of the DCE environment variables and all of the DFS/SMB environment variables.
- Appendix D, "DFS/SMB Performance and Tuning Guidelines" on page 131, describes information about tuning the server and/or client for optimal performance for administrators of DFS/SMB.

Where to Find More Information

Where necessary, this book references information in other books using shortened versions of the book title. For complete titles and order numbers of the books for all products that are part of z/OS, refer to the *z/OS Information Roadmap*, SA22-7500.

Refer to the “Bibliography” on page 169 for a list of the publications to which this book refers and publications that contain additional information about related products.

For information about installing Distributed File Service components, refer to the *z/OS Program Directory* and the *Server Pac: Installing Your Order* documentation.

Information about Distributed File Service configuration on other IBM systems can be found in the configuration guide for those systems.

Softcopy Publications

The z/OS Distributed File Service library is available on a CD-ROM, *z/OS Collection*, SK3T-4269. The CD-ROM online library collection is a set of unlicensed books for z/OS and related products that includes the IBM Library Reader™. This is a program that enables you to view the BookManager® files. This CD-ROM also contains the Portable Document Format (PDF) files. You can view or print these files with the Adobe Acrobat reader.

Internet Sources

The softcopy z/OS publications are also available for web-browsing and for viewing or printing PDFs using the following URL:

```
http://www.ibm.com/servers/eserver/zseries/zos/bkserv/
```

You can also provide comments about this book and any other z/OS documentation by visiting that URL. Your feedback is important in helping to provide the most accurate and high-quality information.

Using LookAt to Look up Message Explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages. You can also use LookAt to look up explanations of system abends. The IBM LookAt development team is investigating other forms of reference information, such as commands.

Using LookAt to find information is faster than a conventional search because LookAt goes directly to the explanation.

LookAt can be accessed from the Internet or from a TSO command line.

You can use LookAt on the Internet at:

```
www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html
```

To use LookAt as a TSO command, LookAt must be installed on your host system. You can obtain the LookAt code for TSO from the LookAt Web site by clicking on the **News and Help** link or from the *z/OS Collection*, SK3T-4269.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id* as in the following:

```
lookat iec192i
```

This results in direct access to the message explanation for message IEC192I.

To find a message explanation from the LookAt Web site, simply enter the message ID and select the release you are working with.

Note: Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*, SA22-7624. For such messages, LookAt prompts you to choose which book to open.

Accessing Licensed Books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link site:

```
http://www.ibm.com/servers/resourcelink
```

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link user ID, password, and z/OS licensed book key code. The z/OS order that you received provides a memo that includes your key code.

To obtain your IBM Resource Link user ID and password, logon to:

```
http://www.ibm.com/servers/resourcelink
```

To register for access to the z/OS licensed books:

1. Logon to Resource Link using your Resource Link user ID and password.
2. Select **User Profiles** located on the left-hand navigation bar.
3. Select **Access Profile**.
4. Select **Request Access to Licensed books**.
5. Supply your key code where requested and select the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed.

After your request is processed you will receive an e-mail confirmation.

Note: You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Logon to Resource Link using your Resource Link user ID and password.
2. Select **Library**.
3. Select **zSeries**.
4. Select **Software**.
5. Select **z/OS**.
6. Access the licensed book by selecting the appropriate element.

Unsupported Services and Commands

The following commands and services are not supported in DFS/SMB:

- The **TapeConfig** configuration file.
- The **FMSLog** log file.
- The following **bos** commands are not supported in DFS/SMB:

  - **bos getdates** command
  - **bos install** command
  - **bos prune** command
  - **bos uninstall** command.

  In addition, the **-newbinary** option is not available in DFS/SMB for any **bos** commands.

- The following **bak** commands cannot be issued against a z/OS **bakserver**. These commands may be issued from z/OS against non-z/OS **bakserver** processes:
  - **bak labeltape** command
  - **bak scantape** command
  - **bak readlabel** command.
- The **fms** command.
- The **dfsgw** gateway function is provided, see the *z/OS Distributed File Service DFS Administration* book for complete details. However, the following **dfsgw** commands cannot be issued from z/OS:
  - **dfsgw add** command
  - **dfsgw apropos** command
  - **dfsgw delete** command
  - **dfsgw help** command
  - **dfsgw list** command
  - **dfsgw query** command.
- The **dfstrace** command.

Commands and services not directly supported in DFS/SMB may be run or requested from other non-z/OS DFS systems. Exceptions are noted where applicable.

Conventions Used in This Book

This book uses the following typographic conventions:

Bold — **Bold** words or characters represent system elements that you must enter into the system literally, such as commands.

Italic — *Italicized* words or characters represent values for variables that you must supply.

`Example Font` — Examples and information displayed by the system are printed using an example font that is a `constant width typeface`.

[ ] — Optional items found in format and syntax descriptions are enclosed in brackets.

{ } — A list from which you must choose an item found in format and syntax descriptions are enclosed by braces.

| — A vertical bar separates items in a list of choices.

< > — Angle brackets enclose the name of a key on a keyboard.

... — Horizontal ellipsis points indicate that you can repeat the preceding item one or more times.

\ — A backslash is used as a continuation character when entering commands from the shell that exceed one line (255 characters). If the command exceeds one line, use the backslash character **** as the last non-blank character on the line to be continued, and continue the command on the next line.

Note: When you enter a command from this book that uses the backslash character (\) make sure you immediately press the Enter key and then continue with the rest of the command. In most cases, the backslash has been used for ease of readability.

$ A dollar sign that appears before a command indicates that the example is shown in shell mode, that is, commands are entered from the shell.

# A number sign is used for the command prompt in instances where *root* authority is needed (in DFS/SMB, **root** refers to a user with a **UID = 0**).

This book uses the following keying convention:

<**Return**> The notation <**Return**> refers to the key on your terminal or workstation that is labeled with either the word "Return" or "Enter," or with a left arrow.

Entering commands

When instructed to enter a command, type the command name and then press <**Return**>.

Chapter 1. Introduction to the Distributed File Service

IBM's Distributed File Service is a base element of z/OS which provides DFS and SMB support that allows users to access and share data in a distributed environment across a wide range of IBM and non-IBM platforms.

DFS support includes DFS client and file server support for DCE. DCE support is provided by the IBM z/OS DCE Base Services element. The DFS implementation is based on source code developed by the Open Software Foundation (OSF). To use the DFS support, the DCE Base Services element of z/OS must be installed, configured, and run on the system.

SMB support includes SMB file and print serving support for Windows clients and file serving for OS/2® clients. SMB file/print serving support depends on IBM z/OS DCE Base Services being installed as part of the z/OS release but does not require the DCE to be configured or run on the system.

The DFS/SMB server running on z/OS can provide support for either the DFS file serving environment for DCE, the SMB File/Print Server environment, or both environments simultaneously.

This chapter discusses the following topics:

- "SMB File/Print Server Support"
- "DFS Server Support for DCE" on page 2
- "DFS Client (DFSCM) Support for DCE" on page 3
- "DCE Local File System Support" on page 3
- "Hierarchical File System (HFS) Support" on page 4
- "z/OS Data Set (RFS) Support" on page 4
- "DFS Commands for DCE in TSO" on page 4.

Note: For information regarding z/OS release installation, refer to the *z/OS Program Directory* and to the *ServerPac: Installing Your Order* documentation.

SMB File/Print Server Support

The Distributed File Service includes functionality to configure and run as SMB File/Print server on a z/OS system.

Important Note to Users

To use the SMB support, IBM z/OS DCE Base Services must be installed but does not need to be configured and run.

The z/OS Distributed File Service Server Message Block (SMB)[1] support provides a server that makes Hierarchical File System (HFS) files and data sets available to SMB clients. The data sets supported include sequential data sets (on DASD), partitioned data sets (PDS), partitioned data sets extended (PDSE), and Virtual Storage Access Method (VSAM) data sets. The data set support is usually referred to as Record File System (RFS) support. The SMB protocol is supported through the use of TCP/IP on z/OS. This communication protocol allows clients to access shared directory paths and shared printers. Personal Computer (PC) clients on the network use the file and print sharing functions that are included in their operating systems. Supported SMB clients include Windows 2000 Professional, Windows 95,

[1] Server Message Block (SMB) is a protocol for remote file/print access used by Windows and OS/2 clients. This protocol is also known as Common Internet File System (CIFS).

Windows 98, Windows NT 4.0 Workstation, Windows 3.11 (Windows for Workgroups), and OS/2 Version 4 (for file access only).

The DFS server running as an SMB file server (an SMB server), exports file data for access by workstation users. Exported file data can be accessed and shared by both SMB workstation users and UNIX System Services users and applications. If the DFS server is also running DFS server in a DCE, the exported file data can also be accessed and shared by DFS clients.

The DFS server running to support SMB print serving provides SMB workstation access to print shares that are defined to z/OS printers using the capabilities of the z/OS Infoprint Server element. For more information on the Distributed File Service SMB File/Print Server support, refer to the *z/OS Distributed File Service SMB Administration* book. For more information on installing the Infoprint Server, refer to Step 19 on page 26 as well as the *z/OS Program Directory* and the *ServerPac: Installing Your Order* documentation.

When the SMB server support for encrypted passwords is specified and hardware encryption capabilities are used, the facilities of the Open Cryptographic Services Facility (OCSF) are used. Refer to Step 2d on page 24 for more information on when OCSF is required. For more information on installing OCSF, refer to Step 20 on page 26 as well as the *z/OS Program Directory* and the *ServerPac: Installing Your Order* documentation.

DFS Server Support for DCE

The Distributed File Service includes functionality to configure and run as a DFS server and DCE related server processes (daemons) on a z/OS system in a DCE cell. The DFS support for DCE allows users to access and share files stored on a DFS file server anywhere in the DCE network.

Important Note to Users

IBM z/OS DCE Base Services must be installed, configured, and running before using the DFS support for DCE.

Using the DFS support for DCE provides the following benefits:

- File location transparency. DFS keeps track of the location of files at all times so users do not have to.
- Data consistency for distributed data through the use of advanced token management.
- Uniform access to enterprise-wide data through unique naming of files.
- Improved security based on Kerberos authentication.
- Increased manageability through the use of distributed databases to track file location and authentication.

The exported file data can include Hierarchical File System (HFS) files, Record File System (RFS) data sets, or OSF's DCE Local File System filesets. The benefits of these file systems are detailed in this chapter.

The DFS server support for DCE includes support for all DCE daemons. In z/OS, the term daemons, server processes, and servers are used interchangeable. The DCE server must also be configured and running to use the DFS server support for DCE.

For more information on DCE cells, refer to the *z/OS DCE Administration Guide,* SC24-5904.

For more information on DFS server address spaces and process (daemons), refer to *z/OS Distributed File Service DFS Administration* book.

For more information on installing and configuring the DFS server, refer to Chapter 3, “Installation” on page 23.

DFS Client (DFSCM) Support for DCE

The Distributed File Service includes functionality to configure and run a DFS client on a z/OS system in a DCE cell. The DFS client is also referred to as the cache manager or **DFSCM**.

> **Important Note to Users**
>
> The DFS Client support requires IBM z/OS DCE Base Services to be installed, configured, and run on the same system.

The **DFSCM** support allows a z/OS user or application to access directories and files exported by DFS servers running in a DCE on either the local z/OS system, another z/OS system, or on non-z/OS systems. To configure or run the **DFSCM** on a system, DCE must also be running on the same system. If DCE is stopped while the **DFSCM** is running, the **DFSCM** cannot continue to run. The DFS/SMB server does not need to be run on the same system with the **DFSCM**.

The **DFSCM** runs as a physical file system with FILESYSTYPE(DFSC) in a colony address space as a host machine kernel extension. Refer to *z/OS UNIX System Services File System Interface Reference*, SA22-7808, for more information on physical file systems and colony address spaces.

Refer to the *z/OS Distributed File Service DFS Administration* book for more information about the **DFSCM**, including the **DFSCM** address space and processes.

Refer to the *z/OS DCE Administration Guide*, SC24-5904, for more information on DCE cells.

Refer to Chapter 3, “Installation” on page 23 for more information on installing and configuring the **DFSCM**.

DCE Local File System Support

The Distributed File Service includes support for the DCE Local File System. The DCE Local File System is part of the Open Software Foundation DFS product. DCE Local File System is a high-performance, log-based file system. It provides enhanced performance and reliability over traditional file systems by providing improved data storage and management.

The DCE Local File System support is implemented as a z/OS kernel extension that allows sharing with data consistency between local OMVS users and POSIX applications, DCE DFS client workstations, and SMB workstations.

In a DCE environment, the DCE Local File System provides support for the extended DFS fileset management functionality of space sharing, setting fileset quotas, moving filesets, creating backup filesets, and limiting access to DCE Local File System data by supporting DCE Access Control Lists (ACLs).

The Distributed File Service extends the DCE Local File System support by allowing SMB clients to access and share the DCE Local File System with DFS clients. If the DFS server is running as only SMB file server, SMB clients can access DCE LFS File System data without requiring DCE to be configured and running on the system. In an SMB environment, the DCE Local File System provides support for space sharing, setting fileset quotas, and DCE Access Control Lists (ACLs).

Hierarchical File System (HFS) Support

Hierarchical File Systems (HFS) can be exported to the DFS namespace * * or as an SMB file share.

In DCE, HFS is referred to as a non-Local File System fileset. The entire HFS file system is registered in the Fileset Location Database machine as one DFS fileset. Additional DFS filesets cannot be created within that HFS file system. There are some limitations when using HFS file systems with DFS. The fileset management functions of creating backup filesets, moving filesets, and fileset replication are not supported for HFS filesets. In addition, only UNIX mode bits can be used to protect HFS files that have been exported to the DFS namespace. Refer to the *z/OS Distributed File Service DFS Administration* book for more information about exporting HFS file systems (non-Local File System partitions) to the DFS namespace.

Usually, an HFS file system has to be locally mounted on the DFS file server machine before it is exported to the DFS namespace. An HFS data set is mounted at its root. If the DFS/SMB Dynamic Mount support is being used then the requirement to mount an exported HFS file system is eliminated.

z/OS Data Set (RFS) Support

z/OS data sets can be exported to the DFS namespace or as a file share to SMB clients. But, there are limitations and restrictions when using RFS filesets in a DFS or SMB file serving environment.

Refer to the *z/OS Distributed File Service DFS Administration* book for information on exporting RFS filesets to the DFS namespace.

Refer to the *z/OS Distributed File Service SMB Administration* book for more information on exporting RFS filesets as file shares to SMB clients.

DFS Commands for DCE in TSO

In DFS, most commands can be run from Time Sharing Option Extensions (TSO/E), the shell, or submitted as batch jobs. To ensure TSO/E users can run these commands, the SIOEEXEC library containing the REXX EXECs for TSO/E must be either allocated to SYSEXEC or concatenated with existing SYSEXEC libraries. This is normally done during the installation process (for details, refer to the *z/OS Program Directory*).

The SIOEEXEC library is supplied with a record format of Fixed Block. Ensure that all libraries in the SYSEXEC concatenation have the same record format.

To ensure that the SYSEXEC library SIOEEXEC is available to be searched, execute the following TSO/E command:

```
EXECUTIL SEARCHDD(YES)
```

You may also, optionally, add the SIOEEXEC library to SYSPROC instead of SYSEXEC. To do so, ensure all libraries in the SYSPROC concatenation have the same record format. Refer to the *z/OS Program Directory* for more information.

Chapter 2. Migration Overview

This chapter discusses migration issues. These issues generally apply whether you are using SMB File/Print support or the DFS support for DCE. Your plan for migrating to a new release of the Distributed File Service should include information from a variety of sources. These sources of information describe topics such as coexistence, service, hardware and software requirements, migration actions, and interface changes.

The following documentation, which is supplied with your product order, provides information about installing your system. In addition to specific information about Distributed File Service, this documentation contains information about all of the z/OS elements.

- *z/OS Planning for Installation*, GA22-7504,

  This book describes the installation requirements for z/OS at a system and element level. It includes hardware, software, and service requirements for both the driving and target systems. It also describes any coexistence considerations and actions.

- *z/OS Program Directory*, GI10-0669

  This document, which is provided with your z/OS product order, leads you through the specific installation steps for Distributed File Service and the other z/OS elements.

- *ServerPac: Installing Your Order*

  This is the order-customized, installation book for using the ServerPac Installation method. Be sure to review "Appendix A. Product Information", which describes data sets supplied, jobs or procedures that have been completed for you, and product status. IBM may have run jobs or made updates to PARMLIB or other system control data sets. These updates could affect your migration.

Within this chapter, you can find information about the specific updates and considerations that apply to this release of Distributed File Service.

- “Steps for Migrating” on page 6

  This section identifies tasks that must be done for any migration. These tasks are not associated with a specific new function, but must be completed to ensure successful migration to the new release.

- “Migration Roadmap” on page 7

  This section identifies the migration paths that are supported with the current level of Distributed File Service. It also describes the additional publications that can assist you with your migration to the current level.

- “Summary of Interface Changes” on page 15

  This section provides a summary of the changes that are made to Distributed File Service user and programming interfaces.

Steps for Migrating

This section describes the migration to a new release of the Distributed File Service.

The recommended steps for migrating to a new release of the Distributed File Service are:

1. Review the supporting migration and installation information for the z/OS release.

   Determine what updates you need for products that are supplied by IBM, system libraries, and non-IBM products. Review the *z/OS Planning for Installation*, GA22-7504, book and the *z/OS Introduction and Release Guide*, GA22-7502, for information about Distributed File Service and other z/OS elements.

2. Develop a migration plan for your installation.

   When planning to migrate to a new release of z/OS, you must consider high-level support requirements such as machine and programming restrictions, migration paths, program compatibility that are described in the *z/OS Program Directory*, the *ServerPac: Installing Your Order* documentation, or the *z/OS Planning for Installation* book, GA22-7504.

   The migration instructions for DFS/SMB, assume that you will install the new release on a system where a previous is already installed, configured, and running.

   Note: In December 1999, the Distributed File Service server was enhanced to also function as an SMB file/print server for workstations that are not in a DCE cell. The SMB support does require DCE to be installed but does not require DCE to be configured or running. Files exported to DCE DFS clients can be accessed and shared with the non-DCE (SMB) clients. Print requests from SMB clients can be routed to the Infoprint Manager on z/OS. The SMB support was made available on OS/390® Version 2 Release 7 and subsequent releases. Refer to the *z/OS Distributed File Service SMB Administration* book.

3. Obtain and install any required Program Temporary Fixes (PTFs) or updated versions of the operating system.

   Call the IBM Software Support Center to obtain the Preventive Service Planning (PSP) upgrade for z/OS Distributed File Service which provides the most current information about PTFs for z/OS Distributed File Service. Check RETAIN again just before testing z/OS Distributed File Service. For information about how to request the PSP upgrade, refer to the *z/OS Program Directory*. Although the *z/OS Program Directory* contains a list of the required PTFs, the most current information is available from the IBM Software Support Center.

4. Before installing the new releases, be sure to review “Additional DFS/SMB Migration Considerations” on page 19.

5. Install the product using the *z/OS Program Directory* or the *ServerPac: Installing Your Order* documentation. Additionally, the information in “Review of Installation and Post Installation Steps” on page 23 can be used to insure that all z/OS release installation steps applicable to the Distributed File Service are completed. Release installation steps with migration considerations are noted in the referenced section.

6. Contact programmers who are responsible for updating applications at your installation to ensure compatibility with the new releases.

   Verify that your installation applications continue to run. No changes should be required for compatibility with the new release. New DFS/SMB options available in the new release can be optionally exercised after the new release is running.

7. Use the new release before initializing major new function.

8. Optionally, customize the support to exercise new function that is available in the release.

Migration Roadmap

This section describes the migration paths that are supported by the current release of the Distributed File Service It also provides information about how you can obtain the DFS/SMB migration information from previous releases.

z/OS Version 1 Release 1 Summary

Table 1 summarizes the updates that have been introduced to Distributed File Service in z/OS Version 1 Release 1 (V1R1). If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 1. Summary of Distributed File Service Updates for z/OS V1R1

For Information About:	Refer to page:
Windows 2000 Professional	9
HFS Automounted File Systems Support (Dynamic Export)	9

OS/390 Version 2 Release 10 Summary

Table 2 summarizes the updates that have been introduced to Distributed File Service in OS/390 Version 2 Release 10 (V2R10). If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 2. Summary of Distributed File Service Updates for OS/390 V2R10

For Information About:	Refer to page:
NT LM 0.12	10
RFS SMB (SMB support for MVS™ data sets)	10
Greater than 32 bit Episode files	10

OS/390 Version 2 Release 9 Summary

Table 3 summarizes the updates that have been introduced to the Distributed File Service in OS/390 Version 2 Release 9 (V2R9). If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 3. Summary of Distributed File Service Updates for OS/390 V2R9

For Information About:	Refer to page:
Encrypted password login	11

OS/390 Version 2 Release 8 Summary

Table 4 on page 8 summarizes the updates that have been introduced to the Distributed File Service in OS/390 Version 2 Release 8 (V2R8). If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 4. Summary of Distributed File Service Updates for OS/390 V2R8

For Information About:	Refer to page:
SMB File/Print Server Support	12

OS/390 Version 2 Release 7 Summary

Table 5 summarizes the updates that have been introduced to the DFS/SMB in OS/390 Version 2 Release 7. If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 5. Summary of Distributed File Service Updates for OS/390 V2R7

For Information About:	Refer to page:
DFS Server Multi-home Support	13
Protected RPC	13
OSF/Transarc compatibility	13

OS/390 Version 2 Release 6 Summary

Table 6 summarizes the updates that have been introduced to the Distributed File Service in OS/390 Version 2 Release 6 (V2R6). If you are migrating from a previous release you should review the information in the detailed section for each item.

Table 6. Summary of Distributed File Service Updates for OS/390 V2R6

For Information About:	Refer to page:
DFSKERN in a Separate Address Space	14
DFS Backup	14
File basis translation	14

z/OS Version 1 Release 1 Overview

If you are migrating from a previous release of OS/390 this sections describes the functions that are introduced for z/OS Version 1 Release 1 (V1R1).

Windows 2000 Professional

Description	The SMB server now supports Windows 2000 Professional as an SMB client.
What This Change Affects	Distributed File Service SMB **Note:** This support was also made available on previous OS/390 releases.
Dependencies	OS/390, z/OS
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

HFS Automounted File Systems Support (Dynamic Export)

Description	The SMB server supports HFS automounted file systems by using a new capability called dynamic export. PC clients are able to access data in HFS file systems that are automatically mounted.
What This Change Affects	Distributed File Service SMB **Note:** This support was also made available on previous OS/390 releases.
Dependencies	OS/390, z/OS
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

OS/390 Version 2 Release 10 Overview

This section describes the new and changed Distributed File Service functions that are introduced for OS/390 Version 2 Release 10 (V2R10).

NT LM 0.12 (includes NT LM 0.12 encrypted password)

Description	The DFS/SMB server supports the LM 0.12 level of the SMB protocol used by Windows NT networking, thereby providing additional password encryption.
What This Change Affects	Distributed File Service SMB
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

RFS SMB (SMB support for MVS data sets)

Description	The DFS/SMB server supports workstation access to data stored in SAM, PDS(E), and VSAM files to further expand the S/390 support for application development.
What This Change Affects	Distributed File Service SMB
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

Greater than 32 bit Episode files

Description	This supports file sizes greater than 32 bits (4 gigabytes).
What This Change Affects	Distributed File Service SMB
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

OS/390 Version 2 Release 9 Overview

This section describes the new and changed Distributed File Service functions that are introduced for OS/390 Version 2 Release 9 (V2R9).

Encrypted Password Login

Description	The SMB server supports encrypted password login. When this function is used, the SMB server invokes OCSF services. PC users are allowed to store their SMB password in RACF&retm. for the encrypted password function.
What This Change Affects	Distributed File Service SMB **Note:** This support was also made available on OS/390 Release 7 and Release 8.
Dependencies	OCSF must be installed
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See "Steps for Migrating" on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

Authorized Programs

The list of authorized programs for the OS/390 V2R9 Distributed File Service has been expanded and includes:

- IOEGRWAG
- IOENEWAG
- IOESALVG
- SMBPW

Refer to the *z/OS Program Directory* for a description of the PARMLIB member updates required for the member IJKTSOxx.

OS/390 Version 2 Release 8 Overview

This section describes the new and changed Distributed File Service functions that are introduced for OS/390 Version 2 Release 8 (V2R8).

SMB File/Print Server Support

Description	SMB (Server Message Block) support is based on X/Open CAE Specification Protocols for X/Open PC Interworking: SMB, Version 2. The SMB File/Print Server processes requests from OS/2 Version 4, Windows 3.11 (Windows for Workgroups), Windows 95, Windows 98, or Windows NT 4.0 workstation users against a shared directory or shared printer. The SMB File Serving support, only, is also available for OS/2 clients. The SMB requests are sent to and received from the SMB server a TCP/IP connection. Shared directories (file shares) are defined to identify HFS data sets exported by the SMB File/Print Server. Shared printers are assigned to JES print queues and printer names defined to the Infoprint Server. The JES queues are written to or queried by the SMB File/Print Server in response to SMB point requests. In turn, the Infoprint Server reads and processes print requests from the JES queue.
What This Change Affects	Distributed File Service SMB **Note:** The SMB File Serving support, only, was made available on OS/390 Release 7.
Dependencies	For SMB Print Serving you need the Infoprint Server installed.
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service SMB Administration*

Authorized Programs

The list of authorized programs for the OS/390 V2R8 Distributed File Service has been expanded and includes:

- IOEBAK
- IOEBOS
- IOECM
- IOEDCERR
- IOEDFSXP
- IOEFTS
- IOEGRWAG
- IOEMAPID
- IOENEWAG
- IOESALVG
- IOESCOUT
- IOEUDBG.

Refer to the *z/OS Program Directory* for a description of the PARMLIB member updates required for the member IJKTSOxx.

OS/390 Version 2 Release 7 Overview

This section describes the new and changed Distributed File Service functions that are introduced for OS/390 Version 2 Release 7 (V2R7).

DFS Server Multi-home Support

Description	The Distributed File Service currently assumes Distributed File Service servers are reachable by a single network interface. The client recognizes and exploits multiple network interfaces for a DFS server host. Multi-homed server capabilities allow administrators to specify up to four interfaces (either hostnames or IP addresses) in the FLDB for each File Server and FLDB machine.
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

Protected RPC

Description	Distributed File Service allows you to set RPC authentication levels for Cache Manager to File Server communications. These levels can be set individually for each Cache Manager and File Server. In addition, you can also set advisory RPC authentication bounds on a per-fileset basis.
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

OSF/Transarc compatibility

Description	This supports DCE Local File System aggregate sizes that are greater than 4GB. The actual maximum size is based on the aggregate blocksize. Aggregates can contain 2G blocks. For a 4K blocksize, the aggregate can be 8TB (4K x 2G). For a 64K blocksize, the aggregate can be 128TB (64K x 2G).
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

OS/390 Version 2 Release 6 Overview

This section describes the new and changed Distributed File Service functions that are introduced for OS/390 Version 2 Release 6 (V2R6).

DFSKERN in a Separate Address Space

Description	This allows the DFSKERN server process to be run in a separate address space. This enables address space termination processing to free file and other resources held by DFSKERN.
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

DFS Backup

Description	This eliminates tape data file open/close overhead by backing up multiple file sets to a single tape data set.
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

File basis translation

Description	This allows the DFS client to access file sets that include both binary and text files. File translation specifications will be supported on a file type (extension) basis.
What This Change Affects	DCE Distributed File Service
Dependencies	OS/390
Coexistence Considerations	DFS clients and servers interact with different platforms and with different releases. It is assumed that the DFS clients and servers are running at the same release level.
Migration Tasks	See “Steps for Migrating” on page 6.
For More Information	*z/OS Distributed File Service DFS Administration*

Summary of Interface Changes

This section summarizes the new and changed interface components of Distributed File Service.

For Information About:	Refer to page:
DFS Files	15
DFS Commands	15
SMB Files	15
SMB Commands	16
DFS/SMB Environment Variables	16

DFS Files

Table 7 lists the new and updated DFS files. See the *z/OS Distributed File Service DFS Administration* for more information.

Table 7. Summary of New and Changed DFS Files

File Name	Release	Description
cm attributes file (cmattr)	OS/390 V2R6	Contains directives that map a file name extension (suffix) to an indication whether the DFS Cache Manager should translate the file data from ASCII to EBCDIC and vice versa (encoding).
cmattr_filesets	OS/390 V2R10	Contains fileset names that limit the effect of the cmattr file. (suffix) to an indication whether the DFS Cache Manager should translate the file data from ASCII to EBCDIC and vice versa (encoding).

DFS Commands

Table 8 lists the new and updated DFS commands. See the *z/OS Distributed File Service DFS Administration* for more information.

Table 8. Summary of New and Changed DFS Commands

Command Name	Release	Description
cm getprotectlevels	OS/390 V2R7	Returns the current DCE RPC authentication level settings for communications between the Cache Manager and File Servers.
cm setprotectlevels	OS/390 V2R7	Adjusts DCE remote procedure call (RPC) authentication levels for communications between the Cache Manager and File Servers.
fts setprotectlevels	OS/390 V2R7	Set advisory DCE remote procedure call (RPC) authentication levels for a specified fileset.
fxd	OS/390 V2R7	Initializes the File Exporter and starts associated kernel daemons.

SMB Files

Table 9 on page 16 lists the new and updated SMB files. See the *z/OS Distributed File Service SMB Administration* for more information.

Table 9. Summary of New and Changed SMB Files

File Name	Release	Description
smbidmap	OS/390 V2R8	Maps an SMB user ID to the user ID. The mapping is used to determine the SMB user's corresponding user ID. This determines access permissions for shared HFS directories and files and owners for print requests sent to shared printers.
smbtab	OS/390 V2R8	Specifies HFS shared directories and shared printers to be made available to PC clients.
Attributes File (rfstab)	OS/390 V2R10	Contains tables describing the attributes used to manipulate RFS files in the SMB server.

SMB Commands

Table 10 lists the new and updated SMB commands. See the *z/OS Distributed File Service SMB Administration* for more information.

Table 10. Summary of New and Changed SMB Commands

Command Name	Release	Description
dfsshare	OS/390 V2R8	An OMVS command that shares (or unshares) HFS directories or printers with SMB clients.

DFS/SMB Environment Variables

Table 11 lists new and changed DFS/SMB environment variables.

Table 11 (Page 1 of 4). Summary of New and Changed DFS/SMB Environment Variables

Environment Variable Name	Release	Description
_IOE_BACKUP_TAPE_CAPACITY	OS/390 V2R6	Specifies the size, in bytes, of the maximum size backup tape data set that is created during backup. It is desirable that this value equal the actual capacity of your tape volumes (or a little less).
_IOE_CM_ATTRIBUTES_FILE	OS/390 V2R6	Specifies the name of the file that contains the definition of file name suffixes that controls whether file data should be translated by the DFS client.
_IOE_CM_ATTRIBUTES_FILESETS	OS/390 V2R10	Specifies the name of the file that contains the definition of filesets that are under control of the cmattr file.
_IOE_CM_DIRCACHE_SIZE	OS/390 V2R6	Specifies the size, in bytes, of the directory cache for the DFS client.
_IOE_DAEMONS_IN_AS	OS/390 V2R6	The DFS environment variable controls the number **DFSCM ioedfsd** process threads set up to handle file requests.
_IOE_DIRECTORY_CACHE_SIZE	OS/390 V2R8	The number of 512 byte blocks used to cache HFS and RFS directory entries.
_IOE_DYNAMIC_EXPORT	z/OS V1R1	The file system is dynamically assigned values and exported when a client crosses a local mount point into a file system that is not known to the SMB server.

Table 11 (Page 2 of 4). Summary of New and Changed DFS/SMB Environment Variables

Environment Variable Name	Release	Description
_IOE_EPI_CACHE_SIZE	OS/390 V2R6	The size, in bytes, of the DCE Local File System metadata buffer cache. This environment variable is used to increase the size of the buffer cache and to potentially improve performance.
_IOE_EXPORT_TIMEOUT	z/OS V1R1	The number of minutes that a file system must be idle before it is dynamically unexported.
_IOE_HFS_ATTRIBUTES_FILE	OS/390 V2R7	Specifies the pathname of the **hfsattr** file that contains the definition of file name suffixes that controls whether file data should be translated by the DFS server.
_IOE_HFS_TRANSLATION	OS/390 V2R7	A DFS server variable that controls the conversion of HFS data to the appropriate data format. Converts incoming data from ASCII ISO 8859-1 to the local code page. Converts outgoing data from the local code page to ASCII ISO 8859-1.
_IOE_INHERIT_TRANSLATION	z/OS V1R1	The translation option of the parent file system will be inherited for a dynamically exported file system.
_IOE_LFS_SYNC_INTERVAL	OS/390 V2R8	The number of seconds between synchronization operations for the DCE Local File System.
_IOE_MVS_DFSDFLT	OS/390 V2R6	The name of the RACF-defined anonymous user that is associated with unauthenticated DCE users attempting access to exported HFS files. This ID must be a RACF-defined with a UNIX segment.
_IOE_MVS_SERVER	OS/390 V2R6	The name of the server used for identity mapping and registering with UNIX as a DFS server.
_IOE_PROTOCOL_RPC	OS/390 V2R8	An environment variable that controls whether the DCE DFS protocol is supported (using DCE RPC).
_IOE_PROTOCOL_SMB	OS/390 V2R8	An environment variable that controls whether the SMB protocol is supported (using TCP/IP).
_IOE_RFS_ALLOC_TIMEOUT	OS/390 V2R10	Specifies the time period (in seconds) that an RFS data set will remain allocated after there has been no access to the data set through the DFS/SMB server. After this time period, the data set will be deallocated and will be available to other applications such as ISPF.
_IOE_RPC_CROSS_MOUNTS	z/OS V1R1	Specifies whether DCE DFS clients can cross or go under local mount points.
_IOE_SMB_BROWSE_INTERVAL	OS/390 V2R8	Specifies the Browser announcement interval (in milliseconds).
_IOE_SMB_CALLBACK_POOL	OS/390 V2R8	Specifies the number of secondary pool threads for processing SMB callback requests.
_IOE_SMB_CLEAR_PW	OS/390 V2R9	Specifies the OS/390 SMB server policy for flowing the SMB password in the clear.
_IOE_SMB_COMPUTER_NAME	OS/390 V2R8	Specifies the name to be used by SMB redirectors (that is, clients) to contact this server.
_IOE_SMB_CROSS_MOUNTS	z/OS V1R1	Specifies whether SMB clients can cross or go under local mount points.

Table 11 (Page 3 of 4). Summary of New and Changed DFS/SMB Environment Variables

Environment Variable Name	Release	Description
_IOE_SMB_DESCRIPTION	OS/390 V2R8	Specifies the description of this server that appears on the PC.
_IOE_SMB_DIR_PERMS	OS/390 V2R9	The permissions for a directory created by the SMB server at the request of an SMB client.
_IOE_SMB_DOMAIN_NAME	OS/390 V2R8	Specifies the name to be used as the Domain name for this server.
_IOE_SMB_FILE_PERMS	OS/390 V2R9	The permissions for a file created by the SMB server at the request of an SMB client.
_IOE_SMB_IDMAP	OS/390 V2R8	Specifies the location of the **smbidmap** file. The **smbidmap** file contains the mapping of SMB user IDs to OS/390 user IDs.
_IOE_SMB_IDLE_TIMEOUT	OS/390 V2R8	Specifies how long (in seconds) an SMB session can remain inactive before it is terminated.
_IOE_SMB_MAIN_POOL	OS/390 V2R8	Specifies the number of primary pool threads for remain inactive before it is terminated.
_IOE_SMB_MAXXMT	OS/390 V2R8	Specifies the maximum server buffer size that is returned on the SMB Server Negotiate response.
_IOE_SMB_NT_SMBS	OS/390 V2R10	Specifies whether new SMBs in the NT protocol dialect are to be accepted from PC clients.
_IOE_SMB_OCSF	z/OS V1R1	Specifies whether OCSF is on or off. This environment variable was also made available on previous OS/390 releases.
_IOE_SMB_OPLOCK_TIMEOUT	OS/390 V2R8	Specifies the Opportunistic Lock Timeout period (in seconds).
_IOE_SMB_OPLOCKS	OS/390 V2R9	Specifies whether the server allows clients to use Opportunistic Locks on files. This allows some SMB clients to cache data at the client.
_IOE_SMB_PRIMARY_WINS	OS/390 V2R8	Specifies the IP address of the Windows Internet Service (WINS) server that this server announces itself to and forwards WINS requests to.
_IOE_SMB_PROTOCOL_LEVEL	OS/390 V2R10	Specifies the highest level of SMB protocol dialect that will be negotiated with the PC client.
_IOE_SMB_RAW	OS/390 V2R8	Specifies whether raw mode is supported in the SMB Server Negotiate response.
_IOE_SMB_SCOPE	OS/390 V2R8	Specifies the Scope ID of the Windows Internet Naming Service (WINS) server.
_IOE_SMB_SECONDARY_WINS	OS/390 V2R8	Specifies the IP address of the Windows Internet Naming Service (WINS) server that this server announces itself to and forwards WINS requests to if the Primary WINS server does not respond.
_IOE_SMB_TOKEN_FILE_MAX	OS/390 V2R8	Specifies the maximum number of files that the SMB token cache should keep tokens for.
_IOE_SMB_WINS_PROXY	OS/390 V2R8	Specifies whether this server can act as a Windows Internet Naming Service (WINS) server proxy.
_IOE_TKCGLUE_CACHE_SIZE	OS/390 V2R10	Specifies the number of file tokens for local HFS access that can be cached.

Table 11 (Page 4 of 4). Summary of New and Changed DFS/SMB Environment Variables

Environment Variable Name	Release	Description
_IOE_TKM_MAX_TOKENS	OS/390 V2R6	Specifies the number of tokens that can be held in the DFS server memory.

Additional DFS/SMB Migration Considerations

This section describes the considerations you should address before migrating to a new release of the Distributed File Service. The section assumes that you are installing the new release on a system where the DFS of SMB support is already configured and running.

Terminology

Refer to "About This Book" on page xv for a definition of terms used in this section that describes the components of the Distributed File Service

SMB File/Print Configuration

If you are using the SMB File/Print support, the migration to a new release of Distributed File Service does not require changes in the SMB configuration on the system where the new release is being installed.

Symbolic Link

Prior to OS/390 Version 2 Release 6, all DFS configuration files resided in a separate HFS data set mounted at **/usr/lpp/dfs/local**. Starting in OS/390 Version 2 Release 6, the DFS configuration files reside in the path **/etc/dfs**. The symbolic link **/opt/dfslocal** is created to reference the **/etc/dfs** directory. Other symbolic links that are created are listed in Table 13 on page 123.

Note: If you have replaced any symbolic links with user files, they will be deleted during DFS installation. You should save the file data before installing DFS. These files are listed in Appendix B, "Directories and Files" on page 123.

/etc HFS Customized Configuration Data

After a new release is installed on a system, the previous configuration data defined in the **/etc** directory can be used when running the new release. For the Distributed File Service DFS/SMB support, the configuration data is in the **/etc/dfs** directory. If you are using DCE, the DCE cell and other related DCE configuration data is in the **/etc/dce** directory.

After the new release is installed into a new **/etc** HFS, the *z/OS Program Directory* or *ServerPac: Installing Your Order* documentation instructs you on how to copy (migrate) the previous **/etc** data into the new **/etc** HFS. After migrating the previous data, the new **/etc** directory reflects new directories, new symbolic links, and the previous customization data.

After you IPL the new release, you are instructed by the *z/OS Program Directory* and the *ServerPac: Installing Your Order* documentation to run the **dfs_cpfiles** program. This program creates configuration files that are new in the release with default information.

Note: The same **/etc/dce** and **/etc/dfs** data can be used for different systems or DCE cells. You must install the new release of DCE and DFS on each system using the appropriate *z/OS Program Directory* and *ServerPac: Installing Your Order* documentation.

Refer to Chapter 3, “Installation” on page 23 for more information.

SecureWay Security Server RACF

RACF or equivalent security system entries are required for the DFS/SMB support and the users who can run them. Refer to “RACF Definitions” on page 29 for a description of the RACF entries required.

If you install the new release using the previous RACF database definitions, no changes are required for the DFS identifier mapping file defined by the **_IOE_MVS_IDMAP** or the **_IOE_SMB_ID_MAP** environment variables. If the new release uses a different RACF database, this mapping file must be updated to define the correct user identifiers. For more information, refer to the *z/OS Distributed File Service DFS Administration* for a description of mapping DCE user IDs to z/OS user IDS or to the *z/OS Distributed File Service SMB Administration* book for a description of mapping SMB work station user IDs to z/OS user IDs.

Exported Data

If you install DFS into a target image that can access the same exported file data as the production image, the preexisting **devtab**, **dfstab**, **rfstab**, and **smbtab** files in the **/opt/dfslocal/var/dfs** path can be used. If you do not want to export the production data files during the testing of the new OS/390 release on the test image, you must update these files to export only test data.

New Configuration or Startup Options

If you are migrating to a new release of the Distributed File Service, optionally update the **/etc/dfs** configuration or **envar** files with new parameters available in the release.

For information on what is new for this release of DFS, refer to “Migration Roadmap” on page 7.

Authorized Program

With OS/390 Version 2 Release 9, the list of authorized programs for the Distributed File Service includes:

- IOEGRWAG
- IOENEWAG
- IOESALVG
- SMBPW (**Note:** This applies to SMB File/Print support only).

Refer to the *z/OS Program Directory* and the *ServerPac: Installing Your Order* documentation for a description of the PARMLIB member updates required for the member IKJTSOxx.

DCE Cell, TCP/IP, and DCE/DFS Configuration

If you are using the DCE support for DFS, when migrating to a new release of the Distributed File Service, it is expected that the DCE cell, DCE configuration, and DFS is already configured on the system and that this configuration remains the same. The migration to a new release of DCE or DFS does not require DCE or DFS to be deconfigured.

If you decide to deconfigure DCE, then DFS must be deconfigured on the system before deconfiguring DCE. Refer to Chapter 7, “Deconfiguring DFS” on page 93 for more information.

If you have DCE or DFS configured on a system, you should not copy the **/etc/dce** or **/etc/dfs** and use them on another system or in another DCE cell. Refer to Appendix B, “Directories and Files” on page 123 for a list of the **/etc/dfs** files.

If you install DFS on a system where DCE and DFS is previously installed, you only need to review and update the **/etc/dfs** configuration files with the optional parameters that are new for this release after the **dfs_cpfiles** program is run during the migration process.

If the target image is in a different DCE cell or has a different TCP/IP address than that for the previous DCE and DFS configuration, it is a new instance of the DFS. DCE and DFS must be installed and configured as if it were a new installation.

For more information, refer to the *z/OS Distributed File Service DFS Administration* book for a description of mapping DCE user IDs to z/OS user IDS or to the *z/OS Distributed File Service SMB Administration* book for a description of mapping SMB work station user IDs to z/OS user IDs.

Test DCE Cell and System

The same **/etc/dce** and **/etc/dfs** data cannot be used for different system or DCE cells. You should install the new release on each system using the appropriate documentation, the *z/OS Program Directory* and the *ServerPac: Installing Your Order*. The release installation includes migrating the previous **/etc** HFS configuration files for use when running the new release. Therefore, after the new release is installed, the existing DCE and DFS configuration is not affected.

Chapter 3. Installation

The Distributed File Service is a base element of z/OS that includes SMB File/Print Server support and DFS support for DCE. Before using either the SMB support or DFS support, you must install the z/OS release including the Distributed File Service and the other base elements, using the appropriate release documentation, the *z/OS Program Directory* or the *ServerPac: Installing Your Order*. During the installation of the z/OS release, you must perform actions to activate either the DFS or SMB support. A review of all the z/OS installation steps and considerations that apply to the Distributed File Service are listed later in this chapter.

Please note:

- To use the SMB support, you only need to install the DCE Base Services base element as part of the z/OS release. To use SMB print serving support, you must be sure to install and configure the z/OS Infoprint Server Feature. To use hardware encryption capabilities for SMB password encryption, you must be sure to install the z/OS Open Cryptographic Services Facility (OCSF) as described in Step 2d on page 24.
- To use the DFS support for DCE you must be sure to install, configure, and run the DCE Base Services element on the system.

After the z/OS release installation and post-installation actions related to the Distributed File Service support are completed, to use either the SMB or the DFS support, you must configure the support on the system. Configuration includes administrative actions such as defining users and administrators; and defining the the client and server processes to start and processing options; and defining the file data for the server to export for work station access. Also, to use DFS support for DCE, you must be sure to configure and run DCE before configuring DFS.

This chapter contains the following information that can assist in the installation, post installation and configuration of the Distributed File Service. The information applies whether you plan to use the SMB support, the DFS support for DCE, and both the SMB and DFS support:

- “Review of Installation and Post Installation Steps”
- “Running the dfs_cpfiles Program” on page 27
- “RACF Definitions” on page 29
- “Installation Verification” on page 30.

Review of Installation and Post Installation Steps

The release installation and post installation actions for the Distributed File Service are identified in this section. The information can be used as a check list to insure that the Distributed File Service is properly installed as part of the z/OS release installation.

To install a new z/OS release, you must follow the instructions in the release documentation which consists of the *z/OS Program Directory* or the *ServerPac: Installing Your Order*. Use the release documentation as a primary source of installation instructions unless you are referred to other sources.

The DFS/SMB related installation actions and considerations are:

1. If the DFS client (**DFSCM**) or Distributed File Service server (**dfskern**) is running on the system, stop the client or server by following the applicable instructions in Chapter 8, “Starting and Stopping DFS Components” on page 107 or in the *z/OS Distributed File Service SMB Administration* book.
2. Be sure to follow all the z/OS release installation instructions in the release documentation applicable to the Distributed File Service. The sequence of steps in the check list may vary from the sequence of

steps presented in the release documentation but they are presented here to make sure you have completed them.

a. **/etc** HFS Considerations:

   - z/OS release installation creates **/etc/dfs** directories, sub-directories, and symbolic links. During post installation processing, configuration files are created in **/etc/dfs** sub-directories by the **dfs_cpfiles** program if the files do not already exist.
   - If you have DCE, DFS, or SMB configured on the system and you are installing a new release, make sure to backup the existing **/etc** HFS file system. After installing the new release into a new **/etc** HFS, you should merge the existing **/etc** HFS file system into the new **/etc** file system so that the content of the files in **/etc/dfs** file system reflect the previous DCE, DFS, or SMB configuration information. The new **/etc** file system can then be used to IPL the new release using the previous DFS and SMB configuration. After IPLing the new release, the **dfs_cpfiles** program can be run to create the new **/etc/dfs** configuration files for the release. Refer to the release documentation for instructions on how to migrate your existing **/etc** HFS file system to the new release.

b. DCE Installation Considerations:

   - If you plan to use only the SMB support, simply install the z/OS Base Distributed Computing Environment (DCE) support as part of the z/OS release. You do not need to configure or run DCE to use the SMB support.
   - If you plan to use the DFS support for DCE, be sure to install, configure, and run DCE of the system where the DFS server or DFS client will be run.
   - Refer to the release documentation for instructions on how to install DCE. Refer to the *z/OS DCE Administration Guide*, for instructions on how to configure DCE.

c. Infoprint Server Installation Considerations:

   If you plan to use the SMB print serving support, you must install and set up the z/OS Infoprint Server. See Step 19 on page 26 for more information.

d. Open Cryptographic Services Facility (OCSF) and Hardware Encryption Considerations:

   Use of OCSF for SMB password encryption support is optional. You only need to use the z/OS Open Cryptographic Services Facility (OCSF) Base component of the Cryptographic Services element if you want to exploit the hardware encryption capability available through the Integrated Cryptographic Services Facility (ICSF) using the Cryptographic Coprocessor Feature or the Integrated Cryptographic Feature (ICRF). Refer to Step 20 on page 26 for more information.

3. Define Target and Distribution Libraries

   Define the target and distribution libraries for the Distributed File Service using the sample job IOEISALC.

4. Set Up HFS Directories

   Make sure you set up the HFS directories by running the IOEISMKD job.

5. Define DDDEFs for DCE Base Services and Distributed File Service

   a. Make sure you define the DCE DDDEFs by running the EUVISDDD job.

   b. Make sure you define DFS DDDEFs by running the IOEISDDD job.

6. Do SMP/E APPLY of the new release.

7. Authorized Program Considerations:

   a. Make sure that the PARMLIB member IKJTSOxx is updated to reflect the authorized programs IOEGRWAG, IOENEWAG, IOESALVG, and IOESMBPW.

b. Update PARMLIB member LNKLSTxx or PROGxx, otherwise define the Distributed File Service SIOELMOD data set as an authorized data set.

8. Optional BPXPRMxx Updates for the DFS Client (**DFSCM**)

   If you plan to use the DFS client for DCE, you should consider making the BPXPRMxx updates that are described in “BPXPRMxx FILESYSTYPE TYPE(DFSC) Entry” on page 58 at this time.

9. Program Execution Environment Considerations

   The Distributed File Service requires the run-time library Language Environment® (LE), SCEERUN, to be available in the program search order by:

   a. Adding the Language Environment supplied run-time library, SCEERUN, to the startup procedures DFS, DFSCM, and DFSKERN using samples supplied in SIOEPROC.

      Note: The DFSCM procedure only applies to the DFS client support for DCE. It does not apply to the SMB support.

   b. Adding SCEERUN data set to your TSO logon proc by concatenating it to ISPLLIB DD statement and/or concatenate it to STEPLIB (the TSOLIB function of TSO/E could be used).

   c. Adding the "export STEPLIB=hlq.SCEERUN" in the **/etc/profile** file.

10. PROCLIB Member Considerations

    The DFS/SMB PROCLIB members in DDNAME=SIOEPROC are:

    a. IOEP0001 which is used to run the DFS/SMB server and other applicable server processes (daemons).

    b. IOEP0002 which is used to run the DFS client as a physical file system with FILESYSTYPE TYPE(DFSC) in a separate colony address space.

    c. IOEP0003 which is used to run the DFS/SMB DFSKERN process in a separate address space from the address space where the other DFS server processes run.

    You must be sure to review and update PROCLIB members for the Distributed File Service and related elements as described in the release documentation.

    Notes:

    a. SIOEPROC must either be fully copied to SYS1.PROCLIB or added to a PROCLIB concatenation accessible to JES.

    b. Be sure to maintain the ALIAS names for these procedures if they are copied from the distribution library to another library. Alias definitions can be maintained by using the IEBCOPY utility when copying a PROCLIB containing aliases.

11. ISPF Considerations:

    Set up ISPF for the Distributed File Service by:.

    a. Adding the libraries DDDEF SIOEMSGE to DDNAME ISPMLIB;

    b. Adding DDDEF SIOEPNLE to DDNAME ISPPLIB;

    c. Adding DDDEF SIOEEXEC to DDNAME SYSEXEC; and

    d. For NLV (National Language) support, adding DDDEF SIOEMSGJ to DDNAME ISPMLIB and DDDEF SIOEPNLJ to DDNAME ISPPLIB.

12. DFS Client Considerations:

    If you are installing the DFS client on a system where the DFS client for DCE is already running, it is recommended that you update the **/opt/dfslocal/etc/CacheInfo** file to specify memory caching before you IPL the new release. Refer to “Completing the DFSCM Installation and Configuration” on page 58 for more information.

13. HFS **/etc** File System Considerations Before IPL

    Refer to Step 2a on page 24 for a description of the **/etc** HFS considerations for the Distributed File Service. Be sure to follow the release documentation for instructions on how to migrate your **/etc** HFS file system for use when you IPL the new release.

14. IPL the new release.

15. DCE Customization Considerations:

    If you plan to use the Distributed File Service DFS client or server support for DCE, insure that you review and complete the following customization actions described in the release documentation:

    a. DCE IDL REXX EXEC Customization
    b. Define DCEKERN to RACF
    c. Define the DCE Administrator User ID to RACF the DCEKERN.START.REQUEST Resource Profile as described under DFS/SMB Customization Considerations.

16. DFS/SMB RACF Customization Considerations:

    To use the DFS client or server and/or the SMB File/Print support, you perform the actions described in “RACF Definitions” on page 29.

    If you use SMB print serving support, refer to the additional RACF requirements identified in Step 19.

    If you want to exploit hardware encryption capabilities for SMB password encryption, refer to the additional RACF requirements identified in Step 20.

17. Create **/etc/dfs** Configuration Files

    Run the **dfs_cpfiles** program to create any configuration files that do not exist in the HFS **/etc/dfs** directory and sub-directories. Refer to “Running the dfs_cpfiles Program” on page 27 for more information.

18. SMB Password Encryption Customization

    If you want to use SMB password encryption support, you must specify **_IOE_SMB_CLEAR_PW=ALLOW** or **NOTALLOWED** in the **/etc/dfs/home/dfskern/envar** file created by the **dfs_cpfiles** program.

    If you do not want to use hardware encryption for SMB password encryption, you should also specify **_IOE_SMB_OCSF=OFF** in the **/etc/dfs/home/dfskern/envar** file. Refer to Step 2d on page 24 for more information on hardware encryption.

    If you want to use hardware encryption or OCSF for SMB password encryption, Step 20 instructs you on the action required.

19. Infoprint Server Considerations:

    If you plan to use the SMB print serving support, you must consider the following:

    a. Since the DFS/SMB server runs as an APF-authorized server, ensure that the DLL libraries that reside in **/usr/lpp/Printsrv/lib** are installed as APF-authorized. The OMVS **extattr +a** command can be used to define a library as APF-authorized.

    b. The Infoprint Server library **/usr/lpp/Printsrv/lib/aopapi.dll** is installed with the program control bit set. This is required if the SIOELMOD is included in the program control environment.

    c. Make sure that the **/etc/dfs/home/dfskern/envar** file is updated to add **/usr/lpp/Printsrv/lib** to the LIBPATH environment variable.

    d. For more information on using the Infoprint Server including any additional RACF requirements, refer to *z/OS Infoprint Server Customization*, S544-5744.

20. OCSF Considerations:

As described in Step 2d on page 24, use of OCSF for SMB password encryption is optional.

If you want to use hardware encryption capabilities or OCSF for SMB password encryption, you need to insure that OCSF is installed and configured.

a. Program Control:

   In a program control environment, use of OCSF services requires the Distributed File Service SIOELMOD distribution library be program controlled. In turn, this requires that the Infoprint Server library to be program controlled as described in Step 19 on page 26. Refer to *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683, for more information on Program Control.

b. Since the DFS/SMB server runs as an APF-authorized server, ensure that the OCSF DLLs are installed as APF-authorized. The OMVS **extattr +a** command can be used to define a library as APF-authorized.

c. Ensure that the **/opt/dfslocal/home/dfskern/envar** file has a LIBPATH that adds the directory (usually the directory **/usr/lib**) that contains the OCSF DLLs.

d. If SMB password encryption support is being used, authorize the user DFS (or other installation defined user ID) to the Open Cryptographic Services Facility (OCSF) CDS.CSSM.CLASS class profile and, if it exists, the BPXSERVER class profile. security considerations for OCSF.

e. Refer to the *z/OS Open Cryptographic Services Facility Application Programming*, SC24-5899, for more information on OCSF DLL libraries, Program Control requirements, and security considerations.

21. Run the installation verification (IVP) procedure. Refer to Step 22 for more information.
22. After installation verification, the DFS client (**DFSCM**) or Distributed File Service server (**dfskern**) may be running. Before performing any post installation configuration, be sure to stop the DFS client or server by following the applicable instructions in Chapter 8, “Starting and Stopping DFS Components” on page 107 or in the *z/OS Distributed File Service SMB Administration* book.
23. Proceed to configure SMB File/Print support described in the *z/OS Distributed File Service SMB Administration* book.
24. Proceed to configure DFS support as described in Chapter 4, “Configuring DFS for DCE” on page 33.

Running the dfs_cpfiles Program

The **/opt/dfsglobal/scripts/dfs_cpfiles** program is a z/OS shell script that creates customizable configuration files in **/opt/dfslocal** subdirectories. **dfs_cpfiles** copies IBM-supplied files from the **/opt/dfsglobal/examples** directory to **/opt/dfslocal** subdirectories. The **dfs_cpfiles** program will create files that do not exist. It will not replace an existing file to preserve any installation configuration data from a previous release.

/opt/dfslocal is a symbolic link to **/etc/dfs**. So, all the files created by the **dfs_cpfiles** program are actually created in the **/etc** file system. Refer to Appendix B, “Directories and Files” on page 123 for more information on the symbolic links defined to identify the configuration files.

Although some configuration files can be created in other directories and identified by **envar** specifications, it is recommended that all the configuration files reside in the **/opt/dfslocal** subdirectories in the **/etc** file system.

To invoke **dfs_cpfiles**:

1. Log in as **root** on the local machine. In DFS/SMB, this means as a user with a **UID = 0**.

2. While in the z/OS shell environment, invoke the DFS/SMB Default Configuration Files Creation Program by entering the following:

   ```
   $ dfs_cpfiles
   ```

3. The following example output shows the output after using **dfs_cpfiles** to create all the customizable DFS configuration files:

```
************************************************************************
**  Create OS/390 Distributed File Service                            **
**  Configuration Files                                               **
************************************************************************

    Create envar Files....
       File /opt/dfslocal/home/bakserver/envar created
       File /opt/dfslocal/home/boserver/envar created
       File /opt/dfslocal/home/butc01/envar created
       File /opt/dfslocal/home/butc02/envar created
       File /opt/dfslocal/home/butc03/envar created
       File /opt/dfslocal/home/butc04/envar created
       File /opt/dfslocal/home/butc05/envar created
       File /opt/dfslocal/home/butc06/envar created
       File /opt/dfslocal/home/butc07/envar created
       File /opt/dfslocal/home/butc08/envar created
       File /opt/dfslocal/home/daemonct/envar created
       File /opt/dfslocal/home/dfscm/envar created
       File /opt/dfslocal/home/dfscntl/envar created
       File /opt/dfslocal/home/dfsexport/envar created
       File /opt/dfslocal/home/dfskern/envar created
       File /opt/dfslocal/home/flserver/envar created
       File /opt/dfslocal/home/ftserver/envar created
       File /opt/dfslocal/home/growaggr/envar created
       File /opt/dfslocal/home/newaggr/envar created
       File /opt/dfslocal/home/repserver/envar created
       File /opt/dfslocal/home/salvage/envar created
       File /opt/dfslocal/home/upclient/envar created
       File /opt/dfslocal/home/upserver/envar created

    Create Miscellaneous Configuration Files....
       File /opt/dfslocal/etc/ioepdcf created
       File /opt/dcelocal/etc/Cacheinfo created
       File /opt/dfslocal/var/dfs/devtab created
       File /opt/dfslocal/var/dfs/dfstab created
       File /opt/dfslocal/var/dfs/rfstab created
       File /opt/dfslocal/var/dfs/smbtab created
       File /opt/dfslocal/var/dfs/cmattr created
       File /opt/dfslocal/var/dfs/hfsattr created
       File /opt/dfslocal/home/dfskern/dfsidmap created
       File /opt/dfslocal/home/dfskern/smbidmap created
```

Notes:

1. The previous example shows the **dfs_cpfiles** messages when all files are created.

   If a file already exists, an example of this message is:

   ```
   File /opt/dfslocal/etc/ioepdcf already exists
   ```

   If an error occurs creating the file, an example of this message is:

   ```
   File /opt/dfslocal/etc/ioepdcf not created
   ```

2. If you are migrating to this release of DFS/SMB from an earlier release of DFS and **dfs_cpfiles** created new customizable configuration files, you may need to add any previous user customization data to the newly created files.
3. If you are migrating to this release of DFS/SMB from an earlier release of DFS and new customizable configuration files were not created by **dfs_cpfiles**, you may want to update pre-existing customizable files with any new customization options available with this release of DFS. Refer to Chapter 2, “Migration Overview” on page 5 for more information on what is new in this release.
4. The files created by the **dfs_cpfiles** program contain only sample entries or comments. The content of these files can be replaced with installation information.
5. The DFS user identity mapping file is identified by the environment variable **_IOE_MVS_IDMAP**. It is recommended that the **/opt/dfslocal/home/dfskern/dfsidmap** file created by the **dfs_cpfiles** program is used by the installation for user identity mapping.

   The SMB user identity mapping file is identified by the environment variable **_IOE_SMB_IDMAP**. It is recommended that the **/opt/dfslocal/home/dfskern/smbidmap** file created by the **dfs_cpfiles** program is used by the installation for user identify mapping.
6. Refer to Chapter 4, “Configuring DFS for DCE” on page 33 for more information on how to customize these files.

After all the configuration files required by this DFS release have been created, you can proceed to perform the additional DFS configuration steps in “Review of Installation and Post Installation Steps” on page 23.

RACF Definitions

DFS/SMB clients and servers must be defined to the z/OS SecureWay Security Server's RACF (or equivalent security product). The clients and servers must have the proper authorization to initialize and perform required system services. Note that access to z/OS file data is performed using the authorization of the user accessing the data or the default authorization defined by the installation. To define DFS/SMB to RACF you must create the following definitions with these exact names.

- Define **DFSGRP** as a group.
- Define **DFS** as a user.
- Define **DFS** as a started task.
- Define **DFSCM** as a started task. This definition is only required for the DFS client for DCE.
- Define **DFSKERN** as a started task.

The following commands can be used to update RACF:

```
ADDGROUP DFSGRP SUPGROUP(SYS1) OMVS(GID(2))
ADDUSER DFS OMVS(HOME(/opt/dfslocal/home/dfsntl) UID(0))  see notes 1 and 2
  DFLTGRP(DFSGRP) AUTHORITY(CREATE) UACC(NONE)
RDEFINE STARTED DFS.** STDATA(USER(DFS))                  see note 3
RDEFINE FACILITY DFSKERN.START.REQUEST UACC(NONE)         see note 3
PERMIT DFSKERN.START.REQUEST CLASS(FACILITY) ID(DFS)
  ACCESS(UPDATE)                                          see note 4
RDEFINE STARTED DFSCM.** STDATA(USER(DFS))                see note 5
SETOPTS CLASSACT(FACILITY)                                see note 6
SETOPTS RACLIST(FACILITY) REFRESH                         see note 7
```

Notes:

1. The home directory for the user ID running the DFS/SMB server can be overridden by the **_EUV_HOME** environment variable value specified on the DFSCNTL PARM as shown in the IOEP0001 (Alias DFS) procedure.
2. The user ID DFS should not be enabled for DCE single sign-on. Also, an equivalent user ID can be used but the use of ID(DFS) is recommended. If a different user ID is used, it must be reflected in all the RACF commands shown.
3. The RDEFINE for DFS and DFSKERN applies to both the DCE DFS and SMB support.
4. Define **DFSKERN** as a started task to run this DFS/SMB server process in an address space that is separate from the address space where the other server processes run. Running **DFSKERN** in a separate address space is the recommended structure for DFS/SMB server processes.
5. The RDEFINE for DFSCM applies to the DFS client support for DCE and is not required for the SMB support.
6. The SETOPTS CLASSACT(FACILITY) command activates the FACILITY class if it is not active.
7. After all z/OS Security Server (RACF) definitions have been made, the FACILITY class must be refreshed if it is RACLISTED. The SETOPTS RACLIST(FACILITY) REFRESH command refreshes the FACILITY class.

For more information about RACF, see the following documentation:

- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS SecureWay Security Server RACF Command Language Reference*, SA22-7687.

Installation Verification

Notes:

1. DCE must be installed as part of the Wave 2 installation. DCE customization, configuration, and DCE Kernel initialization is not required to verify the installation of the Distributed File Service.
2. The term DFS server is used to describe the Distributed File Service server whether it is run to provide distributed file serving support for DCE or SMB file/print serving support.

The installation verification procedure is normally used to start the DFS server for the first time during the installation step of a new release. But, if the DFS server is already running, be sure to stop the DFS server before following the installation verification sets described below.

To ensure the installation completed successfully, perform the following steps:

1. Start the DFS started task:

   ```
   S  DFS,PARM='-nodfs'
   ```

   View SYSLOG and look for the following message:

   ```
   IOEP01103I DFS kernel initialization complete.
   ```

   This message indicates the DFS server started successfully. Additional configuration is required before the server can be used as a DFS (DCE distributed file service) or an SMB file/print server as described below. If the DFS server started task does not initialize successfully, contact the IBM Support Center for help.
2. If the BPXPRMxx entry was updated with the FILESYSTYPE TYPE(DFSC), determine if the DFS client (DFSCM) is started. To do this, view SYSLOG and look for the following message:

```
IOEC04183I dfsd: initialization complete
```

This message indicates that the DFS client detected that DCE is running on the system and the DFS client is ready to route requests to a DFS server. If this message does not appear in SYSLOG, be sure DCE is running on the system.

The following message in SYSLOG indicates a possible error:

```
nn BPXF014D FILESYSTYPE DFSC Terminated, Reply 'R' when ready to restart
```

The possible cause is that DCE was started on the system but subsequently failed. If DCE is running, contact the IBM Support Center for help.

After the installation verification is completed, you should continue with the installation as described in Step 22 on page 27.

Chapter 4. Configuring DFS for DCE

The purpose of this chapter is to assist you in creating and configuring the Distributed File Service. Distributed File Service provides an ISPF dialog to assist you. The use of these panels is explained in this chapter. For information regarding planning considerations that should considered prior to configuring your system, refer to the *z/OS Distributed File Service DFS Administration* book for a description of DCE cells and the DCE global namespace.

For information on setting up the intercell environment, managing intercell naming, and administering a multicell environment, see the *z/OS DCE Administration Guide*.

This chapter provides information on the following topics:

- "Overview of DFS Configuration"
- "DFS Configuration Steps" on page 34
- "Using the DFS Configuration Program (DFSCONF)" on page 39
- "Using DFSCONF to Configure DFS Servers" on page 43
- "Using DFSCONF to Configure the DFS Client" on page 45
- "Completing the DFS File Server Configuration" on page 47
- "Completing the boserver Configuration" on page 47
- "Completing the FLDB Server (flserver) Configuration" on page 47
- "Creating the DFS File Server FLDB Entry" on page 49
- "Completing Fileset Server (ftserver) Configuration" on page 49
- "Completing Replication (repserver) Server Configuration" on page 51
- "Completing Update Client (upclient) Configuration" on page 51
- "Completing Update Server (upserver) Configuration" on page 52
- "Completing the Backup Database Server (bakserver) Configuration" on page 53
- "Completing the DFSCM Installation and Configuration" on page 58
- "Customizing the ioepdcf File" on page 64
- "Customization of the DFS Server and the DFSCM envar Files" on page 65.

Notes:

1. The DCE cell name (*dcecellname*) and DCE host name (*dcehostname*) used for DCE configuration must be the same *dcecellname* and *dcehostname* used to configure DFS.
2. DFS configuration command examples sometimes use **/.:** to denote **/.../***dcecellname*.

Overview of DFS Configuration

Configuring DFS in a DCE cell takes several steps. You should first configure DCE on the system. Then you will decide whether or not you are going to configure the DFS client (Cache Manager) and/or the DFS server on the system. If you decide to configure the DFS server, you also need to decide which of the DFS server processes (daemons) to configure for DCE. Refer to the section that discusses the DFS configuration issues in the *z/OS Distributed File Service DFS Administration* book to assist you in the DFS client and server decisions before proceeding with DFS configuration steps.

DFS Configuration Steps

The DFS configuration steps reference detailed explanations in other sections of this publication. Once you complete a step and its supporting instructions, you should refer to the next step in this list.

1. Before Proceeding with DFS Configuration

   Note: The following important information is required before proceeding with the DFS configuration steps.

   a. **Install DCE on the Host System**

      Insure that DCE is installed on the host system where DFS is configured, following the instructions in the *z/OS Program Directory*.

   b. **Install DFS on the Host System**

      Insure that DFS is installed on the host system where DFS is configured, following the instructions in the *z/OS Program Directory*.

   c. **Insure DCE is Running on the Host System**

      Insure that the host system where DFS will be configured is already configured into a DCE cell and that DCE is running on the host system. Refer to the *z/OS DCE Configuring and Getting Started* book for more information.

      Attention: If you configure a DCE cell with an X.500-style name and plan to run DFS, you must also configure GDS on the host system. If you do not, you cannot access the local cell's space.

   d. **Complete DFS Post Installation Processing**

      Insure that the steps described in Chapter 3, “Installation” on page 23 have been completed successfully.

   e. **Determine DCE Login Information**

      You must log in as **cell_admin** to perform DFS configuration so you will need to know the correct **cell_admin** login password before proceeding. Contact the person who initially configured DCE on the host system or the DCE administrator for this information.

   f. **Determine DCE Cell Name and DCE Host Name**

      You must also know the correct **dcecellname** and **dcehostname** defined during DCE configuration. After DCE is configured on the host system, this information is defined in the **/opt/dcelocal/dce_cf.db** file.

   g. **Add DFS Administrative Principals to DCE Registry**

      This step only applies if DFS servers run on the host system where DFS is being configured.

      If this is not already done, it is recommended that you register at least one user or group principal to be used in all DFS administrative lists. You can register more administrative users or groups at a later time. If you define a group for DFS administrative authorization, you can add users to the group now or more at a later time. It is recommended that the user or group you now select be used in all the DFS administrative lists. Later, different users or groups can be selectively added to the various DFS administrative list files. To register a user or group in a DCE cell, refer to the instructions in the *z/OS DCE Configuring and Getting Started* book or in the *z/OS DCE Administration Guide*.

   h. **Stop DFS and the DFS Client**

      The z/OS installation verifying instructions call for starting the DFS client and server address spaces. It is recommended that you stop DFS processing on the host system during configuration processing.

If DFS (the DFS server) is already running, issue the following system command:

```
stop dfs
```

If the DFS Client is already running, issue the following system command:

```
stop dfscm
```

i. **Validate that DCE is Configured for DFS on the Host System**

   You should validate that DCE is properly configured on the host system where DFS configuration is performed.

   From OMVS on the host system, you can issue the following commands to receive the output shown, if the DCE cell is properly configured for DFS.

   ```
   dcecp -c show rpcentry /.:/subsys/dce/dfs/bak
   dcecp -c show rpcentry /.:/fs
   ```

   If either command results in the message indicating that the name service object is not found, then the DCE cell is not configured for DFS. You must contact the DCE cell administrator to determine the status of the DCE cell (see the *z/OS DCE Configuring and Getting Started* book or the *z/OS DCE Administration Guide*).

2. **Set z/OS shell Flag**

   If you plan to configure the DFS Client (**DFSCM**) on the host system, it is recommended that the UNIX System Services shell's logical flag be set using the **set -o logical** command. See the *z/OS Distributed File Service DFS Administration*book, for more information about displaying the current working directory from the shell with the **DFSCM** configured. Also, see the *z/OS UNIX System Services Command Reference*, SA22-7802, for more information about the **set** command.

3. **Set Up to Use the DFSCONF Command**

   Refer to “Using the DFS Configuration Program (DFSCONF)” on page 39, for setting up information necessary to use the **DFSCONF** command from TSO.

4. **Optionally Modify the DFS envar Files**

   Default DFS process **envar** files are created by **dfs_cpfiles** during DFS post installation (see Chapter 3, “Installation” on page 23). You can use these default **envar** files to configure DFS and customize them at a later time. Instructions on how to modify the DFS **envar** files can be found in “Customization of the DFS Server and the DFSCM envar Files” on page 65.

5. **Use DFSCONF to Configure the DFS Client (DFSCM)**

   Optionally, you can configure the **DFSCM** for use by OMVS users on the host system. If you do not want to configure the **DFSCM** on the host system at this time, proceed to step 6.

   In Distributed File Service, the **DFSCM** does not require any of the DFS servers to be configured and running on the z/OS system on which it is configured and running. But, if you want to configure the DFS client and DFS servers on the host system, it is recommended that you configure both at this time using **DFSCONF**.

   For information on configuring a DFS client, see “Using DFSCONF to Configure the DFS Client” on page 45.

   If you choose to also configure DFS servers using **DFSCONF** at this time, proceed with the following step.

   Note: If no DFS servers will be configured on the host system, after configuring the DFS client using **DFSCONF**, you can now refer to step 10 on page 36, **Complete DFSCM Configuration**, and skip the DFS server configuration steps.

6. **Use DFSCONF to Configure z/OS DFS Servers**

Optionally, you can configure DFS servers on the host system.

It is recommended that you configure all DFS servers you initially want on the host system at this time using **DFSCONF** if not already done previously. For detailed information, see

- “Using DFSCONF to Configure the DFS File Server” on page 44
- “Using DFSCONF to Configure the DFS Backup Server” on page 44
- “Using DFSCONF to Configure the DFS Fileset Location Database Server” on page 45.

7. **Modify the ioepdcf File and Complete boserver Configuration**

   This step only applies to DFS server configuration.

   The default **/opt/dfslocal/etc/ioepdcf** file created during DFS post installation processing by **dfs_cpfiles** (see Chapter 3, “Installation” on page 23) is set up to automatically start only **dfskern** when the DFS server address space is next initialized. It is recommended that you update the **ioepdcf** file to automatically start the **boserver**. This is the recommended setting for subsequent DFS server configuration processing. If you want to change the default **ioepdcf** file settings, refer to “Customizing the ioepdcf File” on page 64 for more information.

8. **Start the DFS Server**

   This step only applies to DFS server configuration.

   Initialize the DFS server address space by issuing the following system command:

   ```
   start dfs
   ```

   Notes:

   a. Only the required DFS server processes **dfskern** and **boserver** will be started if the recommended **/opt/dfslocal/etc/ioepdcf** settings are used (see step 7).

   b. Starting the **boserver** creates empty **/opt/dcelocal/var/dfs/admin.bos** and **/opt/dcelocal/var/dfs/BosConfig** files, if those files do not already exist.

   c. DFS should be running with DFS authorization disabled by the existence of the file **/opt/dcelocal/var/dfs/NoAuth**.

9. **Create Initial DFS Administrative Authorization Lists**

   This step only applies to DFS server configuration.

   You can now define the DFS administrators for the DFS servers you will run on the DCE host system by updating the DFS administrative list file for each server.

   At this time, it is recommended that you only update all the DFS administrative list files on this DCE host system with a single user ID that has **cell_admin** authority. Note, the DCE **cell_admin** user ID inherits no special DFS authorization and has to be explicitly specified in a DFS administrative list file.

   In case you are configuring this DCE host system into an existing DCE cell and another system is acting as a DFS System Control Machine (where the **upserver** is running), the user ID that is specified in the DFS administrative list file on this DCE host should also be one that is specified in the DFS administrative list file on the DFS System Control Machine. Later, you will be instructed when to update the DFS administrative list files with additional users and user groups and to perform updates of the DFS administrative list files on the DCE host system where the **upserver** is running.

   Refer to “Creating the Initial DFS Administrator(s)” on page 72, “Defining a DFS Administrator” on page 70 and “Adding Principals and Groups to Administrative Lists” on page 74 for more information.

10. **Complete DFS Client Configuration**

If the DFS Client (**DFSCM**) was configured on the host system using **DFSCONF**, you must complete configuring and start the **DFSCM** as described in “Completing the DFSCM Installation and Configuration” on page 58.

11. **Complete flserver Configuration**

    There must be a Fileset Location Database Server (**flserver**) running in the DCE cell. If there is no **flserver** running in the DCE cell or if you you want an **flserver** to also run on this DCE host system, follow the instructions for “Completing the FLDB Server (flserver) Configuration” on page 47.

12. **Create FLDB Entry for DFS File Server**

    After an **flserver** is running in the DCE cell (on this host or another host in the DCE cell), you must follow the instructions in “Creating the DFS File Server FLDB Entry” on page 49.

13. **Complete ftserver Configuration**

    If you have configured the DFS File Server using **DFSCONF**, you must configure a DFS Fileset Server (**ftserver**) if you plan to export any file data from this DCE host system. See “Completing Fileset Server (ftserver) Configuration” on page 49 for more information.

14. **Complete repserver Configuration**

    If you have configured the DFS File Server using **DFSCONF** and if you plan to run a DFS Replication Server (**repserver**) on this DCE host system, you must follow the instructions in “Completing Replication (repserver) Server Configuration” on page 51.

    Note: If you plan to define a **root.dfs** fileset for the DCE cell on this DCE host system, you must configure a **repserver** on this DCE host.

15. **Complete upclient Configuration**

    If you have configured the DFS File Server using **DFSCONF** and if you plan to run a DFS Update Client (**upclient**) on this DCE host system, you must follow the instructions in “Completing Update Client (upclient) Configuration” on page 51.

    Note: If an **upserver** is already defined in the DCE cell, you should configure an **upclient** on this DCE host system. Otherwise, you should not configure an **upclient** and configure an **upserver** on this DCE host system as described in step 16.

16. **Complete upserver Configuration**

    If you have configured the DFS File Server using **DFSCONF** and if you plan to run a DFS Update Server (**upserver**) on this DCE host system, you must follow the instructions in “Completing Update Server (upserver) Configuration” on page 52.

    Note: If an **upserver** is not already defined in the DCE cell, you should configure an **upserver** on this DCE host system if you plan to have DFS running on more than one DCE host system in the DCE cell.

17. **Complete bakserver Configuration**

    If you have configured the DFS Backup Server using **DFSCONF** and if you plan to run a DFS Backup Server (**bakserver**) on this DCE host system, you must follow the instructions in “Completing the Backup Database Server (bakserver) Configuration” on page 53.

    Note: If you do not plan to use the DFS backup facility or if a **bakserver** is already defined in the DCE cell, you do not need to define one on this DCE host system.

18. **Complete butc Configuration**

    If you have configured the DFS Backup Server using **DFSCONF** and if you plan to run at least one DFS Backup Tape Coordinator (**butc**) daemon on this DCE host system, you must follow the instructions in “Completing the DFS Backup Tape Coordinator Configuration” on page 55.

Note: If you want to use the DFS backup facility for DCE Local File System data stored on z/OS, you should configure at least one **butc** server on the host system.

19. **Customize the ioepdcf File**

    It is recommended that you can now customize the **ioepdcf** file to indicate that all servers you have configured should be started automatically when the DFS address space is started. (see "Customizing the ioepdcf File" on page 64).

20. **Restart DFS and Start DFS Client**

    It is recommended that you stop and restart DFS processing at this time. For detailed information, see Chapter 8, "Starting and Stopping DFS Components" on page 107.

    a. To stop DFS (all DFS servers running in the DFS address space), issue the following system command:

    ```
    stop dfs
    ```

    b. If the DFS Client is already running, issue the following system command:

    ```
    stop dfscm
    ```

    c. Issue the following system command to start the DFS address space:

    ```
    start dfs
    ```

    d. Restart the DFS Client following the instructions in "Stopping and Restarting the DFSCM" on page 109.

21. **Create Additional DFS Administrators**

    This step only applies to DFS server configuration.

    You can now define additional DFS administrators for the DFS servers you will run on the DCE host system by updating the DFS administrative list file for each server.

    If this DCE host system was added to an existing DCE cell, and there is an **upclient** running on this system, you should update the DFS administrative list files on the DCE host machine where the **upserver** is running. The **upclient** and **upserver** processing will propagate the DFS administrative list file updates back to administrative list files located on this DCE host machine.

    Refer to "Defining a DFS Administrator" on page 70 and "Adding Principals and Groups to Administrative Lists" on page 74, for more information.

22. **Define a root.dfs Fileset for the DCE Cell**

    This step applies to configuring DFS servers on z/OS.

    Each DCE cell requires a **root.dfs** fileset for DFS. If a **root.dfs** fileset does not already exist for the DCE cell into which the host system was configured, one must be created.

    The **ftserver** and **rpserver** must be configured and running on the host system before proceeding to the instructions in Chapter 6, "Exporting Data in DFS" on page 77.

23. **Create and Export a DCE Local File System Aggregate**

    If you want to export file data for general use that is stored in a DCE Local File System aggregate on the host, refer to Chapter 6, "Exporting Data in DFS" on page 77.

24. **Setting Up z/OS User IDs for HFS and RFS Access**

    If you will run the DFS File Server on a z/OS system and plan to export HFS or RFS data sets, see "Mapping z/OS User IDs for HFS and RFS Access" on page 75.

25. **Create and Export General Use HFS Aggregate**

If you want to export file data for general use that is stored in an HFS aggregate on the host, refer to Chapter 6, “Exporting Data in DFS” on page 77.

26. **Create and Export General Use RFS Aggregate**

    If you want to export file data for general use that is stored in an RFS aggregate on the host, refer to Chapter 6, “Exporting Data in DFS” on page 77.

27. **Customize the DFS envar Files**

    Default **envar** files are created by **dfs_cpfiles** during post installation processing if they did not exist (see Chapter 3, “Installation” on page 23). At this time, you may want to follow the instructions in “Customization of the DFS Server and the DFSCM envar Files” on page 65.

Using the DFS Configuration Program (DFSCONF)

Use **DFSCONF** from TSO to configure the Distributed File Service servers and the DFS client. The **DFSCONF** configuration program has interactive panels that prompt and guide you through the details of configuring the Distributed File Service servers and the DFS client.

The Distributed File Service Server Configuration program automates the DCE configuration (and deconfiguration) of the DFS servers and creation of symbolic links required by the DFS Client.

DFSCONF uses the DCE administrative facilities to enter configuration information in the Security Registry and the CDS namespace. These facilities are the Registry Editor, ACL Editor, RPC Control Program, CDS Control Program, and the DCE Control Program. You must be properly authenticated and authorized by the DCE Security Service.

In the DFSCONF panels, you will either be prompted to select from a list of options or to enter a value. To select from a list of options, enter the number corresponding to the desired function in the selection field. For panels that require values, enter the appropriate response in the input field.

Details on using these panels are described in this chapter.

Note: If you have previously configured DFS on this DCE host system, running **DFSCONF** does not alter any DFS administrative file that may exist in the **/opt/dcelocal/var/dfs** directory.

DCE Messaging Subsystem Environment Variables for DFSCONF

The **DFSCONF** configuration program uses the DCE messaging facility. The DCE messaging facility is controlled by the environment variables **_EUV_SVC_MSG_LOGGING** and **_EUV_SVC_MSG_LEVEL**.

It is recommended that when you run **DFSCONF**, **_EUV_SVC_MSG_LOGGING** should be set to **CONSOLE_LOGGING** and **_EUV_SVC_MSG_LEVEL** should be set to **VERBOSE** (the default value).

Refer to “Setting DFSCONF Environment Variables” on page 41 for more information. For more information on specifying the DCE messaging facility environment variables refer to the *z/OS DCE Administration Guide*.

Configuration Log File for DFSCONF

The **DFSCONF** program appends messages to the **dfsconf.log** file that detail the steps that it performs in configuring or deconfiguring DFS on the host system. If the **dfsconf.log** file does not exist, it is created by **DFSCONF**.

By default, the **dfsconf.log** file resides in the home directory of the user running **DFSCONF**. The environment variable **_IOE_CFG_LOG_FILE** can be used to control the location of the **dfsconf.log** file. If the default is not used, be sure that the user running **DFSCONF** has write access to the specified **dfsconf.log** file.

The environment variable, **_IOE_CFG_INFORM_LEVEL** should be set to **2** during DFS configuration. This setting causes each DFS configuration or deconfiguration to generate approximately 7K bytes of message text. Insure that the **dfsconf.log** file resides in the file system that has sufficient free space to contain the messages.

Refer to “DFSCONF Environment Variables” for more information. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example of the content of a **dfsconf.log** file.

DFSCONF Environment Variables

The **DFSCONF** configuration program uses environment variables to set the default values for host information. The environment variables are declared in the **envar** file in the home directory of the administrator running **DFSCONF**. If set, the values of these environment variables are displayed as default values in the DFS configuration panels. If the environment variables are not set, the appropriate information must be entered on the **DFSCONF** configuration panels.

Following is a brief description of the environment variables that apply when running the **DFSCONF** program:

Variable **Description**

_EUV_SVC_MSG_LOGGING Defines where DCE messages are written. The default value **CONSOLE_LOGGING** is used if this is not specified.

_EUV_SVC_MSG_LEVEL Controls the types of DCE messages generated. The default value **VERBOSE** is used if this is not specified.

_IOE_CFG_INFORM_LEVEL Controls the types of DFS messages generated. The default value **1** is used if this is not specified. The value **2** is recommended when running **DFSCONF**.

_IOE_CFG_LOG_FILE Defines the name of the DFS configuration log file where DFS messages are written. The default value **dfsconf.log** is used if this variable value is not specified. By default, the file is located in the home directory of the administrator running **DFSCONF**.

_IOE_CFG_CELL_ID Defines the DCE principal name of the administrator who is running **DFSCONF**. The default value **cell_admin** is used if this is not specified.

NLSPATH Defines the search path used by DCE for the DCE and DFS message catalogs. The default value, **/usr/lib/nls/msg/En_US.IBM-1047/%N**, is used if this is not specified.

Refer to Appendix C, “DFS/SMB Environment Variables” on page 129 for more information on these environment variables.

Important Note About the DCE Cell Name and DCE Host Name

It is critical that the DCE cell name and the DCE host name, specified on the **DFSCONF** configuration panels, are correct. They must be the same names used to configure DCE. DFSCONF displays the DCE cell name and the DCE host name obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**.

Setting DFSCONF Environment Variables: The **DFSCONF** environment variables are declared in the **envar** file in the home directory of the administrator performing the Distributed File Service configuration. Environment variables are set using the following syntax:

```
VARIABLE_NAME=value
```

Following is an example of the entries in the **envar** file for the **DFSCONF** program:

```
_EUV_SVC_MSG_LOGGING=CONSOLE_LOGGING
_EUV_SVC_MSG_LEVEL=VERBOSE
_IOE_CFG_INFORM_LEVEL=2
_IOE_CFG_LOG_FILE=dfsconf.log
_IOE_CFG_CELL_ID=cell_admin
NLSPATH=/usr/lib/nls/msg/En_US.IBM-1047/%N
```

Set Up the DFSCONF Administrator TSO User ID

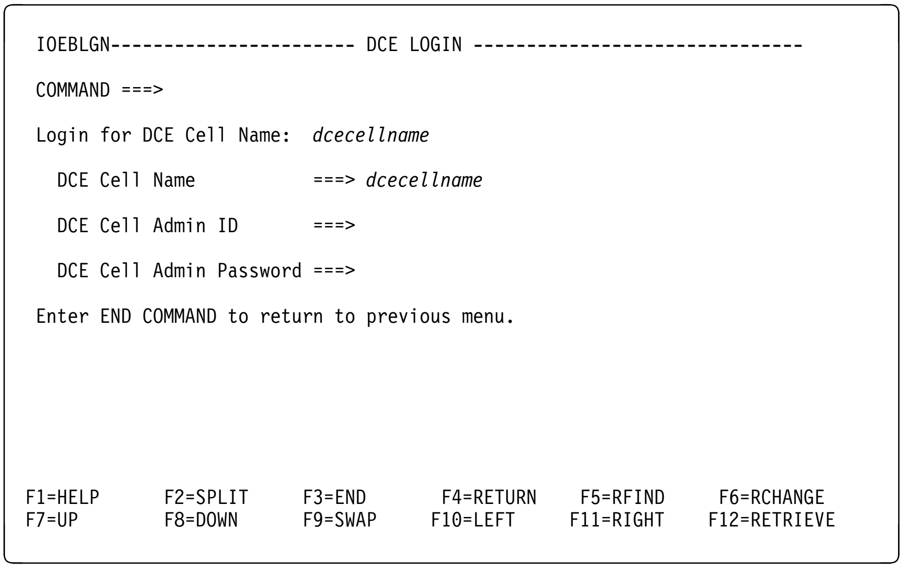
You must set up a TSO user ID that has authority to run the Distributed File Service configuration program. Following are the specifics of this user ID, the **DFSCONF** administrator:

- The **DFSCONF** administrator must have a superuser ID. In z/OS, the superuser or root has a **UID = 0**. A user can be accorded superuser privileges by specifying zero in the UID parameter of the ADDUSER or ALTUSER commands.
- Add the SIOEPNLE data set to the ISPPLIB concatenation.
- Add the SIOEMSGE data set to the ISPMLIB concatenation.
- Add the SIOEXEC library to the SYSEXEC or SYSPROC concatenation.

DCE Login Panel

To configure Distributed File Service on the z/OS host, you must be appropriately authenticated and authorized by the DCE Security Service. If you attempt to perform any of the functions provided by **DFSCONF** that require DCE authentication, **DFSCONF** automatically displays the DCE Login panel. You must then enter the correct Cell Admin ID and password. This ensures that you are logged in to DCE as the administrator who has all the necessary permissions to configure z/OS DFS.

The DCE Cell Admin ID and DCE Cell Admin Password are usually the same as those used during the initial configuration of the cell. However, these may have been changed. Figure 1 on page 42 shows the DCE Login panel.



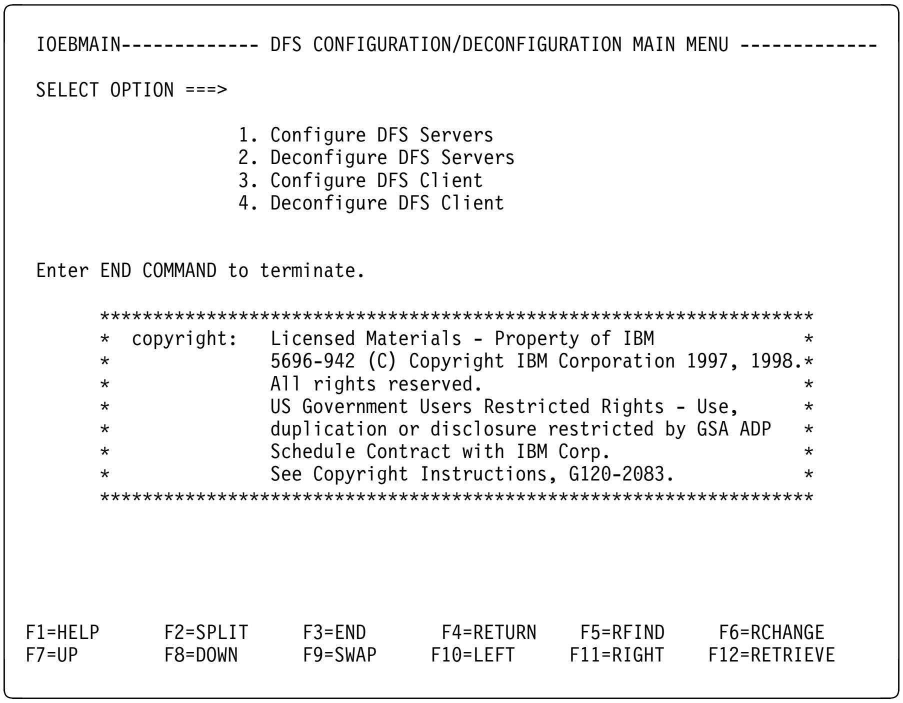
```
IOEBLGN----------------------- DCE LOGIN -------------------------------

COMMAND ===>

Login for DCE Cell Name:  dcecellname

  DCE Cell Name           ===> dcecellname

  DCE Cell Admin ID       ===>

  DCE Cell Admin Password ===>

Enter END COMMAND to return to previous menu.





F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 1. DCE Login Panel

If you have previously set the **_IOE_CFG_CELL_ID** environment variable (note, this needs to be specified in the home directory), its value is displayed in the **DCE Cell Admin ID** field of the panel. You can either accept it or overwrite it with a new value.

DFS Configuration Main Menu

After you run **DFSCONF** from TSO, the DFSCONF Main Menu displays. The DFSCONF Main Menu is shown in Figure 2.



```
IOEBMAIN------------- DFS CONFIGURATION/DECONFIGURATION MAIN MENU -------------

SELECT OPTION ===>

                   1. Configure DFS Servers
                   2. Deconfigure DFS Servers
                   3. Configure DFS Client
                   4. Deconfigure DFS Client


Enter END COMMAND to terminate.

      *******************************************************************
      *  copyright:   Licensed Materials - Property of IBM              *
      *               5696-942 (C) Copyright IBM Corporation 1997, 1998.*
      *               All rights reserved.                              *
      *               US Government Users Restricted Rights - Use,      *
      *               duplication or disclosure restricted by GSA ADP   *
      *               Schedule Contract with IBM Corp.                  *
      *               See Copyright Instructions, G120-2083.            *
      *******************************************************************




F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 2. DFSCONF Main Menu

The menu items are:

Menu Item **Description**

Configure DFS Servers
If you select option **1** you get a list of all the DFS Servers that may be configured. See “Using DFSCONF to Configure DFS Servers.”

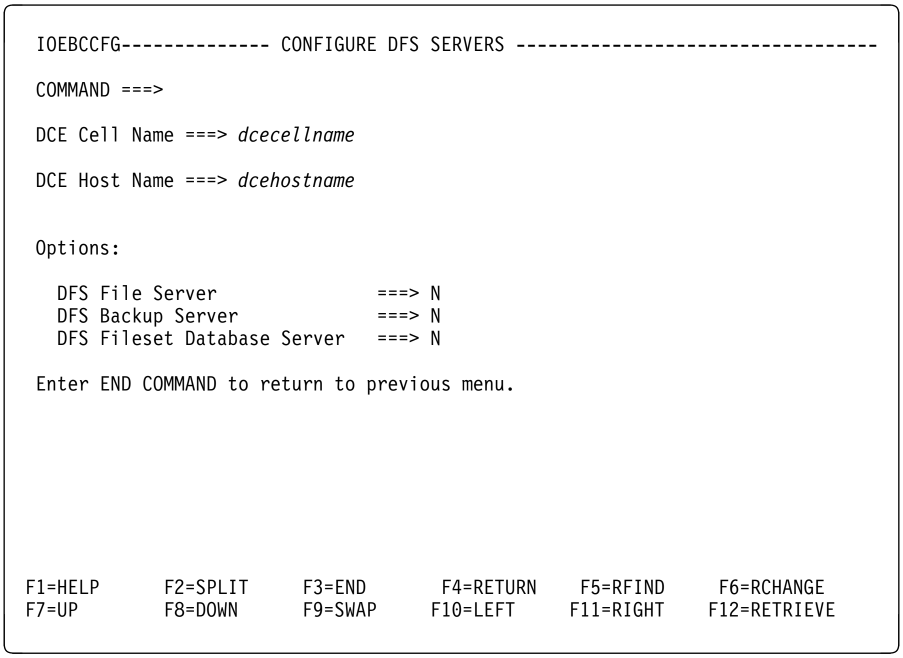
Deconfigure DFS Servers
If you select option **2** you get a list of all the DFS Servers that may be deconfigured. See “Using DFSCONF to Deconfigure DFS Servers” on page 100.

Configure DFS Client
If you select option **3** you perform the DFS Client configuration. See “Using DFSCONF to Configure the DFS Client” on page 45.

Deconfigure DFS Client
If you select option **4** you perform the DFS Client deconfiguration. See “Using DFSCONF to Deconfigure the DFS Client” on page 102.

Using DFSCONF to Configure DFS Servers

Selecting Configure DFS Servers from the DFSCONF Main Menu displays the panel shown in Figure 3.



```
 IOEBCCFG-------------- CONFIGURE DFS SERVERS ----------------------------------

 COMMAND ===>

 DCE Cell Name ===> dcecellname

 DCE Host Name ===> dcehostname


 Options:

   DFS File Server              ===> N
   DFS Backup Server            ===> N
   DFS Fileset Database Server  ===> N

 Enter END COMMAND to return to previous menu.







F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 3. Configuring DFS Servers Panel

The menu items are:

Menu Item **Description**

DCE Cell Name
This is the DCE cell name used to previously configure DCE on this system and should not be changed on this panel. It is displayed for information purposes only. The DCE cell name is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**.

DCE Host Name
This is the DCE host name of the system on which DFS is being configured and should not be changed on this panel. It is displayed for information purposes only. The DCE host name is obtained from the

DCE configuration file **/opt/dcelocal/dce_cf.db**. The DCE host name is case sensitive.

DFS File Server
Enter **Y** to configure the DFS server on this host system. See “Using DFSCONF to Configure the DFS File Server.”

DFS Backup Server
Enter **Y** to configure the DFS Backup Server and the DFS Tape Coordinator on this host system. See “Using DFSCONF to Configure the DFS Backup Server.”

DFS Fileset Database Server
Enter **Y** to configure the DFS Fileset Database server on this host system. See “Using DFSCONF to Configure the DFS Fileset Location Database Server” on page 45.

Note: When you specify the **DCE Cell Name**, it is case sensitive and it is not fully qualified (for example, you can enter **dcecell1**, but not **/.../dcecell1**).

Using DFSCONF to Configure the DFS File Server

The DFS File Server Configuration Program automates the configuration of the DFS File Server. You **must** first configure the DFS File Server before configuring the DFS Backup Server or DFS Fileset Database Server. After configuring the DFS File Server, you can configure or reconfigure the remaining servers in any combination. Therefore, you can specify **Y** on each of the options to have all the servers configured at once.

To initiate the configuration of this server, change the DFS File Server option from **N** to **Y** and press <**Enter**>.

The program begins to issue the DCE administration commands that configures your DFS Server. If not interrupted, and if successfully completed, the program displays the DFS File Server Configuration Ended....SUCCESSFULLY message.

If interrupted, or if the program did not successfully complete configuration, the DFS File Server Configuration Ended....UNSUCCESSFULLY message is displayed. To determine where the error occurred that caused the configuration to fail, examine the **dfsconf.log** file, which by default is in your home directory, or at the location specified by the environment variable **_IOE_CFG_LOG_FILE**. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example of a **dfsconf.log** file. If the **dfsconf.log** file already exists any execution of the DFSCONF program appends any messages to the end of the log file.

After configuring the DFS File Server using **DFSCONF**, additional DFS File Server configuration issues described in “Completing the DFS File Server Configuration” on page 47 must be addressed. But, first, proceed to complete the next step in the “DFS Configuration Steps” on page 34.

Using DFSCONF to Configure the DFS Backup Server

If you want to run the DFS Backup Database Server (**bakserver**) or any DFS Backup Tape Coordinators (**butc01** through **butc08**) on a DCE host system, you should configure the DFS Backup Server using **DFSCONF**.

The Distributed File Service Server Configuration Program (**DFSCONF**) can be used to configure the Backup server (**bakserver** and **butc**) processes. The procedure is identical with that used in configuring other Distributed File Service server processes. It is recommended that the DFS File Server be the first DFS server component configured on your system as described in “Using DFSCONF to Configure the DFS File Server.”

To initiate the configuration of the Backup Server, change the DFS Backup Server option from **N** to **Y** and press <**Enter**>.

The program begins to issue the DCE administration commands that configures your DFS Backup Server. If not interrupted, and if successfully completed, the program displays the DFS Server Configuration Ended....SUCCESSFULLY message.

If interrupted, or if the program did not successfully complete configuration, the DFS Backup Server Configuration Ended....UNSUCCESSFULLY message is displayed. To determine where the error occurred that caused the configuration to fail, examine the **dfsconf.log** file, which by default is in your home directory, or at the location specified by the environment variable **_IOE_CFG_LOG_FILE**. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example output of a **dfsconf.log** file. If the **dfsconf.log** file already exists any execution of the **DFSCONF** program appends any messages to the end of the log file.

After configuring the Backup Server using **DFSCONF**, additional Backup Server configuration issues described in “Completing the Backup Database Server (bakserver) Configuration” on page 53 must be addressed. But, first, proceed to complete the next step in the “DFS Configuration Steps” on page 34.

Using DFSCONF to Configure the DFS Fileset Location Database Server

The Distributed File Service Server Configuration Program (**DFSCONF**) can be used to configure the Fileset Location Database server (**flserver**) process. The procedure is identical with that used in configuring other Distributed File Service server processes. It is recommended that the DFS File Server be the first DFS server component configured on your system as described in “Using DFSCONF to Configure the DFS File Server” on page 44.

To initiate the configuration for this server, change the DFS Fileset Database Server option from **N** to **Y** and press <**Enter**>.

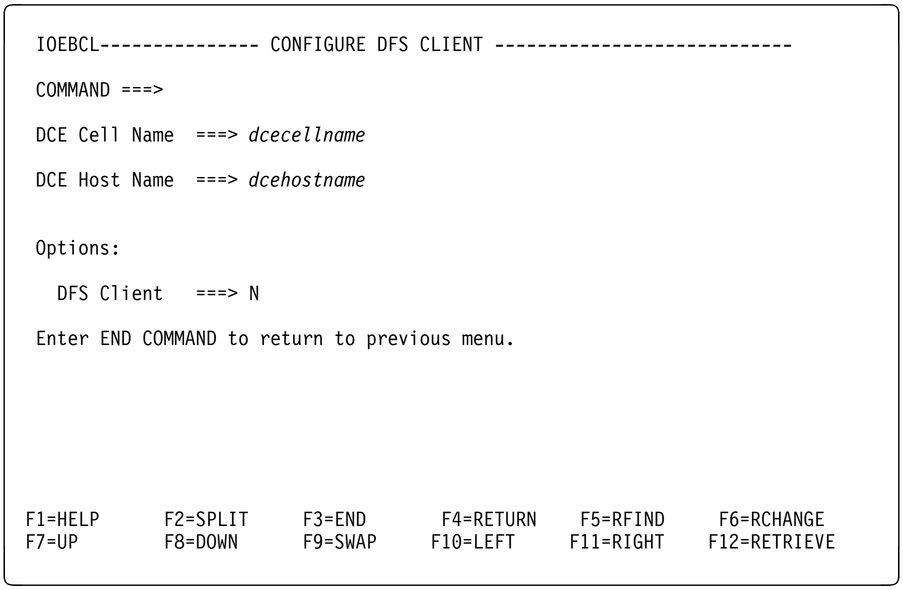
The program begins to issue the DCE administration commands that configures your DFS Fileset Database server. If not interrupted, and if successfully completed, the program displays the DFS Server Configuration Ended....SUCCESSFULLY message.

If interrupted, or if the program did not successfully complete configuration, the DFS Fileset Database Server Configuration Ended....UNSUCCESSFULLY message is displayed. To determine where the error occurred that caused the configuration to fail, examine the **dfsconf.log** file, which by default is in your home directory, or at the location specified by the environment variable **_IOE_CFG_LOG_FILE**. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example output of a **dfsconf.log** file. If the **dfsconf.log** file already exists any execution of the **DFSCONF** program appends any messages to the end of the log file.

After configuring the DFS Fileset Database server using **DFSCONF**, additional DFS Fileset Database server configuration issues described in “Completing the FLDB Server (flserver) Configuration” on page 47 must be addressed. But, first, proceed to complete the next step in the “DFS Configuration Steps” on page 34.

Using DFSCONF to Configure the DFS Client

Selecting Configure DFS Client from the DFSCONF Main Menu displays the panel shown in Figure 4 on page 46.



```
IOEBCL--------------- CONFIGURE DFS CLIENT ----------------------------

COMMAND ===>

DCE Cell Name  ===> dcecellname

DCE Host Name  ===> dcehostname


Options:

  DFS Client   ===> N

Enter END COMMAND to return to previous menu.





F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 4. Configuring the DFS Client Panel

The menu items are:

Menu Item **Description**

DCE Cell Name This is the DCE cell name used to previously configure DCE on this system and should not be changed on this panel. It is displayed for information purposes only. The DCE cell name is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**.

DCE Host Name This is the DCE host name of the system on which DFS is being configured and cannot be changed on this panel. It is displayed for information purposes only. The DCE host name is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**. The DCE host name is case sensitive.

DFS Client To initiate the configuration of the DFS Client, change the DFS Client option from **N** to **Y** and press <**Enter**>.

The Distributed File Service Server Configuration Program (**DFSCONF**) can be used to configure the DFS Client (**DFSCM**).

The program begins to issue the DCE administration commands that configures your **DFSCM**. If not interrupted, and if successfully completed, the program displays the DFS Client Configuration Ended....SUCCESSFULLY message.

If interrupted, or if the program did not successfully complete configuration, the DFS Client Configuration Ended....UNSUCCESSFULLY message is displayed. To determine where the error occurred that caused the configuration to fail, examine the **dfsconf.log** file, which by default is in your home directory, or at the location specified by the environment variable **_IOE_CFG_LOG_FILE**. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example output of a **dfsconf.log** file. If the **dfsconf.log** file already exists any execution of the **DFSCONF** program appends any messages to the end of the log file.

After configuring the DFS Client using **DFSCONF**, additional **DFSCM** configuration issues described in “Completing the DFSCM Installation and Configuration” on page 58 must be addressed. But, first, proceed to complete the next step in the “DFS Configuration Steps” on page 34.

Completing the DFS File Server Configuration

To complete the configuration of the DFS File Server, you must identify the new DCE host system as a DFS file server system to other systems in the DCE cell. You must also complete the configuration of the server components that will comprise the DFS File Server on the DCE host system. This includes updating the associated DFS administrative files and starting the servers.

The DFS configuration steps will instruct you when to perform the additional configurations steps. For your reference, here are the sections where these additional server configurations steps are described:

1. “Completing the FLDB Server (flserver) Configuration”
2. “Creating the DFS File Server FLDB Entry” on page 49
3. “Completing Fileset Server (ftserver) Configuration” on page 49
4. “Completing Replication (repserver) Server Configuration” on page 51
5. “Completing Update Client (upclient) Configuration” on page 51
6. “Completing Update Server (upserver) Configuration” on page 52
7. “Completing the Backup Database Server (bakserver) Configuration” on page 53
8. “Customizing the ioepdcf File” on page 64
9. “Customization of the DFS Server and the DFSCM envar Files” on page 65.

Completing the boserver Configuration

When configuring and running DFS, it is recommended that the **boserver** be automatically started. To automatically start, the **boserver**, update the **boserver** setting in the **ioepdcf** file. Refer to “Customizing the ioepdcf File” on page 64 for more information.

Completing the FLDB Server (flserver) Configuration

After using the Distributed File Service Server Configuration Program, the following additional steps required to configure the Fileset Database Location Database (FLDB) Server (**flserver**) on a DCE host system.

Note: If you are configuring this DCE host system into an existing DCE cell, DFS administrative lists are typically updated on the cell's System Control machine and propagated by the Update Server to other DCE hosts running Update Clients. Refer to Chapter 5, “Defining DFS Administrators and Users” on page 69 for more information on DFS administrative lists and System Control Machines.

The following instructions assume that the system with DCE host name *dcehostname* is the system where the **flserver** and **upserver** will be run.

1. Use the **bos addadmin** command to add the abbreviated DFS server principal (**hosts/***dcehostname***/dfs-server**) of the new DFS server machine on the system with DCE host name *dcehostname* to the **/opt/dcelocal/var/dfs/admin.fl** administrative list file on DCE host system where the **flserver** is run.

   You can use the **dcecp group add** command to add the abbreviated DFS server principal to a security group and include the group in the **admin.fl** administrative list file. Note that DFSCONF processing created the security group **subsys/dce/dfs-fs-servers** and added the member **hosts/***dcehostname***/dfs-server** for this DCE host system to the group. It is recommended that you add this group to the **admin.fl** file using the command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                -adminlist admin.fl -group subsys/dce/dfs-fs-servers -createlist
   ```

Alternatively, you can add the abbreviated DFS server principal for the DCE host system to the **admin.fl** file using the command:

```
$ bos addadmin -server /.:/hosts/dcehostname \
               -adminlist admin.fl \
               -principal hosts/dcehostname/dfs-server -createlist
```

2. Ensure that the **admin.fl** administrative list is also updated with administrator principals and groups in the **/opt/dcelocal/var/dfs** directory on DCE host system where the **flserver** will be being run.

   To update the **admin.fl** administrative list on the system with DCE host name *dcehostname*, issue the following:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                  -adminlist admin.fl \
                  -principal cell_admin -group dfsteam -createlist
   ```

   To verify the **admin.fl** administrative list has been updated on the new database server machine with DCE host name **DCEDFS**, issue the following:

   ```
   $ bos lsadmin /.:/hosts/DCEDFS -adminlist admin.fl

   Admin Users are: user: cell_admin,
   user: hosts/DCEDFS/dfs-server,
   group: dfsteam,
   group: subsys/dce/dfs-fs-servers
   ```

3. Stop and restart the **flserver** process on each system in the DCE cell. Restarting the existing database server processes causes the processes to read the updated RPC server group. This ensures that each Ubik coordinator agrees on the number and identities of the other database server machines of its type, which is vital to Ubik's use of a quorum of database server machines to maintain database consistency. For additional information on Ubik, see the *z/OS Distributed File Service DFS Administration* book.

4. Start the **flserver** database server process on this DCE host machine using the **bos create** command. The following is an example of how to start the **flserver** on a system with the DCE host name of **DCEDFS**:

   ```
   $ bos create /.:/hosts/DCEDFS flserver simple "flserver \
         envar('_EUV_HOME=/opt/dfslocal/home/flserver')/ >dd:flserver 2>&1"
   ```

   The **bos create** command will also instruct the **boserver** to automatically start the **flserver** when the **boserver** is subsequently restarted. For more information on using the **bos create** command, refer to “Starting the boserver Processes from OMVS” on page 112.

5. Verify that there is not already a DFS server entry for the system with DCE host name *dcehostname* by issuing the following command from OMVS which displays all server entries:

   ```
   $ fts lsserverentry -all
   ```

   If the output identifies that a list of sites can not be acquired (usually, since this is the first system in the DCE cell where DFS is configured) or the the **principal=hosts/***dcehostname* identifying this DCE host system is not displayed, then a server entry must be created in the Fileset Location Databases (FLDB).

   To create a DFS server entry in the FLDB with the DCE cell name *dcecellname* and the DCE host name *dcehostname* issue the command:

   ```
   $ fts crserverentry -server /.../dcecellname/hosts/dcehostname \
                       -principal hosts/dcehostname
   ```

6. Synchronize the Fileset Location Databases in the DCE cell.

If the FLDB entry was created, it is recommended that you synchronize the Fileset Location Databases in the DCE cell by issuing the following command from OMVS using the DCE host name (*dcehostname*) for each system in the DCE cell where an **flserver** is configured:

```
$ udebug -rpcgroup /.:/fs -server /.:/hosts/dcehostname
```

For additional detailed information, see *z/OS Distributed File Service DFS Administration* book.

This completes **flserver** configuration. You should proceed to the next configuration step.

Creating the DFS File Server FLDB Entry

It is recommended that you always create a DFS File Server entry in the Fileset Location Database (FLDB). You must create this entry for a DCE host system if you will ever export files using DFS from the DCE host system.

To create the FLDB entry, an **flserver** must be running in the DCE cell. Note that if you previously completed an **flserver** configuration on this DCE host system, you were instructed to create the entry for the DCE host system at that time.

To create the FLDB entry, use the following procedure:

1. Verify that there is not already a DFS server entry with the DCE host name of *dcehostname* by issuing the following command from OMVS to display all server entries:

   ```
   $ fts lsserverentry -all
   ```

   If the output identifies that a list of sites cannot be acquired (usually, since this is the first system in the DCE cell where DFS is configured) or the **principal** = **hosts**/*dcehostname* identifying this DCE host system is not displayed, then a server entry must be created in the FLDB.

   To create a DFS server entry in the FLDB with the DCE cell name of *dcecellname* and the DCE host name of *dcehostname* issue the following command:

   ```
   $ fts crserverentry -server /.../dcecellname/hosts/dcehostname \
                       -principal hosts/dcehostname
   ```

2. If the FLDB entry was created, it is recommended that you synchronize the Fileset Location Databases in the DCE cell by issuing the following command from OMVS using the DCE host name of *dcehostname* for each system in the DCE cell where an **flserver** is configured:

   ```
   $ udebug -rpcgroup /.:/fs -server /.:hosts/dcehostname
   ```

For additional detailed information, refer to the *z/OS Distributed File Service DFS Administration* book.

Completing Fileset Server (ftserver) Configuration

After using the Distributed File Service Server Configuration Program to configure the DFS File Server, the following additional steps required to configure the DFS Fileset Server on a DCE host.

If you are configuring this DCE host system into an existing DCE cell, DFS administrative lists are typically updated on the cell's System Control machine and propagated by the Update Server to other DCE hosts running Update Clients. Refer to Chapter 5, “Defining DFS Administrators and Users” on page 69 for more information on DFS administrative lists and System Control Machines.

Note: The following instructions assume that there is only one system in the DCE cell where DFS is running or that the **upserver** will be run on the same system with the **ftserver**.

1. Use the **bos addadmin** command to add the abbreviated DFS server principal (**hosts/***dcehostname***/dfs-server**) of the new DFS server machine on the system with DCE host name *dcehostname* to the **/opt/dcelocal/var/dfs/admin.ft** administrative list file.

   You can use the **dcecp -c group add** command to add the abbreviated DFS server principal to a security group and include the group in the **admin.ft** administrative list file. Note that DFSCONF processing created the security group **subsys/dce/dfs-fs-servers** and added the member **hosts/***dcehostname***/dfs-server** for this DCE host system to the group. You can add this group to the **admin.ft** file using the command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname -adminlist admin.ft \
                 -group subsys/dce/dfs-fs-servers -createlist
   ```

   Alternatively, you can add the abbreviated DFS server principal for the DCE host system to the **admin.ft** file using the command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname -adminlist admin.ft \
                 -principal hosts/dcehostname/dfs-server -createlist
   ```

2. Ensure that the **/opt/dcelocal/var/dfs/admin.ft** administrative list file is also updated with administrator principals and groups on the DCE host system where the **ftserver** will be run. To update the **admin.ft** administrative list file with the user **cell_admin** and the group **dfsteam** on the system with DCE host name *dcehostname*, issue the following:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname -adminlist admin.ft \
                 -principal cell_admin -group dfsteam -createlist
   ```

   To verify the **admin.ft** administrative list has been updated on the system with DCE host name *dcehostname*, issue the following:

   ```
   $ bos lsadmin /.:/hosts/dcehostname -adminlist admin.ft

   Admin Users are: user: cell_admin,
   user: hosts/dcehostname/dfs-server,
   group: dfsteam,
   group: subsys/dce/dfs-fs-servers
   ```

3. Start the **ftserver** on this DCE host machine using the **bos create** command. The following is an example of how to start the **ftserver** on a system with the DCE host name of **DCEDFS**:

   ```
   $ bos create /.:/hosts/DCEDFS ftserver simple "ftserver \
        envar('_EUV_HOME=/opt/dfslocal/home/ftserver')/ >dd:ftserver 2>&1"
   ```

   For more information on using the **bos create** command, refer to “Starting the boserver Processes from OMVS” on page 112.

4. Verify that there is not already a DFS server entry for the system with DCE host name *dcehostname* by issuing the following command from OMVS which displays all server entries:

   ```
   $ fts lsserverentry -all
   ```

   If the output identifies that a list of sites can not be acquired (usually, since this is the first system in the DCE cell where DFS is configured) or the **principal=hosts/***dcehostname* identifying this DCE host system is not displayed, then a server entry must be created in the Fileset Location Databases (FLDB).

   To create a DFS server entry in the FLDB with the DCE cell name *dcecellname* and the DCE host name *dcehostname* issue the command:

   ```
   $ fts crserverentry -server /.../dcecellname/hosts/dcehostname \
                      -principal hosts/dcehostname
   ```

For additional detailed information, refer to the *z/OS Distributed File Service DFS Administration* book.

This completes **ftserver** configuration. You should proceed to the next configuration step.

Completing Replication (repserver) Server Configuration

After using the Distributed File Service Server Configuration Program, additional steps required to configure and start the Replication Server (**repserver**) on a DCE host system to replicate filesets or house a replica of a fileset. If you plan to define the **root.dfs** fileset for the DCE cell on the host system where DFS is being configured, you must configure the **repserver**.

To complete the initial configuration and start a **repserver** on a DCE host system:

1. Ensure that a FLDB entry for the DCE host system running as a DFS File server machine is created as described in “Creating the DFS File Server FLDB Entry” on page 49.
2. Ensure that the **ftserver** is running on the DCE host system as described in “Completing Fileset Server (ftserver) Configuration” on page 49.
3. Start the **repserver** on this DCE host machine using the **bos create** command. The following is an example of how to start the **repserver** on a system with the DCE host name of **DCEDFS**:

   ```
   $ bos create /.:/hosts/DCEDFS repserver simple "rpserver \
         envar('_EUV_HOME=/opt/dfslocal/home/repserver')/ >dd:rpserver 2>&1"
   ```

   Note: Enter the previous command specifying **repserver** and **rpserver** exactly as shown.

   The **bos create** command will also instruct the **boserver** to automatically start the **repserver** when the **boserver** is subsequently restarted. For more information on using the **bos create** command, refer to “Starting the boserver Processes from OMVS” on page 112.

You must refer to the *z/OS Distributed File Service DFS Administration* book for additional information on how to set up filesets for replication in a DCE cell.

This completes **repserver** configuration. You should proceed to the next configuration step.

Completing Update Client (upclient) Configuration

After using the Distributed File Service Server Configuration Program, additional steps are required to configure and start the Update Client (**upclient**) on a DCE host system. This section explains how to complete the **upclient** configuration.

An **upclient** is run to allow the **upclient** to periodically request an **upserver** to identify the latest level of specific administrative files that exist on the **upserver** DCE host machine. If the latest level of a file does not exist on the **upclient** DCE host system, the **upclient** requests the latest level of the file from the **upserver** and over writes the file that is on the **upclient** DCE host system.

Note: You can run either an **upserver** or **upclient** on a DCE host system. You cannot run an **upserver** and an **upclient** on the same system. Also, the **upserver** or **upclient** must run in the same DCE cell.

Completing the configuration of the **upclient** consists of:

1. Updating the **admin.up** administrative list file on the **upserver** system.

   To identify the DCE host system where the **upclient** will be run to the **upserver**, you must update the **/opt/dcelocal/var/dfs/admin.up** administrative list file on the DCE host system where the **upserver** is running.

For example, if the **upserver** is running on the system with the DCE host name *dcehostnameups* and the **upclient** will be run on the system with the DCE host name *dcehostname*, you can add the abbreviated DFS server principal for the **upclient** system (**hosts/***dcehostname***/dfs-server**) to the **admin.up** file on the **upserver** system using the following command:

```
$ bos addadmin -server /.:/hosts/dcehostnameups \
              -adminlist admin.up -principal hosts/dcehostname/dfs-server
```

2. Ensuring all administrative list files are updated on the **upserver** system before the **upclient** is started.

   Once the **upclient** is started, DFS administrative list files on the system where the **upclient** is running can be overlayed by the interaction of the **upclient** with the **upserver**. You must ensure that all administrative list file updates made during the configuration of the DFS on this DCE host system are reflected in the administrative list files on the system where the **upserver** is running if the **upserver** is exporting the administrative list file. See Chapter 5, “Defining DFS Administrators and Users” on page 69 for more information.

3. Verifying **admin.up** file updates.

   To verify that the **admin.up** administrative list file has been updated on a system with DCE host name *dcehostnameups*, issue the following:

   ```
   $ bos lsadmin /.:/hosts/dcehostnameups -adminlist admin.up

   Admin Users are: user: hosts/DCEDFS/dfs-server,
   group: dfsteam,
   group: subsys/dce/dfs-fs-servers
   ```

4. Starting the **upclient**

   The following command is an example of how to start the **upclient** on the system with DCE host name *dcehostname* when the **upserver** is running on the system with DCE host name *dcehostnameups*:

   ```
   $ bos create /.:/hosts/dcehostname upclient simple "upclient \
         envar('_EUV_HOME=/opt/dfslocal/home/upclient')/ >dd:upclient 2>&1 \
         /.:/hosts/dcehostnameups \
         (/opt/dcelocal/var/dfs/ admin.bos admin.ft admin.up) -f UpCLog"
   ```

   The previous **bos create** command example requests that the **upserver** allow the **upclient** to acquire **admin.bos**, **admin.ft** and **admin.up** administrative list file information if the **upserver** has been started to export these files. The **bos create** command also causes the **boserver** to automatically start the **upclient** when the **boserver** is subsequently restarted.

   Refer to “Starting the boserver Processes from OMVS” on page 112, for more information.

This completes **upclient** configuration. You should proceed to the next configuration step.

Completing Update Server (upserver) Configuration

Note: You can run either an **upserver** or an **upclient** on the host system where DFS is being configured. You cannot run an **upserver** and an **upclient** on the same host system.

After using the Distributed File Service Server Configuration Program, additional steps are required to configure and start the Update Server (**upserver**) on a DCE host system. An **upserver** is run to allow an **upclient** to periodically request an **upserver** to identify the latest level of specified administrative files that exist on the **upserver** DCE host machine. If the latest level of a file does not exist on the **upclient** DCE host system, the **upclient** requests the latest level of the file from the **upserver** and over writes the file that is on the **upclient** DCE host system.

Completing the configuration of the Update Server (**upserver**) consists of:

1. Updating the admin.up File on the **upserver** system.

   Before starting the **upserver**, insure that the **admin.up** file is created (see “Defining a DFS Administrator” on page 70 and “Adding Principals and Groups to Administrative Lists” on page 74 and “System Control Machines and Domains” on page 71).

   To identify the DCE host system where the **upclient** will be run to the **upserver**, you must update the **/opt/dcelocal/var/dfs/admin.up** administrative list file on the DCE host system where the **upserver** is running.

   For example, if the **upserver** is running on the system with the DCE host name *dcehostnameups* and an **upclient** will be run on the system with the DCE host name **DCEDFS**, you can add the abbreviated DFS server principal for the **upclient** system (**hosts/DCEDFS/dfs-server**) to the **admin.up** file on the **upserver** system using the following command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostnameups \
                -adminlist admin.up -principal hosts/DCEDFS/dfs-server
   ```

   The **admin.up** file should contain an entry identifying the abbreviated DFS server principal for each file server machine in the domain where an **upclient** is run (see “Completing Update Client (upclient) Configuration” on page 51).

2. Verifying **admin.up** file updates.

   To verify that the **admin.up** administrative list file has been updated on a system with DCE host name *dcehostnameups*, issue the following:

   ```
   $ bos lsadmin /.:/hosts/dcehostnameups -adminlist admin.up

   Admin Users are: user: hosts/DCEDFS/dfs-server,
   group: dfsteam,
   group: subsys/dce/dfs-fs-servers
   ```

3. Starting the **upserver**

   The following is an example of how to create and start the **upserver** on the system with the DCE host name of *dcehostnameups*:

   ```
   $ bos create /.:/hosts/dcehostnameups upserver simple "upserver \
         envar('_EUV_HOME=/opt/dfslocal/home/upserver')/ >dd:upserver 2>&1 \
         (/opt/dcelocal/var/dfs/ admin.bos admin.up admin.ft)"
   ```

   The previous **bos create** command example requests the **upserver** to export the **admin.bos** and **admin.ft** administrative files to any **upclient** with an entry in the **/opt/dcelocal/var/dfs/admin.up** file on the DCE host system where the **upserver** is running. The **bos create** command also causes the **boserver** to automatically start the **upserver** when the **boserver** is subsequently restarted.

   Refer to “Starting the boserver Processes from OMVS” on page 112 for more information.

This completes **upserver** configuration. You should proceed to the next configuration step.

Completing the Backup Database Server (bakserver) Configuration

This section describes how to complete the configuration of the DFS Backup Server (**bakserver**) and the Backup Tape Coordinator (**butc**) servers. The **bakserver** is not required on a DCE host system when DFS is running. However, if you plan to run a Backup Tape Coordinator (**butc**) server on the system you should configure **bakserver**.

If you are configuring this DCE host system into an existing DCE cell, DFS administrative lists are typically updated on the cell's System Control machine and propagated by the Update Server to other DCE hosts running Update Clients. Refer to Chapter 5, “Defining DFS Administrators and Users” on page 69, for more information on DFS administrative lists and System Control Machines.

Note: The following instructions assume that there is only one system in the DCE cell where DFS is running or that the **upserver** will be run on the same system with the **bakserver**. Also, DFS configuration command examples sometimes use **/.:** to denote **/.../***dcecellname*.

1. Use the **bos addadmin** command to add the abbreviated DFS server principal (**hosts/***dcehostname***/dfs-server**) of the new DFS server machine on the system with the DCE host name of *dcehostname* to the **/opt/dcelocal/var/dfs/admin.bak** administrative list file.

   You can use the **dcecp -c group add** command to add the abbreviated DFS server principal to a security group and include the group in the **admin.bak** administrative list file. Note that DFSCONF processing created the security group **subsys/dce/dfs-fs-servers** and added the member **hosts/***dcehostname***/dfs-server** for this DCE host system to the group. You can add this group to the **admin.bak** file using the following command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                  -adminlist admin.bak -group subsys/dce/dfs-fs-servers -createlist
   ```

   Alternatively, you can add the abbreviated DFS server principal for the DCE host system to the **admin.bak** file using the following command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                  -adminlist admin.bak \
                  -principal hosts/dcehostname/dfs-server -createlist
   ```

2. Ensure that the **/opt/dcelocal/var/dfs/admin.bak** administrative list is also updated with administrator principals and groups on the DCE host system where the **bakserver** will be running. To update the **admin.bak** administrative list on the system with the DCE host name of **DCEDFS**, issue the following command:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                  -adminlist admin.bak \
                  -principal cell_admin -group dfsteam -createlist
   ```

   To verify that the **admin.bak** administrative list has been updated on the system with the DCE host name of **DCEDFS**, issue the following command:

   ```
   $ bos lsadmin /.:/hosts/DCEDFS -adminlist admin.bak

   Admin Users are: user: cell_admin,
   user: hosts/DCEDFS/dfs-server,
   group: dfsteam,
   group: subsys/dce/dfs-fs-servers
   ```

3. Stop and restart the **bakserver** process on each system in the DCE cell where it is running. Restarting the **bakserver** processes causes the updated RPC server group to be read. This ensures that each Ubik coordinator agrees on the number and identities of the other database server machines of its type, which is vital to Ubik's use of a quorum of database server machines to maintain database consistency. You must refer to the *z/OS Distributed File Service DFS Administration* book for additional Ubik information.

4. Start the **bakserver** database server process on this DCE host machine using the **bos create** command. The following is an example of how to start the **bakserver** on a system with the DCE host name of **DCEDFS**:

   ```
   $ bos create /.:/hosts/DCEDFS bakserver simple "bkserver \
        envar('_EUV_HOME=/opt/dfslocal/home/bakserver')/ >dd:bkserver 2>&1"
   ```

Notes:

a. Enter the previous command specifying **bakserver** and **bkserver** exactly as shown.

b. The DFS implementation assumes that the UUID for the **bakserver** is exported when DCE configures the Cell Directory Server (CDS). If the **bakserver** fails to start because the UUID for the **bakserver** is not exported, refer to “Re-Establish the bakserver Object UUID” on page 57 for a description of the recovery actions that you can take.

c. The **bos create** command will also instruct the **boserver** to automatically start the **bakserver** when the **boserver** is subsequently re-started.

For more information on using the **bos create** command, refer to “Starting the boserver Processes from OMVS” on page 112.

5. Synchronize the Backup Databases in the DCE cell.

   To insure that all the DFS backup databases in the DCE cell are synchronized, issue the following command from OMVS using the DCE host name (*dcehostname*) for each system in the DCE cell where a **bakserver** is running. :

   ```
   $ udebug -rpcgroup /.:/subsys/dce/dfs/bak -server /.:/hosts/dcehostname
   ```

For additional detailed information, refer to the *z/OS Distributed File Service DFS Administration* book.

This completes **bakserver** configuration. You should proceed to “Completing the DFS Backup Tape Coordinator Configuration” or to the next configuration step.

Completing the DFS Backup Tape Coordinator Configuration

This section provides information on completing the DFS Backup Tape Coordinator (**butc***nn*) configuration. You must complete the DFS Backup Database Server (**bakserver**) configuration before you complete the Backup Tape Coordinator (**butc**) configuration. Additionally, setting the Backup Tape Coordinator environment variables and creating a VSAM tape backup management file are discussed in this section and should be performed in the order they appear.

If you plan to use the Distributed File Service Backup Tape Coordinator with the Distributed File Service Backup Database Server, complete the following steps:

1. Verify that the Backup Database Server (**bakserver**) is running. Issue the **bos status** command to the DCE host system where the **bakserver** is running, by entering the following:

   ```
   $ bos status /.:/hosts/dcehostname bakserver

   Instance bakserver, currently running normally.
   ```

   If a **bakserver** is not running in the DCE cell, a **bakserver** should be started before proceeding to complete **butc** configuration.

2. The DFS implementation assumes that the object UUID for the **bakserver** is exported during DCE configuration when the Cell Directory Server (CDS) is configured.

   Display the exported object UUID for the **bakserver** by issuing the following command from OMVS:

   ```
   dcecp -c show rpcentry /.:/subsys/dce/dfs/bak

   {003c4de0-9485-6182-07ef-b1e000000000}
   ```

   If the command displays an object UUID (for example, 003c4de0-9485-6182-07ef-b1e000000000), then the object UUID is exported and you can proceed to the next configuration step.

   If an object UUID is not shown and a message indicating that the name service object is not found is displayed, the UUID for the **bakserver** is not exported at this time. If the UUID is not exported, refer to “Re-Establish the bakserver Object UUID” on page 57, for a description of the recovery actions that you can take to allow **butc** configuration to continue.

3. Verify that a **bak addhost** command has been issued for the ID number associated with the **butc***nn* server you want to start. Issue the following command to determine what **butc***nn* servers are defined:

   ```
   $ bak lshosts

   Tape hosts:
       Host /.../dcecellname/hosts/dcehostname, port offset 0
   ```

   The **bak lshosts** command output shown above indicates that the **butc01** server is already defined at offset 0. No tape hosts are displayed if there are no **butc** servers defined on this DCE host system.

   If you want to start a **butc***nn* and it was not displayed by the **bak lshost** command, issue the following **bak addhost** command.

   Note: The numeric offset value shown at the end of the **bak addhost** command is:

   0 to indicate **butc01**
   1 to indicate **butc02**
   2 to indicate **butc03**
   3 to indicate **butc04**
   4 to indicate **butc05**
   5 to indicate **butc06**
   6 to indicate **butc07**
   7 to indicate **butc08**

   For example, if you want to start **butc01** on an DCE host with the DCE host name of *dcehostname*, issue the following command:

   ```
   $ bak addhost /.:/hosts/dcehostname 0

   Adding host '/.../dcecellname/hosts/dcehostname' offset 0
   to tape list ...
   The host name and port information were added to the list of tape
   hosts successfully
   ```

4. Issue the z/OS operator command **modify start** command from the operator console:

   ```
   modify dfs,start butc01

   IOEN00106I DFS daemon BUTC01 is ready for requests.
   ```

   Note: Distributed File Service uses z/OS services to control and dynamically allocate tape drives. Because of this, the **TapeConfig** feature of DFS is not necessary and not available in Distributed File Service.

5. Set the Backup Tape Coordinator environment variables.

   The **/opt/dfslocal/home/butc***nn***/envar** files can be edited to set the environment variables for the **butc***nn* processes, where valid entries for *nn* are 01 through 08. There are four environment variables that may be updated. All have pre-defined default values assigned during installation. These values may be changed if desired.

   The environment variables and default values are:

   _IOE_BUTC_DISKUNIT
   The unit name to be used when dynamically allocating a data definition for disk data sets. This is the name passed to dynamic allocation when allocating the disk device. The default value is **SYSDA**.

_IOE_BUTC_DUMP_HLQ
The high-level qualifier for the dump (backup) data set names. The default value is **DFSBKUP**.

_IOE_BUTC_KSDS
The name of the virtual storage access method (VSAM) DFS tape backup management file. The default value is **DFSBUTC.KSDS.LABEL**. Refer to step 6 for more information.

_IOE_BUTC_TAPEUNIT
The unit name to be used when dynamically allocating a data definition for tape drives. This is the unit name passed to dynamic allocation when allocating the tape device. The default value is **3490**.

At this point, an alias must be created in the z/OS master catalog for the backup data set high-level qualifier. This is specified in the **_IOE_BUTC_DUMP_HLQ** environment variable for the appropriate **butc** process (01 through 08). The default value is **DFSBKUP**. If you choose another name as an alias, the name may be up to 17 characters in length.

For more information on how to set environment variables, refer to the *z/OS Distributed File Service DFS Administration* book. All variables relevant to DFS administration are listed in Appendix C, "DFS/SMB Environment Variables" on page 129.

6. Create a VSAM tape backup management file.

   A VSAM file should be created to maintain a record of each dump processed on the z/OS system. The name of this file must be specified in the environment variable, **_IOE_BUTC_KSDS**. The file should contain the tape name, the dump number, and the tape label. The tape label will contain the tape's expiration date. A new record is added each time a new label is created and written to tape. When the expiration date is reached, the tape becomes expired and is eligible for re-use. The record for the expired tape is deleted from the VSAM data set, the dump information is deleted from the backup database, and the data sets contained in the dump are uncataloged.

   To create the VSAM dataset to contain the tape management backup file identified by the environment variable **_IOE_BUTC_KSDS**, you can copy, edit and submit the IBM supplied sample job statements found in SIOESAMP(BUTCVSAM). To print this VSAM dataset, you can copy, edit and submit the sample job statements found in SIOESAMP(BUTCVSMP).

   Note: If a VSAM data set is not created, an error message will be generated by the **butc** process. The dump process will, however, continue to process correctly and no expired backup records will be deleted from the system.

Re-Establish the bakserver Object UUID

The DFS implementation assumes that the universal unique identifier (UUID) for the **bakserver** is exported when DCE configures the Cell Directory Server (CDS).

You can verify that the UUID for the **bakserver** is exported using the DCE RPC control program (**rpccp**):

```
$ rpccp show entry /.:/subsys/dce/dfs/bak
  objects:
  3396dbb0-1bc1-11d0-a4f6-02608ce88731
>>> no matching binding information found
group members:
  /.../dces390.endicott.ibm.com/hosts/allanon.endicott.ibm.com/self
```

If an object UUID (for example, 3396dbb0-1bc1-11d0-a4f6-02608ce88731) is not displayed in the **rpccp show entry** command output and the message **EUVR12352I Objects not found in server entry** is returned, the UUID for the **bakserver** is not exported at this time. To recover from this condition, you can:

1. Stop other bakservers running in the DCE cell on different DCE host systems.

2. Generate a new UUID using the **uuidgen** command:

   ```
   $ uuidgen
   ```

   The *uuid-value* is displayed by the **uuidgen** command.

3. Issue the **rpcexport command** using the *uuid-value* returned by the **uuidgen** command:

   ```
   $ rpccp export -o uuid-value /.:/subsys/dce/dfs/bak
   ```

4. Re-issue the **rpccp show entry /.:/subsys/dce/dfs/bak** command (see above) to verify that the UUID object is exported.

5. Restart the **bakservers** in the DCE cell. For information on how to start the **bakserver** on a DCE host system, refer to "Starting the boserver Processes from OMVS" on page 112.

Completing the DFSCM Installation and Configuration

This section describes additional configuration topics for the DFS client (**DFSCM**) in a z/OS environment. They include:

- "BPXPRMxx MAXFILEPROC Entry Considerations"
- "BPXPRMxx FILESYSTYPE TYPE(DFSC) Entry"
- "DFSCM Cache Configurations" on page 59
- "DFSCM User .profile File Considerations" on page 63
- "DFSCM Environment Variable Considerations" on page 63.

BPXPRMxx MAXFILEPROC Entry Considerations

MAXFILEPROC(256) is the minimum recommended system parmlib specification for running the DFS client.

BPXPRMxx FILESYSTYPE TYPE(DFSC) Entry

The DFS client runs as a UNIX System Services Physical File System. This section describes the system parmlib member FILESYSTYPE(DFSC) entry required to run the DFS Client (**DFSCM**).

Note: The suffixes *xx* and *yy* are installation defined and uniquely identify system parmlib members. SETOMVS RESET=(*yy*) is a console command that can be used to dynamically add the BPXPRM*yy* member containing the FILESYSTYPE statement for the DFS client. You can update your IEASYS*xx* parmlib member to contain OMVS=(*yy*,*xx*,...) parameter for future IPLs. For more information on BPXPRM*xx* updates, refer to the *z/OS Program Directory*, the *z/OS UNIX System Services Planning*, GA22-7800, and the *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN*, SA22-7591.

The FILESYSTYPE statement for the DFS client is:

```
FILESYSTYPE TYPE(DFSC)
           ENTRYPOINT(IOECMINI)
           PARM('ENVAR("_EUV_HOME=/opt/dfslocal/home/dfscm") /
           >DD:IOEDFSD 2>&1')
           ASNAME(DFSCM)
```

where:

TYPE(DFSC)
: This entry must be be specified as shown or the DFS client will fail in an indeterminate manner. **DFSC** is the well-known name that is used to identify the DFS client (**DFSCM**) as a physical file system.

ENTRYPOINT(IOECMINI)

This entry must be specified as shown or z/OS will not successfully initialize the DFS client.

_EUV_HOME

This environment variable must be specified in the **FILESYSTYPE** statement. A directory other than **/opt/dfslocal/home/dfscm** may be specified but the use of this directory for the DFS client is recommended.

Note: A trailing slash '/' after the **_EUV_HOME** specification is required. **_EUV_HOME** is a DCE runtime option (not a **DFSCM** parameter). A slash identifies the end of the runtime options and the start of the parameters. If the slash '/' is omitted, it indicates that there are no runtime options and **_EUV_HOME** is not defined because it is treated as a **DFSCM** parameter.

>DD:IOEDFSD 2>&1

This entry must be specified as shown. It assigns standard out (**STDOUT**) and standard error (**STDERROR**) for the **DFSCM** DCE login processing to the ddname IOEDFSD. The ddname IOEDFSD is required to be included as started procedure JCL for the DFS client.

ASNAME(DFSCM)

Specifies the name of a procedure in SYS1.PROCLIB that is to be used to start the address space that is initialized by the DFS Client that is run as UNIX System Services physical file system (PFS). The name is 1 to 8 characters and the system converts the name to uppercase. The name you specify is also used for the name of the DFS Client address space. It is recommended that the name DFSCM be specified because the procedure named DFSCM (alias for the IOEP0002) is supplied in the hlq.SIOEPROC library and was used to define the DFS Client as a started task to RACF in the *z/OS Program Directory* instructions. If an alternate name is specified, a new procedure library member must be created with that name and the procedure name defined to RACF.

DFSCM Cache Configurations

This section describes the different configurations available for the DFS client (**DFSCM**) in a z/OS system environment which include:

- “Minimum Configuration of DFSCM”
- “DFSCM Memory Caching”
- “DFSCM Disk Caching” on page 61.

Minimum Configuration of DFSCM: The minimum configuration for the **DFSCM** (or DFS) on a z/OS system is to run with memory caching without any DFS servers. **DFSCM** does not require any DFS servers be running on the z/OS system where **DFSCM** is running.

DFSCM Memory Caching:

To configure a z/OS system to run the **DFSCM** with memory caching, use the following procedure:

1. Create and update the local Hierarchical File System **/opt/dcelocal/etc/CacheInfo** file appropriately.

   The **CacheInfo** file defines the DCE Local File System aggregate to use for a disk cache and the size of a disk or memory cache. The Cache Manager checks this file at initialization to determine this information. (The installation instructions provide details for creating the **CacheInfo** file.)

   The **CacheInfo** file contains the following three fields separated by colons.

   field 1 A directory on the local disk where the Cache Manager mounts the DFS global namespace.

   Note: If **/...** is not specified, symbolic links to the global namespace fail.

field 2 The minor device number for a local DCE Local File System aggregate that serves as the DFS cache for a disk cache. The Cache Manager creates its cache files in this aggregate. There is no default for this entry. The entry must be specified.

Note: Although this aggregate is not used with a memory cache, a placeholder entry must appear in this field even if memory caching is used for the DFS client.

field 3 A definition of the cache size in kilobyte blocks.

Following is an example of a **CacheInfo** file. The file lists the DCE namespace mounted at the global namespace designation (**/...**), the minor device number of **999** for the local DCE Local File System aggregate used for the cache files, and a defined cache size of 512 kilobyte blocks (the aggregate must have this many blocks available on its disk(s)):

```
/...:999:512
```

Notes:

a. A default **CacheInfo** file is usually created in **/opt/dcelocal/etc** during installation by the **dfs_cpfiles** program. Refer to the *z/OS Program Directory* for this release or to the Chapter 3, “Installation” on page 23 for more information on the **dfs_cpfiles** program.

b. The **DFSCM** default is disk caching. But, the **DFSCM** initialization parameter, **-memcache**, is specified as an **_IOE_CM_PARMS** environment variable value in the **/opt/dfslocal/home/dfscm/envar** file created by the **dfs_cpfiles** program during the installation process. Refer to the *z/OS Distributed File Service DFS Administration* book for more information on specifying **DFSCM** startup parameters using the environment variable **_IOE_CM_PARMS**.

2. Create and update the **/opt/dfslocal/home/dfscm/envar** file using the **DFSCM** initialization parameter, **_IOE_CM_PARMS**, to specify **-memcache**.

This step is necessary as the DFS client (**DFSCM**), when initialized by OMVS, reads the **envar** file specified as the **_EUV_HOME** variable value in the parameter library **BPXPRM***xx* **FILESYSTYPE TYPE(DFSC)** entry. A **_EUV_HOME** variable value must be specified in the **BPXPRM***xx* entry for the **DFSCM**. The use of the file **/opt/dfslocal/home/dfscm/envar** is recommended for **DFSCM** processing.

The DCE and DFS Environment Variables that are key to the DFS client are listed in “Customizing the DFSCM envar File” on page 67. For information on other DCE and DFS environment variables, see Appendix C, “DFS/SMB Environment Variables” on page 129.

Note: A z/OS shell configuration script program, **dfs_cpfiles**, is provided in the **/opt/dfsglobal/scripts** directory to copy an example **dfscm.envar** file from the **/opt/dfsglobal/examples** directory to **/opt/dfslocal/home/dfscm**. For more information, see “Running the dfs_cpfiles Program” on page 27.

Following is an example of a **DFSCM envar** file. The first line in the file disables the DFS client usage of the DCE single sign-on function. This controls whether the DFS client will attempt a DCE single sign-on for a user during a file request. The second line in the file controls where **DFSCM** messages are to be sent. In this example, they will be sent to the operator's console. The third line in the file passes initialization parameters to **DFSCM** or options that will take effect when the DFS client is started. In this example, the mounted file system name for the DFS client is set to **IOE_DFS_CLIENT_DATA** (this name is displayed by the UNIX command, **df**). Translation is set to **ON** for ASCII to EBCDIC by the **-translation text** entry. Memory caching is specified for the DFS client. The last line in the example sets the time zone to Eastern Standard Time.

```
_EUV_AUTOLOG=NO
_EUV_SVC_MSG_LOGGING=CONSOLE_LOGGING
_IOE_CM_PARMS=-mountfilesystem IOE_DFS_CLIENT_DATA -translation text -memcache
TZ=EST5EDT
```

Note: The **DFSCM envar** file example shown above reflects the contents of the file after the example **DFSCM envar** file has been copied from the **/opt/dfsglobal/examples** directory.

3. Stop and restart the **DFSCM**. For details on stopping **DFSCM**, see "Starting and Stopping the DFS Client (DFSCM)" on page 107.

DFSCM Disk Caching: The **DFSCM** disk cache resides on a local DCE Local File System aggregate that must be allocated and defined on the local z/OS system. **DFSCM** disk caching does not require the running of any DFS servers on the z/OS system on which **DFSCM** is running.

To configure a z/OS system to run **DFSCM** with disk caching:

1. Update the parmlib member, **BPXPRM***xx*, to add the **FILESYSTYPE TYPE(DFSC)** entry. Refer to "BPXPRMxx FILESYSTYPE TYPE(DFSC) Entry" on page 58 and to the *z/OS Distributed File Service DFS Administration* book for more information.
2. Update the local Hierarchical File System file, **/opt/dcelocal/etc/CacheInfo**, appropriately. Refer to "DFSCM Memory Caching" on page 59 and to the *z/OS Distributed File Service DFS Administration* book for more information.
3. Specify disk caching by adding the **-dcache** entry to the **DFSCM** initialization parameter, **_IOE_CM_PARMS** in the **/opt/dfslocal/home/dfscm/envar** file. (Note: **-dcache** is the default for **DFSCM** processing if **-memcache** is not specified.) Refer to the *z/OS Distributed File Service DFS Administration* book for more information.
4. Allocate and define a local DCE Local File System aggregate.

   The SIOESAMP library member **NEWAGGR** contains sample JCL and control statements that can be modified to allocate VSAM Linear Datasets and initialize a Logical Volume as a DCE Local File System aggregate.

   a. Modify the **/opt/dfslocal/var/dfs/devtab** file. The **devtab** file describes all the Linear Datasets that make up a Logical Volume. It consists of the names of the Linear Datasets that make up the Logical Volume.

   ```
   * Devtab - Example Entry for a logical volume
   define_lfs 999
   DFS.DCELFS.AGGR999.LDS00001
   DFS.DCELFS.AGGR999.LDS00002
   ```

   The previous example defined the device name **/dev/lsf999** that is specified as a PGM=IOENEWAG parameter in the JCL example in item 4c on page 62 that initializes a logical volume as a DCE Local File System aggregate.

   b. Use JCL similar to the following to allocate one or more VSAM Linear Datasets that will make up a Logical Volume. The Linear Datasets may reside on the same or different z/OS disk volumes. The size of the Linear Datasets determines the size of the DCE Local File System. Refer to the SIOESAMP library member **NEWAGGR** for example JCL.

```
//DFSDEFIN JOB ,'DFS Define LDSs',
//         CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*------------------------------------------------------------------
//*
//*  Allocate a VSAM Linear Datasets for use as a Logical Volume
//*
//*------------------------------------------------------------------
//DEFINE   EXEC  PGM=IDCAMS
//SYSPRINT DD     SYSOUT=*
//SYSUDUMP DD     SYSOUT=*
//AMSDUMP  DD     SYSOUT=*
//DASD0    DD     DISP=OLD,UNIT=3390,VOL=SER=PH2020
//DASD1    DD     DISP=OLD,UNIT=3390,VOL=SER=PH2021
//SYSIN    DD     *
  DEFINE CLUSTER (NAME(DFS.DCELFS.AGGR999.LDS00001) VOLUMES(PH2020) -
    LINEAR CYL( 50 0) SHAREOPTIONS(2) )
  DEFINE CLUSTER (NAME(DFS.DCELFS.AGGR999.LDS00002) VOLUMES(PH2021) -
    LINEAR CYL( 50 0) SHAREOPTIONS(2) )
//
```

c. Run JCL similar to the following to initialize the Logical Volume as a DCE Local File System aggregate. Refer to the *xxx*.SIOESAMP(NEWAGGR) library member (where *xxx* is installation dependent) for example JCL. To access the **devtab** file, the PGM=IOENEWAG must be run under **UID = 0** (**root**).

```
//DFSNEWAG JOB ,'DFS NewAggr',
//         CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*------------------------------------------------------------------
//*
//*  Format the Logical Volume as a DCE Local File System aggregate.
//*
//*  This version of newaggr also "loads" the VSAM linear data
//*  set(s) if they have not been previously loaded.
//*
//*  NOTES:
//*       - Note that the "/dev/lvm" *must* be lower case!
//*       - Note that the extra '/' is required because LE
//*         runtime parameters are separated from program
//*         parameters by a '/'.
//*
//*  The syntax of the newaggr command is as follows:
//*
//*  newaggr -aggregate name -blocksize bytes -fragsize bytes
//*       [-initialempty blocks] [-aggrsize blocks]
//*       [-logsize blocks] [-overwrite] [-verbose] [-noaction]
//*
//*------------------------------------------------------------------
//NEWAGGR  EXEC  PGM=IOENEWAG,
// PARM=('//dev/lfs999 8192 1024 -verbose')
//*
//SYSPRINT  DD    SYSOUT=*
//SYSUDUMP  DD    SYSOUT=*
//*
```

Notes:

1) IBM recommends using a **-blocksize 8192** for optimum disk cache performance and storage utilization.

2) No updates to the **/opt/dfslocal/var/dfs/dfstab** file are required since the local DCE File System aggregate used for **DFSCM** disk caching is not exported. Also, **DFSCM** processing completes any initialization of the aggregate to be used for disk caching.

5. Stop and restart the **DFSCM**. For details on stopping the **DFSCM**, see “Starting and Stopping the DFS Client (DFSCM)” on page 107.

DFSCM User .profile File Considerations

To reduce overall processing overhead associated with the **DFSCM** support for the z/OS DCE single sign-on function, it is recommended that users issue the following command after initializing z/OS UNIX:

```
$ cd /...
```

This can be automated by updating the users **/home/***mvs_userid***/.profile** file. Edit the file, adding the following statements:

```
cd /...
cd users_home_directory
```

The first entry will change the directory to the DFS global namespace's pathname prefix, **/...**. The second entry changes the directory to the user's home directory.

Enabling DFSCM to Use DCE Single Sign-on

Distributed File Service supports z/OS DCE's single sign-on function. To enable DCE single sign-on processing for a user accessing the DFS global namespace, **/...**, a Resource Access Control Facility (RACF) segment must first be created. If there are currently no DCE credentials (or they have expired) when the DFS client is invoked, then the DFS client does one of the following:

- Attempts a DFS single sign-on, or
- Sends the file request as an unauthenticated request.

Note: Although RACF is mentioned, any z/OS external security manager (ESM) that has equivalent support can be used instead of RACF.

For detailed information regarding the DFS client usage of the DCE single sign-on function, refer to the the *z/OS Distributed File Service DFS Administration* book.

For user **.profile** file considerations related to DCE single sign-on function, see “DFSCM User .profile File Considerations.”

DFSCM Environment Variable Considerations

To complete the **DFSCM** configuration, you can optionally customize the **DFSCM** envar file as described in Appendix B, “Directories and Files” on page 123.

Customizing the ioepdcf File

On Distributed File Service, the DFS Server **ioepdcf** file contains information for each server process you can run in DFS. This file can be customized to meet your needs. The **ioepdcf** file must be located in the following directory: **/opt/dfslocal/etc**. An example **ioepdcf** file can be found in **/opt/dfsglobal/examples**.

If you are installing Distributed File Service for the first time, you can copy the example file to the **/opt/dfslocal/etc** directory and customize the file. The shell script, **/opt/dfsglobal/scripts/dfs_cpfiles**, automates this procedure (see "Running the dfs_cpfiles Program" on page 27 for more information).

If you have previously installed an earlier version of Distributed File Service, you will be able to maintain your customization by using your existing **ioepdcf** file stored in **/opt/dfslocal/etc**.

The following describes how to customize the **ioepdcf** file for your system.

1. Determine what Distributed File Service processes you want to run.

   All processes running under the control of the Control Task program, **DFSCNTL**, are valid process names that can be specified in the **ioepdcf** file. These processes are:

   - **dfskern**
   - **export**
   - **unexport**
   - **boserver**
   - **butc***nn*. Valid entries for *nn* are 01 through 08.

2. Specify a configuration type for each process you have selected.

   Each process identified in the **ioepdcf** file, must specify a configuration type. There are four valid types available:

Configuration Type	**Explanation**
CONFIGURED=Y	The specified process will start during DFS initialization. Should the process abend or end for any reason, it will automatically be restarted by the DFS Control Process. **Note:** This configuration type is not supported with the **butc** processes.
CONFIGURED=N	The process will not be started by the DFS Control Process. The process cannot be started manually.
CONFIGURED=I	The specified process will be started during DFS initialization **or** when the z/OS Operator command **modify dfs,start all** is issued. The process may also be started manually. The specified process is **not** restarted when it is terminated. **Note:** This configuration type is not supported with the **butc** processes.
CONFIGURED=M	The process will not be started by the DFS Control Process but can be manually started by a z/OS system **modify** command to DFS. The specified process will not restart if it ends for any reason. Always use CONFIGURED=M for **butc***nn* processes.

3. Identify the name of the load module (**LMD**) which corresponds to the name of the member in the SIOELMOD data set in the z/OS partitioned data set (PDS) load library. The SIOELMOD data set was created during installation.

   Each process entry in the **ioepdcf** file must contain the name of the **LMD** which refers to the name of the member in the SIOELMOD data set in the z/OS partitioned data set (PDS) load library. For more information, refer to the *z/OS Program Directory*.

   The following are the PDS member names for the DFS processes started by the DFS Control Task:

boserver The **boserver** daemon load module name. The name, **boserver**, is an alias for the load library entry, **IOEBOSRV**.

butc*nn* The **butc***nn* daemon load module name. There are eight valid load module names for the butc processes: **butc01**, **butc02**, **butc03**, **butc04**, **butc05**, **butc06**, **butc07**, and **butc08**. All are valid aliases for the single load module used for all **butc***nn* processes- **IOEBUTC**. Always use CONFIGURED=M for **butc***nn* processes. The **butc***nn* processes require **bakserver** to be running to complete configuration.

dfskern The **dfskern** daemon load module name. The name, **dfskern**, is an alias for the load library entry, **IOEDFSKN**.

export The **export** process is started by the DFS Control Task and executes the load library entry, **IOEDFSXP**. The **export** process has no alias.

4. Include any special parameters to be passed to the **LMD** when a process is started.

   Each process entry in the **ioepdcf** file includes special parameters that will be passed to the load module when a daemon is started (including Language Environment/370 (LE/370) runtime options). This is also called the argument list. Any runtime overrides such as storage specifications and redirection of output may also be added. The first argument is the home directory for each process. The home directory points the process to the directory holding the environment variable (**envar**) file for the process. Program parameters for the DFS process are preceded with a slash, **/**, in the argument list.

5. Include any special parameters for controlling restart and timeout intervals for your system.

   Additional special parameters that control restart and timeout intervals may also be entered in the **ioepdcf** file. The **ioepdcf** file can also be used to override the default parameter options for the **file exporter** (**fxd**) daemon. Changes are specified in the **dfskern** process entry. For more information on the **ioepdcf** file, refer to the *z/OS Distributed File Service DFS Administration* book.

The following example shows an **ioepdcf** file entry for the **dfskern** process. The configuration type is **Y**, specifying that the process start during DFS initialization. The load module, **LMD**, is identified as **DFSKERN**. In the argument list, **ARG**, **ENVAR** indicates the environment variables to be used. In the example, the home directory is identified as **_EUV_HOME=/opt/dfslocal/home/dfskern**. An LE/370 runtime option is specified: **RPSTG(OFF)**. Parameters for the **fxd** program follow the **/**. The > symbol is a redirection character which indicates that the output will be redirected to the data definition name (ddname) that follows, DFSKERN. The **2**>**&1** entry redirects **STDERR** to the same destination as **STDOUT**, DFSKERN. **RESTART** and **TIMEOUT** values are both set at 300 seconds. The **RESTART** parameter specifies how long the process must be down before DFS control will attempt to restart it. The **TIMEOUT** parameter specifies how long DFS control will wait for the process before it times out.

```
DFSKERN CONFIGURED=Y LMD=DFSKERN \
   ARG="ENVAR('_EUV_HOME=/opt/dfslocal/home/dfskern'),RPSTG(OFF)/ \
   -mainprocs 7 -admingroup subsys/dce/dfs-admin \
   >DD:DFSKERN 2>&1" RESTART=300 TIMEOUT=300
```

Note: The **-admingroup** parameter is required for the file exporter daemon (**fxd**).

Customization of the DFS Server and the DFSCM envar Files

This chapter provides information on the following topics:

- “Customizing the DFS Server envar Files” on page 66
- “Customizing the DFSCM envar File” on page 67.

Customizing the DFS Server envar Files

Before Distributed File Service can be initialized, each DFS process that you choose to start must have a corresponding **envar** file created. On Distributed File Service, the **envar** file for each server process includes information that affects the behavior of the process. The variables for each **envar** may be customized. Setting or changing these variables is optional. Default values are provided if you choose not to set them.

To function properly, the files **must** be stored in the appropriate directories. The following table shows the correct directory for each processes' **envar** file, and the name of the example **envar** supplied in the **/opt/dfsglobal/examples** directory.

Table 12. Directory Locations for envar Files on Distributed File Service

Distributed File Service Process	Directory Where Located in /opt/dfslocal	IBM-Supplied example envar in /opt/dfsglobal/examples
bakserver	**home/bakserver/envar**	**bakserver.envar**
boserver	**home/boserver/envar**	**boserver.envar**
butc*01*	**home/butc01/envar**	**butc01.envar**
butc*02*	**home/butc02/envar**	**butc02.envar**
butc*03*	**home/butc03/envar**	**butc03.envar**
butc*04*	**home/butc04/envar**	**butc04.envar**
butc*05*	**home/butc05/envar**	**butc05.envar**
butc*06*	**home/butc06/envar**	**butc06.envar**
butc*07*	**home/butc07/envar**	**butc07.envar**
butc*08*	**home/butc08/envar**	**butc08.envar**
daemonct	**home/daemonct/envar**	**daemonct.envar**
dfscm	**home/dfscm/envar**	**dfscm.envar**
dfscntl	**home/dfscntl/envar**	**dfscntl.envar**
dfsexport	**home/dfsexport/envar**	**dfsexport.envar**
dfskern	**home/dfskern/envar**	**dfskern.envar**
flserver	**home/flserver/envar**	**flserver.envar**
ftserver	**home/ftserver/envar**	**ftserver.envar**
growaggr	**home/growaggr/envar**	**growaggr.envar**
newaggr	**home/newaggr/envar**	**newaggr.envar**
repserver	**home/repserver/envar**	**repserver.envar**
salvage	**home/salvage/envar**	**salvage.envar**
upclient	**home/upclient/envar**	**upclient.envar**
upserver	**home/upserver/envar**	**upserver.envar**

If you are installing Distributed File Service for the first time, the **envar** files for each process you choose to run must be copied from **/opt/dfsglobal/examples** to the appropriate directory identified in the previous table. A z/OS shell script provided with Distributed File Service, **dfs_cpfiles**, automates the process of copying the IBM-supplied **ioepdcf** and **envar** files from the **/opt/dfsglobal/examples** directories to the appropriate locations if they do not already exist from a previous installation of Distributed File Service. **dfs_cpfiles** is located in the directory **/opt/dfsglobal/scripts**. For more information on this utility, see "Running the dfs_cpfiles Program" on page 27.

If you have previously installed an earlier version of Distributed File Service, you will be able to use your existing **envar** files without additional action. You can also take advantage of additional processes when you install the release, by copying the example **envars** from **/opt/dfsglobal/examples** to the respective directories, **/opt/dfslocal/home/bakserver/envar** or **/opt/dfslocal/home/flserver/envar**. The Distributed File Service Default Configuration Files Creation Program **/opt/dfsglobal/scripts/dfs_cpfiles** can also be used to automate this process (for more information on this utility, see "Running the dfs_cpfiles Program" on page 27).

The following example shows how to customize the settings for the **dfskern** process. This variable sets the number of threads to be started in **dfskern** to service token requests from the glue layer. The number of threads may be set to any number, *n*. The default is **5**.

```
_IOE_TKMGLUE_SERVER_THREADS=n
```

Note: Appendix C, "DFS/SMB Environment Variables" on page 129 provides a description of all environment variables relevant to DFS administration. The appendix provides default values, examples, and descriptions of the environment variables. Refer to the *z/OS Distributed File Service DFS Administration* book, for more information on setting environment variables.

Customizing the DFSCM envar File

When initialized by OMVS, the DFS client (**DFSCM**) reads the environment variable, **envar**, file specified as the **_EUV_HOME** variable value in the system parmlib member **BPXPRM***xx* **FILESYSTYPE TYPE(DFSC)** entry. An **_EUV_HOME** variable value must be specified in the **BPXPRM***xx* entry for the **DFSCM**. The use of the file, **/opt/dfslocal/home/dfscm/envar**, is recommended for the **DFSCM**. Refer to the *z/OS Distributed File Service DFS Administration* book, for more information.

The DCE and DFS environment variables key to the DFS client include:

- **_EUV_AUTOLOG**
- **_EUV_SVC_MSG_LOGGING**
- **_IOE_CM_PARMS**.

See Appendix C, "DFS/SMB Environment Variables" on page 129, for a more comprehensive listing of the environment variables used in DFS.

Chapter 5. Defining DFS Administrators and Users

A ***DFS administrator*** refers to a z/OS user ID used to issue DFS administrator requests to a DFS server running on the same DCE host system or another DCE host system in the DCE cell. A ***DFS user*** refers to a z/OS user ID used to issue file requests that access the DFS namespace. The file requests are routed through the DFS Client to the appropriate DFS server.

All DFS administrators and users must be authenticated and authorized by the DCE Security Service to be able to access the DFS servers. The DCE administrator has to create a DCE account for each DFS administrator and user.

Also, a DFS administrative list file must be created and updated for each DFS server process run on a DCE host system.

If DFS administrators and users also need access to HFS or RFS filesets exported by DFS servers, a relationship (a **mapping**) must be established between their DCE user IDs and their z/OS user IDs on the host system where DFS servers are running and exporting the HFS or RFS filesets.

This chapter discusses the following topics related to defining DCE administrator and user accounts for DFS. This information should be performed in the order that appears.

- "Creating DCE User Accounts for DFS Administrators and Users"
- "Defining a DFS Administrator" on page 70
- "System Control Machines and Domains" on page 71
- "Description of DFS Administrative List Files" on page 72
- "Creating the Initial DFS Administrator(s)" on page 72
- "Disable/Enable DFS Authorization (NoAuth File)" on page 74
- "Creating the admin.bos File" on page 74
- "Adding Principals and Groups to Administrative Lists" on page 74
- "Mapping z/OS User IDs for HFS and RFS Access" on page 75 includes the following sections:
  - "Registering Users With the z/OS Security Subsystem" on page 75
  - "Mapping DCE User IDs to z/OS User IDs" on page 75
- "DFS Anonymous User Considerations" on page 76.

Creating DCE User Accounts for DFS Administrators and Users

The DCE administrator (also known as the cell administrator) must add the DCE user account for the DFS administrator and users to the Security registry using the DCE administrative interface, **dcecp**.

The procedures for creating a DCE account described here assume that the principal, group, and organization have not been created prior to the creation of the account. Skip the relevant steps if the principal, group, and organization have previously been created for the user.

To create a DCE account, follow these steps:

1. Invoke the DCE administrative interface

   ```
   => dcecp
   ```

2. Use the **principal** subcommand to add the principal for the user. For example, if the user is named **smith**:

   ```
   dcecp> principal create smith
   ```

   This assigns a default UNIX number, full name, and object creation quota. If you want to enter values other than the defaults, see the *z/OS DCE Administration Guide* for details.

3. Add the group (or groups) to which the user belongs. This step must only be performed if the group (or groups) have not been previously added to the registry. For example, if the name of the group is **usergrp**:

   ```
   dcecp> group create usergrp -inprojlist yes
   ```

   This assigns a default UNIX number, full name, and includes the group, **usergrp**, in the project list. If you want to enter values other than the defaults, see the *z/OS DCE Administration Guide* for details.

4. Add the principal to the group. For example, to add the principal, **smith**, to the group, **usergrp**:

   ```
   dcecp> group add usergrp -member smith
   ```

5. Add the organization to which the user belongs. This step must only be performed if the organization has not previously been created in the registry. For example, to add the organization, **ibm**:

   ```
   dcecp> organization create ibm
   ```

   The above assigns a default UNIX number and full name. If you want to enter values other than the defaults, see the *z/OS DCE Administration Guide* for details.

6. Add the principal to the organization. For example, to add the principal, **smith**, to the organization, **ibm**:

   ```
   dcecp> organization add ibm -member smith
   ```

7. Add the account for the user. For example, to add the account for a user whose principal is **smith**:

   ```
   dcecp> account create smith -group usergrp -organization ibm \
                 -password xxx -mypwd dce_principal_password
   ```

   Note: In the above example, a user's password, *xxx*, associated with the newly created account and the administrator's password, *dce_principal_password*, must be supplied.

After creating the DCE user account for a DFS administrator, the applicable DFS administrative lists must be updated as described in “Defining a DFS Administrator.”

Additionally, if the DFS administrators and/or users need access to HFS and RFS filesets exported by DFS servers, their DCE user IDs must be mapped to their z/OS user IDs on the host system for the DFS servers. This procedure is described in “Mapping z/OS User IDs for HFS and RFS Access” on page 75.

Defining a DFS Administrator

Before an administrator is added to DFS, the principal has to be authenticated and authorized by the DCE Security Service to be able to perform administrative tasks. A DCE account has to be created by the DCE administrator for the DFS administrator. Creating an account for the DFS administrator follows the same procedure as creating an account for a DFS user. For detailed information regarding this procedure, see “Creating DCE User Accounts for DFS Administrators and Users” on page 69.

Note: The DCE **cell_admin** user ID inherits no special DFS authorization. The user ID **cell_admin** has to be explicitly specified in a DFS administrative list file.

In DFS, administrative lists are used to define the principals and groups that can perform actions affecting specific server processes on a server machine. There is one DFS administrative list for each DFS server process running on a machine. For example, a server's **admin.bos** file defines who has administrative rights to the BOS Server (**boserver**) and determines the users and servers that can manipulate and maintain server processes on that server.

Because administrative lists exist on a per-process and per-machine basis, different groups of principals can have different sets of administrative privileges within a domain. It is often useful to have the same group or user on several lists.

The following sections contain topics related to defining a DFS administrator:

- "System Control Machines and Domains"
- "Description of DFS Administrative List Files" on page 72
- "Creating the admin.bos File" on page 74
- "Creating the Initial DFS Administrator(s)" on page 72
- "Adding Principals and Groups to Administrative Lists" on page 74.

System Control Machines and Domains

A group of DCE host machines in a DCE cell where DFS server processes run is called a ***domain***. There is a System Control Machine for each domain. The **upserver** process runs on the System Control Machine. The administrative list file, for each DFS server process run on any DCE host machine in the domain, is kept up to date and synchronized on each DCE host machine by the interaction of the Update Client (**upclient**) or the Update Server (**upserver**) processes running on the DCE host machines in the domain. Refer to *z/OS Distributed File Service DFS Administration* for more information on System Control machines and domains.

Administrative list files should be updated on the DCE cell's System Control machine where the Update Server (**upserver**) is running. The **upserver** then propagates administrative list file changes to the secondary sites where an Update Client (**upclient**) is running. If an administrative file is updated on a DCE host machine where an **upclient** is running and the updates are not also made on the system where the **upserver** is running, the updates are overlaid on the **upclient** system if the **upserver** is acting as a system control machine for that file.

DFS Administrator Considerations for a DCE Cell

It is recommended that you add the same DFS administrator principals or groups to all the administrative list files in the DCE cell. This allows these DFS administrators to issue requests to any DCE host system in the DCE cell where DFS is running.

DFS Server Principal Considerations for a DCE Cell

When adding a new system with DCE host name *dcehostname* to a DCE cell, the abbreviated DFS server principal **hosts/***dcehostname* for the new DCE host system should be added to the **/opt/dcelocal/var/dfs/admin.fl**, **/opt/dcelocal/var/dfs/admin.ft**, and **/opt/dcelocal/var/dfs/admin.bak** files on the other systems in the DCE cell where DFS is configured. This should be done even if the related DFS server is not being run on a DCE host system at this time. Updating the DFS administrative file on the system where the **upserver** is running enables the update to be propagated to other systems in the DCE cell running an **upclient**. Refer to the following sections for information on how to update these administrative files with the abbreviated DFS server principal:

- "Completing the FLDB Server (flserver) Configuration" on page 47
- "Completing Fileset Server (ftserver) Configuration" on page 49
- "Completing the Backup Database Server (bakserver) Configuration" on page 53.

Description of DFS Administrative List Files

In DFS, administrative lists are used to define the principals and groups that can perform actions affecting specific server processes on a server machine. There is one DFS administrative list for each DFS server process running on a machine. For example, a server's **admin.bos** file defines who has administrative rights to the BOS Server (**boserver**) and determines the servers and/or users that can manipulate and maintain server processes on a host system.

Administrative lists exist on a per-process and per-machine basis so different groups of principals can have different sets of administrative privileges within a domain. But, it is often useful to have the same group or user on several DFS administrative lists.

DFS **bos** commands are used to maintain the administrative list files. Refer to the *z/OS Distributed File Service DFS Administration* book for more information on the DFS **bos** commands.

The default directory for the administrative lists is the configuration directory (**/opt/dcelocal/var/dfs**). If the lists are stored in the default directory, you need to provide only the specific file name (for example: **admin.bak**, **admin.bos**, **admin.fl**, **admin.ft**, or **admin.up**) in the DFS **bos** commands.

The DFS administrative list files are :

- The **admin.bak** file is associated with the Backup Server (**bkserver**).
- The **admin.bos** file is associated with the BOS Server (**boserver**).
- The **admin.fl** file is associated with the Fileset Location Database Server (**flserver**).
- The **admin.ft** file is associated with the Fileset Server (**ftserver**).
- The **admin.up** file is associated with the Update Server (**upserver**).

Administrative lists for server processes are created in one of two ways:

1. When a server is started and a list does not already exist. By default, a server process automatically creates an administrative list and places its list in the configuration directory (**/opt/dcelocal/var/dfs**). An administrative list generated by a process is always empty.

   The BOS server **admin.bos** administrative list file is created before any other administrative list file using this method. Refer to "Creating the admin.bos File" on page 74 for a description of how to establish the **admin.bos** administrative list.
2. Administrative lists other than the **admin.bos** list are created by issuing the **bos addadmin** command with the **-createlist** option. Refer to "Adding Principals and Groups to Administrative Lists" on page 74 for a description of how to create an administrative list file using the **bos addadmin** command.

Refer to the *z/OS Distributed File Service DFS Administration* for more information on **bos** commands and the DFS administrative list files.

Creating the Initial DFS Administrator(s)

During the initial installation of DFS on a host system, an empty **admin.bos** file is created when **boserver** is first started if the file does not exist. Consequently, there may be no **admin.bos** file entries that define a DFS administrators.

The **boserver** authorization checking must be disabled to allow the initial DFS administrators to be defined. When the **boserver** is running, creating the **/opt/dcelocal/var/dfs/NoAuth** file temporarily disables **boserver** authorization.

The initial DFS administrators can be defined by:

1. Logging in as **root**.

   Creating the file directly requires logging into the z/OS system with a **root** userid with **UID = 0**.

2. Creating the **NoAuth** file to disable **boserver** authorization checking by issuing the command:

   ```
   touch /opt/dcelocal/var/dfs/NoAuth
   ```

   The **NoAuth** file must be created in the directory **/opt/dcelocal/var/dfs**. The file is always deleted when the **boserver** is re-started. Additionally, the **NoAuth** file is created by the **bos setauth /.:/hosts/***dcehostname* **off** command and deleted by the **bos setauth /.:/hosts/***dcehostname* **on** command.

3. The **bos status /.:/hosts/***dcehostname* can be issued to determine in **boserver** authorization is disabled. Note that the **bos status** command reports if authorization is disabled but the command output does not explicitly report if authorization is enabled.

   An example of the **bos status** command issued for the **boserver** running on a system with DCE host name *dcehostname* is :

   ```
   $ bos status /.:/hosts/dcehostname
   ```

4. Define the initial DFS administrators while DFS authorization is disabled.

   During the initial setup the initial setup of DFS on a z/OS DCE host system, you should initially add a user and a group with DCE **cell_admin** authority to the **/opt/dfslocal/var/dfs/admin.bos** file which was created when the **boserver** was first started. Note that if DFS was previously configured on the DCE host system, DFS administrators may still be defined in the various DFS administrator list files including the **bos admin** file.

   Before any principal or group can be added to a DFS administrative list file, the DCE principal and group must be defined as described in "Creating DCE User Accounts for DFS Administrators and Users" on page 69.

   The DFS configuration examples assumes that the userid **cell_admin** was previously defined during DCE configuration and that the group **dfsteam** was defined by the command:

   ```
   $ dcecp -c group create dfsteam -inprojlist yes
   ```

   The following command is an example of how to add the userid **cell_admin** and the group **dfsteam** to the **admin.bos** file on a system with DCE host name *dcehostname*:

   ```
   $ bos addadmin -server /.:/hosts/dcehostname \
                  -adminlist admin.bos -principal cell_admin -group dfsteam \
                  -createlist
   ```

   Later you are instructed when to add additional administrators and update the DFS administrative list files on the DCE host system where the **upserver** is running.

5. Enable boserver DFS authorization checking on a system with DCE host name *dcehostname* using the command:

   ```
   $ bos setauth /.:/hosts/dcehostname on
   ```

   After enabling DFS authorization checking using the **bos setauth** command, **boserver** DFS authorization checking can be enabled and disabled while the **DFS** is running using the **bos setauth** on command or creating the **NoAuth** file. Note that DFS authorization checking is re-enabled each time the **boserver** is restarted.

6. Insure that **boserver** authorization checking is enabled on a system with DCE host name *dcehostname* using the command:

```
$ bos status /.:/hosts/dcehostname
```

The **NoAuth** file should subsequently be created and deleted using the **bos setauth** command. For detailed information on the **bos** commands, see the *z/OS Distributed File Service DFS Administration*.

After adding DFS administrators to the **admin.bos** file and enabling **boserver** DFS authorization checking, a DFS administrator can make additional updates to the **admin.bos** administrative list file using the same method employed for the other administrative list files. Refer to "Adding Principals and Groups to Administrative Lists" for details on creating and updating the DFS administrative list files after initially establishing the **admin.bos** list.

Disable/Enable DFS Authorization (NoAuth File)

During the initial installation of DFS on a host system, an empty **/opt/dfslocal/var/dfs/admin.bos** file is created when the **boserver** is first started if the file does not exist. Additionally, each time the **boserver** is started, DFS authorization is enabled.

Before DFS administrators are defined, you can not issue a **bos setauth off** command to disable DFS authorization nor can you update the **admin.bos** file to define DFS administrators using the **bos addadmin** command. So, after the **boserver** is first started, you can disable DFS authorization by creating the **/opt/dcelocal/var/dfs/NoAuth** as described in "Creating the Initial DFS Administrator(s)" on page 72

After initial DFS administrators are defined, the **bos setauth** can be used to disable and enable DFS authorization.

The **/opt/dcelocal/var/dfs/NoAuth** file is created as described in "Creating the Initial DFS Administrator(s)" on page 72 or when a **bos setauth off** command is successful. The **/opt/dcelocal/var/dfs/NoAuth** file is deleted each time the **boserver** is started or when a **bos setauth on** command is successful.

Creating the admin.bos File

Starting the BOS Server **boserver** for the first time on a DCE host system creates an empty **/opt/dfslocal/var/dfs/admin.bos** file if the file does not exist. But, the **admin.bos** file can not be updated to define the DFS administrators unless DFS authorization is disabled. During the initial configuration of DFS, you should follow the instructions described in "Creating the Initial DFS Administrator(s)" on page 72.

Adding Principals and Groups to Administrative Lists

After the **admin.bos** file is created and updated to establish initial DFS administrators as described in "Creating the admin.bos File," you can create the other administrative list files and add administrators or groups. Refer to the *z/OS Distributed File Service DFS Administration* book for more information.

Mapping z/OS User IDs for HFS and RFS Access

For authorized HFS and RFS access on the host system for the DFS servers, DFS administrators and users must be registered with the z/OS security subsystem on the host system for the DFS servers. There must also be a defined relationship (a **mapping**) between their DCE user IDs and their z/OS user IDs.

Included in this and the following sections are topics related to setting up DFS administrator and user access to HFS and RFS filesets which include:

- "Registering Users With the z/OS Security Subsystem"
- "Mapping DCE User IDs to z/OS User IDs"
- "DFS Anonymous User Considerations" on page 76.

Note: Although RACF is discussed in the following sections, any z/OS external security manager (ESM) that has equivalent support can be used instead of RACF.

Registering Users With the z/OS Security Subsystem

Clients of DFS may access HFS or RFS filesets exported by z/OS DFS servers. Access to these resources is controlled by the local z/OS security subsystem (for example, RACF). Thus, all users of the DFS must be registered with the specific security subsystem on the z/OS host where DFS servers are running and exporting HFS or RFS filesets. Refer to the documentation provided by the Security software on your host for information on registering users on the local security subsystem.

Mapping DCE User IDs to z/OS User IDs

The DFS server enables DFS clients to access non-DCE Local File System filesets.

The local security subsystem determines if the client is authorized to use the resource. Because the local security subsystem is not integrated with the DCE Security service, the local security subsystem cannot recognize DCE user IDs.

The administrator has to establish a relationship (that is, a **mapping**) between a user's DCE user ID and z/OS user ID. This section describes methods that Distributed File Service may use to map DCE user IDs to z/OS user IDs. The methods are:

1. Mapping the DCE user IDs to the local z/OS user IDs using the **mapid** utility.

   Note: The mapping procedure in Distributed File Service is identical with that used for z/OS DCE Application Support.

   Refer to the *z/OS Distributed File Service DFS Administration* book for more information.

2. Mapping the DCE user IDs to the local z/OS user IDs using the identity mapping function of the external security manager (ESM) in use on the host system.

   Note: Although RACF is discussed in this chapter, any z/OS external security manager (ESM) that has equivalent support can be used instead of RACF.

   Refer to *z/OS Distributed File Service DFS Administration* book for a description of mapping use IDs and creating the identity mapping file.

3. Optionally establishing a default user ID for unauthenticated access to HFS and RFS filesets.

   Refer to the *z/OS Distributed File Service DFS Administration* book for more information.

Note: In some cases, the DFS client request is denied if the DFS server cannot determine a mapping to a z/OS user ID. For example, if the user is unauthenticated or not mapped or mapped but not in RACF and if **_IOE_MVS_DFSDFLT** is not specified or the **_IOE_MVS_DFSDFLT** user ID is not in RACF the mapping function denies the request.

Refer to Appendix C, "DFS/SMB Environment Variables" on page 129 for a description of **_IOE_MVS_DFSDFLT**.

DFS Anonymous User Considerations

An anonymous user ID is required for unauthenticated access to non-DCE Local File System filesets (for example, HFS or RFS filesets) exported by DFS servers running on the host system. An anonymous user ID is necessary if the DCE-principal-to-z/OS ID mapping fails or if a user has not completed the dce_login process and is, therefore, unauthenticated. If an anonymous user ID is not defined, unauthenticated access to HFS or RFS filesets fails.

The DFS anonymous user ID can be created at this time to allow unauthenticated access to HFS or RFS filesets. If the anonymous user ID is not created at this time, it can be added later.

The following example shows how you can create the DFS anonymous user ID by the Resource Access Control Facility (RACF) using TSO/E:

```
ADDUSER dfsdflt DFLTGRP(dfltgroup) OWNER(owner) NAME('DFS DEFAULT ID') \
  OMVS(UID(dfltuid) HOME(/u))
```

Once created, you must specify the ID in the environmental variable file for **dfskern**. To do so, enter the following, where DFSDFLT is the default ID for the DFS anonymous user ID you have created:

```
_IOE_MVS_DFSDFLT=DFSDFLT
```

Note that you must specify a z/OS Unique Identifier (UID) segment and its group **must** have a z/OS Group Identifier (GID) segment.

Chapter 6. Exporting Data in DFS

Exporting data in DFS requires you to:

1. Create DFS aggregates for the filesets or data sets to be exported by updating the **/opt/dcelocal/var/dfs/devtab** file.

   For DCE Local File System aggregates, you must also allocate VSAM linear data sets and format them so that they can be used for DFS filesets.

2. Export the aggregates by updating the **/opt/dcelocal/var/dfs/dfstab** file.
3. Create the first fileset **root.dfs** for the DCE cell if it does not already exist.
4. Create mount points in the DFS namespace for other filesets to be exported by DFS from this host system.

This chapter describes the procedures for exporting data in Distributed File Service. Topics included in this chapter are:

Creating Aggregates in DFS

To make data available from the DFS file server you have just configured (see “Using DFSCONF to Configure the DFS File Server” on page 44), you must export data and create filesets from a File Server using either z/OS Hierarchical File System (HFS), Record File System (RFS), or DCE Local File System filesets.

The following sections provide information on how to properly create aggregates for HFS, RFS, and DCE Local File System filesets and how to export the aggregates.

Creating Non-Local File System Aggregates in DFS

On Distributed File Service, a Hierarchical File System (HFS) aggregate is the equivalent of a single HFS partition. The HFS aggregate also is seen as a single complete fileset by the DFS File Server. Because of this, exported HFS aggregates (non-Local File System aggregates) can contain only one fileset per partition and the logical volume of any HFS aggregate equals a single HFS partition. Before you can export an HFS data set, it must be mounted locally to the z/OS address space. For details regarding allocating and mounting an HFS file system, see *z/OS UNIX System Services Planning*, GA22-7800.

Once mounted, an entry must be created in the **/opt/dcelocal/var/dfs/devtab** file. This entry maps a minor device number to the HFS file system you wish to export. The minor device number is an identifier for the device you plan to mount. The number can be any integer greater than zero. Each HFS (non-Local File System) aggregate must have a unique minor device number. The minor device number need not be unique across both DCE Local File System and non-Local File System aggregates.

The following is an example of a completed **devtab** file entry. Lines beginning with an asterisk (*) are comment lines. The first entry in the file defines the type of file system, **ufs**. This defines the file system as HFS. The minor device number, **2**, is assigned. This number is a unique identifier and becomes part of the name of the defined logical volume. The logical volume name must be specified in the **/opt/dcelocal/var/dfs/dfstab** file when exporting the HFS partition to DFS (see “Exporting Non-Local File System Data” on page 80). The last entry in the file, **omvs.user.abc**, is the name of the HFS file system you are preparing to export.

```
* HFS Devices
define_ufs 2
omvs.user.abc
```

An RFS (Record File System) aggregate is a z/OS data set name that is either the name of a partitioned data set or is the prefix of a set of data sets and is a non-Local File System aggregate. The whole aggregate is also a single complete fileset to the DFS server.

You must add an entry in **/opt/dcelocal/var/dfs/devtab** which maps a minor device number to the RFS fileset you wish to export. For each RFS fileset add the following entries:

```
* RFS devices
define_ufs 3 rfs
USERA
```

In the previous two examples, lines beginning with an asterisk are comments. The next line defines the type of file system (**ufs**), the minor device number and whether the **ufs** file system is an **hfs** (hierarchical file system) subtype or an **rfs** (record file system) subtype. The default subtype is **hfs**. In this case, we are defining a **ufs** with a subtype of **rfs**.

Note: From now on, we will generally refer to this as an **rfs** fileset. You should keep in mind, however, that it is really a **ufs** fileset with a subtype of **rfs**. When we make comments about a **ufs** fileset, they pertain to both subtypes - **hfs** and **rfs**.

As with a DCE Local File System aggregate, the HFS fileset and the RFS fileset have a minor number describing the device, **2** in the HFS example and **3** in the RFS example. This number can be any integer greater than zero. Each **ufs** fileset must have a unique minor device number. (That is, a **ufs** fileset can have the same minor device number as an **lfs** fileset, but each **ufs** fileset, whether it is an **hfs** subtype or an **rfs** subtype, must have a unique minor device number.) Similarly, two DCE Local File System aggregates cannot have the same minor device number.

The third line in the example is the name of the HFS file system or RFS fileset (prefix) you wish to export. The third line also supports an optional parameter in addition to the data set name in the **devtab** for HFS or RFS filesets to control character data translation.

You may also specify the translation control parameter after the HFS or RFS file system name, on the same line. Valid values for the translation control parameter are:

binary Do not translate data.

text Converts incoming data from ASCII ISO 8859-1 to the local z/OS code page. Converts outgoing data from the local z/OS code page to ASCII ISO 8859-1.

The following example demonstrates the use of the translation control parameter. In this example, the fileset identified as **omvs.user.abc** will not have data translated.

```
* HFS Devices
define_ufs 2
omvs.user.abc binary
```

If the translation control parameter is omitted the default is controlled by the **_IOE_HFS_TRANSLATION** environment variable setting (for HFS filesets) and the **_IOE_RFS_TRANSLATION** environment variable setting (for RFS filesets) in the **dfskern** process, see the *z/OS Distributed File Service DFS Administration* book for more information.

Finally, the **attrfile** *attributes_file* parameter (where *attributes_file* is the name of the attributes file that controls the data set creation, processing, and site attributes for this RFS fileset) is supported on the same line as the data set name in the **devtab** for RFS filesets. The default attributes file for RFS filesets is specified in the **_IOE_RFS_ATTRIBUTES_FILE** environment variable specified in the **dfskern** process.

Note: The use of the translation control parameter overrides the **_IOE_HFS_TRANSLATION** environment variable and the **_IOE_RFS_TRANSLATION** environment variable settings for this specific fileset.

The logical volume defined is **/dev/ufs2**. This unique logical volume name should be used when specifying the device in the **dfstab** file (located in the **/opt/dcelocal/var/dfs** directory) for exporting to DFS.

Creating DCE Local File System Aggregates in DFS

In Distributed File Service, DCE Local File System aggregates are made up of one or more formatted linear data sets (LDS). The data sets for each aggregate are specified in the **/opt/dcelocal/var/dfs/devtab** file and make up a logical volume. The logical volume consists of one or more LDSs that have been formatted to create a single aggregate that will be exported by the **dfskern** process. To define a volume, edit the **/opt/dcelocal/var/dfs/devtab** file.

The following example shows a complete entry in the **devtab** file for a DCE Local File System logical volume. Lines preceded by an asterisk (*) are comments. The type of file system, **lfs**, and the minor device number 1 are defined. The minor device number can be any integer greater than zero. Each logical volume should have a unique minor device number. The volume's minor device number will become part of the aggregate's identifying device name. The minor device number need not be unique across both DCE Local File System and non-Local File System aggregates. Following the definition of the logical volume are the names of three linear data sets that make up the DCE Local File System aggregate:

```
*Devtab- Example Entry for a DCE LFS Logical LFS Volume
define_lfs 1
DFS.DCELFS.AGGR001.LDS00001
DFS.DCEFFS.AGGR001.LDS00002
DFS.DCELFS.AGGR001.LDS00003
```

The logical volume defined will be known as **/dev/lfs1**, with 1 being the device's minor number. This logical volume name should be used when you specify the device in DCE Local File System utilities such as **newaggr** and **growaggr** (for detailed information regarding these utilities, refer to the *z/OS Distributed File Service DFS Administration* book. The logical volume name must also be specified in the **dfstab** file when exporting the volume to **DFS** (see “Exporting DCE Local File System Data” on page 81).

z/OS Job Control Language (JCL) found in the SIOESAMP library member **NEWAGGR** can be used to both allocate LDSs and to create a DCE Local File System aggregate. The SIOESAMP JCL can be edited to reflect the correct linear data set names and logical volume. To access the **devtab** file, the program must run as **UID = 0** (**root**). After using IDC Access Method Services to allocate each Linear Data Set, the **NEWAGGR** JCL then formats the logical volume into a DCE Local File System aggregate.

Exporting File System Data Using DFS

This section discusses exporting non-Local File System data and exporting DCE Local File System data using DFS.

Exporting Non-Local File System Data

The HFS and RFS aggregates to be exported by the **dfskern** process must now be added to the **dfstab** file. This file is located in **/opt/dcelocal/var/dfs**. The **dfstab** file is used by the File Exporter, **fxd**, to determine the aggregates available for exporting. The **dfstab** file describes each aggregate and contains the aggregate device name, the aggregate name, the file system type, the aggregate id, and, for HFS and RFS aggregates, the fileset id.

Edit the **dfstab** file to add the appropriate control line describing the aggregate. For HFS aggregates, this requires an additional step. Since an HFS aggregate is a complete DFS fileset and not created by issuing the **fts** command, an entry must be made in the Fileset Location Database for the fileset. The **dfstab** entry for HFS aggregates requires that a fileset id be assigned when the Fileset Location Database entry is created. Issue the following **fts** command, **fts crfldbentry**, to create the Fileset Location Database entry for the HFS aggregate (for more information regarding this command, see the *z/OS Distributed File Service DFS Administration* book). The fileset name, *fileset*; server, *server*; and aggregate id, *aggrid*, are assigned by the DFS administrator.

```
$ fts crfldbentry -ftname fileset -server server -aggrid aggrid
```

Once the fileset entry is created, the fileset IDs assigned to the fileset's read-write, read-only, and backup filesets will be displayed. The ID for the read-write fileset must be entered in the **dfstab** file for exporting. The device name for an HFS or RFS aggregate is **/dev/ufs***n*, where *n* is the minor device number previously specified in the **devtab** file. The name is any unique name assigned by the administrator. This name **must** be unique. The export program will not mount any aggregate with a duplicate device name.

Edit the **dfstab** file. HFS or RFS entries in the **dfstab** file should always specify an aggregate type of **ufs**. The aggregate ID is a unique ID assigned by the DFS administrator. The read-write fileset ID from the Fileset Location Database entry is entered as the fileset ID. The following fields appear for each entry in the file, in the order listed. Each field must be separated by a minimum of one space or tab; each entry must be on a separate line.

Device Name
The device name of the partition, for example, **/dev/ufs2**.

Aggregate Name
The name to be associated with the exported partition. An aggregate name can contain any characters, but it can be no longer than 31 characters, and it must be different from any other aggregate name in the file. Aggregate names cannot be abbreviated, so you should choose a short, explicit name. For example, the aggregate name of a non-Local File System partition could be **hfs1**.

File System Type
The identifier for the file system type of the partition. For non-Local File System file systems, this must be **ufs**. It must be in lowercase letters.

Aggregate ID
A positive integer to serve as the aggregate ID of the exported partition. The integer must match the aggregate ID specified with the **-aggrid** option of the **fts crfldbentry** command, and it must be different from any other aggregate ID in the **dfstab** file. (If the ID is changed, fileset operations on the partition's fileset will fail.)

Fileset ID
The unique fileset ID number returned by the **fts crfldbentry** command for the fileset on the partition (for example, **0,,1715**). Use the read-write ID number (not the read-only or backup ID number) returned by the command as the value for this field.

The fileset ID is initially represented as 0,,*x*, where *x* is incremented by one until it exceeds 2^{32}. Once 2^{32}. is exceeded, the ID becomes 1,,1. When creating the root fileset, specify the z/OS server name and aggregate name (as specified in the **dfstab** file, see “Exporting Non-Local File System Data” and “Exporting DCE Local File System Data” on page 81) on your z/OS system where the fileset is to reside.

The following is an example of a **dfstab** entry for an HFS aggregate and an RFS aggregate:

```
device        name          type   id   fileset id
/dev/ufs2     hfs1          ufs    102  0,,1715
/dev/ufs3     rfs1          ufs    103  0,,1718
```

If DFS is not running, you can start DFS address spaces and processes (refer to the *z/OS Distributed File Service DFS Administration* book for a description of the DFS server address space). If DFS is already started, issue the following command from the operator's console to begin exporting HFS data:

```
modify dfs,start export
```

A message written by **dfsexport** specifies each aggregate that you have exported and that is attached to the file system. If there are problems in exporting an aggregate, **dfsexport** issues a message identifying the aggregate that failed to attach.

If you plan to export DCE Local File System data, continue to the next section, “Exporting DCE Local File System Data.”

For more information regarding exporting data, refer to the *z/OS Distributed File Service DFS Administration* book.

Exporting DCE Local File System Data

The DCE Local File System aggregates to be exported by the **dfskern** process must also be added to the **dfstab** file. This file is located in **/opt/dcelocal/var/dfs**. The **dfstab** file is used by the File Exporter, **fxd**, to determine the aggregates available for exporting. The **dfstab** file describes each aggregate and contains the aggregate device name, the aggregate name, the file system type, and the aggregate id.

Edit the **dfstab** file to add the appropriate control line describing the aggregate. DCE Local File System entries in the **dfstab** file should always specify an aggregate type of **lfs**. The minor device number, *n*, as specified in the **devtab** file should also be included (see “Creating DCE Local File System Aggregates in DFS” on page 79). The administrator assigns the aggregate name and aggregate ID. The following fields appear for each entry in the file, in the order listed. Each field must be separated by a minimum of one space or tab; each entry must be on a separate line.

Device Name
The device name of the partition, for example, **/dev/lfs1**.

Aggregate Name
The name to be associated with the exported aggregate. An aggregate name can contain any characters, but it can be no longer than 31 characters, and it must be different from any other aggregate name in the file. Aggregate names cannot be abbreviated, so you should choose a short, explicit name (for example, **lfs1**).

File System Type
The identifier for the file system type of the partition. For DCE Local File Systems, this must be **lfs**. It must be in lowercase letters.

Aggregate ID
A positive integer to serve as the aggregate ID of the exported aggregate. The integer must be different from any other aggregate ID in the **dfstab** file. (If the ID is changed, fileset operations on the partition's fileset will fail.)

The following is an example of a **dfstab** entry for a DCE Local File System aggregate:

```
device        name          type   id
/dev/lfs1     lfs1          lfs    6
```

If DFS is not running, you can start DFS address spaces and processes (refer to the *z/OS Distributed File Service DFS Administration* book for a description of the DFS server address space). If DFS is already started, issue the following command from the operator's console to begin exporting DCE Local File System data:

```
modify dfs,start export
```

For more information regarding exporting data in Distributed File Service, refer to the *z/OS Distributed File Service DFS Administration* book.

Setting Up the root.dfs Fileset

To make data available from the DFS file server you have just configured (see “Using DFSCONF to Configure the DFS File Server” on page 44), you must export data and create filesets from a File Server using either HFS, RFS, or DCE Local File System filesets. The first fileset you create should be **root.dfs**. The **root.dfs** fileset can be created on a z/OS or non-z/OS DCE Distributed File Service system. In addition, a server entry must be created for the z/OS DFS Server (this procedure is explained in “Creating the DFS File Server FLDB Entry” on page 49).

Note: DFS must be configured on a host system that has been configured into a DCE cell. If you have already created a **root.dfs** fileset for the DCE cell, skip this section.

The main read/write fileset, **root.dfs**, is required in every DCE cell's file system. It is the first fileset created during DFS configuration on a host system in the cell.

Important Note to Users

The **root.dfs** fileset is the implied fileset for the root of a cell's DFS filespace (**/.../cellname/fs**). It can be a fileset in a DCE Local File System aggregate or in a non-Local File System Aggregate. However, it must be a fileset in a DCE Local File System aggregate, if DFS functionality such as replication is to be available in the DCE cell. Also, on z/OS, the Record File System (RFS) is not supported for **root.dfs**.

Before proceeding to allocate **root.dfs** on a z/OS DCE host system, verify that the DFS (the DFS File Server) including the **ftserver** which is a **boserver** sub-process, has been configured and started on the system. Refer to “DFS Configuration Steps” on page 34, for more information on configuring DFS. Refer to Chapter 8, “Starting and Stopping DFS Components” on page 107, for information on starting and stopping DFS components.

Steps to Define a root.dfs Fileset

This section describes the detailed steps that must be performed to allocate, define, and export a DCE Local File System aggregate on z/OS. It also describes how to define a fileset in the DCE Local File System aggregate to be used as the **root.dfs** fileset.

Note: Defining a **root.dfs** fileset is a special case of creating and exporting a DCE Local File System aggregate for general use.

Refer to the *z/OS Distributed File Service DFS Administration* book for more information about defining filesets and the commands issued in the following instructions.

1. **Logon to z/OS and OMVS**

   Logon to z/OS and OMVS.

   Optionally, logon to z/OS and OMVS as **root** user with **UID = 0**.

   Note: Unless otherwise noted, commands described in this section should be issued from OMVS.

2. **dce_login as cell_admin**

   Login to DCE as a user with DCE cell administrator authorization by issuing the following command:

   ```
   dce_login cell_admin
      or
   dce_login userid
   ```

3. **Verify that there is a server entry for the z/OS DFS Server**

   The following command displays the server entry, in this example, **dcedfs**, from the Fileset Location Database (FLDB). The command is shown as entered from a z/OS UNIX session.

   ```
   $ fts lsserverentry -server dcedfs

   Description for site 'dcedfs':
   dcedfs.endicott.ibm.com (2:0.0.9.130.79.35)
   FLDB quota: 0;  uses: 35:  principal= hosts/DCEDFS; owner=<nil>
   ```

   If a server entry has not yet been created, see "Creating the DFS File Server FLDB Entry" on page 49.

4. **Create VSAM Data Sets for the DCE Local File System Aggregate for root.dfs**

   In order to have **root.dfs** fileset reside on z/OS, you need to allocate VSAM linear data set(s) for the logical volume that will be associated with the device name for the DCE Local File System aggregate that will contain the **root.dfs** fileset.

   To do this, you can make the necessary changes to the example control statements in the installation data set member SYS1.SIOESAMP(NEWAGGR) and submit the NEWAGGR job for execution.

   Note: The following **root.dfs** related examples assume that the data set **OMVS.PRV.DCELFS.AGGR.ROOT.DFS** was created. The data set comprises the logical volume that will be associated with the device name **/dev/lfs1** and the aggregate name **lfs1**.

5. **Create a devtab File for root.dfs**

   To export an aggregate, you need to create and/or update the **/opt/dfslocal/var/dfs/devtab** file. Use OMVS **OEDIT** to update the **devtab** file.

   The following is an example of the contents of a **devtab** file that defines the linear data set name that comprises the logical volume associated with the **root.dfs** fileset and the device name **/dev/lfs1/**.

   Note: The device name **/dev/lfs1/** will be assigned the DCE Local File System aggregate name **lfs1** in the **dfstab** entry (see below).

   ```
   * Logical Volume - DCE LFS (root.dfs) - Device Name /dev/lfs1
   define_lfs 1
   OMVS.PRV.DCELFS.AGGR.ROOT.DFS
   *
   ```

   Refer to the *z/OS Distributed File Service DFS Administration* book, for more information about the **devtab** file.

6. **Create a dfstab File for root.dfs**

   An **/opt/dfslocal/var/dfs/dfstab** file needs to be created and/or updated in order to export the aggregate that will contain the **root.dfs** fileset. Use OMVS **OEDIT** to update the **dfstab** file.

The example **dfstab** file entry shown below for the **root.dfs** fileset:

- Associates the device name **/dev/lfs1/** with the DCE Local File System aggregate name **lfs1**,
- Defines the aggregate type **lfs**,
- Defines the aggregate ID **1**, and
- Identifies that the aggregate should be exported when the file is read by the DFS File Server.

The example of a **dfstab** file entry for the **root.dfs** fileset is:

```
/dev/lfs1           lfs1    lfs  1
```

Refer to the *z/OS Distributed File Service DFS Administration* book, for more information about the **dfstab** file.

7. **Export the root.dfs Aggregate**

   To enable the aggregate and its filesets to be accessed by DFS clients and used for the **root.dfs** fileset, the aggregate needs to be exported.

   To export aggregates that have entries in the **devtab** and **dfstab**, issue the following command from the DFS host z/OS system operator's console:

   ```
   modify dfs,start export
   ```

   Refer to Chapter 8, “Starting and Stopping DFS Components” on page 107 or the *z/OS Distributed File Service DFS Administration* book, for more information about the **modify** command for DFS.

8. **Create the root.dfs Fileset**

   You now need to create and define the **root.dfs** fileset by issuing the following **fts create** command. The command will create the read-write fileset and create the FLDB entries for the fileset. These entries include a unique ID number assigned by the **flserver** for the read-write, read-only (for replication of DCE Local File System filesets), and backup filesets.

   The following shows the syntax for creating the **root.dfs** fileset. The **-server** *machine* is the name of the z/OS server where **root.dfs** resides and **-aggregate** *name* is the name of the aggregate (for more information regarding this command, see the *z/OS Distributed File Service DFS Administration* book).

   The following example assumes that the DCE host name for the z/OS system is **GDLCSST** and creates the fileset named **root.dfs** in the aggregate named **lfs1**. A unique aggregate name (**-aggregate** specification) must be assigned by the location and is also specified in the second field of the associated **dfstab** file entry.

   ```
   fts create -ftname root.dfs -server /.:/hosts/GDLCSST -aggregate lfs1
   ```

   An example of the output from issuing this command would be:

   ```
            readWrite   ID 0,,1  valid
            readOnly    ID 0,,2  invalid
            backup      ID 0,,3  invalid
    number of sites: 1
       server           flags     aggr   siteAge principal      owner

   gdlcsst.endicott.ib RW       lfs1    0:00:00 hosts/GDLCSST  <nil>
         Fileset 0,,40 created on aggregate lfs1 of GDLCSST

   Fileset 0,,1 created on aggregate lfs1 of GDLCSST
   ```

   Note: Because this is the **root.dfs** fileset, it will automatically be mounted as the directory **/:/**. This may take several minutes for this to take effect on your system.

9. **Create the Read/Write Mount Point for root.dfs**

   In order to replicate **root.dfs**, an explicit read/write mount point must be created before replicating **root.dfs**. Issue the following command to create a read/write mount point for the **root.dfs** fileset:

```
fts crmount .rw root.dfs -rw
```

10. **Set Replication Type for root.dfs**

    In order to replicate **root.dfs**, the replication type must be established. Issue the following command to accomplish this:

    ```
    fts setrepinfo -fileset root.dfs -release
    ```

    An example of the output from issuing this command would be:

    ```
    fts setrepinfo: Using default value for maxage of 2:00:00
    fts setrepinfo: Using derived value for failage of 1d0:00:00
    fts setrepinfo: Using default value for reclaimwait of 18:00:00
    ```

11. **Add Replication Site for root.dfs**

    You must add the replication site for **root.dfs** to the Fileset Location Database (FLDB). The following example can be issued for a z/OS host system with the DCE host name of **GDLCSST**:

    ```
    fts addsite -fileset root.dfs -server /.:/hosts/GDLCSST -aggr lfs1
    ```

    An example of the output from issuing this command would be:

    ```
    Added replication site GDLCSST lfs1 for fileset root.dfs
    ```

12. **Initiate Release Replication of root.dfs**

    Initiate replication of **root.dfs** by issuing the following command:

    ```
    fts release -fileset root.dfs
    ```

    An example of the output from issuing this command would be:

    ```
    Released fileset root.dfs successfully
    ```

Notes:

1. After completing the definition of the **root.dfs** fileset, if you are unable to **cd** to ***/.../****dcecellname****/*****fs/.rw** from the DFS client (**DFSCM**) system, issue the following commands from the DFS client:

   ```
   cm flush
   cm flushfile
   cm checkf
   ```

2. On z/OS, if the DFS Client (**DFSCM**) was configured and started, these commands can be issued from OMVS.

Creating Mount Points for Filesets

The **root.dfs** fileset must be defined in the DCE cell's filespace in order to define the mountpoints needed to export any other filesets from a DFS File Server. Refer to “Setting Up the root.dfs Fileset” on page 82 for more information.

A DFS mount point appears and functions like a regular directory but, structurally, it is a symbolic link which indicates the name of the fileset associated with a mount point. Mount points make the contents of a fileset visible and accessible to users in the DCE namespace.

To create mount points for filesets, use the following procedure:

1. Locate and change your directory to the junction to the DFS filespace, **root.dfs**. The junction is defined at **/.../***cellname***/fs**

   ```
   $ cd /.../abc.com/fs
   ```

2. Create directories for related filesets such as *user* for all user data sets or *src* for source files. The following example creates a directory called **user**, a user fileset, and a mount point for the fileset. If the fileset on which the aggregate resides has been exported to the file server, you can access the data in that fileset by changing to the directory of the newly created mount point.

   ```
   mkdir user
   cd user
   fts create user.abc.com lfs1 -verbose
   fts crmount -fileset user.abc.com -dir user.abc
   cd abc
   ```

3. If you plan to use the DCE Local File System replication feature for filesets, create a read-only version of the **root.dfs** fileset. This is necessary as the Cache Manager will traverse a path name to locate a file, beginning with **root.dfs**, and will access the read-only version of the fileset where possible. If *any* fileset in the path name does *not* have a read-only version, the Cache Manager will access the read-write version and will *never* access a read-only version for any filesets it traverses for the remainder of the path.

   Before replicating a **root.dfs** fileset, an explicit read-write mount point must first be created. If this is not done, it becomes impossible to access the read-write version of **root.dfs**. To anticipate the possible need for read-only filesets, create a read-write mount point for **root.dfs**. In the following example, **.rw** is used as the newly created read-write mount point. The **-rw** option specifies the type of mount point as read-write.

   ```
   $ fts crmount .rw root.dfs -rw
   ```

   Once you have replicated **root.dfs**, create additional mount points for other filesets by specifying the read-write path as follows: In the following, *name|ID* specifies the complete name or fileset number of the fileset to be mounted:

   ```
   $ fts crmount -fileset name|ID -dir /:/.rw/user/name|ID
   ```

For more information regarding mount points, refer to the *z/OS Distributed File Service DFS Administration* book.

Steps to Create and Export a DCE Local File System Aggregate

This section describes how to allocate, define, and export a DCE Local File System aggregate on z/OS. It also describes how to define a fileset in the DCE Local File System aggregate.

Refer to the *z/OS Distributed File Service DFS Administration* book, for more information on defining filesets.

1. **Logon to z/OS and OMVS**

   Logon to z/OS and OMVS.

   Optionally, logon to z/OS and OMVS as **root** user with **UID = 0**.

   Note: Unless otherwise noted, commands described in this section should be issued from OMVS.

2. **dce_login as cell_admin**

   Login to DCE as a user with DCE cell administrator authorization by issuing the following command:

   ```
   dce_login cell_admin
      or
   dce_login userid
   ```

3. **Create VSAM Data Sets for the DCE Local File System Aggregate for root.dfs**

To create a DCE Local File System aggregate on z/OS, you need to allocate VSAM linear data set(s) for the logical volume that will be associated with the device name for the aggregate.

To do this, you can make the necessary changes to the example control statements in the installation data set member SYS1.SIOESAMP(NEWAGGR) and submit the NEWAGGR job for execution.

Note: The following examples assume that the data sets OMVS.PRV.DCELFS.AGGR001.LDS00001 and OMVS.PRV.DCELFS.AGGR001.LDS00002 were created for the DCE Local File System aggregate.

4. **Update the devtab File for the DCE Local File System Aggregate**

   In order to export the aggregate, you need to update the **/opt/dfslocal/var/dfs/devtab** file. Use OMVS **OEDIT** to update the **devtab** file.

   Below is an example of the contents of a **devtab** file that defines the linear data set names that makes up the logical volume associated with the device name **/dev/lfs2/**.

   Note: The device name **/dev/lfs2/** will be assigned the DCE Local File System aggregate name **lfs2** in the **dfstab** (see below).

   The following is an example of the contents of a **dfstab** file that defines the logical volumes for two DCE Local File System aggregates with the device names **/dev/lfs1** and **/dev/lfs2**:

```
* Logical Volume - DCE LFS (root.dfs) - Device Name /dev/lfs1
define_lfs 1
OMVS.PRV.DCELFS.AGGR.ROOT.DFS
*
* Logical Volume - DCE LFS - Device Name /dev/lfs2
define_lfs 2
OMVS.PRV.DCELFS.AGGR001.LDS00001
OMVS.PRV.DCELFS.AGGR001.LDS00002
```

5. **Update the dfstab File for the DCE Local File System Aggregate**

   The **/opt/dfslocal/var/dfs/dfstab** file needs to be updated in order to export the aggregate. Use OMVS **OEDIT** to update the **dfstab** file.

   An example of the **dfstab** file entry is shown below:

   - Associates the device name **/dev/lfs2/** with the DCE Local File System aggregate name **lfs2**,
   - Defines the aggregate type **lfs**,
   - Defines the aggregate ID **2**, and
   - Identifies that the aggregate should be exported when the file is read by the DFS File Server.

   The following example **dfstab** file entries define two DCE Local File System aggregates (**lfs1** and **lfs2**):

```
/dev/lfs1          lfs1    lfs  1
/dev/lfs2          lfs2    lfs  2
```

6. **Export the DCE Local File System Aggregate**

   To enable DFS clients to access the filesets in the new DCE Local File System aggregate, the DFS File Server must export the aggregate.

   To export aggregates that have entries in the **devtab** and **dfstab**, issue the following command from the z/OS system operator's console:

```
modify dfs,start export
```

   Or, issue the following command on OMVS while logged on as a **root** user with **UID = 0**:

```
dfsexport lfs2
```

7. **Create a Fileset for the DCE Local File System Aggregate**

The following example command can be issued for a DFS host z/OS system with the DCE host name of **GDLCSST** to create the fileset named **episet1** in the new DCE Local File System aggregate named **lfs2**. The fileset name and aggregate must be unique in the location and is also used as the second field of the associated **dfstab** file entry.

```
fts create -ftname episet1 -server /.:/hosts/GDLCSST -aggregate lfs2 -verb
```

An example of the output from issuing this command would be:

```
------- Creating a new FLDB entry for episet1 -------
Created the FLDB entry for fileset episet1
        readWrite   ID 0,,4   valid
        readOnly    ID 0,,5   invalid
        backup      ID 0,,6   invalid
number of sites: 1
   server           flags     aggr   siteAge principal      owner

gdlcsst.endicott.ib RW       lfs1    0:00:00 hosts/GDLCSST  <nil>

Fileset 0,,4  created on aggregate lfs2 of GDLCSST
```

8. **Create a DFS Mount Point for the DCE Local File System Fileset**

   The following is an example of the commands that can be issued on a DFS host z/OS system to create a mount point for the **episet1** fileset:

   ```
   cd /:/.rw
   fts crmount -fileset episet1 -dir mvsepimtpt
   ```

9. **Set Replication Type for DCE Local File System Fileset**

   In order to replicate the **episet1** fileset defined for the new DCE Local File System aggregate, the following command can be issued:

   ```
   fts setrepinfo -fileset episet1 -scheduled -maxage 1h -failage 2h \
                  -reclaimwait 5h -minrepdelay 5m
   ```

   An example of the output from issuing this command would be:

   ```
   fts setrepinfo: Using given value for maxage of 1:00:00
   fts setrepinfo: Using given value for failage of 2:00:00
   fts setrepinfo: Using given value for reclaimwait of 5:00:00
   fts setrepinfo: Using given value for minrepdelay of 0:05:00
   ```

10. **Add Replication Site for DCE Local File System Fileset**

    Add the replication site for the fileset to the File Set Location Database (FLDB). The following command can be issued for a DFS host z/OS system with the DCE host name of **GDLCSST**:

    ```
    fts addsite -fileset episet1 -server /.:/hosts/GDLCSST \
                -aggr lfs2 -maxsiteage 30m
    ```

    An example of the output from issuing this command would be:

    ```
    Added replication site GDLCSST lfs2 for fileset episet1
    ```

11. **Initiate Schedule Replication for the DCE Local File System Fileset**

    Initiate replication of the fileset defined in the DCE Local File System aggregate. The following command can be issued for a DFS host z/OS system with the DCE host name of **GDLCSST**:

    ```
    fts update -fileset episet1 -server /.:/hosts/GDLCSST -verbose
    ```

    An example of the output from issuing this command would be:

```
Calling REP_UpdateSelf(gdlcsst.endicott.ibm.com, {268458673,,1888327310}
{0,,56}, ...)...
fts update: Repserver on gdlcsst.endicott.ibm.com requested to update
fileset 0,56
```

Steps to Create and Export an HFS Aggregate

This section describes how to allocate, define, and export an HFS data set as a DFS aggregate on z/OS.

Note: An HFS data set defined as a DFS aggregate contains only a single DFS fileset.

Refer to the *z/OS Distributed File Service DFS Administration* book, for more information about defining filesets and the commands issued in the following instructions.

1. **Logon to z/OS and OMVS**

   Logon to z/OS and OMVS.

   Optionally, logon to z/OS and OMVS as **root** user with **UID = 0**.

   Note: Unless otherwise noted, commands described in this section should be issued from OMVS.

2. **dce_login as cell_admin**

   Login to DCE as a user with DCE cell administrator authorization by issuing the following command:

   ```
   dce_login cell_admin
      or
   dce_login userid
   ```

3. **Create a Data Set**

   In order to export an HFS data set, you need to allocate a data set of type **HFS** that will be defined as a DFS aggregate.

   Or, if an HFS data set already exists, then you can export it.

   The examples in this section assume that the name of the HFS data set name used for the DFS aggregate is **OMVS.PRV.HFS.DATASET**.

4. **Update the devtab File for the HFS Aggregate**

   To export the HFS data set as a DFS aggregate, you need to update the **/opt/dfslocal/var/dfs/devtab** file. Use OMVS **OEDIT** to update the **devtab** file.

   The following is an example of the contents of a **devtab** file that defines two DCE Local File System logical volumes and the HFS logical volume **/dev/ufs010** that includes the data set **OMVS.PRV.HFS.DATASET**:

   ```
   * Logical Volume - DCE LFS (root.dfs) - Device Name /dev/lfs1
   define_lfs 1
   OMVS.PRV.EPISODE.AGGR.ROOT.DFS
   *
   * Logical Volume - DCE LFS - Device Name /dev/lfs2
   define_lfs 2
   OMVS.PRV.EPISODE.AGGR001.LDS00001
   OMVS.PRV.EPISODE.AGGR001.LDS00002
   *
   * Logical Volume - HFS  - Device Name /dev/ufs010
   define_ufs 10
   OMVS.PRV.HFS.DATASET
   ```

5. **Create the FLDB Entry for the HFS Aggregate**

Create the Fileset Location Database (FLDB) entry for the HFS aggregate by issuing an **fts crfldbentry** command. The **fts crfldbentry** command assigns the fileset name and creates the fileset and FLDB entry for the HFS aggregate. It also assigns a unique aggregate ID number that must be used as the fourth field of the associated **dfstab** file entry.

Following is an example of the **fts crfldbentry** command when DFS is on a z/OS system with a DCE host name of **GDLCSST**, the fileset name **OMVS.PRV.HFS.DATASET** is selected for the HFS aggregate, and the aggregate ID **10** is specified by the **-aggrid** value. The fileset name (**-ftname** specification) is assigned by the location but the use of the HFS data set name is recommended as shown in the example.

```
fts crfldbentry -ftname OMVS.PRV.HFS.DATASET \
                -server /.:/hosts/GDLCSST -aggrid 10
```

An example of the output from issuing this command would be:

```
        readWrite   ID 0,,22  valid
        readOnly    ID 0,,23  invalid
        backup      ID 0,,24  invalid
number of sites: 1
   server           flags     aggr   siteAge principal      owner

gdlcsst.endicott.ib RW        10     0:00:00 hosts/GDLCSST  <nil>

FLDB entry created for fileset hfsset1 (0,,22) on aggregate 10 of
     GDLCSST
```

Note: The ID number for the **readWrite** entry (**0,,22**) shown above is the number that is to be used in the **dfstab** file entry for the HFS fileset (see step 6, **Update the dfstab File for the HFS Aggregate**).

6. **Update the dfstab File for the HFS Aggregate**

   The **/opt/dfslocal/var/dfs/dfstab** file must be updated to export the fileset assigned to the HFS aggregate. Use OMVS **OEDIT** to update the **dfstab** file.

   The following is an example of the contents of the **dfstab** showing the HFS fileset entry with device name **/dev/ufs010**, aggregate name **ufs10**, and the fileset ID number **0,,22**:

```
/dev/lfs001        lfs1    lfs  1
/dev/lfs002        lfs2    lfs  2
/dev/ufs010        ufs10   ufs  10    0,,22
```

7. **Create an MVS Mount Point for the HFS Aggregate**

   If you want to access the HFS data set used for the DFS aggregate from OMVS, create a mount point for it and mount it in the local file system as you would for any other HFS data set.

8. **Export the HFS Aggregate**

   In order to allow the DFS client to access the HFS aggregate, the aggregate needs to be exported. In order to export aggregates that have entries in the **devtab** and **dfstab**, issue the following command from the operator's console:

```
modify dfs,start export
```

   Or, issue the following command from OMVS while logged on as a **root** user with **UID = 0**.

```
dfsexport ufs10
```

9. **Create a DFS Mount Point for the (HFS) Fileset**

   In order to create a DFS mount point for the fileset assigned to the HFS aggregate, an **fts crmount** command must be issued.

The following is an example of the commands that can be used to mount the HFS fileset named **OMVS.PRV.HFS.DATASET** at the mount point **/:/.rw/mvshfsmtpt**:

```
cd /:/.rw
fts crmount -fileset OMVS.PRV.HFS.DATASET -dir mvshfsmtpt
```

Steps to Create and Export an RFS Aggregate

This section describes how to define and export an RFS data set as a DFS aggregate on z/OS.

Note: An RFS data set defined as a DFS aggregate contains only a single DFS fileset.

Refer to the *z/OS Distributed File Service DFS Administration* book, for more information about defining filesets and the commands issued in the following instructions.

1. **Logon to z/OS and OMVS**

   Logon to z/OS and OMVS.

   Optionally, logon to z/OS and OMVS as **root** user with **UID = 0**.

   Note: Unless otherwise noted, commands described in this section should be issued from OMVS.

2. **dce_login as cell_admin**

   Login to DCE as a user with DCE cell administrator authorization by issuing the following command:

   ```
   dce_login cell_admin
      or
   dce_login userid
   ```

3. **Update the devtab File for the RFS Aggregate**

   To export the HFS data set as a DFS aggregate, you need to update the **/opt/dfslocal/var/dfs/devtab** file. Use OMVS **OEDIT** to update the **devtab** file.

   The following is an example of the contents of a **devtab** file that defines two DCE Local File System logical volumes, an HFS logical volume and the RFS logical volume **/dev/ufs020** that includes the data set **OMVS.PRV.RFS.DATASET**:

   ```
   * Logical Volume - DCE LFS (root.dfs) - Device Name /dev/lfs1
   define_lfs 1
   OMVS.PRV.EPISODE.AGGR.ROOT.DFS
   *
   * Logical Volume - DCE LFS - Device Name /dev/lfs2
   define_lfs 2
   OMVS.PRV.EPISODE.AGGR001.LDS00001
   OMVS.PRV.EPISODE.AGGR001.LDS00002
   *
   * Logical Volume - HFS  - Device Name /dev/ufs010
   define_ufs 10
   OMVS.PRV.HFS.DATASET
   *
   * Logical Volume - RFS  - Device Name /dev/ufs020
   define_ufs 20 rfs
   OMVS.PRV.RFS.DATASET
   ```

4. **Create the FLDB Entry for the RFS Aggregate**

   Create the Fileset Location Database (FLDB) entry for the RFS aggregate by issuing an **fts crfldbentry** command. The **fts crfldbentry** command assigns the fileset name and creates the fileset and FLDB entry for the HFS aggregate. It also assigns a unique aggregate ID value that must be used as the fourth field in the associated **dfstab** file entry.

Following is an example of the **fts crfldbentry** command when DFS is on a z/OS system with a DCE host name of **GDLCSST**, the fileset name **-ftname OMVS.PRV.RFS.DATASET** is selected for the RFS aggregate, and the aggregate ID **-aggrid 20** is assigned. The fileset name (**-ftname** specification) is assigned by the location but the use of the HFS data set name is recommended as shown in the example.

```
fts crfldbentry -ftname OMVS.PRV.RFS.DATASET \
                -server /.:/hosts/GDLCSST -aggrid 20
```

An example of the output from issuing this command would be:

```
        readWrite   ID 0,,19  valid
        readOnly    ID 0,,20  invalid
        backup      ID 0,,21  invalid
number of sites: 1
   server           flags     aggr   siteAge principal      owner

gdlcsst.endicott.ib RW       20      0:00:00 hosts/GDLCSST  <nil>

FLDB entry created for fileset hfsset1 (0,,19) on aggregate 20 of
     GDLCSST
```

Note: RFS aggregate and returns a fileset unique ID number to be used in the **dfstab** file. The ID number for the **readWrite** entry (**0,,19**) shown above is the number that is to be used in the **dfstab** file entry for the HFS fileset (see step 5, **Update the dfstab File for the RFS Aggregate**).

5. **Update the dfstab File for the RFS Aggregate**

   The **/opt/dfslocal/var/dfs/dfstab** file must be updated to export the fileset assigned to the RFS aggregate. Use OMVS **OEDIT** to update the **dfstab** file.

   The following is an example of the contents of the **dfstab** showing the RFS fileset entry with device name **/dev/ufs020**, aggregate name **ufs20**, and the fileset ID number **0,,19**:

```
/dev/lfs1          lfs1    lfs  1
/dev/lfs2          lfs2    lfs  2
/dev/ufs010        ufs10   ufs  10    0,,22
/dev/ufs020        ufs20   ufs  20    0,,19
```

6. **Export the RFS Aggregate**

   In order to allow DFS client to access the HFS aggregate, the aggregate needs to be exported. In order to export aggregates that have entries in the **devtab** and **dfstab**, issue the following command from the operator's console:

```
modify dfs,start export
```

   Or, issue the following command from OMVS while logged on as a **root** user with **UID = 0**.

```
dfsexport ufs10
```

7. **Create a DFS Mount Point for the RFS Fileset**

   In order to create a DFS mount point for the fileset assigned to the RFS aggregate, an **fts crmount** command must be issued.

   The following is an example of the commands that can be used to mount the HFS fileset named **OMVS.PRV.HFS.DATASET** at the mount point **/:/.rw/mvshfsmtpt**:

```
cd /:/.rw
fts crmount -fileset OMVS.PRV.HFS.DATASET -dir mvshfsmtpt
```

Chapter 7. Deconfiguring DFS

Important Note to Users

Before deconfiguring any Distributed File Service Client or Server processes on a z/OS DCE host system, be sure to stop the DFS processes on that system (see Chapter 8, “Starting and Stopping DFS Components” on page 107).

Certain situations require that you deconfigure or remove configuration files for all of DFS or for a particular DFS component from a z/OS DCE host system where DFS is configured.

Selected DFS components may require deconfiguration. If you want to reconfigure a particular component with new parameters, you must deconfigure it, removing the existing configuration, before setting up the new configuration. Or, for example, if a configuration of a component failed and it is only partially configured, you must remove the partial configuration before attempting configuration again.

Situations may require that you deconfigure all DFS components from the server. For example, if you want to transfer a server from one cell to another, you must remove the configurations for the old cell from the server before setting up the configurations for the new cell.

You may want to deconfigure an entire DCE cell. If you choose to do this, you should also unregister the DCE cell's name from the global namespace. Before deconfiguring DCE, you should always deconfigure the DFS Client and Servers.

Exercise caution in deconfiguring DFS components, especially if you are removing components which perform services required by other components. Deconfiguring a component partially or completely disables other components which are dependent upon it. For example, deconfiguring all Fileset Location Database servers, the basis of any cell's DFS system, disables the DFS functions for the cell.

To deconfigure you must set up a TSO user ID (see “Set Up the DFSCONF Administrator TSO User ID” on page 41) and make sure you are logged in to DCE (see “DCE Login Panel” on page 41) as a user with DCE cell administrator (**cell_admin**) authority.

DFS should be deconfigured prior to deconfiguring DCE on the host system. After DFS is deconfigured, you can deconfigure DCE. After DCE is reconfigured, you can then configure DFS on the host system following the steps described in Chapter 4, “Configuring DFS for DCE” on page 33. Special actions are required if DFS is configured on a host system and DCE is deconfigured and configured on the same host system with a different DCE host name (see “DFS Deconfiguration Steps for DCE Host Name Change” on page 104).

Note: Before any DFS deconfiguration is done, you should review the “DCE Deconfiguration Considerations for DFS” on page 94 and “Update (upserver) Deconfiguration Considerations” on page 94.

The actions associated with deconfiguring a DFS component are described in this chapter. The chapter includes:

- “DCE Deconfiguration Considerations for DFS” on page 94
- “Update (upserver) Deconfiguration Considerations” on page 94
- “DFS Component Deconfiguration” on page 95
- “Deconfigure Backup Database Server (bakserver)” on page 96
- “Deconfigure the Fileset Location Database Server (flserver)” on page 97
- “Deconfigure the Fileset Server (ftserver)” on page 99
- “Deconfigure the DFS File Server” on page 99
- “Using DFSCONF to Deconfigure DFS Servers” on page 100

- "Using DFSCONF to Deconfigure the DFS Client" on page 102
- "Removing DFS Servers from a Host System" on page 103
- "Remove the DFS Client from a Host System" on page 103
- "DFS Deconfiguration Steps for DCE Host Name Change" on page 104.

Note: It is recommended that you follow the deconfiguration steps outlined in "DFS Component Deconfiguration" on page 95.

DCE Deconfiguration Considerations for DFS

The following considerations apply to deconfiguring DFS on a z/OS system:

1. If you need to reconfigure (deconfigure and configure) DCE on a z/OS system, and DFS is also configured on the z/OS system, you need to reconfigure both DFS and DCE. It is recommended that the reconfiguration steps be done in the following order:
   - Deconfigure DFS
   - Deconfigure DCE
   - Configure DCE
   - Configure DFS.
2. If DCE is reconfigured while DFS remains configured on a z/OS DCE host system, it is recommended that DFS be reconfigured on that system.
3. It is recommended that you do not change the DCE host name when reconfiguring the DCE client if DFS is configured on a z/OS system. Special DFS reconfiguration steps are required if the DCE host name is changed (see "DFS Deconfiguration Steps for DCE Host Name Change" on page 104).
4. After DFS is configured on a DCE host system, you can not reconfigure all the DCE Security Servers or all DCE Cell Directory Servers in a DCE cell without destroying the DFS authorization. The content of all the DFS administrative list file and ACL data for DCE Local File System files is invalidated. File data may not be accessible through DFS even if DFS is reconfigured. You should copy DCE Local File System data to another file system before you deconfigure the DCE Security Servers in a DCE cell.

   If all the DCE Security Servers and DCE Cell Directory Servers reconfigure, you can attempt to re-establish DFS operations by:

   a. Deconfiguring the DFS servers using DFSCONF if this was not done prior to deconfiguring DCE.

   b. Configuring the DFS servers using DFSCONF.

   c. Deleting the DFS administrative list files and re-creating the them as described in Chapter 5, "Defining DFS Administrators and Users" on page 69. All the required DFS administrator principals and groups must also be added to the the DFS administrative list files.

   d. Redefining all DFS file ACL data as described in the *z/OS Distributed File Service DFS Administration* book.

Update (upserver) Deconfiguration Considerations

The **upserver** deconfiguration considerations in this section only apply if you have an **upserver** running on the z/OS host system. Before deconfiguring all DFS servers or just the DFS **upserver** from a z/OS DCE host system, you may need to configure the **upserver** on another DCE host system in the DCE cell if DFS runs on more than system in the DCE cell.

You can ignore these **upserver** related instructions if:

- DFS is configured on a single DCE host system in the DCE cell.
- There is no **upserver** running in the DCE cell.
- The **upserver** is running on another DCE host system in the DCE cell.
- If the **upserver** remains configured on this z/OS system after other components of DFS are deconfigured.

If the DFS **upserver** is running on the z/OS system where DFS including the **upserver** is being deconfigured, the following steps must be taken before any further DFS deconfiguration steps are performed:

1. Stop the **upserver** on the z/OS host system.
2. Stop the Update Clients on the other DFS host systems in the DCE cell.
3. Configure and start an **upserver** on another DCE host system in the DCE cell.
4. Start an **upclient** on the other DCE host systems in the DCE cell (on a system other than where the **upserver** is now running), identifying the new **upserver** when an **upclient** is started.

DFS Component Deconfiguration

This section identifies the DFS components that can be deconfigured on a z/OS DCE host system and the sequence in which DFS component deconfiguration can be performed.

Deconfigure the DFS Client (DFSCM)

The DFS Client (**DFSCM**) can be deconfigured independent from DFS servers.

To deconfigure the **DFSCM** on the z/OS system with DCE host name *dcehostname*, the following steps should be performed:

1. Use **DFSCONF** to deconfigure the **DFSCM** (see “Using DFSCONF to Deconfigure DFS Servers” on page 100).
2. If you plan to remove the **DFSCM** from the DCE host system, you may want to perform the additional steps described in “Remove the DFS Client from a Host System” on page 103.

Deconfigure DFS Servers

All DFS servers are dependent on the configuration of the DFS File Server on the DCE host system. The DFS File Server should be deconfigured last.

The Backup Database Server (**bakserver**) including the Backup Tape Coordinator (**BUTC**) Servers, the Fileset Location Database Server (**flserver**), and the Fileset Server (**ftserver**) can be deconfigured independently and only depend on the configuration of the DFS File Server on the DCE host system.

It is recommended that DFS servers be deconfigured as described in the sections referenced in the following list:

1. Deconfigure the Backup Database Server (**bakserver**) including the Backup Tape Coordinator (**BUTC**) Servers (see “Deconfigure Backup Database Server (bakserver)” on page 96).
2. Deconfigure the Fileset Location Database Server (**flserver**) (see “Deconfigure the Fileset Location Database Server (flserver)” on page 97).
3. Deconfigure the Fileset Server (**ftserver**) (see “Deconfigure the Fileset Server (ftserver)” on page 99).
4. Deconfigure the DFS File Server (see “Deconfigure the DFS File Server” on page 99).

5. If you plan to remove DFS servers from the DCE host system, you may want to perform the additional steps described in “Removing DFS Servers from a Host System” on page 103.

After DFS is deconfigured using **DFSCONF**, any DFS administrative list files and the **BosConfig** file still exist in the directory **/opt/dcelocal/var/dfs**. If you reconfigure DFS, these files are re-used by DFS unless you rename, move, or delete them.

Deconfigure Backup Database Server (bakserver)

To deconfigure the Backup Database Server (**bakserver**) and any Backup Tape Coordinator (**BUTC**) servers, the following steps should be performed:

1. Insure that the **boserver** is running on the host system.
2. Insure that the **bakserver** is running on the host system or in the DCE cell.
3. If any of the **butc01** through **butc08** servers are running the host system, issue the following z/OS operator command:

   ```
   modify dfs,stop butcnn
   ```

   where: *nn* equals **01** through **08**.
4. Login as a user ID with DCE **cell_admin** authority from OMVS.
5. Issue the following command for each **BUTC**nn server configured on the DCE host system:

   ```
   bak rmhost -tcid n
   ```

   where *n* = the value 0-7 that identifies the **BUTC** process. Note that the **BUTC** process identifier values **0** through **7** relate to the **BUTC01** through **BUTC08** servers.

   See “Completing the DFS Backup Tape Coordinator Configuration” on page 55 for more information on identifying **BUTC** servers.

   For additional detailed information about the **bak rmhost** command, refer to the *z/OS Distributed File Service DFS Administration* book.
6. Optionally update the DFS Backup Tape Management file. For more information, refer to “Completing the DFS Backup Tape Coordinator Configuration” on page 55, step 6 on page 57.
7. Optionally update the **BUTC**nn **envar** files to reflect the IBM supplied values. For more information, refer to “Completing the DFS Backup Tape Coordinator Configuration” on page 55, step 5 on page 56.
8. If the **bakserver** is running on the host system, issue the following command from OMVS:

   ```
   bos stop /.:/hosts/dcehostname bakserver
   ```
9. Issue the following command:

   ```
   bos delete /.:/hosts/dcehostname bakserver
   ```

   For additional detailed information about the **bos delete** command, refer to the *z/OS Distributed File Service DFS Administration* book.
10. Use **DFSCONF** to deconfigure the Backup Database Server (**bakserver**) on the z/OS DCE host system (see “Using DFSCONF to Deconfigure DFS Servers” on page 100).
11. Complete the **bakserver** deconfiguration (see “Completing the Backup Database Server Deconfiguration” on page 97).

Completing the Backup Database Server Deconfiguration

The section describes additional deconfiguration steps that must be performed after using **DFSCONF** to deconfigure the Backup Database Server (**bkserver** or **bakserver**) (see “Using DFSCONF to Deconfigure DFS Servers” on page 100).

The additional steps that are required to complete the deconfiguration of the (**bkserver**) are:

1. Use the **dcecp -c rpcentry show /.:/subsys/dce/dfs/bak** command to update the entry for the RPC server group **/.:/subsys/dce/dfs/bak**. The command forces CDS to update information that it caches from the entry for the group in the namespace.
2. Stop and restart the Backup Database Server process (**bakserver**) on each database server of that type. Restarting the existing database server processes causes the processes to read the updated RPC server group. This ensures that each Ubik coordinator agrees on the number and identities of the other database servers of its type, which is vital to Ubik's use of a quorum of database servers to maintain database consistency.
3. Remove the DFS server principal for the **admin.bak** list.

   If there is only a single DCE host system in the DCE cell where DFS has been configured or if there is no DFS **upserver** running on another DCE host system in the DCE cell, you can skip this step. Refer to “Update (upserver) Deconfiguration Considerations” on page 94 for more information.

   If there is an **upserver** running on another DCE host system in the DCE cell, use the **bos rmadmin** command to cause the DCE cell System Control machine where the **upserver** is running to remove the abbreviated DFS server principal **hosts/***dcehostname***/dfs-server** for this DCE host system from it's **admin.bak** administrative list. The **upserver** then distributes the updated list to the Update Clients in the DCE cell.

   If you previously chose to add the full or abbreviated DFS server principal to a security group and included the group in the **admin.bak** list, you can use the **dcecp group remove** command to remove the DFS server principal from the group. Note that **DFSCONF** has already removed the abbreviated DFS server principal **hosts/***dcehostname***/dfs-server** from the security group **subsys/dce/dfs-bak-servers**.
4. Remove the **admin.bak** administrative list from the **/opt/dcelocal/var/dfs** directory on the database server to be removed. Modify the Update Server as necessary if the list is propagated from the cell's System Control machine.

For additional detailed information, refer to the *z/OS Distributed File Service DFS Administration* book.

After performing the actions described in this section, you should proceed to the next step described in “DFS Component Deconfiguration” on page 95.

Deconfigure the Fileset Location Database Server (flserver)

To deconfigure the Fileset Location Database Server (**flserver**) on the z/OS system with DCE host name *dcehostname*, the following steps should be performed:

1. Insure that the **boserver** is running on the host system.
2. Login as a user ID with DCE **cell_admin** authority from OMVS.
3. If the **flserver** is running on the host system, issue the following command from OMVS:

   ```
   bos stop /.:/hosts/dcehostname flserver
   ```
4. Issue the following command:

   ```
   bos delete /.:/hosts/dcehostname flserver
   ```

For additional detailed information about the **bos delete** command, refer to the *z/OS Distributed File Service DFS Administration* book.

Note: After issuing the **bos delete** command for the **flserver**, if another **flserver** does not exist in the DCE cell, the **fts** commands shown for deconfiguring the DFS File Server (see "Deconfigure the DFS File Server" on page 99) can not be issued.

5. Use **DFSCONF** to deconfigure the Fileset Location Database Server (**flserver**) on the z/OS DCE host system (see "Using DFSCONF to Deconfigure DFS Servers" on page 100).
6. Complete the **flserver** deconfiguration (see "Completing the Fileset Location Database Server Deconfiguration").

Completing the Fileset Location Database Server Deconfiguration

The section describes additional deconfiguration steps that must be performed after using **DFSCONF** to deconfigure the Fileset Location Database Server (**flserver**), (See "Using DFSCONF to Deconfigure DFS Servers" on page 100 for more information.)

The additional steps that are required to complete the deconfiguration of the **flserver** are:

1. Use the **dcecp rpcentry show** command on the Fileset Database Server to update the entry for the appropriate RPC server group from CDS. The command forces CDS to update information that it caches from the entry for the group in the namespace.
2. Stop and restart the Fileset Database Server process (**flserver**) on each database server of that type. Restarting the existing database server processes causes the processes to read the updated RPC server group. This ensures that each Ubik coordinator agrees on the number and identities of the other database servers of its type, which is vital to Ubik's use of a quorum of database servers to maintain database consistency.
3. Remove DFS server principal for the **admin.fl** list

   If there is only a single DCE host system in the DCE cell where DFS has been configured or if there is no DFS **upserver** running on another DCE host system in the DCE cell, you can skip this step. Refer to "Update (upserver) Deconfiguration Considerations" on page 94 for more information.

   If there is an **upserver** running on another DCE host system in the DCE cell, use the **bos rmadmin** command to cause the DCE cell System Control machine where the **upserver** is running to remove the abbreviated DFS server principal **hosts/***dcehostname***/dfs-server** for this DCE host system from it's **admin.fl** administrative list. The **upserver** then distributes the updated list to the Update Clients in the DCE cell.

   If you previously chose to add the full or abbreviated DFS server principal to a security group and included the group in the **admin.fl** list, you can use the **dcecp group remove** command to remove the DFS server principal from the group. Note that **DFSCONF** has already removed the abbreviate DFS server principal **hosts/***dcehostname***/dfs-server** from the security group **subsys/dce/dfs-fs-servers**.
4. Remove the **admin.fl** administrative list from the **/opt/dcelocal/var/dfs** directory on this DCE host system. Modify the Update Server as necessary if the list is propagated from the cell's System Control machine.

For additional detailed information, refer to the *z/OS Distributed File Service DFS Administration* book.

After performing the actions described in this section, you should proceed to the next step described in "DFS Component Deconfiguration" on page 95.

Deconfigure the Fileset Server (ftserver)

To deconfigure the DFS Fileset Server (**ftserver**), the steps shown in this section should be performed.

Note: You can skip the steps that require an **fts** command to be issued if the Fileset Location Database Server (**flserver**) has already been deconfigured on this host system and there is no other **flserver** running in the DCE cell. Refer to "Deconfigure the Fileset Location Database Server (flserver)" on page 97 for more information.

The steps to deconfigure the **ftserver** on a z/OS system with DCE host name *dcehostname* are:

1. Insure that the **boserver** is running on the host system.
2. Insure that the **ftserver** is running the host system.
3. Login as a user ID with DCE **cell_admin** authority from OMVS.
4. List fileset entries from the Fileset Location Data Base (FLDB) for the host using the following command:

   ```
   fts lsfldb -server /.:/hosts/dcehostname
   ```
5. Delete each fileset entry from the FLDB for *dcehostname* using the following command:

   ```
   fts delfldbentry -fileset abc.fileset -server {ipaddress | /.:/hosts/dcehostname}
   ```
6. Delete the server entry from the FLDB for *dcehostname* using the following command:

   ```
   fts delserverentry -server {ipaddress | /.:/hosts/dcehostname}
   ```
7. Delete the DFS customizable files described in Table 16 on page 126.
8. If the **ftserver** is running on the host system, issue the following command from OMVS:

   ```
   bos stop /.:/hosts/dcehostname ftserver
   ```
9. Issue the following command:

   ```
   bos delete /.:/hosts/dcehostname ftserver
   ```

For additional detailed information about the **bos** and **fts** commands, refer to the *z/OS Distributed File Service DFS Administration* book.

After performing the actions described in this section, you should proceed to the next step described in "DFS Component Deconfiguration" on page 95.

Deconfigure the DFS File Server

To deconfigure the DFS File Server the steps shown in this section should be performed.

Note: You can skip the steps that require an **fts** command to be issued if the Fileset Location Database Server (**flserver**) has already been deconfigured on this host system and there is no other **flserver** running in the DCE cell. Refer to "Deconfigure the Fileset Location Database Server (flserver)" on page 97 for more information.

The steps to deconfigure the DFS File Server on a z/OS system with DCE host name *dcehostname* are:

1. Login as a user ID with DCE **cell_admin** authority from OMVS.
2. Verify that the DFS server entry for DCE host name *dcehostname* still exists by issuing the command:

   ```
   $ fts lsserverentry -all
   ```
3. If it exists, delete the server entry from the FLDB for *dcehostname* using the following command:

```
$ fts delserverentry -server {ipaddress | /.:/hosts/dcehostname}
```

4. Use **DFSCONF** to deconfigure the DFS File Server (see “Using DFSCONF to Deconfigure DFS Servers”).

After performing the actions described in this section, you should proceed to the next step described in “DFS Component Deconfiguration” on page 95.

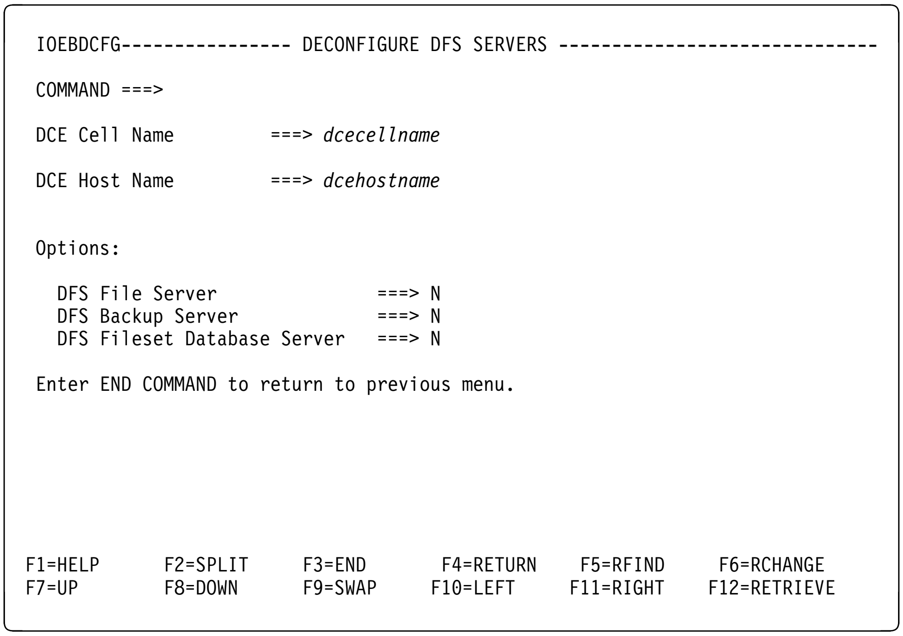
Using DFSCONF to Deconfigure DFS Servers

Note: Refer to “Using the DFS Configuration Program (DFSCONF)” on page 39 for information on how to use **DFSCONF** to DCE login as a cell administrator and select the option to **Deconfigure DFS Servers** from the DFSCONF Main Menu.

Using the **DFSCONF** panels to deconfigure your system automates some of the deconfiguration process. It is recommended that you use the **DFSCONF** panels to deconfigure DFS on your z/OS DCE host system system.

If you plan to completely deconfigure your system, it is recommended that the Backup Tape Coordinator, Backup Database server, and Fileset Database server be deconfigured before the DFS File Server is deconfigured. The DFS Client can be deconfigured before or after any of the DFS servers are deconfigured.

Selecting Deconfigure DFS Servers from the DFSCONF Main Menu displays the panel shown in Figure 5.



```
 IOEBDCFG---------------- DECONFIGURE DFS SERVERS ------------------------------

 COMMAND ===>

 DCE Cell Name         ===> dcecellname

 DCE Host Name         ===> dcehostname


 Options:

   DFS File Server              ===> N
   DFS Backup Server            ===> N
   DFS Fileset Database Server   ===> N

 Enter END COMMAND to return to previous menu.







F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 5. Deconfiguring DFS Servers Panel

The menu items are:

Menu Item	**Description**
DCE Cell Name	This is the DCE cell name used to previously configure DCE on this z/OS system and is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**. It normally should not be changed on this panel and is displayed for information purposes only.
DCE Host Name	This is the DCE host name currently defined by the DCE configuration on this z/OS system and is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**. It normally should not be changed on this panel but if you want to deconfigure the DFS servers using a different DCE host name you can change it. The DCE host name value is case-sensitive.
DFS File Server	To deconfigure the DFS File Server, change the option from **N** to **Y**. To initiate the deconfiguration of this server, press <**Enter**> after selecting other options.
Backup Server	To deconfigure the Backup Database Server (**bakserver**), and Backup Tape Coordinator (**butc**) servers, change the option from **N** to **Y**. To initiate the deconfiguration of this server, press <**Enter**> after selecting other options.
DFS Fileset Location Database Server	To deconfigure the Fileset Location Database server (**flserver**), change the option from **N** to **Y**. To initiate the deconfiguration of this server, press <**Enter**> after selecting other options.

After deconfiguration processing is initiated (by pressing <**Enter**>) for the selected servers, the servers are deconfigured in order, starting with the bottom option working up the list.

The program begins issuing the DCE administration commands that deconfigure the server (see Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117).

If not interrupted, and if successfully completed, the program displays a server Deconfiguration Ended....SUCCESSFULLY message for each selected server.

If interrupted, or if deconfiguration was not successful for a server, a server Deconfiguration Ended....UNSUCCESSFULLY message is displayed for the server.

To determine where the error occurred that caused the deconfiguration to fail, examine the **dfsconf.log** file, which by default is in your home directory, or at the location specified by the environment variable **_IOE_CFG_LOG_FILE**. Refer to Appendix A, “Example DFS Server Configuration Log File (dfsconf.log)” on page 117 for an example output of a **dfsconf.log** file. If the **dfsconf.log** file already exists, **DFSCONF** appends messages to the end of the log file.

When **DFSCONF** processing is complete, if you want to now (re-)configure the DFS server or servers on the DCE host system, refer to “DFS Configuration Main Menu” on page 42.

If you intend to remove the DFS Fileset Location Database Server (**flserver**) from this DCE host system, refer to “Completing the Fileset Location Database Server Deconfiguration” on page 98.

If you intend to remove the DFS Backup Database Server (**bkserver** or **bakserver**) from this DCE host system, refer to “Completing the Backup Database Server Deconfiguration” on page 97.

If you want to remove all DFS servers from the DCE host system, after completing the instructions in “Completing the Fileset Location Database Server Deconfiguration” on page 98 and “Completing the Backup Database Server Deconfiguration” on page 97, you should refer to “Removing DFS Servers from a Host System” on page 103.

After following the previous instructions in this section, DFS server deconfiguration is complete.

Using DFSCONF to Deconfigure the DFS Client

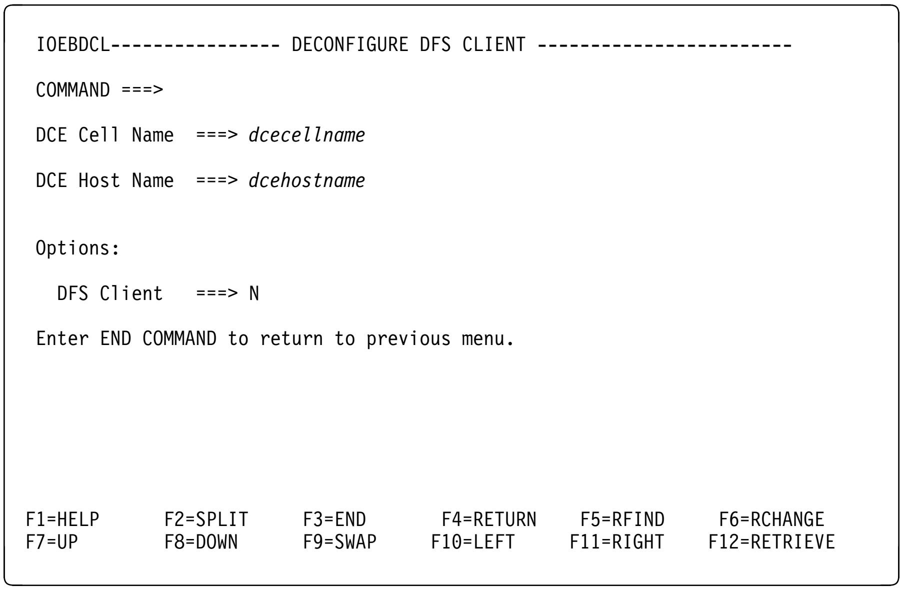
Note: Refer to “Using the DFS Configuration Program (DFSCONF)” on page 39 for information on how to use **DFSCONF** to DCE login as a cell administrator and select the option to **Deconfigure DFS Client** from the DFSCONF Main Menu.

The DFS client (**DFSCM**) should be deconfigured to remove it from a DCE host system.

Note: If you intend to simply change the form of **DFSCM** caching (e.g from memory caching to disk caching), there is no need to deconfigure the **DFSCM** before changing the disk configuration values. See “DFSCM Disk Caching” on page 61 for more information.

As the **DFSCM** does not require any DFS servers to be running on the same z/OS system, the deconfiguration process for **DFSCM** is handled separately from the DFS servers.

Selecting DFS Client from the DFSCONF Main Menu displays the panel shown in Figure 6.



```
 IOEBDCL---------------- DECONFIGURE DFS CLIENT ------------------------

 COMMAND ===>

 DCE Cell Name  ===> dcecellname

 DCE Host Name  ===> dcehostname


 Options:

   DFS Client   ===> N

 Enter END COMMAND to return to previous menu.






F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6. Deconfiguring the DFS Client (DFSCM) Panel

The menu items are:

Menu Item **Description**

DCE Cell Name

This is the DCE cell name used to previously configure DCE on this z/OS system and is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**. It normally should not be changed on this panel and is displayed for information purposes only.

DCE Host Name	This is the DCE host name currently defined by the DCE configuration on this z/OS system and is obtained from the DCE configuration file **/opt/dcelocal/dce_cf.db**. It normally should not be changed on this panel but if you want to deconfigure the DFS client using a different DCE host name you can change it. The DCE host name value is case-sensitive.
DFS Client	To initiate the deconfiguration of the **DFSCM**, change the DFS Client option from **N** to **Y** and press <**Enter**>.

Note: After the **DFSCM** is deconfigured as described above, you should not restart the DFS client as described in “Starting and Stopping the DFS Client (DFSCM)” on page 107 before the **DFSCM** is (re)configured as described in Chapter 4, “Configuring DFS for DCE” on page 33.

When DFSCONF processing is complete, if you want to now (re-)configure the **DFSCM** on the DCE host system, refer to “DFS Configuration Main Menu” on page 42.

Or, if you intend to remove the **DFSCM**, from your system, refer to “Remove the DFS Client from a Host System.”

Removing DFS Servers from a Host System

To remove DFS servers from a z/OS DCE host system, you should:

1. Deconfigure all the DFS servers as outlined in “DFS Component Deconfiguration” on page 95
2. Insure all the DFS customizable files described in Table 16 on page 126 are deleted
3. Delete any DCE LFS filesets allocated on the z/OS host system for exporting to DFS clients.

Remove the DFS Client from a Host System

After deconfiguring DFS Client (**DFSCM**) as described in “Deconfigure the DFS Client (DFSCM)” on page 95, you can perform the following actions to remove the **DFSCM** from a host system:

If you intend to remove the **DFSCM** from your system, perform the following actions:

1. Remove the **BPXPRM***xx* **FILESYSTYPE TYPE(DFSC)** entry in the data set that is allocated to the IEFPARM DD statement in the z/OS system start-up procedure. Once removed, z/OS does not attempt to start the DFS client during the next z/OS initialization (IPL).
2. Delete the **/opt/dcelocal/etc/CacheInfo** file.
3. Delete the **/opt/dfslocal/home/dfscm/envar** file.
4. If a DCE Local File System aggregate was allocated during the initial configuration for DFS client disk caching, de-allocate the aggregate or re-use the aggregate for other purposes as needed.

Note: After the **DFSCM** is fully deconfigured and removed from your system, you should not restart the DFS client as described in “Starting and Stopping the DFS Client (DFSCM)” on page 107 before z/OS is re-IPL'ed.

DFS Deconfiguration Steps for DCE Host Name Change

To change a DCE host name for a z/OS system, DCE must be reconfigured. If DFS is configured on a z/OS DCE host system, it is recommended that DFS be deconfigured before DCE is deconfigured. Then DCE and DFS can then be configured using the new DCE host name.

Additionally, certain DFS administrative files must be changed and re-created to reflect the new DCE hostname. This section describes how to change the DCE hostname and the DFS configuration to reflect the new DCE host name. It includes the following:

- “Recommended Reconfiguration Steps When Changing DCE Host Name”
- “DCE Host Name Change While DFS is Configured”
- “DCE Host Name Change While DFS is Configured.”

Unless otherwise noted, more information on the commands and files referred to in this section can be found in the *z/OS Distributed File Service DFS Administration* book.

Also, it is recommended that the command parameter specification **/.:/hosts/***dcehostname* should be used instead of the *ipaddress* parameter whenever possible.

If you need more information on deconfiguring DCE and reconfiguring DCE refer to the *z/OS DCE Configuring and Getting Started* book. If you need more information on the DCE commands shown in this section (for example, **dcecp** and **cdscp**), refer to the *&euvmk001.*.

For additional information on how to deconfigure DFS using **DFSCONF**, see Chapter 7, “Deconfiguring DFS” on page 93.

Recommended Reconfiguration Steps When Changing DCE Host Name

The recommended sequence of DFS and DCE reconfiguration steps are:

1. Use **DFSCONF** to deconfigure DFS using the current DCE host name.

   Note: The DCE host name displayed on the DFSCONF panels is the DCE host name that must be used for the DFS deconfiguration in this scenario.

2. Deconfigure DCE.
3. Configure DCE using a new DCE host name.
4. Configure DFS using the new DCE host name.

   Note: The new DCE host name that must be used for DFS configuration is displayed appropriately on DFSCONF panels.

5. Complete DFS deconfiguration by removing the DFS entries from various configuration files that contain the previous DCE host name and creating new configuration files entries that contain the new DCE host name (see “DCE Host Name Change While DFS is Configured”).

DCE Host Name Change While DFS is Configured

Although not recommended, if DCE is deconfigured and reconfigured with a new DCE host name (*newdcehostname*) while DFS remains configured on a z/OS system with the previous DCE host name (*olddcehostname*), DFS does not operate properly.

Note: Make sure you review, “DCE Deconfiguration Considerations for DFS” on page 94, before you proceed with the following.

To restore DFS to an operational status, DFS must be reconfigured on the z/OS DCE host system using the following sequence of DFS reconfiguration steps:

1. **Deconfigure DFS**

   Use **DFSCONF** to deconfigure all DFS components that are configured on the z/OS system.

   Note: The DCE host name displayed on the DFSCONF panels is the *newdcehostname*. It must be replaced with the *olddcehostname* used for the DCE configuration. You can ignore any errors that occurred while running **DFSCONF**.

2. **Configure DFS**

   Use **DFSCONF** to configure all DFS components that are required on the z/OS system.

   Note: It is the *newdcehostname* that must be used for this DFS configuration. Change the DCE host name on the panel if the new DCE host name is not displayed.

3. **Start DFS Servers**

   At this time, you must start the DFS servers that were previously configured on this DCE host system using the z/OS system operator command **start dfs**. Since the DFS administrative list files in the directory **/opt/dfslocal/var/dfs** and the **/opt/dcelocal/var/dfs/BosConfig** file should be intact from the previous configuration of DFS, the DFS servers initialize successfully.

4. **Complete DFS deconfiguration**

   You must complete the DCE host name change for DFS by removing the DFS entries from various configuration files that contain the previous DCE host name, and by creating new configuration file entries that contain the new DCE host name.

   The steps to complete DFS deconfiguration and reconfiguration after DCE is configured using the new DCE host name are listed below. Some of the steps to establish the DFS configuration file entries for the *newdcehostname* are the same as the additional steps used to complete the configuration of the DFS components that are described in Chapter 4, "Configuring DFS for DCE" on page 33.

   The steps to complete DFS deconfiguration and reconfiguration after a DCE host name change are:

   a. Verify the correct DCE host name by issuing the following command:

      ```
      dcecp -c host catalog | grep newdcehostname
      ```

      Note: The **dcecp** command is described in the *z/OS DCE Configuring and Getting Started* book.

   b. You must determine the TCP/IP address (*ipaddress*) that corresponds to the *newdcehostname* using the following command:

      ```
      cdscp show object /.:/hosts/newdcehostname/self | grep ncadg
      ```

      Note: The **cdscp** command is described in the *z/OS DCE Configuring and Getting Started* book. Also, it is recommended that you remember the *ipaddress* in case you may required it at a later time.

   c. Using the TCP/IP address list fileset entries from the Fileset Location Data Base (FLDB) for the DCE host system using the following command:

      ```
      fts lsfldb -server ipaddress
      ```

      Note: You must remember the fileset names and fileset IDs for the filesets displayed if you have HFS or RFS filesets and you did not name your HFS or RFS filesets using the MVS data set name (*mvsdatasetname*) allocated for the fileset. In step 4i on page 106, you need to refer to this list to determine the correct fileset name to use for HFS or RFS filesets.

   d. Delete each fileset entry from the FLDB for the DCE host system using the following command:

      ```
      fts delfldbentry -fileset abc.fileset -server ipaddress
      ```

e. Remove the entry from the Fileset Location Database (FLDB) for the DFS servers that were configured on the z/OS host system using the *olddcehostname*. Delete the server entry using the following command:

```
fts delserverentry -server ipaddress
```

f. Add the server entry for *newdcehostname* using the following command:

```
fts crserverentry -server /.:/hosts/newdcehostname -principal hosts/newdcehostname
```

g. Synchronize the FLDB with the fileset IDs using the following command:

```
fts syncfldb -server /.:/hosts/newdcehostname
```

Note: The **fts syncfldb** command sets HFS and RFS fileset names to the MVS data set name (*mvsdatasetname*) allocated for the fileset. HFS and RFS fileset names that are different from the MVS data set name are not propagated by the **fts syncfldb** command.

h. Display FLDB to see the current fileset names and fileset IDs using the following command:

```
fts lsfldb
```

Note: The HFS and RFS fileset names displayed are the MVS data set names from the **devtab** file (**/opt/dcelocal/var/dfs/devtab**).

i. If you do not name your HFS or RFS filesets using the MVS data set name (*mvsdatasetname*) allocated for the fileset, you must now rename the HFS or RFS filesets.

First, you must determine the fileset name that was previously used for each fileset before DFS was reconfigured. In step 4c, you listed and saved the previous fileset entries from the FLDB for the DCE host. This list is referred to as the `previous FLDB list`. In step 4h, you displayed the current fileset entries from the FLDB for all DCE hosts in the cell where DFS is exporting filesets. This display is referred to as the `current FLDB list`. Now, to rename the HFS or RFS filesets follow these steps:

1) In the `current FLDB list`, look in the **aggr** column to determine the aggregate name.

2) Look in the **dfstab** file to determine if the aggregate name has a file system type of **ufs**.

3) If the aggregate has a file system type of **ufs**, locate the aggregate name in the `previous FLDB list` and use the associated fileset name as the *previousfilesetname*. The *previousfilesetname* is used as the **-newname** specification below.

4) In the `current FLDB list` entry for this aggregate name, you can determine the current fileset name which was set to the *mvsdatasetname* by the **fts syncfldb** command issued in step 4g. The *mvsdatasetname* is used as the **-oldname** specification below.

5) Now, you can rename the HFS or RFS fileset in the FLDB using the following command:

```
fts rename -oldname mvsdatasetname -newname previousfilesetname
```

Chapter 8. Starting and Stopping DFS Components

This chapter describes the procedures necessary to start and stop the DFS client (**DFSCM**) and the DFS servers on Distributed File Service. On Distributed File Service, the DFS client is implemented as a physical file system running as a colony address space, **DFSCM**. The DFS servers run in the DFS address space, **DFS**. A separate **DFSKERN** address space may optionally exist to provide better recovery capability for the server processes running in the **DFS** address space.

This chapter discusses the following topics:

- "Starting and Stopping the DFS Client (DFSCM)"
- "Starting and Stopping the DFS Servers" on page 109.

Starting and Stopping the DFS Client (DFSCM)

This section discusses the DFS client (**DFSCM**) address space and various ways of starting and stopping the DFS client on Distributed File Service. It contains the following topics:

- "Who Can Start and Stop DFSCM?"
- "DFSCM Initialization"
- "Starting the DFS Client (DFSCM)" on page 108
- "Stopping and Restarting the DFSCM" on page 109.
- "DFSCM Relationship to DCE and DFS Servers" on page 109.

Who Can Start and Stop DFSCM?

A user with z/OS operator privileges is the only one who can start or stop the DFS client (**DFSCM**).

DFSCM Initialization

The **DFSCM** is started as part of z/OS initialization and can be stopped and restarted without stopping z/OS. Other Distributed File Service servers do not need to be running on the local z/OS system with the **DFSCM**. **DFSCM** is a colony address space that is automatically started by OMVS as a physical file system during system IPL when OMVS is started. The DFS client is identified as a physical file system in the system parmlib **BPXPRM***xx* member **FILESYSTYPE TYPE(DFSC)** entry that is read during system initialization.

Note: The variable, *xx*, is a suffix used to uniquely identify system parmlib members.

On z/OS, the **FILESYSTYPE TYPE(DFSC)** entry in the system parmlib member **BPXPRM***xx* causes z/OS to initialize the **DFSCM**. Information is passed from the **FILESYSTYPE TYPE(DFSC)** entry to the **DFSCM**. Refer to "BPXPRMxx FILESYSTYPE TYPE(DFSC) Entry" on page 58 for more information.

In z/OS, the DFS client runs in the colony address space, **DFSCM**. **DFSCM** daemons **ioecmini**, **ioedfsd**, and **ioelogin** run as processes within the **DFSCM** address space and can not be controlled separately.

When initializing the **DFSCM**, z/OS UNIX issues the **ioecmini** process. The **ioecmini** process performs several functions including:

- Starting the **ioedfsd** process which in turn, creates various worker threads.
- Mounting the HFS file system name specified in the system parmlib member **BPXPRM***xx* **FILESYSTYPE TYPE(DFSC)** entry for the DFS client.
- Returning to z/OS UNIX before it has completed initialization.

- Waiting for DCE to initialize before completing initialization and notifying z/OS UNIX that the DFS client is available as a physical file system.

After **DFSCM** notifies z/OS UNIX that the DFS client is available as a physical file system, **ioecmini** is subsequently invoked by z/OS UNIX in cross memory mode by a program call as a result of a file system interface request when an OMVS user or application references a directory or file in the DFS (**/...**) namespace. More information on the DFS namespace follows.

The **ioecmini** process queues work for the **ioedfsd** process which does not run in cross memory mode. The **ioedfsd** process and its threads perform the necessary work to service the request and returns information to the **ioecmini** process which in turn, returns the information to z/OS UNIX and the OMVS user or application. Additionally, requests are queued to the **ioelogin** process by the **ioedfsd** process to perform DCE login services.

Important Note to Users

The **ioedfsd** process is also unique in z/OS in that it combines the functions of the traditional Cache Manager **dfsd** and **dfsbind** processes. The **dfsd** and **dfsbind** command options are described in the *z/OS Distributed File Service DFS Administration* book.

During initialization, the **DFSCM** also reads parameters from the local HFS **/opt/dcelocal/etc/CacheInfo** file and the **/opt/dfslocal/home/dfscm/envar** file. A parameter specified in the **CacheInfo** file is overridden by the same parameter specified in the **envar** file. Refer to “Completing the DFSCM Installation and Configuration” on page 58, for more information on the **CacheInfo** and **envar** file.

Important Note to Users

DFSCM information specified in the system parmlib member **BPXPRM***xx* only takes effect for the **DFSCM** when OMVS is started. Also, the **DFSCM _EUV_HOME** environment variable must be specified in the system parmlib **BPXPRM***xx* **FILESYSTYPE TYPE(DFSC)** entry.

Initialization processing for the **DFSCM** waits until DCE initialization processing is completed. The initialization process for the **DFSCM** assumes that the local directory, **/...**, exists and mounts the DFS global namespace, identified locally by a z/OS Hierarchical File System (HFS) file system name, at this directory. The HFS file system does not need to physically exist but the file system name should be unique as it identifies the DFS global namespace on the local z/OS host system. Any information previously mounted or accessed using the **/...** pathname is overlaid and unavailable while the **DFSCM** is running. Refer to the *z/OS Program Directory* or to the *z/OS Distributed File Service DFS Administration* book for more information on the local HFS file system used for DFS.

Starting the DFS Client (DFSCM)

DFSCM is started automatically, during the system IPL, as part of OMVS initialization. After the initialization has completed successfully you receive the following message:

```
IOEC04183I DFSD: initialization complete
```

In z/OS, you can customize and control several DFS client features by specifying **DFSCM** initialization parameters in the **/opt/dcelocal/etc/CacheInfo** file and **_IOE_CM_PARMS** environment variable (located in **/opt/dfslocal/home/dfscm/envar**). For more information, see “DFSCM Initialization” on page 107 and “Completing the DFSCM Installation and Configuration” on page 58.

Stopping and Restarting the DFSCM

To **stop** the **DFSCM** address space, use the **stop** operator command. (Refer to *z/OS MVS System Commands* book, SA22-7627, for more information on the **stop** operator command.)

```
stop dfscm
```

To **restart** the **DFSCM** address space, reply to the z/OS operator console message:

```
*nn BPXF014D FILESYSTYPE DFSC TERMINATED.  REPLY 'R' WHEN READY TO RESTART.
```

DFSCM Relationship to DCE and DFS Servers

To configure and use the **DFSCM**, DCE must be running on the same host system. **DFSCM** waits for DCE to start before completing initialization. If DCE stops while the **DFSCM** is running, the **DFSCM** processing fails.

Starting and Stopping the DFS Servers

Note: In z/OS DFS, the DFS servers are sometimes referred to as ***daemons***.

This section describes the procedures necessary to start and stop the DFS servers or daemons on Distributed File Service. For an overview of the DFS servers and processes, refer to the *z/OS Distributed File Service DFS Administration* book.

This section includes the following topics:

- "Who Can Start and Stop DFS Server and boserver Processes?"
- "Order of Starting DFS Daemons" on page 110
- "Starting DFS Server with System START Command" on page 110
- "Controlling DFS Server Processes with System MODIFY Command" on page 111
- "Stopping the DFS Server with System STOP Command" on page 110
- "Starting the boserver Processes from OMVS" on page 112
- "Stopping All the boserver Processes from OMVS" on page 115
- "DFS Server Relationship to DCE" on page 115.

Who Can Start and Stop DFS Server and boserver Processes?

There are two types of users who can start or stop the DFS server z/OS address space and the processes controlled by **DFSCNTL**, (see "Controlling DFS Server Processes with System MODIFY Command" on page 111):

- A user with z/OS operator privileges.
- A user who has update privilege to the **DFSKERN.START.REQUEST** RACF facility. This facility is created during the installation of DFS. For more information on this RACF facility, refer to the *z/OS Program Directory.*.

The **boserver** process itself is controlled by the **DFSCNTL** process. But it, in turn, controls sub-processes (see "Starting the boserver Processes from OMVS" on page 112, for more information on these processes). To start or stop the **boserver** sub-processes from OMVS using the **bos create** or **bos stop** command, the issuer must be logged into DCE with DCE principal or DCE group that is listed in the **admin.bos** file on the DCE host where the **boserver** sub-process is running. (Note, you can also issue **bos** commands with the **-localauth** option when you are logged on the z/OS system as **root** user with **UID=0**. For more information, refer to the *z/OS Distributed File Service DFS Administration*.)

Order of Starting DFS Daemons

When DFS starts, the DFS daemons that are started automatically are controlled by the configuration type in the **/opt/dfslocal/etc/ioepdcf** file.

When DFS daemons are started manually, the successful start-up of some daemons depends on the availability of services provided by other daemons. This implies that the DFS daemons must be started in a particular order.

The following is the sequence in which DFS daemons should be started on a DCE host system:

1. **dfskern**
2. **export**
3. **boserver**
4. **bakserver**
5. **butc***nn*
6. other DFS servers controlled by the **boserver** can be started in any order.

For example to start the **boserver**, the **dfskern** daemon must already be up and running.

If the DFS daemons are started using the **modify dfs,start all** z/OS system operator command, DFS ensures that the correct starting order is followed.

Note that if the **/opt/dfslocal/etc/ioepdcf** file indicates that a **butc** daemon should be started automatically, and a **bakserver** is not already running in the DCE cell, the **butc** daemon fails to start automatically. If the **bakserver** is run on the same DCE host system with a **butc** daemon, the **butc** daemon must be started using the **modify dfs,start butc***nn* command.

Make sure that the passwords of the DFS daemons are valid before starting them up. If the passwords have expired, the daemons cannot be started. For more information, refer to the *z/OS Distributed File Service DFS Administration*.

Starting DFS Server with System START Command

The z/OS system command, **start**, is used to start DFS in Distributed File Service (for detailed information regarding this command, refer to the *z/OS Distributed File Service DFS Administration* book).

To start **dfs**, issue the following command:

```
start dfs
```

A message written by **dfsexport** specifies each aggregate that you have exported and that is attached to the file system. If there are problems in exporting an aggregate, **dfsexport** issues a message identifying the aggregate that failed to attach.

Stopping the DFS Server with System STOP Command

To stop the DFS address space, use the **stop** operator command to ensure the normal shutdown of the address space.

To stop the DFS address space and all DFS daemons, enter the following z/OS operator commands:

```
modify dfs,start unexport (optional)
stop dfs
```

Note: You can issue the **modify** command with the **unexport** option to perform an orderly shutdown of the DFS server. The **modify dfs,start unexport** command insures updates to files made by DFS clients are written to the DFS server and makes the files in exported aggregates unavailable to the DFS clients. It is not necessary to issue the command if a normal DFS shutdown is being performed using the **stop dfs** command. It is shown here for your information. The **modify dfs,start export** command can be used to instruct the DFS file server to re-export aggregate file data. Refer to "Controlling DFS Server Processes with System MODIFY Command" for more information.

Controlling DFS Server Processes with System MODIFY Command

Once the DFS address space is started, the DFS Control Task, **DFSCNTL**, is running. Processes under its control can be stopped, restarted, or queried by issuing the z/OS system command, **modify** (for detailed information regarding this command, refer to the *z/OS Distributed File Service DFS Administration* book). Processes that run under **DFSCNTL** are: **dfskern**, **dfsexport** (which includes the **export** and **unexport** processes), **boserver**, and **butc01** through **butc08**.

Note: In Distributed File Service, the following subprocesses run under the control of the **boserver** process: **bkserver**, **flserver**, **ftserver**, **rpserver**, **upserver**, and **upclient**. These processes are started and stopped differently (see "Starting the boserver Processes from OMVS" on page 112 and "Stopping All the boserver Processes from OMVS" on page 115).

For example, to start the **butc01** daemon, enter the following:

```
modify dfs,start butc01
```

To start all the daemons, enter the following:

```
modify dfs,start all
```

Do not use the **modify** command to start additional DFS daemons while the DFS address space is still initializing. During initialization, **DFSCNTL** attempts to start all the processes that run under its direct control that have been configured on the z/OS host system and have the appropriate configuration type in the **/opt/dfslocal/etc/ioepdcf** file (for more information, see "Customizing the ioepdcf File" on page 64). If you issue the **modify** command while DFS is initializing, the DFS daemons may be started out of order or stopped erroneously. This may lead to unexpected errors during initialization and cause DFS to end abnormally.

Note: You must wait until DFS has issued a log message indicating that DFS initialization has completed before using additional **modify** commands.

To stop the DFS daemons that run under **DFSCNTL** (**dfskern**, **export**, **unexport**, **boserver**, **butc01** thru **butc08**) but not the DFS address space, use the **stop all** command. For example:

```
modify dfs,stop all
```

The **stop all** command causes **DFSCNTL** to stop all daemons that it controls. The **boserver** is stopped, causing all daemons controlled by it to also be stopped.

You can use the **modify dfs** system command to stop any specific daemon that runs under **DFSCNTL**. For example, to stop the **boserver** and all its sub-processes, enter:

```
modify dfs,stop boserver
```

Starting the boserver Processes from OMVS

The **boserver** process, like all processes that run under the control of the Control Task, can be started automatically during system initialization. This can be done by specifying a **Y** for the **boserver's** configuration type in the **/opt/dfslocal/etc/ioepdcf** file (for more information, see "Customizing the ioepdcf File" on page 64). Processes that run under the control of the **boserver** are started using the **bos create** command and are automatically started when the **boserver** starts. These processes are:

- **bkserver**
- **flserver**
- **ftserver**
- **rpserver**
- **upserver**
- **upclient**.

Starting the **boserver** creates empty **/opt/dfslocal/var/dfs/BosConfig** and **/opt/dfslocal/var/dfs/admin.bos** files if the files do not exist.

Prior to starting any of the processes controlled by the **boserver**, consult the following:

- If you plan to start the **bkserver**, refer to "Completing the Backup Database Server (bakserver) Configuration" on page 53.
- If you plan to start the **flserver**, refer to "Completing the FLDB Server (flserver) Configuration" on page 47.
- If you plan to start the **ftserver**, refer to "Completing Fileset Server (ftserver) Configuration" on page 49.
- If you plan to start the **rpserver**, refer to "Completing Replication (repserver) Server Configuration" on page 51.
- If you plan to start the **upclient**, refer to "Completing Update Client (upclient) Configuration" on page 51.
- If you plan to start the **upserver**, refer to "Completing Update Server (upserver) Configuration" on page 52.

Important Note to Users

Do not run the **upserver** and **upclient** daemons on the same z/OS machine.

To start any of the **boserver's child** processes, use the **bos create** command. To direct **STDOUT** and **STDERR** from the **child** processes to their own output DD name, a redirection parameter is needed for the **bos create** command. The parameters must also include the definition of an environment variable to define the program's home directory. This allows the server running under DCE to find the proper **krb5ccname** file and to perform its own dce_login. The path for the program must be substituted with the partitioned data set member name of the program to be run.

Besides starting the **child** process, the **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file and cause the **child** process to be started each time the **boserver** is restarted.

In the following example, of starting the **ftserver**, **/.:/hosts/DFSMVS** identifies the z/OS host running the DFS server started task. The **ftserver simple** entry specifies the server process name and process type. The **ftserver** entry immediately following the first **"** (double quote) is the name of the member in the SIOELMOD data set in the z/OS MVS partitioned data set (PDS) load library (the SIOELMOD data set was created during Distributed File Service installation). For more information, refer to the *z/OS Program Directory*. The portion of the command in double quotes specifies the program, **ftserver**, to be executed, sets a runtime parameter, **envar**, and redirects **STDOUT** for this process to the z/OS ddname **dd:ftserver**.

In Distributed File Service, **envar**, the runtime parameter, must include **_EUV_HOME** which points to that process' home directory. The slash, **/**, before **>dd:ftserver** separates the runtime parameters from the Distributed File Service program parameters. A redirection parameter to the server's own output ddname must be specified in Distributed File Service. In this example, the redirection of **STDOUT** to **DDNAME=FTSERVER** and the redirection of **STDERR** to **STDOUT** is specified by: **dd:ftserver 2**>**&1**. It is important to note that Distributed File Service may, depending on the processes to be created, require additional parameters.

```
$ bos create /.:/hosts/DFSMVS ftserver simple "ftserver \
    envar('_EUV_HOME=/opt/dfslocal/home/ftserver')/ >dd:ftserver 2>&1"
```

The process automatically begins on z/OS after issuing this command and whenever the **boserver** is restarted.

Example of Creating, Starting, and Stopping the ftserver:

An example command to create and start the **ftserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos create /.:/hosts/DFSMVS ftserver simple "ftserver \
  envar('_EUV_HOME=/opt/dfslocal/home/ftserver')/ >dd:ftserver 2>&1"
```

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **ftserver** process starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **ftserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS ftserver
```

An example command to restart the **ftserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS ftserver
```

Example of Creating, Starting, and Stopping the bakserver:

An example command to create and start the **bkserver** process (also known as the **bakserver**) on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos create /.:/hosts/DFSMVS bakserver simple "bkserver \
  envar('_EUV_HOME=/opt/dfslocal/home/bakserver')/ >dd:bkserver 2>&1"
```

Note: Enter the previous command by specifying **bakserver** and **bkserver** exactly as shown.

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **bakserver** starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **bakserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS bakserver
```

An example command to restarting the **bakserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS bakserver
```

Example of Creating, Starting, and Stopping the flserver:

An example command to create and start the **flserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos create /.:/hosts/DFSMVS flserver simple "flserver \
  envar('_EUV_HOME=/opt/dfslocal/home/flserver')/ >dd:flserver 2>&1"
```

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **flserver** process starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **flserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS flserver
```

An example command to restart the **flserver** process on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS flserver
```

Example of Creating, Starting, and Stopping the repserver: An example command to create and start the **rpserver** process (also known as the **repserver**) on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos create /.:/hosts/DFSMVS repserver simple "rpserver \
  envar('_EUV_HOME=/opt/dfslocal/home/repserver')/ >dd:rpserver 2>&1"
```

Note: Enter the previous command by specifying **repserver** and **rpserver** exactly as shown.

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **repserver** starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **repserver** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS repserver
```

An example command to restart the **repserver** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS repserver
```

Example of Creating, Starting, and Stopping the upserver: Here is an example of creating and starting an **upserver** process and enabling it to act as the system control machine for the specified administrative files. In this example, the administrative files (**admin.bak**, **admin.bos**, **admin.ft**, **admin.fl**, and **admin.up**) all reside in the directory **/opt/dcelocal/var/dfs/**. To create and start an **upserver** on a z/OS system with a DCE host name of **DFSMVS**, issue the following command from OMVS:

```
$ bos create /.:/hosts/DFSMVS upserver simple "upserver \
  envar('_EUV_HOME=/opt/dfslocal/home/upserver')/ >dd:upserver 2>&1 \
  /opt/dcelocal/var/dfs/ admin.bak admin.bos admin.ft admin.fl \
  admin.up"
```

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **upserver** process starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **upserver** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS upserver
```

An example command to restart the **upserver** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS upserver
```

Example of Creating, Starting, and Stopping the upclient: Here is an example of creating and starting the **upclient** process when the **upclient**, the **boserver**, and the **ftserver** are on the z/OS system with a DCE host name of **DFSMVS** and the **upserver** is on the system with a DCE host name of *dcehostnameups*. To create and start the **upclient** from OMVS, issue the command:

```
$ bos create /.:/hosts/DFSMVS upclient simple "upclient \
  envar('_EUV_HOME=/opt/dfslocal/home/upclient')/ >dd:upclient 2>&1 \
  /.:/hosts/dcehostnameups /opt/dcelocal/var/dfs/admin.bos  \
  /opt/dcelocal/var/dfs/admin.ft -f UpCLog"
```

The **bos create** command updates the **/opt/dfslocal/var/dfs/BosConfig** file. As a result, the **upclient** process starts on the z/OS system and whenever the **boserver** is restarted.

An example command to stop the **upclient** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos stop /.:/hosts/DFSMVS upclient
```

An example command to restart the **upclient** on the system with a DCE host name of **DFSMVS** from OMVS is:

```
$ bos start /.:/hosts/DFSMVS upclient
```

Stopping All the boserver Processes from OMVS

Processes controlled by **boserver** can be shut down using the **bos stop** and **bos shutdown** commands from OMVS. The following command instructs the BOS Server running on **fs3** to stop all **child** processes and continue to run itself:

```
$ bos shutdown /.../abc.com/hosts/fs3
```

For more information, refer to the *z/OS Distributed File Service DFS Administration* book.

DFS Server Relationship to DCE

To configure and use the DFS server, DCE must be running on the same host system. The DFS server cannot successfully start if DCE is not running. If DCE processing fails after the DFS server is initialized, the DFS server processing fails.

Appendix A. Example DFS Server Configuration Log File (dfsconf.log)

This appendix shows an example of the contents of a z/OS DFS configuration log file **dfsconf.log** that is created in the home directory of the user running DFSCONF. The example shows the output of a DFS server configuration, DFS server deconfiguration, DFS client configuration and DFS client deconfiguration in that order. The output shown here is for reference only and the actual output that is created by DFSCONF in the **dfsconf.log** file may not be exactly as shown.

```
-------------------------------------------------------------------------
Notice:  DFSCONF started at Fri Dec 19 15:37:13 1997.

Notice:  Option IOEBMAIN.1 selected.
IOED01033I DFS configuration initiated at Fri Dec 19 15:37:18 1997.

Notice: Panel Data:
        <cellname>    = !nomyn.endicott.ibm.com!
        <host name>   = !DCEDFS7!
        <dfs server>  = !1!
        <backup tape> = !1!
        <backup db>   = !1!
        <fileset db>  = !-2024565684!

IOED01016I Login for cell_admin at Fri Dec 19 15:37:22 1997.
dce_login cell_admin
EUVS24588I Attention - change current password.
EUVS24577I Login successful.
dce_login cell_admin admin

Beginning configuration of DFS server machine.

IOED01022I Configure Initial Registry Information

IOED01019I Adding necessary principals to registry database for DCEDFS7.
dcecp -c group catalog

dcecp -c group create subsys/dce/dfs-admin -inprojlist yes

dcecp -c group catalog

dcecp -c group create subsys/dce/dfs-fs-servers -inprojlist yes


dcecp -c group list subsys/dce/dfs-admin

dcecp -c group add subsys/dce/dfs-admin -member cell_admin

IOED01023I Creating Security Group
IOED01019I Adding necessary principals to registry database for DCEDFS7.
dcecp -c principal create hosts/DCEDFS7/dfs-server

IOED01020I Creating the dfs-server principal for DCEDFS7.
rgy_edit -up

IOED01021I Modify ACL for DCEDFS7.
dcecp -c acl modify
```

```
 /.../nomyn.endicott.ibm.com/sec/principal/hosts/DCEDFS7/dfs-server
 -add {group subsys/dce/dfs-admin rcDnfmag}

IOED01024I Configure Initial Name Service Information
Modifying the namespace for DFS server operation
dcecp -c directory create /.:/subsys/dce/dfs

Setting ACLs on the new entries for DFS server operation
dcecp -c acl modify /.:/subsys/dce/dfs
 -ic -add {group subsys/dce/dfs-admin rwcidta}

dcecp -c acl modify /.:/subsys/dce/dfs
 -io -add {group subsys/dce/dfs-admin rwcdt}
dcecp -c acl modify /.:/subsys/dce/dfs
 -add {group subsys/dce/dfs-admin rwcdta}

dcecp -c acl modify /.:/fs -e
 -add {group subsys/dce/dfs-admin rwcdt}

dcecp -c acl modify /.:/fs -e
 -add {group subsys/dce/dfs-fs-servers rwcdt}

Setting Security ACLs
acl_edit
 -addr 50d6e004-2b8f-1f97-95dc-001234567890@ncacn_ip_tcp:9.130.79.230
  group/subsys/dce/dfs-admin -m group:acct-admin:rctDnfmM

acl_edit
 -addr 50d6e004-2b8f-1f97-95dc-001234567890@ncacn_ip_tcp:9.130.79.230
  group/subsys/dce/dfs-fs-servers -m group:acct-admin:rctDnfmM

acl_edit
-addr 50d6e004-2b8f-1f97-95dc-001234567890@ncacn_ip_tcp:9.130.79.230
 group/subsys/dce/dfs-fs-servers -m group:subsys/dce/dfs-admin:rctDnfmM

Configure Registry Information              .
dcecp -c group add subsys/dce/rpc-server-group
 -member hosts/DCEDFS7/dfs-server

Update Name Service Database
Exporting the DFS server endpoint mapper host binding.
dcecp -c rpcentry export /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self
 -i {e1af8308-5d1f-11c9-91a4-08002b14a0fa,3.0}
 -b {ncadg_ip_udp:9.130.79.230ﬄ135“}

Setting ACL for DFSTRACE Command Suite
dcecp -c acl modify /.../nomyn.endicott.ibm.com/hosts/DCEDFS7
 -add {user hosts/DCEDFS7/dfs-server rwdtcia}

End configuration of DFS File Server.


Beginning configuration of DFS BAK server machine.

dcecp -c group catalog

dcecp -c group create subsys/dce/dfs-bak-servers -inprojlist yes
dcecp -c rpcgroup list /.../nomyn.endicott.ibm.com/subsys/dce/dfs/bak

dcecp -c rpcentry create /.../nomyn.endicott.ibm.com/subsys/dce/dfs/bak

dcecp -c rpcprofile add /.../nomyn.endicott.ibm.com/cell-profile
 -member /.../nomyn.endicott.ibm.com/subsys/dce/dfs/bak
 -i {eb814e2a-0099-11ca-8678-02608c2ea96e,4.0} -a bak

dcecp -c acl modify /.:/subsys/dce/dfs/bak -e
 -add {group subsys/dce/dfs-bak-servers rwct}

acl_edit
 -addr 50d6e004-2b8f-1f97-95dc-001234567890@ncacn_ip_tcp:9.130.79.230
  group/subsys/dce/dfs-bak-servers -m group:acct-admin:rctDnfmM

acl_edit
 -addr 50d6e004-2b8f-1f97-95dc-001234567890@ncacn_ip_tcp:9.130.79.230
  group/subsys/dce/dfs-bak-servers -m group:subsys/dce/dfs-admin:rctDnfmM

dcecp -c rpcgroup add /.../nomyn.endicott.ibm.com/subsys/dce/dfs/bak
 -member /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self

dcecp -c group add subsys/dce/dfs-bak-servers
 -member hosts/DCEDFS7/dfs-server

End configuration of DFS BAK server.

Beginning configuration of DFS FLDB server machine.

dcecp -c rpcgroup add /.../nomyn.endicott.ibm.com/fs
 -member /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self

dcecp -c group add subsys/dce/dfs-fs-servers
 -member hosts/DCEDFS7/dfs-server

End configuration of DFS FLDB server.

IOED01060I This configuration was successful for this machine.

Notice:  DFSCONF ended at Fri Dec 19 15:43:29 1997.
-------------------------------------------------------------------------


Notice:  DFSCONF started at Fri Dec 19 15:16:31 1997.
-------------------------------------------------------------------------

Notice:  Option IOEBMAIN.2 selected.
IOED01032I DFS deconfiguration initiated at Fri Dec 19 15:16:39 1997.

Notice: Panel Data:
        <cellname>    = !nomyn.endicott.ibm.com!
        <host name>   = !DCEDFS7!
        <dfs server>  = !1!
        <backup tape> = !1!
        <backup db>   = !1!
        <fileset db>  = !-2024565684!
```

```
IOED01016I Login for cell_admin at Fri Dec 19 15:16:43 1997.
dce_login cell_admin
EUVS24588I Attention - change current password.
EUVS24577I Login successful.
dce_login cell_admin admin


Beginning deconfiguration of DFS FLDB server machine.

dcecp -c rpcgroup remove /.:/fs
 -member /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self

dcecp -c group remove subsys/dce/dfs-fs-servers
 -member hosts/DCEDFS7/dfs-server


End deconfiguration of DFS FLDB server.


Beginning deconfiguration of DFS BAK server machine.

dcecp -c rpcgroup remove /.../nomyn.endicott.ibm.com/subsys/dce/dfs/bak
 -member /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self

dcecp -c group remove subsys/dce/dfs-bak-servers
 -member hosts/DCEDFS7/dfs-server

End deconfiguration of DFS BAK server.

Beginning deconfiguration of DFS server machine.

dcecp -c group remove subsys/dce/rpc-server-group
 -member hosts/DCEDFS7/dfs-server

dcecp -c account delete hosts/DCEDFS7/dfs-server

dcecp -c keytab
 remove /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/config/keytab/self
 -member hosts/DCEDFS7/dfs-server

dcecp -c principal delete hosts/DCEDFS7/dfs-server

dcecp -c rpcentry unexport /.../nomyn.endicott.ibm.com/hosts/DCEDFS7/self
 -i {e1af8308-5d1f-11c9-91a4-08002b14a0fa,3.0}


End deconfiguration of DFS File server machine.

IOED01061I This machine has been successfully deconfigured.

Notice:  DFSCONF ended at Fri Dec 19 15:20:20 1997.
-----------------------------------------------------------------------

Beginning configuration of DFS Client machine.

IOED01060I This configuration was successful for this machine.

Notice:  DFSCONF ended at Thu Jan  8 15:33:08 1998.
```

```
-------------------------------------------------------------------------

Notice:  DFSCONF started at Thu Jan  8 15:32:34 1998.

Notice:  Option IOEBMAIN.4 selected.
IOED01032I DFS deconfiguration initiated at Thu Jan  8 15:32:48 1998.

Notice: Panel Data:
        <cellname>    = !gdlcsst.endicott.ibm.com!
        <host name>   = !GDLCSST!
        <client>      = !1!


Beginning deconfiguration of DFS Client machine.

IOED01061I This machine has been successfully deconfigured.

-------------------------------------------------------------------------
```

Appendix B. Directories and Files

This appendix lists the important z/OS HFS files and subdirectories shipped as part of the Distributed File Service product. In addition, symbolic links created during the installation process are identified.

Files and Directories in /opt/dfsglobal

Note: The symbolic link **/opt/dfsglobal** refers to the directory **/usr/lpp/dfs/global**.

Table 13 lists the directories and the files in the **/opt/dfsglobal** directory. It also gives a short description of the directories.

Table 13. Files and Directories in /opt/dfsglobal

File or Directory	Created by	Description
/opt/dfsglobal/bin	Installation	Contains commands: bak, bos, dce_error, dfsexport, fts, growaggr, mapid, newaggr, salvage, scout, and udebug.
/opt/dfsglobal/examples	Installation	Contains example files for devtab, dfstab, ioepdcf, and rfstab. Also contains the example envar files for the DFS client and server processes. The directory also contains the following program files: **asc2eb.cmd** and **eb2asc.cmd**. The **asc2eb.cmd** program file is used to convert ASCII characters to EBCDIC characters. The **eb2asc.cmd** program file is used to convert EBCDIC characters to ASCII characters.
/opt/dfsglobal/lib/nls/msg/En_US.IBM-1047	Installation	Contains DFS message catalogs.
/opt/dfsglobal/src/COPYRIGHT.DFS	Installation	Contains the Distributed File Service copyright information.
/opt/dfsglobal/script/dfs_cpfiles	Installation	Contains the z/OS shell script for the z/OS Distributed File Service default configuration files creation program.

Symbolic Links to /etc/dfs Created During Installation

Table 14 lists the frequently used symbolic links created during installation.

Table 14 (Page 1 of 2). Symbolic Links Created During Installation

Symbolic Link	Linked File
/bin/bak	../usr/lpp/dfs/global/bin/bak
/bin/bos	../usr/lpp/dfs/global/bin/bos
/bin/cm	../usr/lpp/dfs/global/bin/cm
/bin/dfsexport	../usr/lpp/dfs/global/bin/dfsexport
/bin/fts	../usr/lpp/dfs/global/bin/fts
/bin/growaggr	../usr/lpp/dfs/global/bin/growaggr
/bin/mapid	../usr/lpp/dfs/global/bin/mapid

Table 14 (Page 2 of 2). Symbolic Links Created During Installation

Symbolic Link	Linked File
/bin/newaggr	../usr/lpp/dfs/global/bin/newaggr
/bin/salvage	../usr/lpp/dfs/global/bin/salvage
/bin/scout	../usr/lpp/dfs/global/bin/scout
/bin/udebug	../usr/lpp/dfs/global/bin/udebug
/etc/ioepdcf	../etc/dfs/etc/ioepdcf
/opt/dcelocal/etc/CacheInfo	../../../etc/dfs/etc/CacheInfo
/opt/dcelocal/var/dfs	../../../etc/dfs/var/dfs
/opt/dfslocal	../etc/dfs
/opt/dfsglobal	../usr/lpp/dfs/global

Directories in /opt/dfslocal

Notes:

1. The symbolic link **/opt/dfslocal** refers to the directory **/etc/dfs**.
2. Prior to OS/390 Version 2 Release 6, some DFS symbolic links were created to link to files in the **/usr/lpp/dfs/local** path. These symbolic links must now link to files in the **/etc/dfs** path.

Table 15 lists the directories in **/opt/dfslocal** and a brief description of each.

Table 15 (Page 1 of 2). Directories in /opt/dfslocal

Name (Note: /opt/dfslocal-->/etc/dfs)	Created by	Description of Directories
/opt/dfslocal/etc	Installation	Contains the daemon configuration (ioepdcf) file.
/opt/dfslocal/home	Installation	Contains the home directories of the Distributed File Service daemons (see the following entries).
/opt/dfslocal/home/bakserver	Installation	Home directory of the bakserver process.
/opt/dfslocal/home/boserver	Installation	Home directory of the boserver process.
/opt/dfslocal/home/butc01	Installation	Home directory of the butc01 process.
/opt/dfslocal/home/butc02	Installation	Home directory of the butc02 process.
/opt/dfslocal/home/butc03	Installation	Home directory of the butc03 process.
/opt/dfslocal/home/butc04	Installation	Home directory of the butc04 process.
/opt/dfslocal/home/butc05	Installation	Home directory of the butc05 process.
/opt/dfslocal/home/butc06	Installation	Home directory of the butc06 process.
/opt/dfslocal/home/butc07	Installation	Home directory of the butc07 process.
/opt/dfslocal/home/butc08	Installation	Home directory of the butc08 process.
/opt/dfslocal/home/daemonct	Installation	Home directory of the daemonct process.
/opt/dfslocal/home/dfscm	Installation	Home directory of the dfscm process.
/opt/dfslocal/home/dfscntl	Installation	Home directory of the dfscntl process.
/opt/dfslocal/home/dfsexport	Installation	Home directory of the dfsxport process.
/opt/dfslocal/home/dfskern	Installation	Home directory of the dfskern process.

Table 15 (Page 2 of 2). Directories in /opt/dfslocal

Name (Note: /opt/dfslocal-->/etc/dfs)	**Created by**	**Description of Directories**
/opt/dfslocal/home/flserver	Installation	Home directory of the flserver process.
/opt/dfslocal/home/ftserver	Installation	Home directory of the ftserver process.
/opt/dfslocal/home/growaggr	Installation	Home directory of the growaggr command.
/opt/dfslocal/home/newaggr	Installation	Home directory of the newaggr command.
/opt/dfslocal/home/repserver	Installation	Home directory of the rpserver process.
/opt/dfslocal/home/salvage	Installation	Home directory of the salvage command.
/opt/dfslocal/home/upclient	Installation	Home directory of the upclient process.
/opt/dfslocal/home/upserver	Installation	Home directory of the upserver process.
/opt/dfslocal/lib/nls/msg	Installation	Contains the DFS message catalogs.
/opt/dfslocal/syscall	Installation	Used to contain the socket for AFS_SYSCALL.
/opt/dfslocal/var/dfs	Installation	Contains administration and configuration files.
/opt/dfslocal/var/dfs/adm	Installation	Contains log files for the DFS client and server processes.
/opt/dfslocal/var/dfs/backup	Installation	Contains error and log files for the bakserver and butc processes

Customizable Files in /opt/dfslocal

Table 16 on page 126 identifies the files in the **/opt/dfslocal** subdirectories.

Notes:

1. The symbolic link **/opt/dfslocal** refers to the directory **/etc/dfs**.
2. The symbolic link **/opt/dfsglobal** refers to the directory **/usr/lpp/dfs/global**.

Many of these files are copied from the IBM supplied files in the **/opt/dfsglobal/examples** directory to the applicable operational directory by the **/opt/dfsglobal/scripts/dfs_cpfiles** program which is run as described in Chapter 3, “Installation” on page 23. These files can be updated by an administrator using an editor such as **OEDIT** to specify environment variable (**envar**) values and other configuration information for the DFS server and DFS client (DFSCM) processes.

Other files are usually created and maintained by administrator commands such as the **bos addadmin** command. These files should only be updated using the commands.

There are also some files in **/opt/dfslocal** subdirectories that are created and maintained by the DFS server and DFSCM process during normal operations. These files should not be directly updated or customized.

Finally, some of the **/opt/dfslocal** files apply to either the DFS support for DCE, the SMB File/Print support, or to both DFS and SMB.

Table 16 on page 126 describes the **/opt/dfslocal** files in this manner.

Table 16 (Page 1 of 2). DFS/SMB Customizable Files

File (Note: /opt/dfslocal-->/etc/dfs)	Created By	Customizable	Applicability
/opt/dfslocal/etc/CacheInfo	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/etc/ioepdcf	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/home/bakserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/bakserver/krb5ccname	bakserver processing	NO	DFS
/opt/dfslocal/home/boserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/boserver/krb5ccname	boserver processing	NO	DFS
/opt/dfslocal/home/butc01/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc01/krb5ccname	butc01 processing	NO	DFS
/opt/dfslocal/home/butc02/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc02/krb5ccname	butc02 processing	NO	DFS
/opt/dfslocal/home/butc03/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc03/krb5ccname	butc03 processing	NO	DFS
/opt/dfslocal/home/butc04/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc04/krb5ccname	butc04 processing	NO	DFS
/opt/dfslocal/home/butc05/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc05/krb5ccname	butc05 processing	NO	DFS
/opt/dfslocal/home/butc06/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc06/krb5ccname	butc06 processing	NO	DFS
/opt/dfslocal/home/butc07/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc07/krb5ccname	butc07 processing	NO	DFS
/opt/dfslocal/home/butc08/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/butc08/krb5ccname	butc08 processing	NO	DFS
/opt/dfslocal/home/daemonct/envar	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/home/dfscm/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/dfscm/krb5ccname	DFSCM processing	NO	DFS
/opt/dfslocal/home/dfscntl/envar	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/home/dfsexport/envar	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/home/dfskern/dfsidmap	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/dfskern/envar	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/home/dfskern/smbidmap	dfs_cpfiles Program	OEDIT	SMB
/opt/dfslocal/home/flserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/ftserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/growaggr/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/newaggr/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/repserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/salvage/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/upclient/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/home/upserver/envar	dfs_cpfiles Program	OEDIT	DFS
/opt/dfslocal/syscall/rendezvous	Server processing	NO	DFS & SMB
/opt/dfslocal/var/dfs/admin.bak	bos addadmin command	bos addadmin command	DFS
/opt/dfslocal/var/dfs/admin.bos	bos addadmin command	bos addadmin command	DFS
/opt/dfslocal/var/dfs/admin.fl	bos addadmin command	bos addadmin command	DFS
/opt/dfslocal/var/dfs/admin.ft	bos addadmin command	bos addadmin command	DFS
/opt/dfslocal/var/dfs/admin.up	bos addadmin command	bos addadmin command	DFS

Table 16 (Page 2 of 2). DFS/SMB Customizable Files

File (Note: /opt/dfslocal-->/etc/dfs)	Created By	Customizable	Applicability
/opt/dfslocal/var/dfs/cmattr	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/var/dfs/devtab	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/var/dfs/dfstab	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/var/dfs/hfstab	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/var/dfs/NoAuth	touch command	OEDIT	DFS
/opt/dfslocal/var/dfs/rfstab	dfs_cpfiles Program	OEDIT	DFS & SMB
/opt/dfslocal/var/dfs/smbtab	dfs_cpfiles Program	OEDIT	SMB

Appendix C. DFS/SMB Environment Variables

Environment variables affect the behavior of the DFS/SMB components. In the DFS/SMB, the **envar** file of a client or server process can be created to contain the declarations of the environment variables.

The DCE environment variables affect the administration of DFS. For a complete listing of the environment variables that are applicable to the Distributed File Service refer to the Appendix in the *z/OS Distributed File Service DFS Administration* book. You can also refer to the *z/OS Distributed File Service SMB Administration* book for more information on the Distributed File Service environment variables specific for SMB support.

Appendix D. DFS/SMB Performance and Tuning Guidelines

This appendix describes performance and tuning guidelines.

Who Should Read This

This information is provided for administrators of the Distributed File Service (DFS/SMB) server or client for z/OS. Customers of DFS/SMB that run large installations may need to tune the server or the client for optimal performance.

Introduction

The Distributed File Service (DFS/SMB) performance, like most file system clients and servers, is dependent on many factors. DFS/SMB support provides performance information to help the administrator determine sources of bottlenecks and determine the affect of a hardware or software change on DFS/SMB performance. Environment variables and the output of QUERY commands provide DFS/SMB with many tuning knobs that can be adjusted. Proper settings for DFS tuning options is essential.

In this section, the DFS server refers to the file server that runs to support either z/OS DCE DFS or SMB clients (or both). The DFS client refers to the z/OS DCE DFS client or a DCE DFS client running on a different system or platform.

This document provides you with a description of the various pieces of performance information you can extract from the DFS/SMB file server or DFS client. You can use this information in determining the source of a bottleneck. This document also provides information on the various DFS file server tuning options, how to set them, and how the settings may affect performance or storage consumption. Percentages listed in this appendix are guidelines. Attainable values may depend on your particular installation. Figure 7 on page 132 displays the DFS/SMB server components.

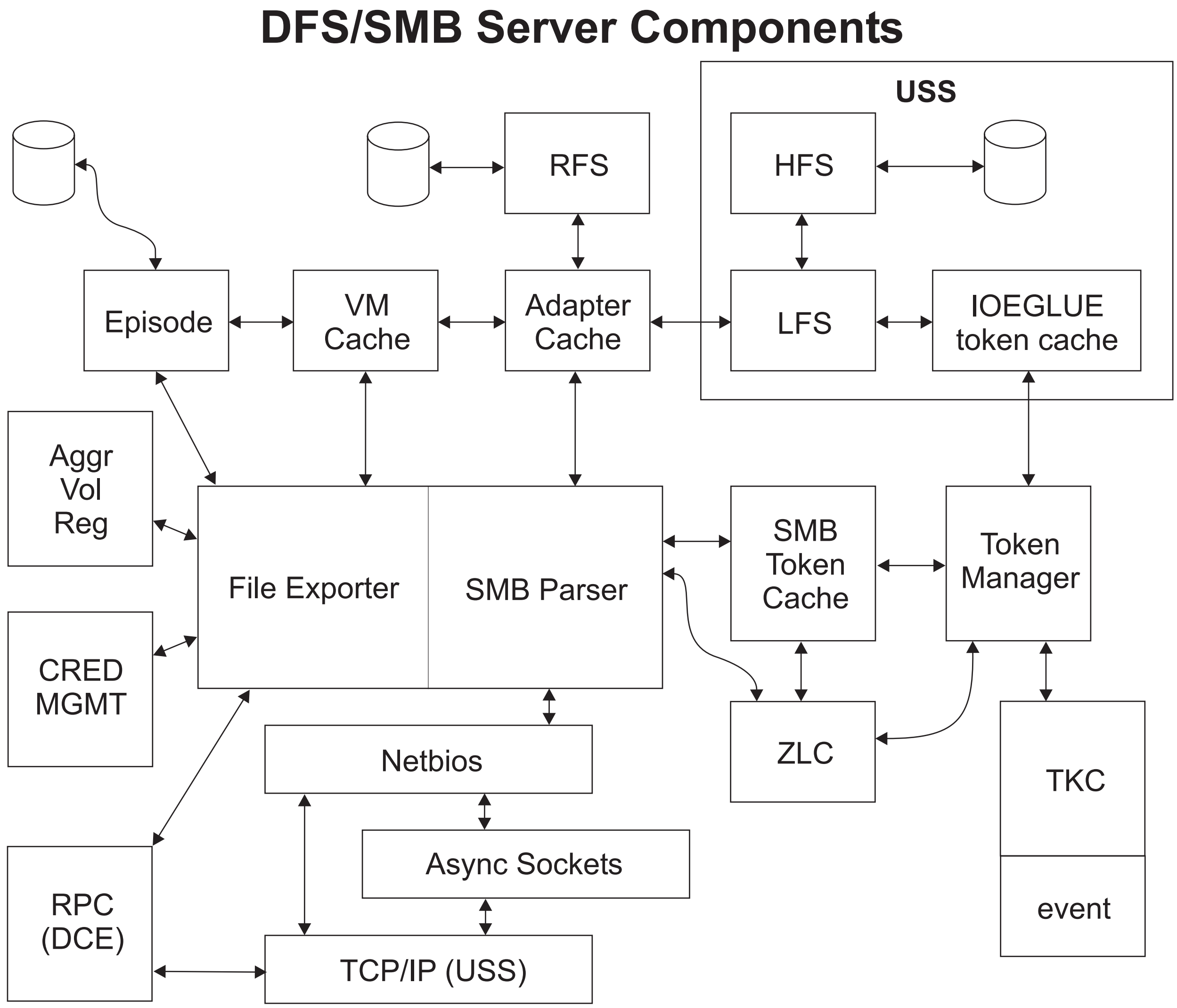


Figure 7. DFS/SMB Server Components

Tuning Considerations and Recommendations

DFS/SMB is a large and complex product with many components. z/OS Distributed File Service provides a QUERY command that provides information for each of these components. It can be used by IBM service personnel if a performance problem is reported. A subset of this information is useful to an administrator in gauging DFS performance and determining the affect of a change to a tuning option.

Tuning DFS/SMB means:

- Determining the size of thread pools responsible for handling work
- Determining the amount of storage to reserve for various types of caches
- Balancing the data I/O load among disks and the networks attached to the machine.

Many of the DFS default settings or values work for most installations, but there are some defaults that may not be sufficient for larger installations. The most important tuning options are described here.

Threading: If DFS has a unit of work that needs to be dispatched on a processing thread and there are no available threads, that work needs to be placed on a queue to wait for an available service thread. The queue wait time increases server or client response time, therefore ensuring that DFS has enough defined processing threads. DFS default values for the number of processing threads is sufficient for smaller installations, but may be inadequate for larger production use. Each processing thread requires a certain amount of storage, therefore increasing the number of threads increases the DFS storage requirements.

In the sections that follow, the tuning options that control threading are described.

_IOE_SMB_MAIN_POOL
This dfskern environment variable controls the number of threads available to process incoming SMBs for the DFS file server. This is an important tuning option for customers running SMB workloads.

_IOE_RFS_WORKER_THREADS
This dfskern environment variable controls the number of threads available to process RFS file system dataset open or close requests. This is important for customers using RFS on a larger scale.

_IOE_CM_REQUEST_THREADS
This dfscm environment variable controls the number of threads available to process local user requests that requires an RPC. This is important for DFS client workloads where a large number of users are accessing the DFS file space at one time.

-mainprocs
This dfskern command line option controls the number of threads available to process remote DCE/RPC requests for the DFS file server. This is an important tuning option for customers running DCE/DFS workloads. It also controls the number of threads available to process asynchronous file RPCs for DFS client workloads.

Caching: DFS clients cache data to reduce the need to make RPC calls to remote servers. RPCs take a "long" time to complete relative to the amount of time it takes to process a client request when the information is in the DFS client cache. Ensuring the caches are large enough to support the given workload is important to providing good performance for DFS client workloads.

DFS servers cache file system data in memory to reduce the need to perform disk I/O and to allow disk I/O to be performed more efficiently. DFS clients that use disk caching also need to determine the amount of cache storage for the LFS utility aggregate used to house the client cache files. Therefore the LFS cache options for the server are also relevant to DFS client installations that use disk caching.

Providing sufficient caches is essential to good performance. The following are some DFS client tuning options:

-blocks Number of 1K blocks in the DFS client file cache.

-files Number of files to cache. This is the number of client files used for disk caching, not relevant for memory cache clients.

-stat This sets the size of the file status information cache.

-memcache
This parameter indicates if a memory cache is used instead of a disk cache. The default is disk cache.

-namecachesize
This is the size of the name lookup cache.

_IOE_CM_DIRCACHE_SIZE
This dfscm environment variable sets the size of the client cache used to store directory data.

For file servers or clients using disk caching, the following tuning options are important:

_IOE_VM_CACHE_SIZE
This environment variable indicates how much file data to cache for DFS servers or DFS clients using disk caching. This is applicable to all server workloads, SMB or DCE, and with all file systems, HFS, RFS, and LFS.

_IOE_EPI_CACHE_SIZE
This environment variable indicates how much file system metadata to cache in memory for DFS servers or DFS clients using disk caching. This is applicable to all DFS client disk cache workloads and all server LFS workloads.

bufhigh This dfskern environment variable indicates how much data to store in the RFS dataset cache. It is used for RFS workloads. High RFS file read/write activity may require an increase in the size of this cache.

I/O Balancing: There are no server tuning options that can be used to balance I/O among disks or networks. That is a function of the physical placement of the hardware and where file systems reside. However, the DFS/SMB QUERY commands can possibly point out a bottleneck in a particular disk or the overall network. I/O response times reported by DFS/SMB may show the affect of a bottleneck or the affect of the removal of a bottleneck.

QUERY Command

The DFS file server and client provide a query command that is used to capture various performance information. The output of the QUERY command goes to the system log and the job output. The syntax of the command for the DFS file server is:

```
F DFS,SEND DFSKERN,QUERY,<REPORT>
```

The syntax of the query command for the DFS client is:

```
F DFSCM,QUERY,<REPORT>
```

Where <*REPORT*> is the name of a DFS performance report. The following is a list of all the available DFS/SMB performance reports:

Reports Available for the DFS/SMB Server

The following reports are for the DFS/SMB Server.

AFS4INT This report shows the number and type of remote DCE/DFS client RPC requests made to the server, the average DFS file server response times to process those requests, and the amount of bytes transferred to or from the file server.

SMBINT For the DFS file server, this report shows the number and type of remote Windows client SMB file requests made to the server, the average DFS file server response and CPU times to process those requests, and the TCP/IP send and receive times that can be used to determine network overhead in DFS file server response times.

SMBPRT This report shows the number and type of calls made to the Infoprint Server and the average response time and CPU time of the Infoprint Servers per call. This is used to determine the performance of the Infoprint Server.

SMBCOMM
This report shows the TCP/IP asynchronous socket calls made by the file server for SMB client communications and the amount of queuing for incoming SMB requests.

VM This report shows the storage usage and performance of the DFS file data cache that is stored in virtual memory. It is used to reduce disk I/O rates and physical file system calls.

ADAPTERS
This report shows the storage usage and performance of the DFS file server file status, directory contents, file name lookup, and file security cache used by both the HFS and RFS physical file systems. It also shows the number of calls made by the DFS protocols (DCE or SMB) to the HFS and RFS physical file systems, the number of calls to the DFSMS HFS file system, and the average response time of each call.

RFS This report shows the access method counts and response times for the RFS physical file system.

LFS This report shows the number of calls made by the DFS protocols to the local file system (Episode) along with the performance of the Episode metadata cache. It also shows I/O rates to each disk, the amount of data transferred, the number of I/O waits, and average I/O wait time.

TKM This report shows the token manager statistics for the DFS file server. It includes the size of the token manager cache, the token manager request rates, the number of token revoke RPC calls to clients, and the average response time for a token revoke RPC call.

SMBTKC This report shows the storage usage and performance of the SMB token cache used to obtain tokens for remote SMB clients. This ensures that SMB and DCE client caches remain synchronous. It also shows SMB opportunistic lock callbacks and average callback response time.

LOCKING
This report shows the DFS lock facility statistics. It includes the number of lock waits, the average lock wait time, and the time threads sleep waiting for certain specific events.

STORAGE
This report shows the total storage used by the DFS file server. It includes storage obtained from the LE heap, storage obtained from MVS subpools, and TCB owned storage.

ALL This option shows all of the reports described above.

Reports Available for the DFS Client

The following reports are for the DFS Client.

AFS4INT This report shows the number and type of remote DCE/DFS RCP requests made by the client, the average DFS file server response and CPU times to process those requests, the amount of bytes transferred to or from the client, and the RPC counts and times for remote servers and servers on the local machine.

CMSERV This report shows the external call rate, DFS client response times for each type of call, and client service thread queueing performance.

CMCACHE
This report shows the storage usage and performance of the DFS/SMB client file status, file data, directory data, name lookup caches, and DFS client background I/O statistics.

VM This report shows the storage usage and performance of the DFS server file data cache that is stored in the server's virtual memory. It is used to reduce disk I/O rates and physical file system calls. This report is only meaningful if the z/OS client uses disk caching.

LFS This report shows the number of calls made by the DFS protocols (DCE or SMB) to the local file system (Episode) along with the performance of the Episode metadata cache. It also shows I/O rates to each disk, the amount of data transferred, and the number of I/O waits, and average I/O wait time. This report is only meaningful if the z/OS client uses disk caching.

LOCKING
This report shows the DFS lock facility statistics. It includes the number of lock waits, the average lock wait time, and the time threads sleep waiting for certain specific events.

STORAGE
This report shows the total storage used by the DFS/SMB client. It includes storage obtained from the LE heap, storage obtained from MVS subpools, and TCB owned storage.

ALL This option shows all of the reports described above. Following is an example of the command syntax using the ALL option:

```
F DFS,SEND DFSKERN,QUERY,ALL
```

RESET Command

The RESET command clears the statistics monitored by the DFS/SMB file server or client. The QUERY command always displays the statistics since client or server startup or since the last RESET. This allows an administrator to query performance just for a given time period of day, such as during peek usage times. The syntax for the server is:

```
F DFS,SEND DFSKERN,RESET,<REPORT>
```

The syntax for the client is:

```
F DFSCM,RESET,<REPORT>
```

where <*REPORT*> is the name of a report as described above. The following example clears the statistics kept for the AFS4INT report.

```
F DFSCM,RESET,AFS4INT
```

Data Normalization

When analyzing performance information, you can normalize a performance metric with respect to the external request rate. For example, the SMBINT report records the number of SMBs received by the file server and the average SMB response time. Additionally, DFS/SMB provides many other fields that show count and average response times. When analyzing data, it is often easier to normalize a statistic per the average request response time. Following is an example.

From the SMBINT report the following was shown from a QUERY command:

```
Total SMB calls    5649396
Average DFS Response Time per SMB      2.158
```

Additionally, from the same QUERY command, the LOCK report showed the following:

```
Total waits for locks:                      1366750
Average lock wait time:          0.535 (msecs)
```

Therefore, to get average DFS/SMB lock wait time per SMB you could perform the following calculation:

```
Avg. Lock Waits per SMB  =   Total waits for locks    =  0.242
                            ------------------------
                             Total SMB calls

Avg. DFS lock wait time per SMB  = Avg. Lock Waits per SMB   X    Average lock wait time
                                 = 0.522
```

DFS lock wait time is approximately 25% of the overall response time in the previous example.

This same technique can be used to determine I/O waits per SMB or DCE RPC call for the LFS file system, or HFS call time per SMB or DCE/RPC by looking at data from the LFS and ADAPTERS report, respectively.

DFS Server Tuning

This section discusses Distributed File Service server tuning.

Workloads

DFS file server performance is controlled by many factors. Some factors are external, such as the requests the clients present to the server. The client workload determines much of the file server performance characteristics. For example, are there many large file reads or writes?, do the clients request many metadata operations such as file renames, deletions, hard links, and so forth. Other factors include the file read to write ratio (is the workload write intensive?). The type of the client (DCE or SMB) also determines the workload seen by the server, along with the location of the client (is the client on same machine as the server or not?).

Hardware

Hardware greatly affects file server performance. Bottlenecks in disk I/O could greatly increase server response times. An I/O bottleneck could exist because of improper balancing of I/O amongst the server's disks. It could also be a system's disks, channels, or control units cannot handle the disk I/O rates presented by the various physical file systems that DFS/SMB supports. DFS file server cache sizes may also affect disk I/O rates or result in increased or decreased I/O waits (DFS/SMB uses asynchronous I/O whenever possible).

Another possibility for contention is the network. Network transfer rates can represent a large portion of the end user response time. It could be, greater than the client and server portion of the end user response time. Network performance is especially important if large files are constantly read from or written to the file server.

SMB Workloads

SMB (Windows) clients often cache less data than DCE clients. For example, they do not cache file status or directory contents nearly as often as DCE clients do, therefore SMB workloads characteristically send much more file lookups and directory reads than DCE clients. While SMB workloads have much higher message rates per client user task or command, DFS/SMB caches data to turn around SMB requests as fast as possible.

Tuning Options: The following tuning options are available that control the type of SMBs that flow from clients to servers. These are all environment variables that would be specified in the dfskern environment variable file.

_IOE_SMB_RAW

This option controls whether the client can use the raw mode SMBs that send data in large amounts to the server. Enabling raw mode improves large file read and write performance but the administrator can disable this if desired. Enabling raw mode allows the client or server to send or receive up to 64K of data in one packet. Additionally, the transmission is optimized. For some networks disabling raw mode improved performance and therefore DFS/SMB allows the administrator to disable raw mode transmission. The possible values are:

ON This is the default. Clients send raw mode SMBs to the server.

OFF This disables raw mode and no client sends raw mode SMBs to the server.

_IOE_SMB_MAXXMT
This option controls how much data can be sent in one SMB. This is specified as a whole integer and the default is 65535. This variable controls how much data a client or the server sends in one SMB and determines how large a single file read or write request can be. Any time a client desires to send more data in one SMB it uses raw mode transmission for the read or write. The maximum possible value is 65535.

Note: A good test to determine the optimal settings for your network would be to make a simple test to copy large files to or from an idle server and track the response time. The test could be repeated with different options enabled at the server.

_IOE_SMB_OPLOCKS
This option controls whether clients are allowed to request an opportunistic lock when a file is opened for read or write. Opportunistic locks allow clients to cache the file data in certain cases, cache byte range lock requests, and perform read-ahead and write-behind optimizations. It is recommended that you run with oplocks enabled, which is the default. The possible values are:

ON This is the default.

OFF Sets oplocks off. Clients will not cache files and will not perform read-ahead or write-behind.

_IOE_SMB_PROTOCOL_LEVEL
This option determines the level of protocol negotiated with remote clients. It is the dialect that the client and server use to communicate with each other. The default is the NT protocol which is the highest level available. The following values are allowed:

NT This is the highest protocol level and the default.

LANMAN This is the next highest level and can be set by the administrator.

_IOE_SMB_NT_SMBS
This option determines the NT capabilities for the NT protocol. The possible values are:

ON Turns on the NT capabilities. This is the default.

OFF Turns off the NT capabilities. For some workloads, performance is better when NT capabilities are turned off.

Diagnostics: The following sample output is from the SMBINT report. It can be used by an administrator to determine the amount and type of file related calls made to the DFS file server by SMB clients.

```
                            Average DFS Time (msecs)
SMB Call                        Count   Response Time        CPU Time
------------------------  ----------   -------------    ------------
smb_mkdir                          20         144.581          31.926
smb_rmdir                          20          96.246          19.172
smb_close                        1779           3.310           2.195
smb_unlink                        207         151.634          35.641
smb_mv                             15         133.500          14.208
smb_getattr                      1872          11.284           0.778
smb_setattr                       469           8.206           0.872
smb_write                        8644         343.186          12.129
smb_chkpath                        45           6.176           0.516
smb_readBraw                      154           6.149           4.993
smb_writeBraw                      21         274.119          63.503
smb_setattrE                      404          46.993           0.544
smb_getattrE                      445           0.556           0.445
smb_lockingX                       37           0.305           0.245
smb_openX                        2008          30.339           6.214
smb_readX                       20883           4.295           1.850
smb_trans2                       2219          10.424           0.636
T2: find first                    941          12.764           0.366
T2: find next                      20           0.173           0.151
T2: qfsinfo                       344           0.030           0.027
T2: qpathinfo                     540           5.778           0.183
smb_findclose                      20           3.793           3.670
smb_negprot                         0           0.000           0.000
smb_sesssetupX                      0           0.000           0.000

Total SMB calls      39262
Average DFS Response Time per SMB     82.411
Average DFS CPU Time per SMB           4.442

TCP/IP Session read count:       47751    Avg Time:         0.136 (msecs
TCP/IP Session write count:      39301    Avg Time:         0.252 (msecs

Average TCP/IP call time per SMB       0.418
```

This report shows the workload presented to the server from the SMB clients. It shows the type and performance of each SMB and an overall average. Additionally, the session read and write counts show performance of the TCP/IP calls made to read or write data on the client socket. This time is included in the overall SMB response time but is not controllable by the DFS file server.

The administrator could use this information in the report to determine the affect of the setting of a tuning option or a hardware change on server SMB performance. Excessive TCP/IP send/receive times relative to the average SMB response time may indicate a network bottleneck. If the TCP/IP send plus receive time is greater than 50% of the overall SMB average response time, a network bottleneck is probably present.

Note: Only the SMB types that were received by the server since the last statistics reset are shown.

The following sample output is from the SMBPRT report. The SMBPRT report shows the calls made to the Infoprint Server on behalf of print requests received by the DFS/SMB server, the average response time, and CPU time of each call. This information can be used to determine the performance of the Infoprint Server and the portion of the DFS/SMB response time represented by Infoprint Server calls.

```
                            Average Print Time (msecs)
Infoprint Server API           Count   Failed   Response Time       CPU Time
------------------------  ----------   ------   -------------    ------------
AbortPrintFile                     0        0           0.000           0.000
BeginEnumJobs                     30        0           1.395           0.496
BeginEnumPrinters                  0        0           0.000           0.000
CancelJob                          1        0          23.560           2.879
ClosePrintFile                     3        0          31.473           2.806
CreatePrintFile                    3        0          15.222           2.475
EndEnumJobs                       30        0           0.042           0.041
EndEnumPrinters                    0        0           0.000           0.000
EnumJobs                          60        0           0.574           0.314
EnumPrinters                       0        0           0.000           0.000
GetJobInfo                         8        0           1.422           0.798
GetPrinterInfo                   110        3           1.886           0.530
HoldJob                            0        0           0.000           0.000
InitAPI                            2        1         558.078          37.522
ReleaseJob                         0        0           0.000           0.000
SetTerminationHandler              1        0           0.016           0.016
TermAPI                            0        1           0.000           0.000
WritePrintFile                    55        0           0.271           0.253

Total Infoprint Server calls        303
Average Response Time per call      5.251
Average CPU Time per call      0.684
```

The calls to `CreatePrintFile`, `ClosePrintFile`, and `WritePrintFile` occur when files are being printed by PC clients on Infoprint Server managed printers. The response time listed is the average time the print request actually spent in the Infoprint server, processing the request, or waiting for a request to complete.

Having a PC client printer queue window open results in a high volume of `GetPrinterInfo` and `xxxEnumJobs` requests to be made to the Infoprint Server. A high value for this number can represent the fact that these windows are left open for extended periods of time. Doing this can lead to unnecessary network traffic, and network contention.

Each of these remote printer queue requests uses a thread represented by the dfskern environment variable **_IOE_SMB_MAIN_POOL** to execute. The system administrator should use this information to ensure that enough threads are present to process these requests along with the file related requests from PC clients. If this number is too low, it can appear that the response time of the File portion of the SMB server is slow, but actually the file related requests are being queued instead of immediately handled. See the SMBINT and SMBCOMM reports for more information on how the SMB requests are actually being received and handled by the SMB server.

SMB Service Threads

The DFS server has two thread pools to service incoming SMB requests. If all threads are busy then incoming requests are queued until a service thread becomes available. There is a primary pool that is used to service most SMB requests and a secondary pool to handle callbacks from clients and is used to ensure that client callbacks have available threads to process them. An administrator tunes the size of the primary pool.

Tuning Options: The following environment variables control the size of the DFS file server main and secondary thread pools for SMB. The storage requirements of an SMB processing thread is shown in the SMBCOMM report.

_IOE_SMB_MAIN_POOL

This is a whole number indicating the number of threads to assign to the DFS file server main thread pool. The default is 14. This is shown as pool number 0 in the SMBCOMM report.

_IOE_SMB_CALLBACK_POOL

This is a whole number indicating the number of threads to assign to the DFS file server secondary thread pool. The default is 2. This is shown as pool number 1 in the SMBCOMM report.

The default may not be good for all your workloads. If you have a large number of active clients you want to increase the size of the main thread pool. The following section provides a diagnostic aid that can help an administrator determine the optimal thread pool size.

Diagnostics: The following sample output is from the SMBCOMM report. This report indicates the incoming requests to the DFS/SMB file server and the amount of queuing that occurs.

```
             SMB Asynchronous I/O Statistics
             -------------------------------
Service Thread Stack Size=76K   Number of Pools=2
Pool:   0   Threads:    40     Pool:   1   Threads:     2

        SRB Accepts :          0          SRB Requests :     387008
         SRB Queued :          5
          Schedules :          0               Accepts :          0
              Reads :          0                Readvs :          0
              Recvs :     386875             Recvfroms :        136
             Writes :          0               Writevs :          0
              Sends :          0               Sendtos :          0
            Cancels :          0         Cancelsockets :          0
```

The number of threads in the pool along with the size of each thread's stack is shown to allow for an approximate determination of the amount of storage required if a thread pool is adjusted in size. The `SRB Accepts` field indicates the number of new clients connecting since the statistics were reset (or since startup). The `SRB Requests` field is a raw count of client requests received by the server. The `SRB Queued` field indicates the number of requests that needed to be queued due to lack of service threads. The administrator should ensure that the percentage of requests queued is not more than 5% of the total requests since this represents increased response time to process SMB requests.

DCE Workloads

The most fundamental report available for determining the server DCE workload is the AFS4INT report. There are no server tuning options available that affects the rate or type of requests made to the server. However, larger client caches would reduce RPC rates to the server. The AFS4INT report shows response times and CPU times for each request along with the total number of file bytes sent or received by the server. These response times are *DFS only*, they do not include time spent by DCE RPC runtime sending or receiving data. DFS server response times are mainly affected by workload and disk I/O response times. For most z/OS systems tested, overall response time ranged from approximately two to thirty milliseconds on average. The affect of any kind of server change, including a tuning adjustment or new hardware, is reflected in this overall report. Note that this command can also be used for problem determination. You could issue this command to determine if a DFS file server is receiving any RPC requests from clients or not. This is useful if there is a network or configuration problem.

The following sample output is from the AFS4INT report.

```
                        AFS4 Interface RPC Call Counts

                                          Average DFS Time (msecs)
RPC Call                     Count   Response Time        CPU Time
------------------------  ----------   -------------    ------------
SAFS_Lookup                  77055           2.093           0.593
SAFS_LookupRoot               1815          10.055           0.658
SAFS_FetchStatus             15382           0.905           0.378
SAFS_StoreStatus             21266           1.037           0.451
SAFS_FetchData               33742          20.536           0.700
SAFS_StoreData               23296          17.276           1.650
SAFS_CreateFile               7669           2.848           1.547
SAFS_RemoveFile               8234           7.044           2.020
SAFS_Rename                    160           4.449           1.792
SAFS_Readdir                  6396           9.796           0.667
SAFS_MakeDir                   788          28.222           2.523
SAFS_RemoveDir                 947          11.319           1.630
SAFS_Link                        0           0.000           0.000
SAFS_Symlink                     2          14.735           1.614
SAFS_FetchACL                    0           0.000           0.000
SAFS_StoreACL                    0           0.000           0.000
SAFS_GetToken                21023           1.541           0.296
SAFS_ReleaseTokens            4953           0.320           0.305
SAFS_SetContext               2037          23.513           0.455
SAFS_SetParams                   0           0.000           0.000
SAFS_GetTime                 36419           0.082           0.081
SAFS_MakeMountPoint              0           0.000           0.000
SAFS_BulkKeepAlive               0           0.000           0.000
SAFS_BulkFetchVV                 0           0.000           0.000
SAFS_GetStatistics            7776           0.024           0.022
SAFS_BulkFetchStatus             0           0.000           0.000

Total RPC calls     268960
Average DFS Response Time per RPC      5.901
Average DFS CPU Time per RPC      0.656

Derived Fetch/Store Statistics:

Total Fetched Bytes 766433K
Total Stored Bytes 955937K

IOEN00119I DFSKERN: SEND command - QUERY,AFS4INT completed successfully.
```

DCE Service Threads

There are no DFS reports available to show service thread or request queuing for DCE workloads because DCE RPC manages the service threads. It must ensure there are enough service threads to process incoming requests or the requests are delayed waiting for an available thread.

There are two thread pools used to process DCE RPC requests. The primary pool is the main pool and the secondary pool is used to prevent deadlock when clients are called back to reclaim tokens. The administrator sets the size of the main pool. The size of the thread pools are controlled by the following two startup parameters for DFSKERN. These are specified on the DFSKERN command line that is listed in the **ioepdcf** file.

-mainprocs
This is the size of the main thread pool. The default is 4.

-tokenprocs
This is the size of the secondary thread pool. The default is 2.

Each thread requires an amount of 48K storage unless overridden with LE runtime options.

If DCE RPC runtime detects that the settings are not appropriate (excessive queuing), new threads are created on a temporary basis (up to three times the setting of each pool) to handle the requests.

However, thread creation and deletion is expensive so it is recommended that an administrator provides an ample setting for the thread pools.

Token Management

This section discusses token management.

Token Revokes: Token management is important in any DFS product because it is used to guarantee client cache consistency for both SMB and DCE remote clients. DCE clients obtain tokens from the file server and return them as needed. Whenever the file server needs to reclaim a token from a client it issues a TokenRevoke RPC to reclaim it. The amount of token revokes needed and the response time of those revokes affect performance. For example, token revokes should be small relative to the total number of RPCs present in the AFS4INT report for a DCE workload.

Central Token Manager: The DFS file server central token manager is a cache of tokens with a fixed maximum size. The maximum size is tailorable by the administrator. When setting this size the administrator should consider that every DCE client obtains at least one token for each fileset it references and keep that token for long periods of time. Because of this, systems with a large number of filesets may need to increase the size of the central token manager cache. When the central token manager runs out of tokens it performs garbage collection to reclaim a large amount of tokens from clients. This garbage collection is expensive and should be minimized during the garbage collection period. It is possible to ensure garbage collection does not happen at all.

Local User Glue: Local user access to file systems exported by DFS (either to DCE or SMB clients) also obtain tokens to ensure remote client caches remain correct. The DFS glue code caches tokens in the UNIX System Services kernel address space. For all local user access to file systems exported by DFS, the DFS glue code gets control and ensures needed tokens are obtained before the request is processed by the file system. If the tokens are in the cache then no call to the central token manager is needed. Administrators can set the size of this cache. When a token is needed from the central token manager a task switch is required. There are a pool of threads in DFS file server address space reserved to process local user token requests and the administrator can set the size of this thread pool.

SMB Token Cache: Ensuring proper file sharing semantics and client cache consistency, DFS uses an SMB token cache to obtain tokens for SMB requests from the central token manager. Windows (SMB) clients use a form of caching called opportunistic locking. These opportunistic lock requests are mapped to DFS token requests. The server caches SMB tokens from the central token manager to reduce pathlengths. Opportunistic locks may require a callback to the client. The frequency of these callbacks and their average round-trip response time are shown in the SMBTKC report. The administrator can control the size of this cache by using tuning options. Although the SMB token cache can be set larger than the central token manager cache, the SMB token cache never uses more tokens than are available in the central token manager cache. The central token manager does not allow the cache maximum to be exceeded.

Tuning Options: The central token manager cache maximum size can be set by using the following tuning option:

_IOE_TKM_MAX_TOKENS

This is the maximum number of tokens that the central token manager can hold. The default is 10,240. The size of each token (in bytes) is shown in the TKM report.

The following tuning options can be used to control the number of threads available to process local HFS user token requests. This is important if HFS file systems are updated concurrently from remote DFS clients and local HFS users.

_IOE_TKMGLUE_SERVER_THREADS

The number of threads available for processing requests from local HFS users to get tokens from the central token manager. The default is 5.

_IOE_TKCGLUE_CACHE_SIZE

The maximum number of files that the DFS glue code caches tokens for. The more files that are cached, there is not as much of a need to obtain tokens from the central token manager. The DFS glue provides this cache in the UNIX Systems Services kernel address space. The default value is 1000.

The following tuning option can be used to control the size of the SMB token cache:

_IOE_SMB_TOKEN_FILE_MAX

The SMB token cache allows the administrator to set the number of files that can have tokens cached. The default is 4096. Although the maximum size is allowed to be exceeded (the central token manager really determines the maximum number of tokens and files with tokens), the SMB token cache tries to re-use its internal file structures before exceeding the maximum size. Setting the SMB token cache too small is permitted, though pathlengths are larger because of the execution of the file reuse logic trying to keep the number of files or tokens cached below the maximum. The storage for an internal file and token structure is shown in the SMBTKC report.

Diagnostics: The following sample output is from the TKM report.

```
                 Token Management Statistics
                 ---------------------------
Token Ceiling:      262144       Allocated Tokens:       7168
Max Tokens:          17408       In Use:                  111
Min Tokens:            128       Free Tokens:            7057
GC Invocations:          0       GC Failures:               0
Tokens Obtained:    220035       Tokens Returned:      208916
Tokens Revoked:       2237       Revokes Refused:          95
Max Files:           18496       Files Allocated:        5120
Min Files:            3082       Files Freed:            5107
Files In Use:         3095       Files Recycled:        57964
Max Filesets:        18496       Filesests in use:         43

Token struct size: 80   File struct Size: 120   Fileset struct size: 168

                         TKN4 Interface RPC Call Counts

      Average TKN RPC Response Time (msecs)
RPC Call                       Count   Response Time       CPU Time
------------------------  ----------   -------------    ------------
TKN_TokenRevoke                 2094          16.221           1.045
TKN_AsyncGrant                     0           0.000           0.000
TKN_Probe                          0           0.000           0.000
TKN_InitTokenState               211          32.989           1.100

Total TKN RPC callbacks       2305
Average Response Time per TKN RPC     17.756
Average CPU Time per TKN RPC      1.050

IOEN00119I DFSKERN: SEND command - QUERY,TKM completed successfully.
```

The TKM report shows the maximum number of tokens allowed in the cache (`Max Tokens`) which is the administrator setting. The `Allocated Tokens` field indicates how many tokens are allocated and `In Use` indicates how many are in use by clients at the given time. Similarly, the `Max Files`, `Allocated Files`, and `Files in Use` indicate the file maximum, the number currently allocated in memory, and the number in use to hold client tokens. The most important fields in this report are `GC Invocations` and `GC Failures`. These numbers should be as close to 0 as possible. (0 is optimal). Garbage collection should not be invoked more than a few times per day or peek period. It is expensive, especially for DCE workloads, because it requires large RPC requests made to many clients at once to reclaim tokens. The bottom of the report shows RPC callbacks to reclaim tokens (`TKN_TokenRevoke`). These counts should be small, relative to the number of DCE client RPC requests (found in AFS4INT report), not more than a few percent of the total number of DCE RPC requests.

The following sample output is from the SMBTKC report.

```
                SMB Token Cache Statistics
                --------------------------

            Allocated        In-use
            ----------     ----------
Hosts               3              3
Files             190             68      (Max: 4096, Struct size: 152)
Tokens            189             67      (Struct size: 72)
Open Holds         21              0      (Oplocked: 0)
Lock Holds         21              0
Data Holds         55              1

                 Count                                 Count
               ----------                            ----------
Token Obtains:        239        Token Returns:           212
Fast Revokes:         217        Slow Revokes:              0
File Opens:          1779        Compatibility:             0
Oplocks:             1779        Exclusive Grants:          0
Batch Grants:           0        Pre-CIFS Grants:        1509
Level2 Grants:          0        Failed Opens:              0
File Locks:             0        Permitted Checks:      29702
Tkset Adds:         43277        Longterm Holds:           40

Total Callbacks to clients:         72      Response Waits:         72
Average Callback Response Time: 20.335 (msecs)
```

This SMBTKC report shows a number of statistics related to SMB token cache performance. The important fields are the `Total Callbacks To Clients` and `Average Callback Response Time`. The counts should not be much more than a few percent of the total number of SMBs that are shown on the SMBINT report. The `Files` allocated and in-use count should be examined. If these values are greater than the maximum then the SMB token cache is low on tokens. Server response times are increased slightly because of increased pathlength. If there are no DCE clients then the SMB token cache number of files can be set to the central token manager maximum files setting. With DCE clients it is more difficult but the central token manager is always the limiting factor.

Virtual Memory File Cache

The DFS/SMB file server has a cache that is used to contain file data to reduce disk I/Os and perform read-ahead and write-behind of file data. This cache is hereafter referred to as the virtual memory (VM) cache. The administrator can tailor the maximum number of files to cache and how much storage is used to contain the cache. This disk cache is like most disk caches. It has least recently used (LRU) algorithms used to keep to most recently used file data in memory.

This cache is used for all DFS/SMB server file systems: HFS, RFS, and LFS. Because of the nature of RFS, the cache is mainly used in a write-through mode and less write optimizations are made for RFS. The proper tuning of this caching system is essential, especially for SMB client workloads and for all LFS workloads, regardless of client type. Following are some guidelines for tuning this cache for the different types of workloads and file systems.

SMB Workloads: Windows (SMB) clients caches directory and file status information in some cases. It performs limited file data caching but the data is not cached as efficiently as DCE/DFS clients. Therefore, the VM cache is important to these workloads. As with all data caching systems, the base measure of performance is the file read hit ratio. This is the percentage of times that a read request is satisfied from the cache rather than requiring a disk I/O. This ratio for SMB workloads should be at least 80-90%. The VM cache provides diagnostics and tuning options that allows the administrator to determine the performance of the cache and alter the cache performance. The HFS file system also caches data in the address space, but the VM cache is used for HFS to reduce calls to HFS and hence reduce pathlength. A production DFS/SMB file server should be run with a larger VM cache for SMB workloads than the current 1M default.

DCE/DFS Workloads: DCE/DFS clients aggressively cache data. You will see a very low read hit ratio for these types of workloads. However, that does not mean the cache is not important. Another function of the cache is to provide the ability to perform write-behind to disk to allow for faster file write performance. This requires cache buffers to store the data while it is pending a disk I/O. If the cache is too small relative to the amount of data being stored at a given time, then as cache buffers are recycled, synchronous I/O waits will need to be performed to free buffers for a new file write request. The VM cache diagnostics provides the administrator an indication if this is occurring. The VM cache needs to be larger if there is much concurrent activity from the local DFS/SMB client because that bypasses DCE/RPC and hits the server much faster with file write requests than remote clients. This write-behind optimization is especially important for LFS workloads.

Tuning Options: The basic VM cache tuning options control how much storage is used for the cache and the maximum number of files allowed to be cached. The number of files can be set large because each in-memory structure that represents a file in the VM cache is not too large. The size of each file structure is shown in the VM report.

_IOE_VM_MAX_FILES
This is the maximum number of files to cache. The default is 4096.

_IOE_VM_CACHE_SIZE
This is the maximum size in bytes of the VM cache. The user can prefix the size with 'K' or 'M' to indicate kilobytes or megabytes, respectively. The default size is 1M. The following example sets the cache size to 256 megabytes.

```
_IOE_VM_CACHE_SIZE=256M
```

Diagnostics: The following sample output is from the VM report. This report shows the performance of the virtual memory caching system since last reset or since server startup (whichever is most recent).

```
                   Virtual Memory Caching System Statistics
                   ----------------------------------------

External Requests:
------------------
Reads          37871     Fsyncs          251     Opens        119818
Writes         35362     Truncates     12723     Unmaps         3805
Asy Reads       2882     Getattrs     283760     Schedules      7610

File System Reads:
------------------
Reads Faulted       28075     (Fault Ratio 74.13%)
Writes Faulted          0     (Fault Ratio  0.00%)
Read Waits           1829     (Wait Ratio  4.83%)
Total Reads         30766

File System Writes:
-------------------
Scheduled Writes         35362     Sync  Waits         223
Error Writes                 0     Error Waits           0
Page Reclaim Writes          0     Reclaim Waits       168
Write Waits                  3     (Wait Ratio  0.01%)

File Management: (File struct size=128)
---------------------------------------
Total Files        4096     Free              3447     Pending             0
Unreferenced        649     Referenced           0     File Waits          0
Lookups     119818     Hits       80213 (Hit Ratio   66.95)
Misses       39605     Reuses         0 (Reuse Ratio  0.00)

Page Management (Segment Size = 64K) (Page Size = 4K)
----------------------------------------------------
Total Pages        8192     Free                10     Pending             0
Unreferenced       8182     Referenced           0     Page Waits          0
Steal Invocations      15640     Steals From Self          0
Files Stolen From      40164     Steals From Others      40164
Waits for Reclaim         17
```

The following fields from the VM report are important:

External Requests
These are the number of requests presented to the VM cache system. The most significant fields are `Reads` and `Writes` which represent read and write requests made to the cache.

Reads Faulted
This field is the read miss ratio. It is the opposite of hit ratio. Obtaining the hit ratio you subtract the miss ratio from 100%. In this case, the hit ratio would be 100-74 = 26%.

Note: The hit ratio is low, as expected, in this DCE client workload example.

Reclaim Waits
This is the number of times a thread had to be suspended when reclaiming cache pages for an in-progress I/O (an I/O for file data written in write-behind mode). This number should be low relative to the number of external requests. It represents a thread waiting on disk I/O. A higher number means slower general response time.

File Waits
This is the number of times a thread had to be suspended waiting to obtain an in-memory VM cache file structure. It means that all the VM cache files are in-use and represents an extreme shortage of VM cache file structures. The preferred value is 0.

Total Pages
This is the total number of pages allocated to the VM cache. A page is 4K in size. In this case there are 8192 pages, therefore, 32M.

Page Waits

This is the number of times a thread had to wait for a free page in the cache to become available. It represents cases where there is an extreme shortage of pages in the cache. This number should be low, preferably 0, and should not be more than 1% of the total external requests.

Waits for Reclaim

The VM cache allows only one thread at a time to reclaim pages from the cache for use for a new file read or write request. If another thread requires a page and it finds another thread is already running reclaim processing, it waits. The occurrence of this depends on how many concurrent threads are running at a time and how often reclaim needs to run. This count should be low (not more than a few percent of the total number of external requests).

In the above VM report, performance of the VM cache for a DCE client workload is very good based on the indicators. The administrator can use these indicators and set the cache size to obtain an optimal size of the cache for their environment.

If `File Waits` is high then the number of files should be increased by setting the **_IOE_VM_MAX_FILES** environment variable to a higher number. If the number of `Waits For Reclaim` or `Page Waits` is high then the maximum storage should be increased by setting the **_IOE_VM_CACHE_SIZE** environment variable to a higher number. Refer to Appendix C, “DFS/SMB Environment Variables” on page 129 for more information on these environment variables.

HFS/RFS Caches

The HFS/RFS file systems share a cache used to store file status information such as owner ID, file change time, file permissions, directory contents, and directory name to vnode mappings. A vnode in DFS/SMB is much like a UNIX inode. It is the data structure used to represent a file system object such as a directory or file. The RFS file system uses MVS access methods to manipulate datasets while HFS is accessed through z/OS UNIX System Services. This cache hides HFS/RFS implementations that do not use the same interfaces as the rest of DFS/SMB and provides pathlength reduction because it reduces calls substantially to MVS access methods or HFS. The performance of this cache is essential for RFS performance. Formal tests have shown it provides a throughput boost for HFS (up to 39% for some workloads) when this cache was originally added to z/OS V1R3.0 Distributed File Service.

Status: File status is cached in an extension to the vnode. By caching this status calls are reduced to z/OS UNIX System Services or z/OS dataset access methods. By caching vnode handles the number of calls are reduced to z/OS UNIX System Services to obtain and return handles. The size of this cache is controlled by the administrator.

Permissions: The HFS/RFS cache stores user permissions to file system objects. This is done by querying the permissions from SAF or z/OS UNIX System Services. By saving the permissions, calls to z/OS UNIX System Services or SAF are reduced.

Directory Contents: The directory cache stores directory contents in DFS/SMB format. Use of this cache reduces lookup and directory read calls made to z/OS dataset access methods or HFS.

Name Lookup Cache: The name lookup cache saves directory entry name to associated vnode mappings reducing calls to obtain vnode handles.

Tuning Options: An important tuning option for the shared HFS/RFS cache is the size of the directory cache. A larger cache allows directories to be cached and results in less calls to the associated physical file system routines. However, for large production systems using HFS or RFS, a large vnode cache may be desirable.

_IOE_VNODE_CACHE_SIZE

This is the number of vnodes provided in the cache. The default is 4096. More vnodes are provided, if needed, resulting in less calls to the HFS or RFS physical file system to read file status, file permissions, and obtaining vnode handles. The size of a vnode structure is shown in the ADAPTER report.

_IOE_DIRECTORY_CACHE_SIZE

This is the number of 512 byte blocks that can be used to store directory contents and name to vnode mappings. The default is 2048 blocks, which is one megabyte of storage.

Diagnostics: The following sample output is from the ADAPTER report. The ADAPTER command can be used to determine the performance of the shared HFS/RFS cache and the performance of HFS.

```
                          Adapter Caching Statistics
                          --------------------------
Vnode Cache Size = 6144 vnodes, structure size = 488 bytes
Vnode lookups = 1368121, hits = 1326732, ratio = 97%
Vnode invalidations = 41342
Directory Cache Size = 2048 (512 byte) blocks, Free = 729
Directory buffer refreshes = 0
Directory buffer reads = 5494594, hits = 5494594, ratio=100%
Get attributes calls = 2451296, hits = 2394971, ratio=98%
Name cache lookups = 5300598, hits = 5054509, avoided=246089, ratio=100%
Avoided set attributes calls = 1462
Access checks = 8145852, hits = 8063074, ratio=99%
Access cache invalidates = 82731


                            HFS Adapter Vnode Op Counts

Vnode Op                       Count   Vnode Op                       Count
------------------------  ----------   ------------------------  ----------
xoefs_hold                   1158433   xoefs_readdir                 143600
xoefs_rele                   7651388   xoefs_create                   41387
xoefs_inactive                     0   xoefs_remove                   41343
xoefs_getattr                2043702   xoefs_rename                    8684
xoefs_setattr                 316196   xoefs_mkdir                        0
xoefs_access                  519908   xoefs_rmdir                        0
xoefs_lookup                 5313507   xoefs_link                         0
xoefs_getvolume                    0   xoefs_symlink                      0
xoefs_getlength                    0   xoefs_readlink                     0
xoefs_afsfid                 8149223   xoefs_rdwr                         0
xoefs_fid                          0   xoefs_fsync                    14362
xoefs_vmread                  150510   xoefs_translate                    0
xoefs_vmwrite                 101034

Total HFS Vnode Ops   25653277
```

```
                           Average HFS Time (msecs)
VFS Call                      Count   Response Time       CPU Time
------------------------  ----------   -------------    ------------
v_access                      82778           1.443           0.021
v_lookup                          0           0.000           0.000
v_get                             0           0.000           0.000
v_getattr                     56325          18.862           0.024
v_setattr                    356120           7.694           0.065
v_create                      41387          41.396           0.196
v_link                            0           0.000           0.000
v_mkdir                           0           0.000           0.000
v_rdwr                       265906          49.819           0.171
v_readdir                         0           0.000           0.000
v_readlink                        0           0.000           0.000
v_rel                         41342           3.549           0.041
v_remove                      41343          26.775           0.154
v_rename                       8684          28.960           0.349
v_rmdir                           0           0.000           0.000
v_rpn                             0           0.000           0.000
v_export                          0           0.000           0.000
v_symlink                         0           0.000           0.000
v_fstatfs                     10140           0.059           0.041


Total VFS calls                 904025
Average HFS Response Time per Call        22.553
Average HFS CPU Time per Call         0.101

                           RFS Adapter Vnode Op Counts

Vnode Op                      Count   Vnode Op                     Count
------------------------  ----------   ------------------------  ----------
xrda_hold                         0   xrda_readdir                    40
xrda_rele                         0   xrda_create                    219
xrda_inactive                     0   xrda_remove                    197
xrda_getattr                  17692   xrda_rename                     10
xrda_setattr                   1479   xrda_mkdir                      20
xrda_access                   21456   xrda_rmdir                      20
xrda_lookup                    7555   xrda_link                        0
xrda_getvolume                    0   xrda_symlink                     0
xrda_getlength                    0   xrda_readlink                    0
xrda_afsfid                   50687   xrda_rdwr                        0
xrda_fid                          0   xrda_fsync                       0
xrda_vmread                    1591   xrda_translate               10059
xrda_vmwrite                   8468

Total RFS Vnode Ops     119493


                              RFS Adapter VFS Op Counts

VFS Op                        Count   VFS Op                       Count
------------------------  ----------   ------------------------  ----------
r_get                             0   r_readdir                       20
r_rel                           222   r_create                       219
r_getattr                        57   r_remove                       197
r_setattr                      1259   r_rename                        10
r_access                        478   r_mkdir                         20
r_lookup                          5   r_rmdir                         20
r_export                          0   r_link                           0
r_fstatfs                         2   r_symlink                        0
r_rpn                             0   r_readlink                       0
                                      r_rdwr                       10059

Total RFS Operations           12568
```

The following sections provide explanations of the ADAPTER report.

Adapter Caching Statistics: This is an important section of the report because it details performance of the caches.

Vnode Cache Size
This is the size of the vnode cache. It determines how many files or directories status and file permissions can be cached in the HFS/RFS cache. It also displays the size of an HFS/RFS vnode.

Vnode lookups
This is the number of searches for vnodes based on the inode (file number) and the hit ratio. The higher the hit ratio, the better.

Directory Cache Size
This is the size of the directory cache.

Directory buffer reads
This is the number of directory buffer read requests and the percentage of time the directory buffer was in storage.

Get attributes calls
This is the number of calls made to obtain file status information and the percentage of time valid attributes were found in the cache.

Name cache lookups
This is the number of calls made to find the vnode associated with a given file name. The percentage of times the vnode was found without a call to the HFS or RFS physical file system is also shown.

Access checks
This is the number of permission checks made and the percentage of times the permissions for a user were found cached.

The hit ratios of the various caches should be approximately 80% or more. If the hit ratios are low then the size of the vnode or directory caches could be increased.

HFS Adapter Vnode Op Counts: This section of the report shows the number of each type of call made by the DCE or SMB protocol servers to the HFS adapter. Many of these calls do not result in an HFS call due to the presence of the shared cache.

Average HFS Time: This section shows the average response time and CPU time in milliseconds for each HFS physical file system call. The administrator determines the portion of the overall DFS/SMB response time that is represented by HFS calls, if desired. Changes to disk configurations would affect the performance of these response times because disk I/O wait time would be the predominant factor in these response times. Large HFS call times relative to the average DFS server response time could indicate DASD I/O bottlenecks.

RFS Adapter Vnode Op Counts: This is the number of calls made to the the DCE or SMB protocol servers to the RFS adapter. Many of these calls do not result in an RFS file system call due to the presence of the shared cache.

RFS Adapter VFS Op Counts: This is the number of calls made to the DFS/SMB RFS physical file system. Response times are not given for these calls because the DFS/SMB product provides the support for RFS. The RFS report shows the z/OS access method response times.

HFS

In addition to the shared HFS/RFS cache, the bulk of the performance and tuning issues are a function of the HFS file system. The appropriate HFS publications should be used to tune the HFS file system.

LFS (Episode)

The performance of Episode, like any file system, is dependent on I/O response times. Episode attempts to perform as much asynchronous I/O as possible to speed up response times. It uses logging techniques to provide fast and reliable recovery and reduces file system metadata I/O substantially.

Aggregate Log File: Every Episode aggregate has a log file that is fixed in size. This size is set by the NEWAGGR program. The default is 1% of the total disk size. This may be too small for small disks with many concurrent transactions. The size of the log file can only be set at aggregate format time by NEWAGGR. If the log file size needs to be changed the aggregate has to be reformatted.

Aggregate Block Size: In oder to obtain optimal aggregate block size performance, a default block size of 8K is recommended. It should be used when formatting the aggregate with the NEWAGGR program.

Metadata Cache: The Episode file system has a cache for file system metadata, which includes directory contents and the data of small files (smaller than the aggregate block size). The setting of this cache size is important to Episode performance because Episode references the file system metadata frequently. Synchronous reads of metadata increases I/O rates to disk and server response times.

Tuning Options The setting of the VM cache size is important for LFS performance. The administrator should set the VM cache tuning options in conjunction with the LFS tuning options.

_IOE_EPI_CACHE_SIZE
This sets the size, in bytes, of the LFS buffer cache used to store file system metadata and small file contents. You can can append a 'K' or 'M' to the size to indicate kilobytes or megabytes, respectively. The default is 1M.

_IOE_LFS_IO_SEEK
This allows the administrator to enable a seek scheduling algorithm for LFS aggregates where I/O is optimized by disk location. This changes the order from a first-come first-serve basis to one based on traditional elevator algorithms for non-priority I/O (I/O that is not being waited on by a thread). The possible values are ON or OFF. The default is ON. The following example sets the size of the LFS buffer cached to 10 megabytes.

```
_IOE_EPI_CACHE_SIZE=10M
```

Diagnostics: The following sample output is from the LFS report. This report shows the performance of the LFS file system, the I/O rates to disk, and the average I/O wait time.

```
                                LFS Vnode Op Counts

 Vnode Op                      Count   Vnode Op                      Count
 ------------------------  ----------   ------------------------  ----------
 efs_hold                          0   efs_readdir                  486909
 efs_rele                          0   efs_create                   140363
 efs_inactive                      0   efs_remove                   140580
 efs_getattr                 5435375   efs_rename                    29422
 efs_setattr                  594838   efs_mkdir                         0
 efs_access                  1122105   efs_rmdir                         0
 efs_lookup                  5357251   efs_link                          0
 efs_getvolume                     0   efs_symlink                       0
 efs_getlength                     0   efs_readlink                      0
 efs_afsfid                 10295933   efs_rdwr                          0
 efs_fid                           0   efs_fsync                     48720
 efs_vmread                    46430   efs_waitIO                   210106
 efs_vmwrite                  239265   efs_cancelIO                  71672

 Total LFS Vnode Ops   24218969
```

```
                              LFS Caching Statistics

Buffers    (K bytes)  Requests     Hits    Ratio  Updates
---------- --------- ---------- ---------- ----- ----------
      4096     32768    8975442    8419314 93.8%    9552884

                      LFS Directory Cache Statistics

Dir Blocks (K bytes)  Requests     Hits    Ratio  Deletes
---------- --------- ---------- ---------- ----- ----------
       128      1024   10138937   10121974 99.8%          0


                  I/O Summary By Type
                  -------------------

Count       Waits       Cancels     Merges      Type
----------  ----------  ----------  ----------  ----------
      1474        1598           0           0  File System Metadata
    136416       51562           0        2931  Log File
    361717      105865         310           4  User File Data


                  I/O Summary By Circumstance
                  ---------------------------

Count       Waits       Cancels     Merges      Circumstance
----------  ----------  ----------  ----------  ------------
     47502       47502           0           0  LFS cache read
       213         213           0           0  VM cache direct read
         0           0           0           0  Log file read
         0           0           0           0  LFS cache async delete write
         0           0           0           0  LFS cache async write
    104355       10307           0           1  LFS cache lazy write
         0           0           0           0  LFS cache sync delete write
         0           0           0           0  LFS cache sync write
    209648       47843         310           3  VM cache direct write
         0           0           0           0  LFS cache file sync write
       725         723           0           0  LFS sync daemon write
         0           0           0           0  LFS aggregate unexport write
         0           0           0           0  LFS buffer block reclaim write
         0           0           0           0  LFS buffer allocation write
         0           0           0           0  LFS fileset quiesce write
       749         875           0           0  LFS buffer log file full write
    136415       51562           0        2931  Log file write
         0           0           0           0  LFS buffer shutdown write
                     LFS I/O by Currently Attached Aggregate

Minor
Device MV  Mode       Reads     K bytes      Writes     K bytes   Dataset Name
------ --- ----  ----------  ----------  ----------  ----------   ------------
     1   N  R/W           0           0           0           0   DFS.EPISODE.AGGR001.LDS00001
   401   N  R/W        5854       46888       54963     1334992   IOEDFSKN.LFS.SET41
   402   N  R/W        3958       31944       37204      887544   IOEDFSKN.LFS.SET42
   301   N  R/W        5745       45960       53338     1304256   IOEDFSKN.LFS.SET31
   302   N  R/W        3873       31040       36571      881288   IOEDFSKN.LFS.SET32
   201   N  R/W        6005       48040       56573     1370048   IOEDFSKN.LFS.SET21
   202   N  R/W        4218       33744       39778      950968   IOEDFSKN.LFS.SET22
   100   N  R/W           0           0           0           0   IOEDFSKN.LFS.TEST3
   101   N  R/W        5697       46080       53302     1296728   IOEDFSKN.LFS.SET11
   102   N  R/W        3949       31984       37275      892400   IOEDFSKN.LFS.SET12
   501   N  R/W        4206       33928       39886      955864   IOEDFSKN.LFS.SET51
   502   N  R/W        4209       33840       39762      951864   IOEDFSKN.LFS.SET52
------           ----------  ----------  ----------  ----------
    12                47714      383448      448652    10825952   *TOTALS*


Total number of waits for I/O:     159025
Average I/O wait time:              6.861 (msecs)
```

The following sections provide explanations of the LFS report.

LFS Vnode Op Counts: This is the count of each type of LFS vnode operation. It shows the request rate from the DCE protocol exporter.

LFS Caching Statistics: This section of the report shows the performance of the LFS buffer cache. Because LFS allows aggregates to be blocked at different sizes (4K, 8K, etc..), the cache may grow or shrink in size depending on how many buffers for disks that are blocked larger than 8K in memory. If all aggregates are blocked at 8K then the size of the cache is fixed in size. The `Buffers` field indicates the number of buffers in memory and the `(K bytes)` column indicates total storage.

The `Hits` and `Ratio` columns are the base indicators of performance of the cache. The Ratio should show a hit ratio of 90% or more. LFS reads from the metadata cache frequently, so even if the cache is small the hit ratio is high (100% is preferred). Because LFS hits the cache so frequently per operation, a better indication of LFS I/O performance is provided in "LFS I/O by Currently Attached Aggregate"

LFS Directory Cache Statistics: The LFS directory cache is a layer above the LFS buffer cache that is used to simplify coding of directory management. This is done because each Episode aggregate could be blocked at a different size, but Episode directories are always logically blocked at 8K bytes. This cache is not tailorable by the end user and it is expected that you would not need to change the size of this cache. It is fixed in size at 1M. The contents of this cache are obtained from the LFS buffer cache, when needed.

I/O Summary: The I/O summary sections show I/O by type and circumstance (why the I/O is being performed).

LFS I/O by Currently Attached Aggregate: This report shows the read and write I/O rates to every LFS aggregate, the minor device number, R/W indicator, name of the first dataset for the aggregate, and read and write counts. The `MV` column indicates if the aggregate spans more than one dataset.

The read and write counts show I/O rates to disk. This can be used to determine the affect of a change, such as modifying one of the file server cache sizes. It is also used to determine if there is an I/O imbalance among the disks.

The number of I/O waits and average I/O wait time is also displayed. This should not be confused with DASD response time from reports such as RMF. LFS attempts to perform as much I/O as it can asynchronously, so LFS waits on fewer I/Os than it issues. The I/O wait count is incremented each time a thread or unit of work needs to be suspended waiting for I/O completion. It is the bottom line performance measurement of LFS. Any change to the disk I/O configuration along with cache sizes affects this number. The administrator can use this report to determine the average number of I/Os per RPC and the average I/O wait time per RPC. This should help an administrator balance I/O and determine if a disk or set of disks is a bottleneck.

RFS

The RFS file system maps POSIX based file I/O and maps them to z/OS dataset access method calls. It uses z/OS access methods to store file status information into z/OS catalogs and retrieves the information later. The important factors regarding RFS performance is the number of access method and z/OS dynamic allocation calls made and their corresponding response times. DFS/SMB attempts to perform I/O efficiently to and from the corresponding datasets. It tries to read/write a large number of blocks when performing I/O and attempts to overlap processing with I/O as much as possible. However, much of RFS is gated by the access method performance and disk I/O performance and is therefore the most important measure of performance. Additionally, because of the nature of z/OS datasets, certain file access patterns such as random file access and file update patterns of Windows NT machines, RFS may be forced to perform small I/Os to the datasets or may perform synchronous I/O.

Tuning Options: The adapter cache tuning options and VM cache tuning options are important for RFS performance because the larger those caches are the smaller the number of access method calls RFS needs to make. The following RFS tuning options are important:

_IOE_RFS_WORKER_THREADS
This sets the number of worker threads required to process RFS dataset open/close requests. z/OS has a restriction where a dataset must be closed by the same task that opened the dataset. Because the DCE or SMB protocol exporters manage the threads to process incoming requests there is no control available to RFS to ensure a close request is processed by the same thread that opened the dataset. DFS/SMB handles this restriction by making available a pool of threads used exclusively for dataset open/close processing. The default size of this pool is one thread.

bufhigh Many RFS processing options are specified in an RFS attributes file stored in HFS. An important option is the size of the cache used to store BSAM dataset blocks. The VM cache caches the file contents in POSIX byte stream format. BSAM record datasets require additional control information imbedded in the disk blocks, therefore I/O to the datasets must use disk blocks properly formatted for the dataset. RFS provides a buffer cache used to hold these blocks in raw dataset format. A sufficient size for it should be provided. This size should be set to the average number of bytes being read/written at a moment in time by the server. The default is 2M.

blksize Another RFS processing option is the *blksize* dataset creation parameter. This can be specified on the user command line and/or the RFS attributes file. This option is used for creation of BSAM files. Records should either be large, or should be grouped into large blocksizes for efficient I/O performance. z/OS allows applications to specify a block size of 0 which means z/OS will determine optimal block size, this is the recommended setting by DFS/SMB for BSAM datasets (sequential datasets and PDS or PDS/E members).

recordsize
This sets the size of the records for VSAM datasets. z/OS DFS will read/write up to 64K of data at a time to a VSAM dataset. However, I/O is more efficient with larger records.

space Another RFS processing option allows the administrator or user to specify the primary and secondary extent sizes of a dataset. Dataset creation and initial write takes longer if secondary extents are needed. It is best to use only a primary extent if possible, or at least make the secondary extents large enough so contents of the dataset are not scattered.

Diagnostics: The following sample output is from the RFS report.

```
                          FILBLK Management Statistics
                          ----------------------------
Active NAMEBLKs = 108
Active FILBLKs = 66
FILBLK inactivations = 0
FILBLK invalidations = 0

                             Logical I/O Statistics
                             ----------------------
Logical Cache High = 4194304 (4096K) (4M)
Logical Cache Used = 0 (0K) (0M)
Cache Window Per File = 16
Logical Read Requests = 7752
Read-for-size Requests = 0
Read-for-offset Requests = 5
Out-of-sequence Reads = 0
Logical Write Requests = 12404
Out-of-sequence Writes = 0
Writes With Holes = 0
Logical File Syncs = 48
Logical File Sync Closes = 19
```

```
             RFS Open/Close Thread Statistics
             --------------------------------
Number of threads: 5  (stack size = 16K)
Open requests:           809   Queued:          0  (0.0%)
Close requests:          797   Queued:         24  (3.0%)
AsyClose requests:        12   Queued:          1  (8.3%)


                             Physical I/O Statistics
                             -----------------------
Buffer Cache High = 2097152 (2048K) (2M)
Buffer Cache Used = 0 (0K) (0M)
Buffer Cache Steal Percent = 20%
Buffer Cache Trims = 249
Buffer Steals = 1108
Buffer Closes = 12

Dynamic Allocation Calls = 289
Dynamic Unallocation Calls = 267
                            Call Counts By Access Method
Access                                 GETs        PUTs
Method         OPENs     CLOSEs       READs      WRITEs      CHECKs       WAITs     *Total*
-------  ----------  ---------  ----------  ----------  ----------  ----------   ----------
BPAM              40         40           0          15          30           15 |       140
BSAM             263        263         509         704        1276         1213 |      4228
QSAM              20         20          20           0           0            0 |        60
VSAM             526        526         640        4062         903            0 |      6657
-------  ----------  ---------  ----------  ----------  ----------  ----------   ----------
*Total*          849        849        1169        4781        2209         1228 |     11085

                       Average Time Per Call By Access Method
Access                                GETs        PUTs
Method        OPENs      CLOSEs      READs      WRITEs      CHECKs       WAITs      *Avg*
------   ----------   ---------   ----------   ----------   ----------  ----------  ----------
BPAM           4.901       7.276        0.000        0.163        0.015       2.708 |     3.790
BSAM          10.202      23.315        0.200        0.186        0.120       4.688 |     3.521
QSAM          11.428       2.705        0.008        0.000        0.000       0.000 |     4.714
VSAM          44.727      42.300        1.894        0.891        0.694       0.000 |     7.696
-------  ----------  ---------    ----------   ----------   ----------  ----------  ----------
*Avg*         31.371     112.676       1.124       0.785       0.353          4.664 |     8.630

                      Average CPU Time Per Call By Access Method
Access                                GETs        PUTs
Method        OPENs      CLOSEs      READs      WRITEs      CHECKs       WAITs       *Avg*
------   ----------  ---------  ----------  ----------  ----------  ---------    ----------
BPAM           1.785      1.621       0.000       0.100       0.015       0.031  |     0.991
BSAM           2.097      1.705       0.112       0.104       0.055       0.028  |     0.292
QSAM           2.488      1.206       0.008       0.000       0.000       0.000  |     1.234
VSAM          13.534      9.203       1.628       0.495       0.581       0.000  |     2.334
-------  ----------  ---------  ----------  ----------  ----------  ----------   ----------
*Avg*          9.177      26.340       0.940      0.436       0.269       0.028        1.532
```

The important fields in the RFS report are the access method counts, response times, and the open/close service threads statistics.

RFS Open/Close Thread Statistics: The open/close statistics show the `Number of threads` available to process open/close requests, the number of requests, and the number of requests that needed to be queued (because all of the service threads were busy). If a large portion of open/close requests need to be queued then response times increase because of open/close queue wait time when reading/writing files or directories. The size of each thread's program stack is shown to allow an approximation of the required storage if the number of service threads is changed.

Physical I/O Statistics: The `Buffer Cache Steals` and the `Buffer Cache Trims` indicates cases where storage needed to be stolen from other files in the cache allowing an I/O to be performed for a given file. A high steal rate relative to the total number of RFS `xrda_vmread` and `xrda_vmwrite` calls (from the ADAPTER report) might indicate that the RFS buffer cache could be too small and the **bufhigh** RFS tuning option might need adjustment.

Access Method Statistics: RFS performance is determined by the response time of the z/OS access methods. DFS tries to read/write to files in large (64K) amounts but certain client access patterns might prevent that. An administrator can determine from this report and the reports that detail external client requests the relative amount of access method calls per DCE/RPC or SMB and determine what portion of the response time the access methods use.

Locking and Serialization

The DFS file server and client provides information on DFS internal lock performance.

I/O Wait Time Affects Lock Wait Time: In certain cases the file systems require a lock to be held over an I/O. I/O waits are long relative to processor speed. The slower the disk response time the longer the I/O waits. Therefore, disk I/O wait time affects not only those threads waiting on the I/O, but also other threads if the I/O waiter is holding a lock. Improving I/O response time often improves lock wait time and then overall file server performance.

File Op Queuing: There are certain events that the DFS file server and client monitor. One event is file operation queuing for SMB clients. The SMB protocol exporter queues file write operations and truncation operations on an internal file handle. This is done to ensure client requests are presented to the physical file system in order. This queuing does not introduce additional serialization since the physical file system locks the file in write mode when performing a write operation. However, file operation queue waits are part of the overall server response time. Hence, the locking and serialization diagnostics provide an indication of file operation queuing. This queue time is directly related to physical file system performance related to disk response times and the efficiency of physical file system file write algorithms.

Diagnostics: The following sample output is from the LOCK report.

```
                                     Locking Statistics

Lock Obtains:        587193954    Fastpath:  577430213  (98.3%)
Lock Releases:       658325735    Fastpath:  649381697  (98.6%)
Lock Upgrades:               0    Fastpath:          0  (0.0%)
Lock Downgrades:        140400    Fastpath:     140400  (100.0%)
Non-block attempts:     533426    Success:      529193  (99.2%)
Work units run:       71131781    Sync:       70996757  (99.8%)

Untimed sleeps:      16766   Timed Sleeps:        4769   Wakeups:     443450

Total waits for locks:                         8900506
Percent of lock obtains/upgrades requiring wait: 1.5%
Average lock wait time:          0.552 (msecs)

Total monitored sleeps:                           7048
Average monitored sleep time:          5.347 (msecs)
```

```
                Top 15 Most Highly Contended Locks
Thread Wait     Async Disp.     Pct.      Description
-----------     -----------     -----     -----------
    4977280               0     55.1%     LFS vnode main lock
    1959305               0     21.7%     LFS main transaction lock
    1272775               0     14.1%     LFS main equivalence class lock
     254672               0      2.8%     SMB time of day services lock
     154204           79264      2.6%     SMB AS queue lock
      81346               0      0.9%     LFS log map lock
      34953           33162      0.8%     SMB TKC global file LRU and free list lock
      42055               0      0.5%     LFS volume handle lock
      24416               0      0.3%     LFS buffer lock
      18548               0      0.2%     VM cache all file lock
          0           17195      0.2%     OSI Global queue of threads waiting for locks
      14504               0      0.2%     LFS allocation handle lock
      12283               0      0.1%     LFS vnode lock
      11319               0      0.1%     SMB CS handle table lock
       8014               0      0.1%     LFS file zero lock


                Top 10 Most Common Thread Sleeps
Thread Wait     Pct.      Description
-----------     -----     -----------
       7048    100.0%     SMB CS File Operation Queue Wait
          0     0.0%     VM Page Wait
          0     0.0%     VM File Wait
          0     0.0%     VM Page Reclaim Wait
          0     0.0%     SMB TKC Tkset Revoke Wait for Tkset Holds
          0     0.0%     SMB TKC Tkset Wait For Pending Revoke
          0     0.0%     SMB TKC Get Token Wait
          0     0.0%     SMB TKC Open-in-progress Wait
          0     0.0%     SMB TKC Callback Response Wait
          0     0.0%     SMB CS Oplock Break Read-Raw Wait
```

The important fields from the LOCK report are the `Total waits for locks`, `Average lock wait time`, `Total monitored sleeps`, `Average monitored sleep time`. These fields indicate average amount of time a thread processing a RPC or SMB request must wait on a lock or for access to a shared resource. In the previous example the `SMB CS File Operation Queue Wait` is not too large and therefore the disk performance and the file system performance were good. File operation queue waits occur more frequently and the average wait time is longer when a disk is overloaded and occurs infrequently when the disk is not a bottleneck.

Storage Usage

The DFS/SMB file server and client provides a command that reports DFS storage usage of the runtime heap. Storage DFS/SMB allocates directly from base z/OS storage subpools. Therefore all DFS allocated storage is shown with the exception of storage allocated for thread stacks. There is no way for DFS/SMB to track that information, though there are LE runtime options that shows you that information.

Diagnostics: The following sample output is from the STORAGE report.

```
                            OSI Storage Statistics
                            ----------------------

 Current allocated storage size: 49515778 (48355K) (47M)
 Number of storage allocations: 722977
 Number of storage frees: 721684

                            MVS Obtained Storage Statistics
                            -------------------------------

 Current allocated storage size above 16M line: 303387036 (296276K) (289M)
 Number of storage allocations above 16M line: 0
 Number of storage frees above 16M line: 0

 Current allocated storage size below 16M line: 1092 (1K) (0M)
 Number of storage allocations below 16M line: 0
 Number of storage frees below 16M line: 0

                 TCB Owned Storage
                 -----------------
 Lock storage allocations: 157
 Lock storage allocated:   11304
 Non-lock storage allocations: 804
 Non-lock storage allocated:   2317976
```

The `OSI Storage Statistics` indicates current DFS/SMB program use of heap storage. MVS and TCB owned storage statistics indicate the number of bytes allocated directly from z/OS storage subpools.

DFS Client

DFS client performance is tied to its cache performance and the performance of DCE RPC.

Workload and Threading

The DFS client has a number of caches to reduce RPC rates to file servers. DFS/SMB clients are called on the same task (TCB) as the OMVS user, and then runs in cross memory mode. The DFS/SMB client attempts to process the request fully on the user's TCB, but in some cases, when an RPC is needed, the request needs to be switched to a DFS/SMB client task to perform the RPC. There is a pool of service threads that can be controlled by the administrator that are used when a task switch is necessary to complete the processing of a request. Task switches are expensive, especially given the request rate to a production DFS/SMB client. The more often the data resides in a DFS client cache the more likely a task switch is not needed. Sufficient service threads need to be made available to process requests that could not be satisfied on the user's TCB to ensure that requests are not waiting in the service thread request queue (which results in larger response times).

Tuning Options: The administrator can set the size of the DFS/SMB client service thread pool with the following environment variable:

_IOE_CM_REQUEST_THREADS
This is the number of service threads available to process requests that could not be fully processed on the user's task. The default number is 10.

Diagnostics: The following sample output is from the CMSERV report. The CMSERV report provides details on the overall client workload, the client's average request response time, and the performance of the service thread pool.

```
                    DFS/MVS Client External Calls
                    -----------------------------
Operation            Count       Fastpath     % Fast       Avg Time
---------         ----------    ----------    ------      ----------
cm_opens             1010          992          98%          0.030
cm_closes            1010          719          71%         12.655
cm_reads             4061         4061         100%          0.144
cm_writes             851          849         100%          0.412
cm_ioctls               5            5         100%          0.010
cm_getattrs          1660         1660         100%          0.019
cm_setattrs             0            0           0%          0.000
cm_accesses            46           46         100%          0.015
cm_lookups           5033         4459          89%          1.695
cm_creates            193            0           0%         18.193
cm_removes            212            0           0%         35.365
cm_links                5            0           0%          5.904
cm_renames             51            0           0%          8.545
cm_mkdirs              35            0           0%         53.043
cm_rmdirs              35            0           0%         13.081
cm_readdirs           154          104          68%          1.700
cm_symlinks            20            0           0%         13.665
cm_readlinks           25            0           0%          0.213
cm_fsyncs               0            0           0%          0.000
cm_truncs               0            0           0%          0.000
cm_lockctls             0            0           0%          0.000
cm_inactives          247          247         100%          0.006
cm_recoverys            0            0           0%          0.000
---------         ----------    ----------    ------      ----------
*TOTALS*            14653        13142          90%          2.501

Number of service threads: 5 (stacksize=48K)
Requests: 1517    Queued: 4 (0.3%)
```

This example shows that most requests were satisfied directly on the user's task with no task switch necessary (`Fastpath` and `%Fast` columns show how often a task switch is avoided).

The `Number of service threads` indicates the number of defined threads and the amount of storage required for a client service thread. The `Requests` and `Queued` fields indicate how many requests from UNIX Systems Services needed to be switched to a service thread task and how often a request needed to be queued when all service threads were busy. The percentage of requests queued should be small because queue wait time directly increases end user response times.

Client Caches

The DFS/SMB client has a number of caches used to significantly reduce RPC rates to servers. Specifically, the client caches file status information (such as size, permissions), user ACL permissions (authority a user has to a file system object), directory name to local vnode mappings (mapping names to file system vnodes), directory contents, file contents, RPC binding handles, and DFS tokens. The sizes of most of these caches can be controlled by the administrator. The sizes of some are dependent on the settings of others.

File Caching: Server file contents are cached locally in client files that are either stored in memory only (memory cache clients) or are stored in a special purpose LFS aggregate (disk cache). The size of each file will at most equal the client chunk size. This controls the granularity of each file and the amount of file data that the client transmits or receives to remote servers in a single RPC. Note that a single server file is often split and stored in multiple client files in the client disk cache. The administrator can specify the size of the cache either by simply specifying the total size of the cache (the client divides total size by chunk size to obtain number of files) or can specify the number of files that can be cached (the client multiplies the number of files by the chunk size to get total bytes).

The DFS client saves two other files (either in memory or on the special purpose LFS aggregate). The first contains a mapping of each client file to the remote server file and offset within the remote server file.

The other file contains cached fileset information. The mapping of local client files to remote server files is called the **CacheItems** file and this file is essentially an array of mappings. The client saves in memory (if a disk cache is used) a number of these mappings for faster access. These mappings are called a *dcache*.

Status Caching: The client caches the status, user file permissions and DFS tokens for files to reduce RPC rates to servers. The number of files whose status can be cached can be set by the administrator. The size of a status cache entry is shown in the CMCACHE report.

Directory Caching: The contents of directories are cached if the raw directory size is 8K or less. The administrator can control how much memory is used to cache directory contents.

Name Lookup Cache: The lookup operation is the most common operation performed by applications or by the kernel. The client caches a mapping of parent directory/object name/vnode mappings that allows a lookup to be performed without the need of an RPC. Furthermore, the client caches negative lookups. That is, the client also remembers names that do not exist in directories because at times the same name is looked up repeatedly. These actions significantly reduce RPC rates. The administrator can control the size of the lookup cache. The size of each name mapping entry is shown in the CMCACHE report.

Tuning Options: The following IOEDFSD startup parameters control the sizes of the various caches.

-blocks Number of 1K blocks in the file cache.

-files Number of files to cache. This is the number of files used for disk caching. It is not relevant for memory cache clients.

-stat This sets the size of the status cache. The default is 300.

-memcache
This parameter indicates if a memory cache is to be used instead of a disk cache. The default is disk cache.

-dcache The number of disk cache file descriptors stored in memory. As mentioned above, the client stores server file contents in local files on a special purpose LFS aggregate for disk cache. The cache items file lists each client file, what server file, and what portion of that file the local client file stores. By caching the descriptors in memory, the cache items file needs to be read less often. It is not recommended that this option be set by an administrator.

-chunksize
This controls the size of each client file used to store remote server files and how much data can be read or written in a single RPC. A large chunk size is preferred. The default is 64K for disk cache and 8K for memory cache.

-namecachesize
This is the size of the name lookup cache. The default is 256.

The following environment variable controls client caching:

_IOE_CM_DIRCACHE_SIZE
This controls the amount of storage the DFS client uses to store directory contents. The default is 1M.

Diagnostics: The following sample output is from the CMCACHE report. It shows the performance of the client caches.

```
                    DFS Client Cache Performance
                    ----------------------------
Status Cache Statistics
-----------------------
Total Entries          300     Entry Size 376 bytes.
Total Searches        4226     Hits       3964 (Ratio 93.80%)
Total Flushes            0

Data Cache Statistics
---------------------
Total Files            512     Chunk Size 64K bytes.
Total Entries          512
Total Searches        5588     Hits       5150 (Ratio 92.16%)
Total Recycles           0
Total Bytes Read From Cache           0 (KB)
Total Bytes Written To Cache          0 (KB)

Directory Name Lookup Statistics
--------------------------------
Total Entries          256     Entry Size 80 bytes.
Total Searches        5589     Hits       4457 (Ratio 79.75%)

Token Management Statistics
---------------------------
Status Tokens Acquired        265     Released        212     Revoked          0
Data Tokens   Acquired        265     Released        212     Revoked          0

Connection Statistics
---------------------
Binding Handles Created         19     Waited        538

Directory Cache Statistics
--------------------------
Total Entries           28     Entry Size 8192 bytes.
Total Reads            114     Hits         64 (Ratio 56.14%)
Reads for directories too large to cache: 0
```

The following sections provide explanations of the CMCACHE report.

Status Cache Statistics: This section shows the number of status cache entries in the cache (`Total Entries`) and the status cache hit ratio (`Hits/Ratio` fields). The hit ratio should be 80% or higher. A miss requires one or more DCE/RPCs to be issued to obtain the status.

Data Cache Statistics: This section shows the number of data cache files (`Total Files`) and the size of each "file" (given by `Chunk Size`). For memory cache, each file is in memory and the total cache storage equals the `Total Files` times the `Chunk Size`. For disk cache clients, not all cache files are stored in memory, but a certain number of cache descriptors, which indicate where the files are on disk are in memory. This is the `Total Entries` fields. The important statistic is the `Total Searches` and the `Hits/Ratio` fields. A high cache hit ratio is recommended because a miss requires an RPC.

Directory Name Lookup Statistics: This section indicates the number of directory lookups and the percent of time those lookups were found in the name lookup cache. The `Total Entries` is the size of the cache and the `Total Searches` and `Hits/Ratio` fields indicate the percentage of time the name was found in the cache. A high hit ratio is recommended because a miss requires one or more DCE/RPCs made to the file server.

Directory Cache Statistics: This section shows the current cache size and indicates the number of directory buffer reads and the percent of time the directory data was cached at the client, (`Total Reads` and `Hits/Ratio`,respectively). A high hit ratio is recommended because a miss requires a DCE/RPC. In this case, the directories that are larger than 8K are not cached by the client and always result in remote RPC calls. If your installation has many directories larger than 8K your hit ratio is lower and increasing directory cache size does not help. The field `Reads for directories too large to cache` indicates how often this occurs. An acceptable directory cache hit ratio would be at least 60% or higher.

RPC I/O Performance

DFS makes DCE RPC calls whenever it cannot satisfy a request locally. Performance of the DFS client is often directly related to the performance of the DFS client cache and the response times of the DCE RPC requests (which includes remote server response time). If the client can satisfy a request locally, without any RPC, then the client response time is fast (less than 1 millisecond). However, when an RPC is required, response time is much larger.

Local Server: If the server resides on the same system as the client then the client bypasses DCE to make the RPC call and makes the call directly using a local UNIX socket. This eliminates any wire transmission and significantly reduces pathlengths. Therefore, if the server is local, the response time is often dominated by the server response time itself.

Background Threads: When performing file read/write requests, the client uses a pool of background I/O threads to perform read-ahead and write-behind. By processing large file read/write requests on separate threads in parallel with user application processing, large file read/write performance is improved substantially. If the pool of threads is busy then a read/write request will either be queued waiting for a background thread to become free, which adds queue wait time to the response time, or will be performed by the user application task synchronously. The administrator can set the size of the background I/O thread pool and monitors performance of the thread pool.

Tuning Options: The only tuning option is the size of the background I/O thread pool which is specified in the DFS client startup parameters.

-mainprocs
This sets the number of background I/O threads. The default is 5.

Diagnostics: The following sample out is from the AFS4INT report. This report shows performance of the background thread pool, the count and average response times of RPCs made to the local server on the same system as the client, and the count and average response times of RPCs made to remote servers.

```
                         AFS4 Interface RPC Call Counts

             Average RPC Time (msecs)
RPC Call                       Count   Response Time
------------------------  ----------   -------------
AFS_Lookup                         1          15.447
AFS_LookupRoot                     2         168.591
AFS_FetchStatus                    1          14.589
AFS_StoreStatus                    1          39.415
AFS_FetchData                      1          14.023
AFS_StoreData                      0           0.000
AFS_CreateFile                     0           0.000
AFS_RemoveFile                     0           0.000
AFS_Rename                         0           0.000
AFS_Readdir                        1          47.987
AFS_MakeDir                        0           0.000
AFS_RemoveDir                      0           0.000
AFS_Link                           0           0.000
AFS_Symlink                        0           0.000
AFS_FetchACL                       0           0.000
AFS_StoreACL                       0           0.000
AFS_GetToken                       1          14.714
AFS_ReleaseTokens                  0           0.000
AFS_SetContext                    20          40.866
AFS_SetParams                      1          12.683
AFS_GetTime                        0           0.000
AFS_MakeMountPoint                 0           0.000
AFS_BulkKeepAlive                  0           0.000
AFS_BulkFetchVV                    0           0.000
AFS_GetStatistics                  0           0.000
AFS_BulkFetchStatus                0           0.000

Total RPC calls         29
Average Response Time per RPC     45.288

Derived Fetch/Store Statistics:

Total Fetched Bytes 119K
Total Stored Bytes 0K

                   AFS4 Interface LOCAL RPC Call Counts

       Average Local RPC Time (msecs)
RPC Call                       Count   Response Time
------------------------  ----------   -------------
L.AFS_Lookup                     566           5.725
L.AFS_LookupRoot                   0           0.000
L.AFS_FetchStatus                  1           0.827
L.AFS_StoreStatus                  0           0.000
L.AFS_FetchData                    7          35.541
L.AFS_StoreData                  545          48.077
L.AFS_CreateFile                 193          16.510
L.AFS_RemoveFile                 212          34.112
L.AFS_Rename                      51           7.959
L.AFS_Readdir                     49           3.341
L.AFS_MakeDir                     35          52.045
L.AFS_RemoveDir                   35          12.435
L.AFS_Link                         5           3.351
L.AFS_Symlink                     20          12.995
L.AFS_FetchACL                     0           0.000
L.AFS_StoreACL                     0           0.000
L.AFS_GetToken                     6          10.953
L.AFS_ReleaseTokens                8           1.377
L.AFS_BulkFetchStatus              0           0.000


Total Local RPC calls       1733
Average Response Time per LOCAL RPC     24.980
```

```
          Background Thread Statistics
          ----------------------------
Number of threads:   16   (stack size=32K)
Number of background requests              493
Number of background requests queued        24 (4.8%)
Number of background requests failed         5
Number of background requests waited         0
```

The following describes the sections of the AFS4INT report.

AFS4 Interface RPC Call Counts: This reports shows the average response time of remote procedure calls. Two tables are shown, one for remote server calls and one for calls made to a file server that reside on the same physical machine as the client (if one exists). Normally, local RPCs are much faster than remote RPCs since no wire transmission is made. The response time of a remote procedure call is not controllable by the client. Network performance and file server performance determines that. This information is shown merely so an administrator can determine what affect RPC response time is having on local user response time. The administrator can only set larger caches in order to reduce RPC calls made by using DFS tuning options. Faster network hardware improves RPC response times and network bottlenecks would be reflected in these numbers.

Background Thread Statistics: The `Number of background requests` and `Number of background requests queued` indicate how many times a file read/write request was sent to a background I/O thread for processing and how often the request had to be queued. It is recommended, in this case, to have a low amount of queueing because background queue waits contribute to increased end user response time and RPC requests are themselves waiting for RPC completion (which are long term waits).

Disk Caching Considerations

All of the information described above is relevant for clients that use disk caching rather than memory caching. However, there are additional considerations when using a disk cache. File data is stored on a utility LFS aggregate, then the performance of LFS is very relevant to the performance of the client disk cache. Also, because file data is stored in the virtual memory cache associated with LFS, the performance of the VM cache is important to disk cache performance. The description of LFS and VM cache performance in the **Server Tuning Guidelines** section is relevant for clients with disk cache. The local user client access hits the VM cache and LFS much harder than a server will because there are no RPC delays between requests. This makes performance of these components much more critical.

The performance of the VM cache is critical for client disk cache performance. The administrator should set the size of the VM cache to ensure a recommended hit ratio of at least 80%. If this is accomplished this allows performance of disk cache clients to approach memory cache clients but with much less memory used. Additionally, the LFS utility aggregate are hit heavily. It is recommended to place this aggregate on a disk that is otherwise unused or has a low amount of activity not related to the LFS utility aggregate. Finally, the size of the LFS metadata cache should be set large enough to ensure at least a 90% hit ratio.

Locking

The DFS client performs most processing on a UNIX System Services user task. This task runs in synchronous cross memory mode. This means that standard POSIX locking cannot be used and the processing code used for UNIX System Services tasks uses a different lock package than the rest of the DFS product. Appended to the LOCK report that is shown for the server is a report indicating lock waits and average lock wait time for this code.

The following sample output is from the LOCK report.

```
            DFS XMEM Lock Statistics
            ------------------------
TREAD lock waits:           0    avg. time        0.000 (msecs)
TWRITE lock waits:          0    avg. time        0.000 (msecs)
TLOCK lock waits:           0    avg. time        0.000 (msecs)
Sleeps:                    17    avg. time       10.961 (msecs)
-----------------    ----------            -------------
Total waits                17    avg. time       10.961 (msecs)
Wakeups:                    0    Wait Failures:          0
```

Some Final Considerations

Production sized distributed file system performance is affected by many factors. The network and disk hardware are a large portion of end user response time. A bottleneck in either the network or the disk systems will reduce end user response time. Tuning DFS properly places less of a strain on both the network and disk systems.

Eliminating a bottleneck might uncover other bottlenecks. For example, assume that a particular file system environment has a severe network bottleneck. If the network bottleneck is fixed you might actually notice that DFS response times increase. The reason is because the faster the RPCs or SMBs arrive to the server the longer it takes the server to process them. When the network is very slow, the server often has little work to do and therefore has threads available to process work. It is also less likely that a disk I/O wait will occur. When work is arriving quickly to a server (particularly from the same client) there is a good chance of lock contention and CPU waits. But, there is a greater chance of disk I/O waits and the disk I/O wait time increasing. This is because the more work a server is processing, the more likely that when a thread needs to wait on an I/O to disk there are other disk I/Os ahead of it on the device queue. This makes response times longer because not only is the thread waiting for the disk, but it is waiting for other I/Os to the disk to complete first. This is seen in a DCE/RPC workload. If the client and server reside on the same machine, RPCs are bypassed. If you write large files from a DCE/DFS client to a DFS server on the same machine, from the end user's point of view, response time is much better than writing large files from another client located on a different physical machine. If you view the DFS statistics for each run you will see that DFS response times are greater for the local user case. The elimination of the network overhead makes requests arrive at the DFS server sooner and when a disk I/O wait occurs it is likely other I/Os are ahead of it in the queue and lock contention occurs more frequently.

Therefore, eliminating one bottleneck might uncover another. The faster the network, the more important DFS server tuning becomes.

Appendix E. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described 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:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States or other countries or both:

BookManager
Library Reader
OS/2
RACF
IBM
Language Environment
OS/390
Resource Link
IBMLink
MVS
MVS/ESA
z/OS

Microsoft Windows, Windows NT, and the Windows logo are trademarks of of Microsoft Corporation, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Bibliography

This section lists and provides a brief description of each publication in the Distributed File Service library. Also listed are publications from the DCE library that may be useful.

Distributed File Service Publications

This section lists and provides a brief description of each publication in the Distributed File Service library.

Administration

- *z/OS Distributed File Service Customization*, SC24-5916

  This book helps system and network administrators configure the Distributed File Service.

- *z/OS Distributed File Service DFS Administration*, SC24-5915

  This book introduces the DFS concepts to system and network administrators and provides an in-depth understanding of Distributed File Service, its uses and benefits. This book also provides reference information for the commands and files used by system and network administrators to work with Distributed File Service

- *z/OS Distributed File Service SMB Administration*, SC24-5918

  This books provides guidance and reference information for system and network administrators to use when they work with the Server Message Block (SMB) support of the Distributed File Service base element of z/OS. SMB is a protocol for remote file/print access used by Windows.

Reference

- *z/OS Distributed File Service Messages and Codes*, SC24-5917

  This book provides detailed explanations and recovery actions for the messages, status codes, and exception codes issued by the Distributed File Service.

DCE Publications

This section lists and provides a brief description of each publication in the DCE library.

Overview

- *z/OS DCE Introduction*, GC24-5911

  This book introduces DCE. Whether you are a system manager, technical planner, z/OS system programmer, or application programmer, it will help you understand DCE, and evaluate the uses and benefits of including DCE as part of your information processing environment.

Planning

- *z/OS DCE Planning*, GC24-5913

  This book helps you plan for the organization and installation of DCE. It discusses the benefits of distributed computing in general, and describes how to develop plans for a distributed system in a DCE environment.

Administration

- *z/OS DCE Configuring and Getting Started*, SC24-5910

  This book helps system and network administrators configure DCE.

- *z/OS DCE Administration Guide*, SC24-5904

  This book helps system and network administrators understand DCE, and tells how to administer it from the batch, TSO, and shell environments.

- *z/OS DCE Command Reference*, SC24-5909

  This book provides reference information for the commands that system and network administrators use to work with DCE.

- *z/OS DCE User's Guide*, SC24-5914

  This book describes how to use DCE to work with your user account, use the directory service, work with namespaces, and change access to objects that you own.

Application Development

- *z/OS DCE Application Development Guide: Introduction and Style*, SC24-5907

  This book assists you in designing, writing, compiling, linking, and running distributed applications in DCE.
- *z/OS DCE Application Development Guide: Core Components*, SC24-5905

  This book assists programmers in developing applications using application facilities, threads, remote procedure calls, distributed time service, and security service.
- *z/OS DCE Application Development Guide: Directory Services*, SC24-5906

  This book describes the z/OS DCE directory service and assists programmers in developing applications for the cell directory service and the global directory service.
- *z/OS DCE Application Development Reference*, SC24-5908

  This book explains the DCE Application Program Interfaces (APIs) that you can use to write distributed applications on DCE.

Reference

- *z/OS DCE Messages and Codes*, SC24-5912

  This book provides detailed explanations and recovery actions for the messages, status codes, and exception codes issued by DCE.

SecureWay Security Server Publications

This section lists and provides a brief description of books in the SecureWay Security Server library that may be needed for the DCE SecureWay Security Server and for RACF interoperability.

- *z/OS SecureWay Security Server DCE Overview*, GC24-5921

  This book describes the DCE security server and provides a road map for DCE security server information in the DCE library.
- *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683.

  This book explains RACF concepts and describes how to plan for and implement RACF.
- *z/OS SecureWay Security Server RACF Command Language Reference*, SA22-7687.

  This book contains the functions and syntax of all the RACF commands.

Index

Readers' Comments

z/OS
Distributed File Service
Customization

Publication No. SC24-5916-00

You may use this form to report errors, to suggest improvements, or to express your opinion on the appearance, organization, or completeness of this book.

Date: ________________________________

____

____

____

____

____

____

____

____

____

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note

Report system problems to your IBM representative or the IBM branch office serving you.
U.S. customers can order publications by calling the IBM Software Manufacturing Solutions at **1-800-879-2755**.

In addition to using this postage-paid form, you may send your comments by:

FAX	1-607-752-2327	Internet	pubrcf@vnet.ibm.com
IBMLink	GDLVME(PUBRCF)		

Would you like a reply? ___**YES** ___ **NO** If yes, please tell us the type of response you prefer.

___ Electronic address: ____

___ FAX number: ________________________________

___ Mail: (Please fill in your name and address below.)

____
Name

____
Company or Organization

____
Phone No.

____
Address

____

____

Readers' Comments
SC24-5916-00

Cut or Fold
Along Line

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

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

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

POSTAGE WILL BE PAID BY ADDRESSEE

Department G60
International Business Machines Corporation
Information Development
1701 North Street
ENDICOTT NY 13760-5553

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

Cut or Fold
Along Line

Program Number: 5694-A01

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

SC24-5916-00

Spine information:

IBM z/OS Distributed File Service DFS/SMB Customization