Open System Services Management and Operations Guide

Abstract

This guide describes how to manage and operate the HP NonStop™ operating system Open System Services (OSS) environment.

Product Version

OSS Monitor T8622H02, T8622G12

Supported Release Version Updates (RVUs)

This guide supports G06.27 and all subsequent G-series RVUs and H06.06 and all subsequent H-series RVUs until otherwise indicated by its replacement publication.

Part Number	Published
527191-004	May 2006

Document History

Part Number	Product Version	Published
527191-002	OSS Monitor T8622H01, T8622G11	July 2005
527191-003	OSS Monitor T8622H01, T8622G12	September 2005
527191-004	OSS Monitor T8622H02, T8622G12	May 2006

Open System Services Management and Operations Guide

2. Operating the OSS Environment

3. Understanding the OSS File System

4. Managing Servers

4. Managing Servers (continued)

5. Managing Filesets

5. Managing Filesets (continued)

6. Managing OSS Files

7. Managing Terminal Access

8. Managing Security

9. Managing With the Shell

10. Managing OSS Devices

11. Managing Problems

12. Open System Services Monitor

Figures

Figures (continued)

Tables

Tables (continued)

What's New in This Guide

Guide Information

Abstract

This guide describes how to manage and operate the HP NonStop™ operating system Open System Services (OSS) environment.

Product Version

OSS Monitor T8622H02, T8622G12

Supported Release Version Updates (RVUs)

This guide supports G06.27 and all subsequent G-series RVUs and H06.06 and all subsequent H-series RVUs until otherwise indicated by its replacement publication.

Part Number	Published
527191-004	May 2006

Document History

Part Number	Product Version	Published
527191-002	OSS Monitor T8622H01, T8622G11	July 2005
527191-003	OSS Monitor T8622H01, T8622G12	September 2005
527191-004	OSS Monitor T8622H02, T8622G12	May 2006

New and Changed Information

The product version for the OSS monitor changed at the H06.05 RVU from T8622H01 to T8622H02.

The H06.06 RVU adds support for OSS files larger than 2 gigabytes. Descriptions of new APIs and new file size limits apply to systems running H06.06 and later H-series RVUs only. They do not apply to systems running G-series RVUs.

Changes to this edition of the manual are:

- A discussion of OSS file sizes and underlying file formats has been added to Relating OSS Files, Filesets, and Disk Volumes on page 3-8.
- The new file size limit for a pax archive, 8 gigabytes, for H06.06 and later RVUs has been added to Backing Up and Restoring OSS Files on page 6-11.
- Information about the -W spl flag has been added to Specifying a Default Printer on page 10-2.

- In Appendix A, Messages, messages that are described in the *Operator Messages Manual* were removed. For descriptions of these messages, see the *Operator Messages Manual*.
- In Appendix D, Falling Back to a Previous Release Version Update, information about falling back to release version updates (RVUs) prior to G06.12 has been removed.
- In Appendix E, Environment Limits, a description of file size formats and file size limits has been added.

About This Guide

The HP NonStop operating system Open System Services (OSS) environment enables users on HP NonStop servers to integrate an operating system similar to the UNIX operating system into their work environment. You manage and operate the OSS environment primarily from the Guardian environment.

What This Guide Is About

This guide describes how to manage and operate the OSS environment. It describes only what is unique to the OSS environment, its management, and its operation. This guide does not describe Guardian programs and commands that have been enhanced to accommodate the OSS environment.

Who Should Read This Guide?

This guide is for system managers and operators of the OSS environment. System managers set policy, perform system and shell configuration, and perform operations that require the use of the super ID (255,255 in the Guardian environment, 65535 in the OSS environment) or membership in the super group (255,*nnn*), such as managing users and groups. System operators carry out system support and maintenance as described in the *NonStop S-Series Operations Guide.*

Because Open System Services is managed and operated primarily through the Guardian environment, the audience for this guide consists of Guardian system managers and operators and those who have acquired proficiency at managing or operating a Guardian system. At the very least:

- System managers should complete the HP NonStop system education and training course *NonStop S-Series Problem Management* or have equivalent work experience, as well as be proficient at using a security product on a NonStop S-series or NonStop NS-series server, such as the Safeguard product. System managers should have a basic knowledge of the UNIX operating system and the Korn shell.
- Operators should complete the HP NonStop system education and training courses *NonStop S-Series Configuration and Change Management* and *NonStop S-Series Production Management* or have equivalent work experience.

What This Guide Does Not Cover

This guide does not describe general Guardian management or operation procedures. It also does not describe how to use Guardian programs that you would use to manage or operate Guardian systems, even though these programs could be applied to systems that have the OSS environment running on them.

What Is in This Guide?

This guide contains information and procedures for managing and operating the OSS environment. It is divided into the following sections:

- Section 1, Introducing Open System Services, presents an overview of how to manage and operate the OSS environment. It includes a task table that refers you to the appropriate documentation for specific kinds of tasks.
- Section 2, Operating the OSS Environment, describes how to operate the OSS environment. This section includes procedures for managing subsystems and processes, maintaining files needed in the OSS environment, and performing backups and restores.
- Section 3, Understanding the OSS File System, describes the structure and naming conventions within the OSS file system.
- Section 4, Managing Servers, describes how to manage server processes and functions using OSS Monitor SCF commands and, where necessary, HP Tandem Advanced Command Language (TACL) commands.
- Section 5, Managing Filesets, describes how to manage filesets using OSS Monitor SCF commands and (where necessary) TACL commands, including:
  - How to use the OSS Monitor SCF DIAGNOSE FILESET command, which invokes the FSCK fileset integrity checker.
  - How to use the Guardian Catalog Volume Tool (CVT) utility, which enables you to manage fileset catalogs.
- Section 6, Managing OSS Files, describes the actions necessary to maintain data files within the OSS file system.
- Section 7, Managing Terminal Access, describes how to configure user access to the OSS environment for terminal users.
- Section 8, Managing Security, summarizes security features, user and user-group characteristics, and means for protecting your system. It also describes how to specify initial working directories and initial programs.
- Section 9, Managing With the Shell, describes how to manage objects in the OSS environment using the OSS version of the Korn shell. The shell is the command interface to the OSS environment.
- Section 10, Managing OSS Devices, discusses OSS devices and describes how to configure printers in the OSS environment.
- Section 11, Managing Problems, briefly reviews problem-reporting procedures in the OSS environment.
- Section 12, Open System Services Monitor, provides a reference to the Subsystem Control Facility (SCF) product module for the OSS Monitor and the SCF commands you can use to manage the OSS environment.

- Appendix A, Messages, describes OSS EasySetup, CVT, FSCK, OSS message-queue server, OSS Monitor, and OSS subsystem messages.
- Appendix B, Manually Setting Up an OSS Environment, summarizes the procedures to configure and start a new OSS environment.
- Appendix C, OSS Management Utilities, describes the OSSTTY and OSS EasySetup product utilities and files.
- Appendix D, Falling Back to a Previous Release Version Update, provides information necessary to fall back to earlier releases of OSS software.
- Appendix E, Environment Limits, provides guidance on current maximum values for various OSS features.

Related Reading, Training, and Services

This subsection describes prerequisite and additional reading and training for this guide. Prerequisite reading and training are items that a reader of this guide must complete before using this guide.

The following subsections briefly provide a context for each publication.

Prerequisite Reading and Training

Users of the *Open System Services Management and Operations Guide* must be familiar with the *Open System Services User's Guide* and know how to use the *Open System Services Shell and Utilities Reference Manual*.

For OSS installation instructions using automated setup tools, see the *Open System Services Installation Guide*.

For system operators, the prerequisite reading and training are:

- *Guardian User's Guide*
- The Independent Study Program for *Basic Tandem Operator Tasks*
- These HP NonStop system education and training courses:
  - *Concepts and Facilities*
  - *NonStop S-Series Problem Management*
  - *Open System Services (OSS) Operations and Management*

For system managers, the prerequisite reading and training are the same as for system operators, plus:

- *Introduction to NonStop S-Series Servers*
- *Introduction to NonStop Operations Management*
- *Safeguard Administrator's Manual*
- *Safeguard Audit Service Manual*
- *Safeguard Reference Manual*

- *SCF Reference Manual for G-Series RVUs*
- *SCF Reference Manual for H-Series RVUs*
- *Security Management Guide*
- *Software Internationalization Guide*
- The courses *NonStop S-Series Configuration and Change Management* and *NonStop S-Series Production Management*

Additional Reading

Problem reporting often involves using the Event Management Service (EMS) and the Guardian VPROC utility. These topics are discussed in the following publications:

- *EMS Manual*
- *Guardian User's Guide*

Any publication about system configuration management, the print spooler, the Korn shell, the UNIX operating system, or UNIX system administration is helpful. The following subsections list some publications on UNIX system administration, the Korn shell, and UNIX security. The lists are not exhaustive.

Installation

- *DSM/SCM User's Guide*

Configuration

- *DNS Configuration and Management Manual*
- *SCF Reference Manual for the Kernel Subsystem*
- *Storage Subsystem Configuration and Management Manual*
- *TCP/IP Configuration and Management Manual*
- *TCP/IP (Parallel Library) Configuration and Management Manual*
- *TCP/IPv6 Configuration and Management Manual*
- *Telserv Manual*

Periodic Task Management

- *Backup and Restore 2.0 Manual*
- *Guardian Disk and Tape Utilities Reference Manual*
- *NetBatch Manual*
- *NetBatch Plus Reference Manual*

Print Spooler

- *AWAN 3883/4/5 Access Server Configuration and Management Manual*
- *Asynchronous Terminals and Printer Processes Configuration and Management Manual*

- *Spooler Utilities Reference Manual*
- *System Generation Manual for G-Series RVUs*
- *Telserv Manual*
- *WAN Subsystem Configuration and Management Manual*

UNIX System Administration

- Frisch, Æleen. *Essential System Administration, Second Edition*. O'Reilly & Associates, Inc. Sebastopol: 1995
- Nemeth, Evi, Garth Snyder, Scott Seebass, and Trent R. Hein. *UNIX® System Administration Handbook, Second Edition.* Prentice-Hall. Englewood Cliffs: 1995
- Reiss, Levi and Joseph Radin. *UNIX System Administration Guide*. Osborne McGraw-Hill. Berkeley: 1993

Korn Shell

- Rosenblatt, Bill. *Learning the Korn Shell*. O'Reilly & Associates, Inc. Sebastopol: 1993
- Valley, John. *UNIX® Desktop Guide to the Korn Shell.* Hayden Books. Carmel: 1992

UNIX Security

- Arnold, N. Derek. *UNIX Security. A Practical Tutorial*. McGraw-Hill, Inc. New York: 1993
- Farrow, Rick. *UNIX System Security*. Addison-Wesley Publishing Company: Menlo Park: 1991
- Garfinkel, Simson and Gene Spafford. *Practical UNIX & Internet Security, 2nd Edition*. O'Reilly & Associates, Inc. Sebastopol: 1996
- Wood, Patrick and Stephen Kochan. *UNIX System Security*. Hayden Books. Carmel: 1990

Unsupported Utilities

HP is not responsible for the proper functioning of unsupported utilities or facilities and will not respond to product reports about them. Such utilities and facilities include those in the OSS `/bin/unsupported` directory. Use these utilities and facilities at your own risk.

Acknowledgment

The Portable Archive Interchange (pax) utility software was developed by Mark H. Colburn and is sponsored by the USENIX Association.

Copyright (c) 1989 Mark H. Colburn.
All rights reserved.

The pax utility, distributed to HP free of charge, is used to support the tar, cpio, and pax user interfaces furnished with the OSS environment and is documented in this guide and the *Open System Services Shell and Utilities Reference Manual*.

Notation Conventions

The subsections that follow describe:

- Hypertext Links
- General Syntax Notation
- The notation used in message descriptions
- The notation used to indicate changes within this guide

Hypertext Links

Blue underline is used to indicate a hypertext link within text. By clicking a passage of text with a blue underline, you are taken to the location described. For example:

- This is a hyperlink to Notation for Messages on page xxii.

General Syntax Notation

The following list summarizes the notation conventions for syntax presentation in this manual.

UPPERCASE LETTERS. Uppercase letters indicate keywords and reserved words; enter these items exactly as shown. Items not enclosed in brackets are required. For example:

```
MAXATTACH
```

lowercase italic letters. Lowercase italic letters indicate variable items that you supply. Items not enclosed in brackets are required. For example:

```
file-name
```

computer type. Computer type letters within text indicate C and Open System Services (OSS) keywords and reserved words; enter these items exactly as shown. Items not enclosed in brackets are required. For example:

```
myfile.c
```

italic computer type. *Italic computer type* letters within text indicate C and Open System Services (OSS) variable items that you supply. Items not enclosed in brackets are required. For example:

```
pathname
```

[ ] Brackets. Brackets enclose optional syntax items. For example:

```
TERM [\system-name.]$terminal-name
```

```
INT[ERRUPTS]
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list may be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
FC [ num  ]
   [ -num ]
   [ text ]
```

```
K [ X | D ] address
```

{ } Braces. A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list may be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LISTOPENS PROCESS { $appl-mgr-name }
                  { $process-name  }
```

```
ALLOWSU { ON | OFF }
```

| Vertical Line. A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
INSPECT { OFF | ON | SAVEABEND }
```

… Ellipsis. An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
M address [ , new-value ]...
```

```
[ - ] {0|1|2|3|4|5|6|7|8|9}...
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
"s-char..."
```

Punctuation. Parentheses, commas, semicolons, and other symbols not previously described must be entered as shown. For example:

```
error := NEXTFILENAME ( file-name ) ;
```

```
LISTOPENS SU $process-name.#su-name
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must enter as shown. For example:

```
"[" repetition-constant-list "]"
```

Item Spacing. Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
CALL STEPMOM ( process-id ) ;
```

If there is no space between two items, spaces are not permitted. In the following example, there are no spaces permitted between the period and any other items:

```
$process-name.#su-name
```

Line Spacing. If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
ALTER [ / OUT file-spec / ] LINE

   [ , attribute-spec ]...
```

Notation for Messages

The following list summarizes the notation conventions for the presentation of displayed messages in this manual.

Bold Text. Bold text in an example indicates user input entered at the terminal. For example:

```
ENTER RUN CODE
?123
CODE RECEIVED:      123.00
```

The user must press the Return key after typing the input.

Nonitalic text. Nonitalic letters, numbers, and punctuation indicate text that is displayed or returned exactly as shown. For example:

```
Backup Up.
```

lowercase italic letters. Lowercase italic letters indicate variable items whose values are displayed or returned. For example:

```
p-register
process-name
```

[ ] Brackets. Brackets enclose items that are sometimes, but not always, displayed. For example:

```
Event number = number [ Subject = first-subject-value ]
```

A group of items enclosed in brackets is a list of all possible items that can be displayed, of which one or none might actually be displayed. The items in the list might be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
proc-name trapped [ in SQL | in SQL file system ]
```

{ } Braces. A group of items enclosed in braces is a list of all possible items that can be displayed, of which one is actually displayed. The items in the list might be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
obj-type obj-name state changed to state, caused by
{ Object | Operator | Service }
```

```
process-name State changed from old-objstate to objstate
{ Operator Request. }
{ Unknown.          }
```

| Vertical Line. A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
Transfer status: { OK | Failed }
```

% Percent Sign. A percent sign precedes a number that is not in decimal notation. The % notation precedes an octal number. The %B notation precedes a binary number. The %H notation precedes a hexadecimal number. For example:

```
%005400
%B101111
%H2F
P=%p-register E=%e-register
```

Change Bar Notation

A change bar (as shown to the right of this paragraph) indicates a difference between this edition of this guide and the preceding edition. Change bars highlight new or revised information.

1 Introducing Open System Services

The Open System Services (OSS) environment provides a user and programming interface similar to that of the UNIX operating system. The OSS environment combines the benefits of the UNIX operating system with the features of the HP NonStop operating system.

Open System Services differs from the UNIX operating system in that almost all management and operations activities are performed through Guardian environment commands. The OSS environment is managed in a similar fashion to the Guardian environment.

Note. The term “management” is used in this guide instead of “administration” because “management” is the term used in the Guardian environment.

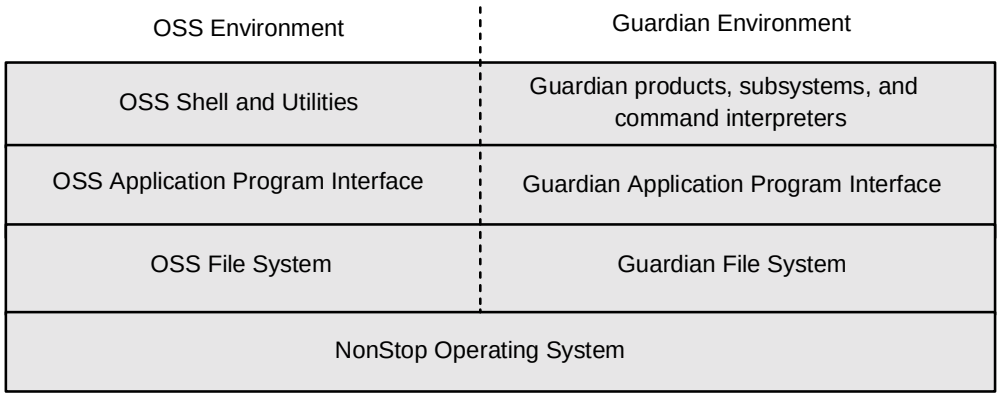
This section presents an overview of OSS management and operations. It briefly describes:

The Operating System Environments

The OSS environment coexists with the Guardian environment on a NonStop S-series or NonStop NS-series system, as shown in Figure 1-1 on page 1-2. Environments are sometimes called “personalities” in other HP documentation.

You use management tools in both environments. You can use features of each environment from the other environment.

Figure 1-1. The Operating System Environments



Management Tools

Your primary management tool is the Subsystem Control Facility (SCF) module for the OSS Monitor, used from the Guardian environment. Reference information for that module appears in Section 12, Open System Services Monitor.

Any Guardian-environment command that you can enter at an HP Tandem Advanced Command Language (TACL) prompt can also be entered from the OSS environment using the `gtacl` command at an OSS shell prompt. See the `gtacl(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for additional information about that command.

You might find it convenient to manage the OSS environment from the OSS environment. Therefore, you should know something about OSS commands and utilities.

Note. Before using an OSS command, read the appropriate reference page either online or in the *Open System Services Shell and Utilities Reference Manual* to make sure that the command behaves in the way you expect. OSS commands conform to the XPG4 standards, but some OSS commands and utilities might have different options and behavior from the version of the UNIX operating system that you are familiar with.

The *Open System Services User's Guide* contains tables that show the approximate correspondence between commands supported in the Guardian environment and the UNIX commands and utilities supported in the OSS environment.

The OSS command set differs from other implementations of UNIX shell commands and utilities. Many OSS commands contain HP extensions that display Guardian environment information or allow actions that do not exist in the UNIX environment.

Management and Operations Tasks

The only OSS management and operations tasks that must occur within the OSS environment are performing backups, configuring printer aliases, and defining default profiles; the rest can occur from the Guardian environment. Table 1-1 lists common OSS management and operations tasks, along with the location of information on performing those tasks.

If a task is not listed, it is a task that affects the NonStop operating system and includes the Guardian environment as well as the OSS environment. For information about performing these tasks, see Guardian documentation.

Many of the general user tasks described in the *Open System Services User's Guide* are also useful for a system manager or operator. See that guide when the Guardian documentation does not describe a task you need to perform.

Table 1-1. Management and Operations Tasks (page 1 of 3)

Task	Subtask	See
Auditing	configuring	ADD FILESET Command on page 12-7 and ALTER FILESET Command on page 12-20
	filesets	Auditing a Fileset on page 5-12, the *Security Management Guide*, and the *Safeguard Audit Service Manual*
	related SCF commands	Section 12, Open System Services Monitor
	shell commands	Auditing of OSS Shell Commands on page 8-26, the *Security Management Guide*, and the *Safeguard Audit Service Manual*
Configuring	printers and printer aliases	Managing Printers in the OSS Environment on page 10-1
	servers	Configuring a Server on page 4-29
	terminal and server access	Section 7, Managing Terminal Access, the *Telserv Manual*, and either the *TCP/IP Configuration and Management Manual* or the *TCP/IPv6 Configuration and Management Manual*
	user access	Managing Users and Groups on page 8-9, How Users Gain Access to the OSS Environment on page 7-1, the *Telserv Manual*, and either the *TCP/IP Configuration and Management Manual* or the *TCP/IPv6 Configuration and Management Manual*
	persistent processes	Starting the OSS Monitor as a Persistent Process on page 2-9, Making OSS Application Processes Persistent with the Kernel Subsystem on page 2-23, and the *SCF Reference Manual for the Kernel Subsystem*

Table 1-1. Management and Operations Tasks (page 2 of 3)

Task	Subtask	See
Installing	reference pages	Appendix B, Manually Setting Up an OSS Environment, Updating the whatis Database Files on page 6-10, OSSSETUP Utility on page C-11, or the *Open System Services Installation Guide*
	software	Appendix B, Manually Setting Up an OSS Environment, Installing New Product Files on page 6-4, or the *Open System Services Installation Guide*
Maintaining	reference pages and the `whatis` database	Updating the whatis Database Files on page 6-10
	HP software files	Removing Obsolete OSS Files and Directories on page 6-9
Managing	interprocess communication (IPC) facilities	Managing OSS Interprocess Communication Facilities on page 2-34, and the *Open System Services Shell and Utilities Reference Manual*
	OSS environment activities	Section 9, Managing With the Shell, and the *Open System Services Shell and Utilities Reference Manual*
	OSS files	Section 6, Managing OSS Files
	OSS filesets	Section 5, Managing Filesets
	OSS fileset catalogs	Managing and Repairing Fileset Catalog Files on page 5-40
	OSS fileset storage pools	Creating a Storage Pool on page 5-6 or Removing a Disk Volume From a Storage-Pool File on page 5-22
	OSS processes	Monitoring OSS Processes on page 2-20, Managing OSS Processes on page 2-22, and the *Open System Services Shell and Utilities Reference Manual*
	periodic tasks	Scheduling Periodic Tasks on page 2-34, the *NetBatch Manual*, and the *NetBatch Plus Reference Manual*
	printers	Managing Printers in the OSS Environment on page 10-1
	servers	Section 4, Managing Servers
	user definitions and aliases	Managing Users and Groups on page 8-9, the *Safeguard Administrator's Manual* and the *Safeguard Reference Manual*, or an administrator's guide for a third-party product
	user groups	Managing Users and Groups on page 8-9, the *Safeguard Administrator's Manual* and the *Safeguard Reference Manual*, or an administrator's guide for a third-party product

Table 1-1. Management and Operations Tasks (page 3 of 3)

Task	Subtask	See
Mounting	an OSS fileset	Starting (Mounting) or Restarting Filesets on page 5-7
Starting	an OSS fileset	Starting (Mounting) or Restarting Filesets on page 5-7
	the OSS environment	ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34, Starting the OSS Monitor on page 2-7, or STARTOSS Utility on page C-14
	a server	Starting a Server on page 4-36 and Starting OSSTTY on page C-1
Stopping	an OSS fileset	Stopping (Unmounting) a Fileset on page 5-13
	the OSS environment	Starting and Stopping the OSS Environment on page 2-1 or STOPOSS Utility on page C-16
	a server	Stopping a Server on page 4-43 and Starting OSSTTY on page C-1
Unmounting	an OSS fileset	Stopping (Unmounting) a Fileset on page 5-13
Trouble-shooting	an OSS fileset	Troubleshooting Filesets on page 5-39, and Appendix A, Messages
	a server	Troubleshooting a Server on page 4-50, and Appendix A, Messages
Using	CVT	Managing and Repairing Fileset Catalog Files on page 5-40
	FSCK	Checking and Repairing Fileset Integrity on page 5-24
	OSS Monitor	Section 12, Open System Services Monitor
	SCF	Using OSS Monitor Commands on page 2-13, and the *SCF Reference Manual for G-Series RVUs*
	VPROC	Section 11, Managing Problems, and the *Guardian User's Guide*

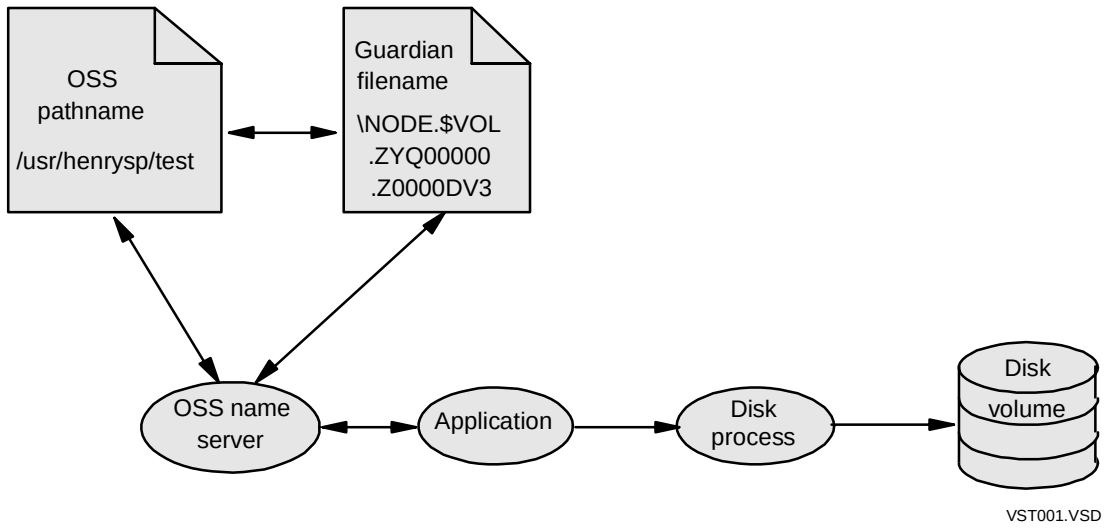
OSS File System Concepts

You use the OSS file system in the same way as you use a UNIX file system. The OSS file system provides the same functionality as a UNIX file system, but the OSS system is implemented differently internally. Because the OSS environment must work with the Guardian environment and the NonStop operating system, the OSS environment has the following major differences from the standard UNIX operating system:

- Disk management through the NonStop operating system. The portion of the NonStop operating system software that performs read, write, and lock operations on disk volumes is known as the disk process, shown in Figure 1-2 on page 1-6.

- OSS pathnames, which have underlying Guardian filenames. The mapping between OSS pathnames and Guardian filenames is known as filename resolution, and it is done by an OSS name server.
- The `/G` directory, which contains filesets for local files in the Guardian namespace, and the `/E` directory, which contains files on other nodes in the network.
- The `/dev` directory being used in a special way.

Figure 1-2. Guardian Filenames and OSS Files



OSS Files

OSS data files are stored under directories. Directories are grouped together for storage purposes; each group of directories is administered as an entity called a fileset.

Every OSS data file (called a disk file or a regular file) has a unique pathname and an underlying Guardian filename. An OSS name server process translates OSS file pathnames to and from Guardian filenames. The OSS name server also maintains the file and directory catalogs for the OSS environment; these catalogs contain uniquely numbered data structures for each file and directory, called inode numbers and link numbers.

Figure 1-2 shows that the OSS name servers resolve Guardian filenames and OSS pathnames to each other, then provide the information used by the system on behalf of the application program to communicate with the disk process, which provides access to the file on disk. Furthermore, each OSS filename points to an underlying Guardian file ID, such as Z0000DV3 in the figure.

For further information about the OSS file system, see Section 3, Understanding the OSS File System.

The /G Directory

The /G directory provides OSS names for Guardian files on your local NonStop S-series or NonStop NS-series node. Each Guardian filename has a corresponding OSS name of the form /G/*volume*/*subvolume*/*fileID*, where *volume*, *subvolume*, and *fileID* are case-insensitive; for example, $SYSTEM.SYS00.CONFLIST becomes /G/system/sys00/conflist.

The /G directory itself is reserved for HP use. You cannot put anything in this directory. You also cannot add a directory at the /G/*volume* level of the directory hierarchy.

If you add or delete files below the /G/*volume* level of this directory, you increase or decrease the amount of disk space used. However, the increase or decrease affects only the disk volumes that contain the Guardian files, which can be different from the volumes that contain OSS files. Therefore, if users delete files from the /G directory, they might not free space for OSS files and can lose files that they intended to keep.

The /G directory is described in more detail in Section 3, Understanding the OSS File System.

The /E Directory

The /E directory provides OSS names for OSS and Guardian files on remote NonStop S-series or NonStop NS-series nodes. Each remote filename has a corresponding OSS name of the form:

- /E/*nodename*/*pathname* for OSS files
- /E/*nodename*/G/*volume*/*subvolume*/*fileID* for Guardian files

where *nodename*, *volume*, *subvolume*, and *fileID* are case-insensitive.

For example:

- The OSS file /tmp/datafile on \NODE2 becomes /E/node2/tmp/datafile
- The Guardian file $SYSTEM.SYS00.CONFLIST on \NODE2 becomes /E/node2/G/system/sys00/conflist.

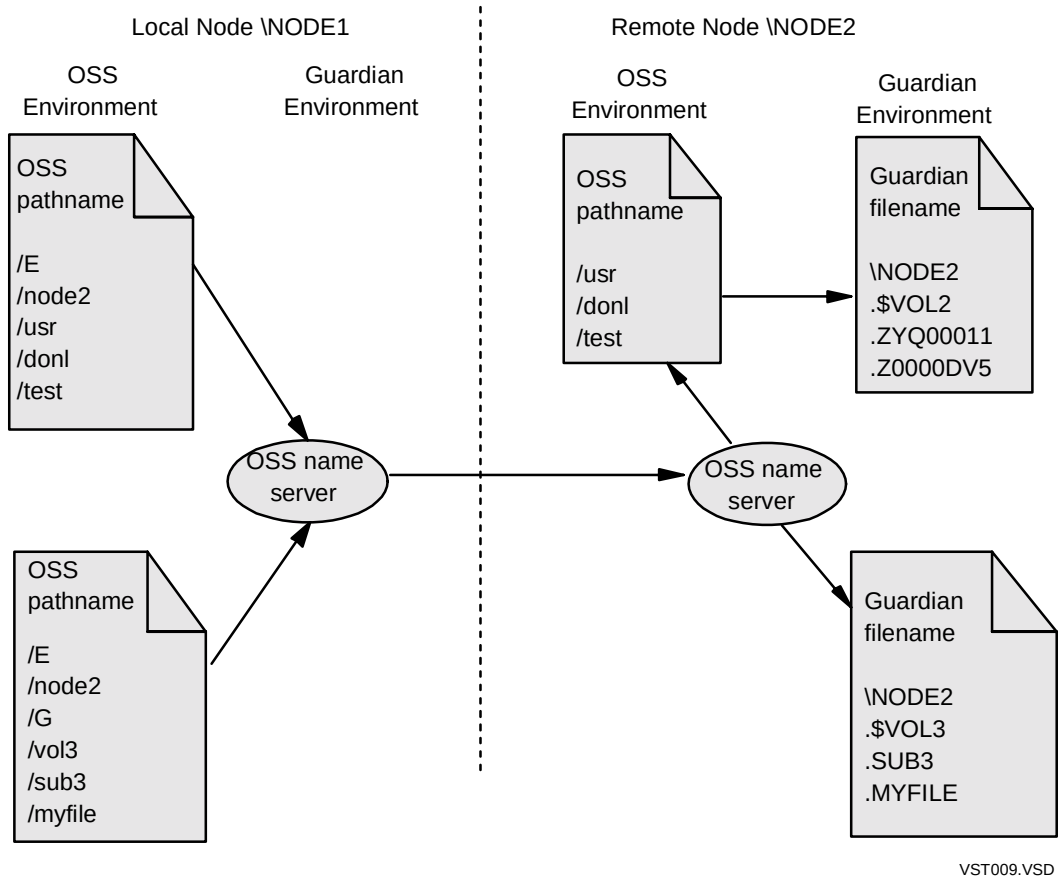
The /E directory itself is reserved for HP use. You cannot put anything in this directory.

If you add or delete Guardian files below the /G/*volume* level of this directory, you increase or decrease the amount of disk space used on the corresponding NonStop S-series or NonStop NS-series node. However, the increase or decrease affects only the disk volumes that contain the Guardian files—which can be different from the volumes that contain OSS files.

If you add or delete OSS files below the /*nodename* level of this directory, you increase or decrease the amount of disk space used on the corresponding NonStop S-series or NonStop NS-series node. However, the increase or decrease affects only the disk volumes that contain the OSS files, which can be different from the volumes that contain Guardian files.

Figure 1-3 shows how the /E directory on your local node allows remote file access through an Expand network.

Figure 1-3. Pathname Resolution for Remote File Access Through the Guardian Expand Network



The /E directory is described in more detail in Section 3, Understanding the OSS File System.

The /dev Directory

The /dev directory is the device directory. It contains only the following files:

`tty` The current controlling terminal for the application that is running.

`null` A data sink. Anything successfully sent to /dev/null disappears.

The /dev directory itself is provided for compatibility with existing UNIX software. Do not put anything in this directory.

Devices are added and configured in the Guardian environment; therefore, they do not appear in the /dev directory.

Components to Be Managed

You need to manage the software described in the following subsections:

Input/Output Utilities

Input/output utilities include:

OSSTTY Servers

Guardian administrative applications or OSS applications at your site might require you to make one or more copies of an OSSTTY server available for redirecting OSS standard files. The concept of redirection is described in Redirecting OSS Standard Files on page 6-27; the command to start a copy of OSSTTY is described in Starting OSSTTY on page C-1. Other than deciding how many copies of OSSTTY you need to run simultaneously, there are no management tasks associated with OSSTTY.

Terminal Helper Servers

Both OSSTTY and Telserv terminal processes can perform nonblocking input and output through the OSS file system because such communication is monitored and managed using a terminal helper process.

A terminal helper process named $ZTT*nn* runs in each processor (*nn* indicates the processor number). The terminal helper process starts automatically when the processor starts; if a terminal helper process terminates abnormally, it takes down the processor in which it runs.

The processor running a terminal helper server process can be shut down without first stopping the process, but you should stop all applications using that processor for terminal input or output first. There are no management tasks associated with the terminal helper server processes.

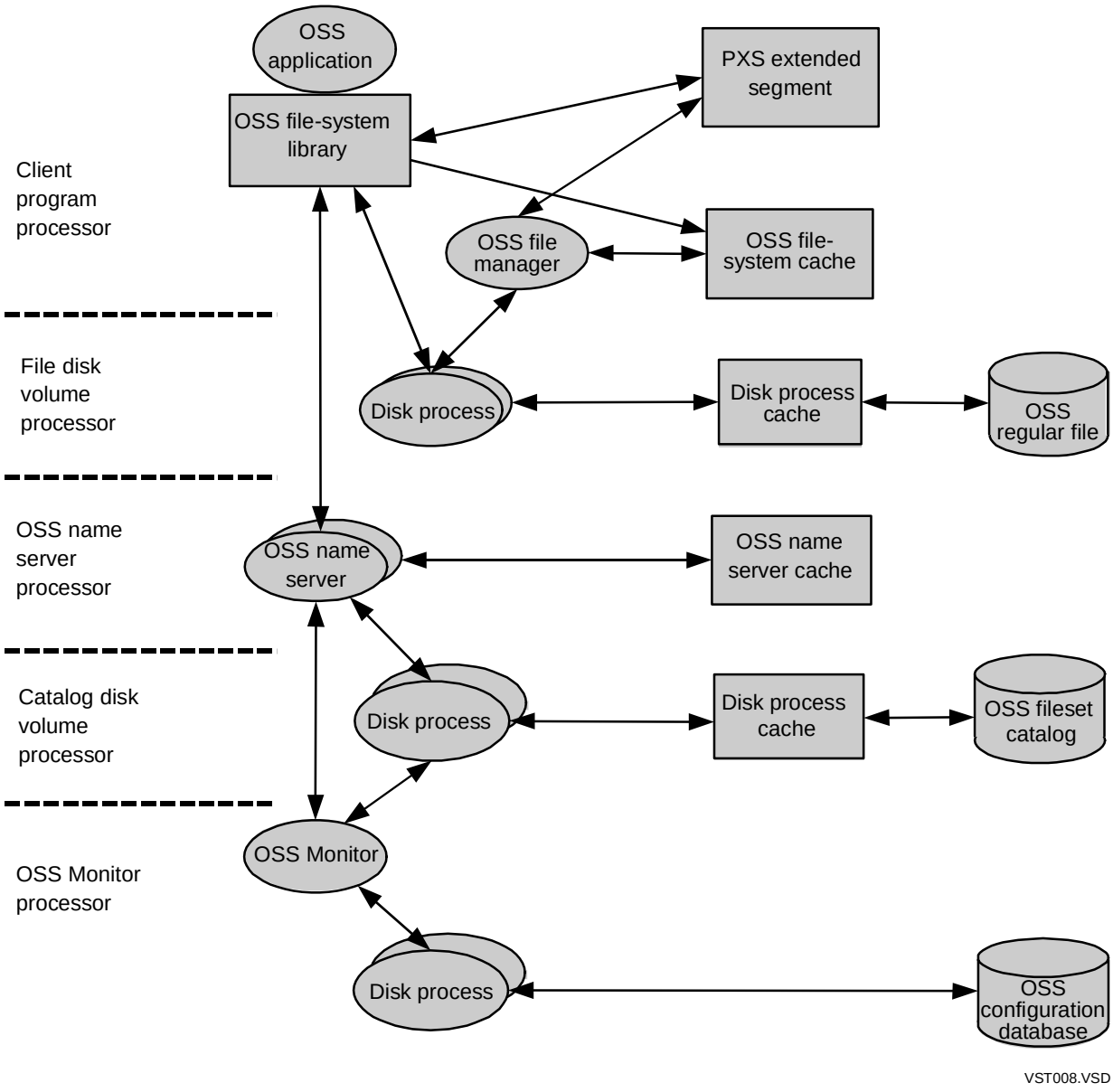
OSS Security

Security for the OSS environment is configured and managed through the optionally licensed Safeguard product. The use of Safeguard is described in Section 8, Managing Security.

OSS File-System Components

The software components that work together to provide access to an OSS file are shown in Figure 1-4 on page 1-10. Many of these components are configured and managed using the interfaces described in this guide.

Figure 1-4. OSS File-System Components



An application running in the OSS environment makes calls for access to an OSS file through function calls in the OSS file-system library. The OSS file-system library is present in each processor. The library consists of functions in a shared run-time library (SRL) and in the system library. OSS file-system library functions are described in the

Open System Services System Calls Reference Manual and in the *Open System Services Library Calls Reference Manual*.

When the application opens an OSS file, an OSS name server locates the correct file within the correct fileset, based upon configuration information you have supplied to the OSS Monitor for that fileset's catalog.

An OSS file manager process named $ZFM*nn* runs in each processor (*nn* indicates the processor number). The OSS file manager starts automatically when the processor starts; if the OSS file manager terminates abnormally, it takes down the processor. Its processor can be shut down without first stopping the process, but you should stop all applications with open OSS files first.

The OSS file manager automatically:

- Allocates and initializes the PXS extended segment and OSS file-system cache
- Satisfies OSS regular file cache-related requests from disk processes

The PXS extended segment is used to share OSS file-system data structures within that processor. The PXS extended segment is created and managed without system manager or operator intervention.

The OSS file-system cache is used to buffer data transfers between the OSS disk files and an OSS application. Whether this cache is used for a specific file depends on:

- Options used by an application to open the file
- The setting of the OSSCACHING flag for the disk process of the disk volume that contains that file

See Changing OSS File Caching for the Disks of a Fileset on page 5-18 for more information on configuring this cache.

Disk processes also can use data caches, as shown in Figure 1-4 on page 1-10. The use of disk process caching determines whether OSS file-system caching can be used, as mentioned above. See the *Storage Subsystem Configuration and Management Manual* for additional information about disk process caching.

An OSS name server also maintains a cache of the most recently resolved names for its filesets. This cache has a default size of 4096 inode values and 4096 link values. Other cache sizes can be specified through Subsystem Control Facility (SCF) commands, as described under Configuring an OSS Name Server on page 4-29.

OSS resources such as the PXS extended segment, data cache, OSS name server inode cache, and link cache, can also be monitored using Measure. See the *Measure Reference Manual* for information about the OSSCPU and OSSNS entities.

Interprocess Communication Facilities

Open System Services provides a set of interprocess communication (IPC) facilities identical to those of the UNIX operating system. The OSS IPC facilities are separate from the IPC facilities supplied within the Guardian environment.

OSS IPC facilities include the facilities described in the following subsections:

- OSS Shared Memory and Semaphores on page 1-14
- OSS Message Queues on page 1-15
- Pipes and FIFOs on page 1-15
- OSS Sockets on page 1-15

Figure 1-5 on page 1-13 shows these facilities with their approximate equivalents in the Guardian environment.

Figure 1-5. Interprocess Communication Facilities (page 1 of 2)

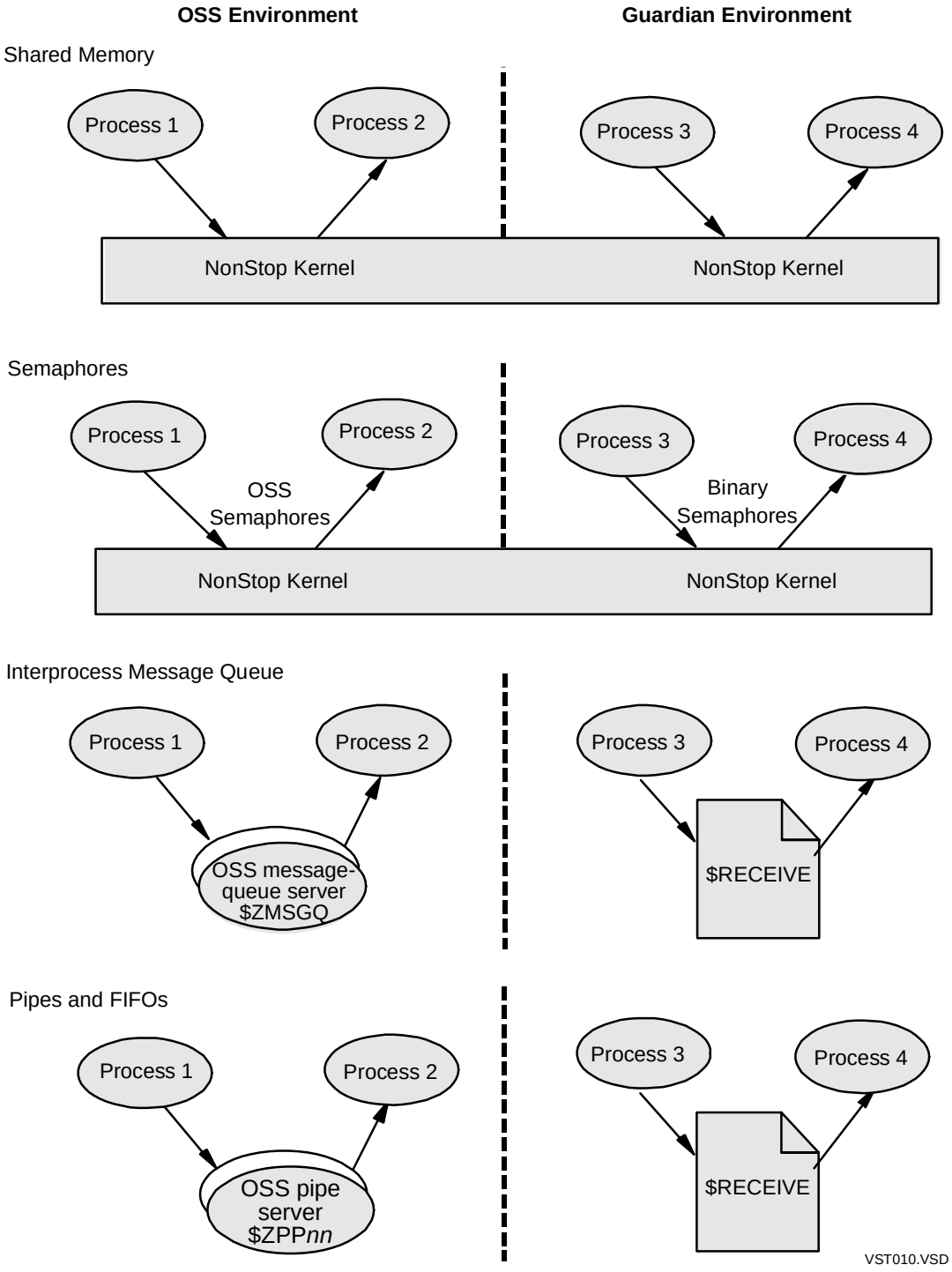
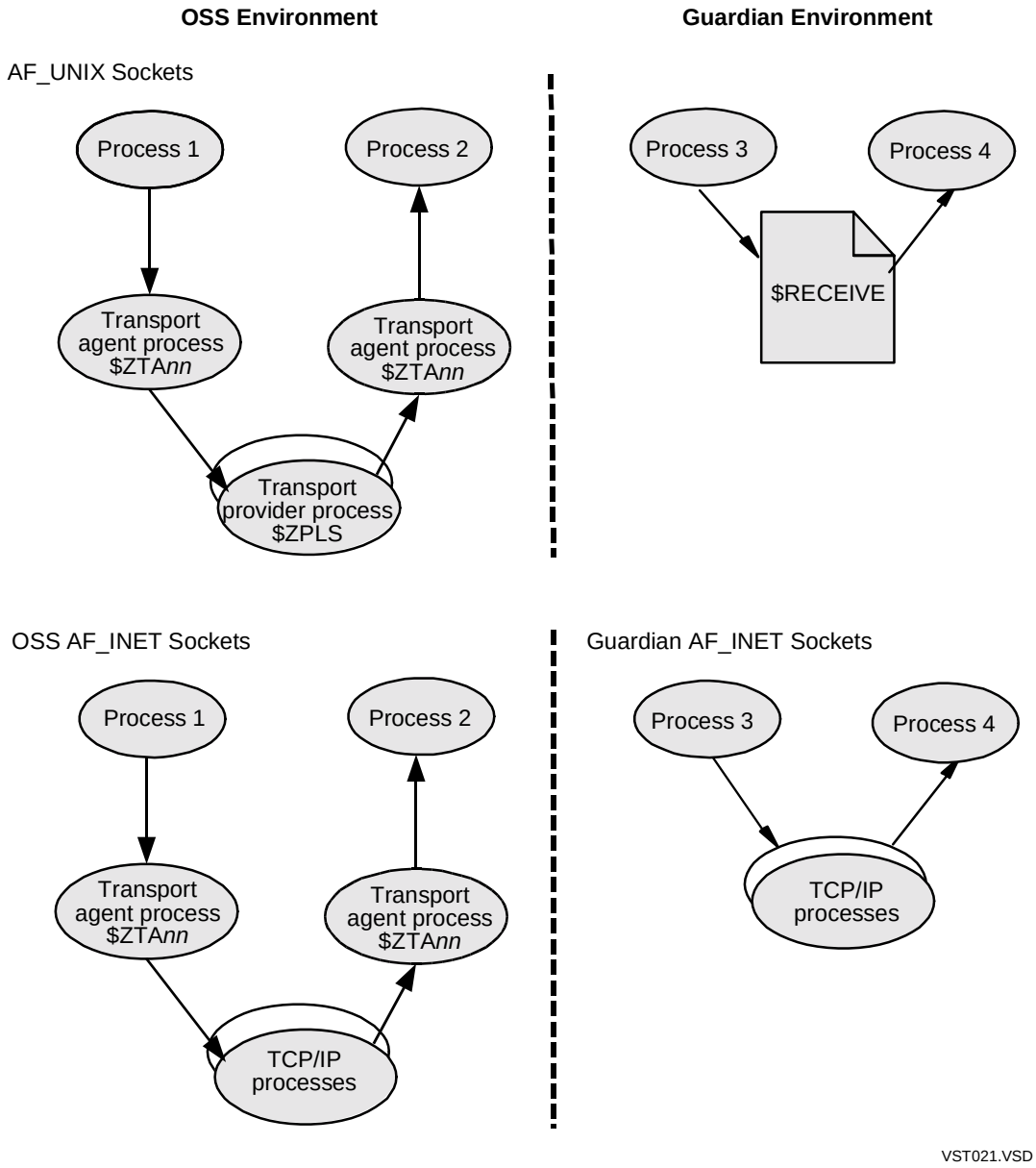


Figure 1-5. Interprocess Communication Facilities (page 2 of 2)



OSS Shared Memory and Semaphores

Semaphores allow one process to signal another about events such as the status of data in memory shared by the processes. OSS shared memory and semaphores are implemented using NonStop operating system features and do not require any installation or configuration actions before use. The Guardian environment has separate facilities for shared memory and semaphores that are not discussed in this guide.

OSS shell commands allow you to manage OSS message queues, OSS shared memory, and OSS semaphores; these commands are discussed further in Section 2, Operating the OSS Environment.

OSS Message Queues

Message queues are linked lists of messages used by programmers to pass data from one process to another. In the OSS environment, the OSS message-queue server process named $ZMSGQ manages the message queues.

Message queues cannot be used unless the OSS message-queue server is running. The OSS message-queue server can be started by the OSS SCF Monitor START SERVER Command, as described in Starting the OSS Message-Queue Server.

Pipes and FIFOs

Pipes are unnamed connections between two OSS application processes that are used to send or receive data. FIFOs are pipes with names. Pipes and FIFOs are special files in UNIX terminology, rather than regular files.

An OSS pipe server process named $ZPP*nn* runs in each processor (*nn* indicates the processor number). The OSS pipe server supports the transfer of data between OSS processes that use pipes or FIFOs between processors.

Like the OSS file manager, the OSS pipe server starts automatically when the processor starts; if the OSS pipe server terminates abnormally, it takes down the processor. Its processor can be shut down without first stopping the process, but you should stop all applications with open OSS pipes or FIFOs first.

Because you do not need to manage the OSS pipe server, pipes and FIFOs are not discussed further in this guide and the OSS pipe server is not shown in Figure 1-4 on page 1-10.

OSS Sockets

In addition to the OSS message-queue server and the OSS name servers, your system needs servers to provide an OSS application program with access to OSS sockets. OSS sockets facilities are separate from the sockets facilities provided within the Guardian environment.

There are two kinds of OSS sockets, named after the address families used to send and receive data through them. OSS sockets include:

- `AF_UNIX` sockets, sometimes called local or UNIX domain sockets
- `AF_INET` or `AF_INET6` sockets, sometimes called Internet domain sockets

When `AF_UNIX` sockets are used or `AF_INET` sockets are used with HP NonStop TCP/IP:

- OSS sockets route data between application processes using a process called a transport agent and a process called a transport provider

- Multiple transport-provider processes can be used to create separately addressed IP subnetworks within a node

When AF_INET sockets are used with NonStop Parallel Library TCP/IP:

- No transport-agent process or transport-provider process is involved in data routing.
- A transport-agent process must still be started in each processor to initialize the OSS sockets interface for that processor.
- One transport-provider process must be running in the system. The transport-provider process provides configuration compatibility for open and close operations on sockets and for nonsensitive SCF commands.

When AF_INET sockets or AF_INET6 sockets are used with NonStop TCP/IPv6:

- No transport-agent process or transport-provider process is involved in data routing.
- A transport-agent process must still be started in each processor to initialize the OSS sockets interface for that processor.
- At least one transport-provider process must be running in the system.
- The transport-provider process provides configuration compatibility for open and close operations on sockets and for nonsensitive SCF commands.
- Multiple transport-provider processes can be used to create separately addressed IP subnetworks within a node. This feature is called logical network partitioning; such subnetworks are configured and controlled through the SCF interface for the NonStop TCP/IPv6 subsystem.

Transport-Agent Processes:

- The transport-agent process initializes global sockets structures and tables when it starts. A transport-agent process that routes data (for NonStop TCP/IP or AF_UNIX sockets) among OSS sockets application processes does so for a single processor. Each processor has its own copy of the transport-agent process.
- HP provides a transport-agent process, $ZTA*nn*, where *nn* is the processor number. The transport-agent process starts automatically when the processor comes up. You can bring down its processor without first stopping the process, but you should stop all applications with open OSS sockets before bringing down a processor. You manage the transport agent through the SCF module of the OSS Monitor, as described in Section 4, Managing Servers.

Transport-Provider Processes:

- A transport-provider process provides routing services for one sockets address family. Each node has one or more transport-provider processes.

- HP provides the OSS sockets local server, $ZPLS, as a transport-provider process for OSS AF_UNIX sockets. You manage the OSS sockets local server through the SCF module of the OSS Monitor, as described in Section 4, Managing Servers.
- HP provides processes for each of its TCP/IP implementations as transport-provider processes for OSS and Guardian AF_INET and AF_INET6 sockets. You manage an AF_INET or AF_INET6 transport-provider process through the SCF commands for the TCP/IP subsystem that you use for specific sockets.

AF_INET or AF_INET6 Sockets:

- Internet domain sockets allow application programs to communicate with each other or with terminals using the underlying TCP/IP processes that also provide Telserv terminal access for your system. AF_INET sockets provide access through Internet Protocol (IP) version 4 addresses; AF_INET6 sockets provide access through IP version 6 addresses.
- For Internet domain sockets using NonStop TCP/IP, the socket application communicates with the $ZTA*nn* process. The $ZTA*nn* process in turn provides Internet access through the transport-provider process of TCP/IP.
- The AF_INET or AF_INET6 transport-provider process is named $ZTC*n*, $ZSAM*n*, or another name chosen by your site. The OSS sockets software assumes that the default name is $ZTC0 when no other process name is specified by an appropriate method. This guide uses $ZTC0 for discussions that involve the Internet domain transport-provider process.

  Note. A copy of the transport-provider process is not necessarily given the default name of $ZTC0. However, as long as either the DEFINE =TCPIP^PROCESS^NAME matches the running transport-provider process or the OSS application selects the transport-provider process name by calling `socket_transport_name_set()`, OSS sockets function properly.

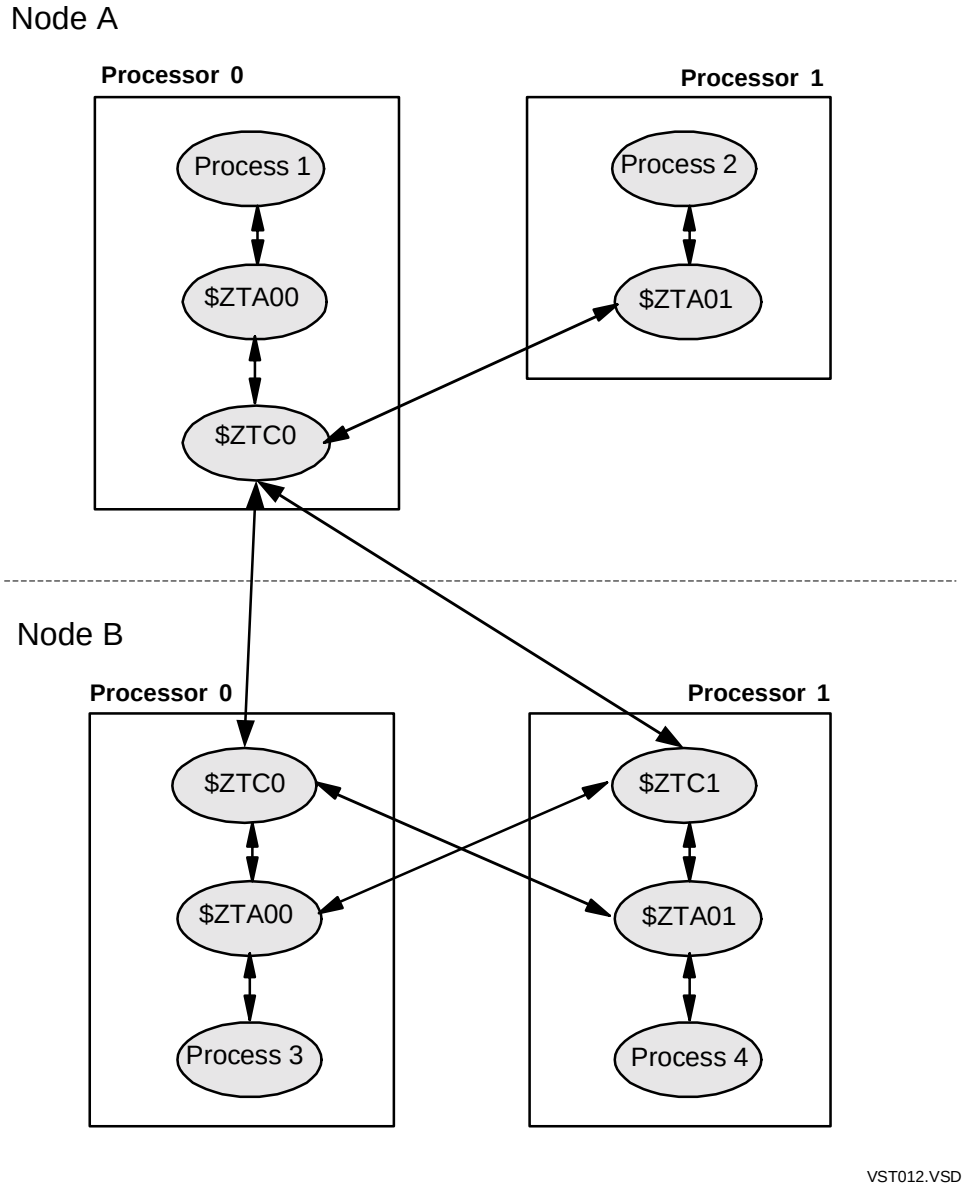
  See the:

  - *TCP/IP Configuration and Management Manual*
  - *TCP/IP (Parallel Library) Configuration and Management Manual*
  - *TCP/IPv6 Configuration and Management Manual*

  for information on the configuration and management of the corresponding transport-provider process.

- Figure 1-6 on page 1-18 shows the processes related to providing OSS AF_INET sockets for NonStop TCP/IP. The diagram shows two systems, referred to as Node A and Node B, each with two processors. In Node A, only one of the processors has a transport-provider process ($ZTC0), whereas in Node B, each processor has its own transport-provider process ($ZTC0 and $ZTC1). Each processor has its own transport-agent process ($ZTA*nn*).

Figure 1-6. OSS AF_INET Sockets Servers for NonStop TCP/IP



The application processes (Processes 1 through 4) communicate with each other through the transport-provider ($ZTC*n*) processes, with data being carried back and forth by the transport-agent processes. Transport-agent processes do not communicate with each other directly, but through the transport-provider.

Note. Whenever any application process sends a message to another process, the message is always sent through the transport-provider processes and through the network, regardless of which processors or nodes they are running on. Even when two processes are running on the same processor, a message from one to the other always goes to the transport-provider process first.

For example, given the situation pictured in Figure 1-6 on page 1-18, suppose Process 2 sends a message to Process 4. The following occurs:

1. The message is forwarded by $ZTA01 (the transport-agent process on processor 1 of Node A, where Process 2 is running) to $ZTC0 (the transport-provider process that supports the IP address used by Process 2).
2. The message is sent over the network and is received on Node B by $ZTC0 (the transport-provider process that supports the IP address used by Process 4).
3. The message is forwarded to $ZTA01 (the transport-agent process on processor 1 of Node B, where Process 4 is running).
4. The message is delivered to Process 4 when that process is ready to accept the message.

- For NonStop Parallel Library TCP/IP, Internet domain socket application processes exchange data through their embedded library code; however, other processes not managed as part of the OSS environment must be running. The NonStop Parallel Library TCP/IP component TCPMON, which runs as a process with the name $ZPTM*n* (where *n* is its processor number), must be running in a processor that runs an OSS sockets application program. See the *TCP/IP (Parallel Library) Configuration and Management Manual* for more information on data flow between sockets applications.
- For NonStop TCP/IPv6, Internet domain socket application processes exchange data through their embedded library code; however, other processes not managed as part of the OSS environment must be running. The NonStop TCP/IPv6 component TCP6MON, which runs as a process with the name $ZPTM*n* (where *n* is its processor number), must be running in a processor that runs an OSS sockets application program. See the *TCP/IPv6 Configuration and Management Manual* for more information on data flow between sockets applications.

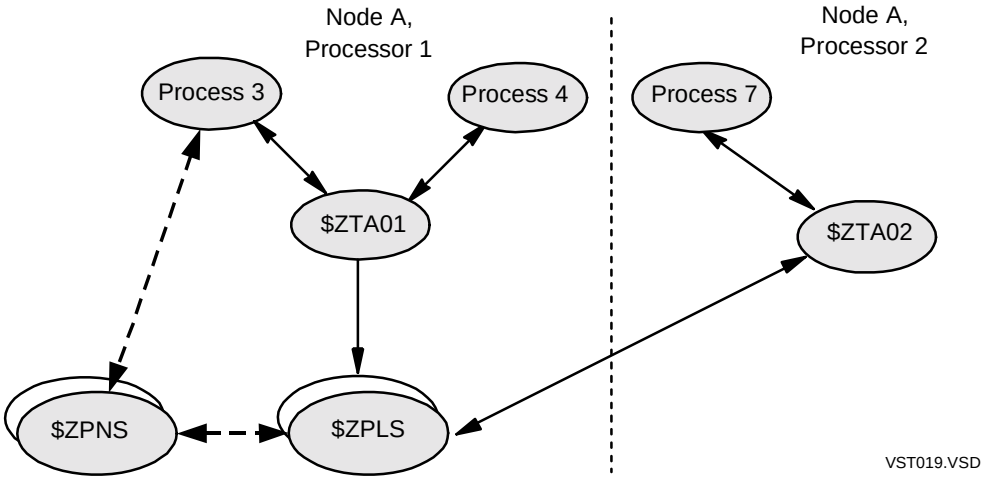
AF_UNIX Sockets:

- `AF_UNIX` sockets allow an application program to use an OSS socket as if it were a named disk file or a named pipe.
- For `AF_UNIX` sockets, the OSS sockets local server, $ZPLS, is the transport provider process and $ZTA*nn* is the transport agent process. `AF_UNIX` sockets initially require an OSS name server to provide addressing information.
- Figure 1-7 on page 1-20 shows these processes. In this case, the OSS name server ($ZPNS) for the root fileset provides the addressing information used by the OSS sockets local server for an `AF_UNIX` socket when that socket is created by Process 3 in a directory that is part of the root fileset.

  In the figure, Processes 3 and 4 use `AF_UNIX` sockets to communicate, as do Processes 3 and 7. When Process 3 sends a message to Process 4, the message is forwarded to $ZTA01, then to $ZPLS, then to $ZTA01, and then to Process 4.

Similarly, when Process 3 sends a message to Process 7, the message is forwarded to $ZTA01, then to $ZPLS, then to $ZTA02, and then to Process 7.

Figure 1-7. OSS AF_UNIX Sockets Servers



2

Operating the OSS Environment

You operate the Open System Services environment from the Guardian environment. As a system operator, when you are operating the OSS environment, there is little you need to do differently from operating the Guardian environment. This section describes what you do to operate the OSS environment that is unique to the OSS environment.

The primary sets of operating tasks are as follows:

Starting and Stopping the OSS Environment

The OSS environment was started when the system first came up. An application program can run in the OSS environment without any additional action by using a startup mechanism that assigns it an OSS process ID. Therefore, you actually cannot stop the OSS environment.

However, for a program or an interactive user to start an OSS shell, use an OSS utility, or use an OSS file, the OSS file system must be running. For most users, starting and stopping the OSS file system is equivalent to starting and stopping the OSS environment.

You start the OSS file system by starting the root fileset. The OSS file system runs as long as the root fileset for your local node is started (mounted). Stopping the OSS Monitor does not stop the OSS file system.

Possible Ways to Start the OSS File System

Starting the OSS file system is a multiple-step process:

1. The OSS Monitor must be started at least once.
2. All filesets needed by users must be started (mounted), beginning with the root fileset.
3. All servers managed by the OSS Monitor should be started.

These steps can be performed in several ways:

- Using the automatic startup service AUTOSTART feature described under Automatic Startup Service on page 2-2. Configuring and using this feature properly

provides the most availability for OSS systems and requires the least user intervention if a failure occurs.

- Using the STARTOSS utility if your system was initially configured by using the OSSSETUP utility, and the OSSINFIL file has been properly maintained. See STARTOSS Utility on page C-14 for more information.
- Using individual commands described under Managing the OSS Subsystem on page 2-6 and in other sections of this guide.

The OSS Monitor was probably started when Open System Services was installed. If not, follow the process described under Starting the OSS Monitor on page 2-7.

Automatic Startup Service

The automatic startup service allows you to configure:

- each fileset and its OSS name server
- the OSS message-queue server
- the OSS local sockets server

so that they are automatically restarted whenever a system load occurs, when one of those server processes fails, or when a processor failure and reload occurs that affects one of those server processes.

The automatic startup service is configured by setting the subsystem AUTOSTART attribute to either MANUAL or AUTO.

When MANUAL (the default) is chosen, the OSS Monitor performs automatic remounting of filesets based only on the current state of the fileset when a failure occurred. This behavior is described in Automatic Restart of Filesets During OSS Monitor Startup and Automatic Restart of Filesets After OSS Name Server Failure on page 5-10; this behavior is compatible with the behavior of release version updates preceding G06.17. The OSS message-queue server and OSS local sockets server are not automatically started after a system load or if circumstances prevent their recovery when running as fault-tolerant process pairs.

When AUTO is chosen, the OSS Monitor attempts to start the objects configured for automatic startup based upon the configured value for the DESIREDSTATE attribute of each object. Objects configured with a desired state of STARTED and not manually stopped after the previous system load are started when:

- The OSS Monitor starts for the first time after a system load
- A server process managed through the OSS Monitor fails
- A processor starts that is configured for use by an automatically started server process

Attempts to restart a server can be controlled using the AUTORESTART and MAXWAITTIME attributes for it, as described in ADD SERVER Command on page 12-16 and ALTER SERVER Command on page 12-28.

Possible Ways to Stop the OSS File System

You can stop the OSS file system by unmounting the root fileset, which effectively stops the OSS environment. This step can be performed either by:

- Using the STOPOSS utility. STOPOSS might also stop processes you want to continue running; see the STOPOSS Utility on page C-16 for considerations when using this command.
- Entering individual commands by using the procedures described under Manually Stopping the OSS File System and the OSS Environment on page 2-3 and in Section 5, Managing Filesets.

You might need to stop the OSS file system to perform maintenance operations such as checking fileset integrity. Afterwards, you must restart the OSS file system, either by using STARTOSS (see STARTOSS Utility on page C-14) or as described under Manually Restarting the OSS File System and the OSS Environment on page 2-6.

Manually Stopping the OSS File System and the OSS Environment

Stopping the OSS file system and the OSS environment requires the following steps:

1. Before stopping the OSS file system, warn your current users and all new users that you are about to do so. Use a method described under Sending Warnings to Users on page 2-5.
2. Identify and stop all applications currently using the OSS environment:
   a. To identify all executing OSS processes, enter the OSS shell command:

      ```
      ps -e
      ```

      The output includes:

      - The process file pathname (of the program file or shell script being run) so you can probably map a specific process to an application
      - The OSS process ID (PID) for each process

   b. Perform application-specific procedures to stop each process, reentering the `ps -e` command periodically to determine how many processes remain to shut down:

      - If the application supports a shutdown command, use that command to stop all its processes. This might involve Guardian PATHCOM or another command interpreter instead of a command in the application itself.
      - If the application does not support a shutdown command, its processes might have been coded to terminate gracefully (performing data cleanup, normal file closes, and state cleanup) when normal shutdown signals are received. Shutdown signals can vary from application to application;

however, the most commonly used signal can be sent by entering the OSS shell command:

```
kill PID1 PID2 PID3 ...
```

where *PID1*, *PID2*, and *PID3* are OSS process IDs displayed by the `ps` command.

To avoid specifying the individual PIDs, which can make command entry a lengthy and potentially error-prone step, write an OSS shell script to extract the PID numbers from the `ps` command output and pipe those numbers into the `kill` command.

- When an application process ignores normal shutdown signals, you can use force shutdown by entering:

  ```
  kill -s KILL PID1 PID2 PID3 ...
  ```

  where *PID1*, *PID2*, and *PID3* are OSS process IDs displayed by the `ps` command.

For more information about the `ps` and `kill` commands, see the `ps(1)` and `kill(1)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual*.

3. Stop all servers managed by the OSS Monitor by entering the following Subsystem Control Facility (SCF) commands:

   ```
   ASSUME PROCESS $ZPMON
   STOP SERVER *
   ```

4. Stop (unmount) all filesets mounted on the root fileset and the root fileset itself by entering the following SCF commands:

   ```
   ASSUME PROCESS $ZPMON
   STOP FILESET *
   ```

   The wildcard form of the STOP FILESET command stops all filesets in the reverse of the order that they were started (mounted).

5. After the last fileset stops, stop the OSS Monitor process:

   - If the OSS Monitor is running as a standard process, enter the following at a TACL prompt:

     ```
     STOP $ZPMON
     ```

   - If the OSS Monitor is running as a persistent process, enter the following at an SCF prompt:

     ```
     ABORT PROCESS $ZZKRN.#ZPMON
     ```

Note. The process device identifier #ZPMON is the convention used throughout this guide; your site might use another naming convention such as #OSMON or #OSSMN. The process name $ZPMON is required by the OSS Monitor process itself; however, the process device identifier used within the Kernel subsystem is not required to be #ZPMON.

Sending Warnings to Users

Use the OSS shell `wall` command or an OSS shell script to warn users of the OSS shell about a shutdown. Follow your site's broadcast message procedures to warn users of Guardian environment processes that might be using OSS files.

The shell script in Figure 2-1 on page 2-5 gathers the terminal names for all logged-in users into the shell variable named `list`, then echoes a message to each terminal name in `list`. The message is displayed on all terminals with an active OSS shell.

Alternatively, you can capture the message in a file and use the OSS shell `wall` command to broadcast it to all logged-in users. For example, if you capture the message portion of the script from Figure 2-1 into a file named `warn.msg` in your current working directory, then, to send that message to all logged-in users of the OSS shell, you would enter:

```
/bin/wall warn.msg
```

Figure 2-1. Sample Broadcast Message for Stopping the OSS File System

```
#!/bin/sh
list=`who | awk '{print $2}'`

for T in $list
do
  (
echo "****************************************"
echo "*    Broadcast message at `eval date`   *"
echo "*                 WARNING!               *"
echo "*             OSS environment            *"
echo "*         stopping in five minutes.      *"
echo "*    Please save your files and exit.    *"
echo "****************************************"
  ) > $T
done
```

You can also put commands such as the script in Figure 2-2 on page 2-5 in the `/etc/profile` file to warn users who are starting an OSS shell. In Figure 2-2, *`time`* is the time you put the message into the `/etc/profile` file. The message is displayed every time a user logs in. Remember to remove the message after you restart the OSS file system.

Figure 2-2. Sample Login Warning for Stopping the OSS File System

```
echo "****************************************"
echo "*    Broadcast message at time          *"
echo "*            WARNING!                   *"
echo "*        OSS environment                *"
echo "*    stopping in five minutes.          *"
echo "*        Please exit now.               *"
echo "****************************************"
```

For information about shell scripts, see the *Open System Services User's Guide*. For information about the `wall` command, see the `wall(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

You should also follow your site's broadcast message procedures to warn users who log in through TACL to use OSS files from Guardian environment processes.

Manually Restarting the OSS File System and the OSS Environment

Restarting the OSS file system or OSS environment should be a rare occurrence. Follow these steps:

1. If the OSS Monitor is not running, start it as described under Starting the OSS Monitor on page 2-7.
2. Unless you have configured the root fileset for automatic restart, check the integrity of the root fileset if you have not done so recently. Use the SCF DIAGNOSE FILESET command, as described in Checking and Repairing Fileset Integrity on page 5-24.
3. If you have not configured the root fileset for automatic restart, restart the OSS file system by starting the root fileset with the SCF START FILESET Command.
4. If you have not configured filesets for automatic restart, restart all other filesets in the order of their mount points within the OSS file system; proceed in top-down order, beginning with the filesets mounted on the root fileset.
5. If you have not configured servers for automatic restart, restart all servers managed by the OSS Monitor as described in Starting a Server on page 4-36.

Managing the OSS Subsystem

Managing the OSS subsystem involves the tasks described in:

To a system manager, the OSS subsystem consists of the OSS Monitor and all processes started by or through the OSS Monitor. That view of the OSS subsystem is used throughout this guide.

To a system operator, the OSS subsystem consists of all processes in the OSS environment that log operator messages. Only one subsystem identifier exists for all

messages recorded using the Event Management Service (EMS) through either of the following:

- The OSS shell `logger` command
- Program calls that use the `syslog()` function and related functions

For more information about the `logger` command, see the `logger(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. For more information about the `syslog()` function, see the `syslog(3)` reference page either online or in the *Open System Services Library Calls Reference Manual*.

EMS messages logged through the `logger` command or the `syslog()` function are described in the *Operator Messages Manual*.

The OSS subsystem as described in this guide produces the informative and diagnostic messages discussed in Appendix A, Messages. Those messages are returned when you use either the SCF commands for the OSS Monitor or the CVT utility, as discussed under Managing and Repairing Fileset Catalog Files on page 5-40.

You start the OSS subsystem by starting the OSS Monitor at least once. The OSS Monitor was probably started when the OSS environment was installed. If you need to start or restart the OSS Monitor, see Starting the OSS Monitor on page 2-7.

OSS Monitor commands can be used on the subsystem itself, on server objects, or on fileset objects. Use on the subsystem itself is discussed in the following subsections. Using OSS Monitor commands on servers and filesets is discussed in later sections of this guide.

Although you cannot stop the entire subsystem without stopping its individual components and the OSS file system, you can temporarily stop the OSS Monitor. See Stopping the OSS Monitor on page 2-15.

Starting the OSS Monitor

The OSS Monitor can be started as either a normal process or a persistent process. Starting the OSS Monitor as a persistent process also allows you to enable the automatic startup service for selected OSS servers and filesets after a system load or restart.

Before starting the OSS Monitor, note that:

- The security process $ZSMP should be running before the OSS Monitor is started. If $ZSMP is not running, security auditing, user aliases, and supplementary groups are not available for the OSS environment.
- If your node number has changed since the last time the OSS Monitor was started, you should perform the maintenance task described for the ZOSSFSET file on page 4-12.

Beginning with the G06.15 release version update (RVU), if your system was ordered preconfigured or your initial OSS configuration was performed by using the

OSSSETUP utility, you can start $ZSMP by using the STARTOSS utility; see STARTOSS Utility on page C-14 for more information. Alternatively, you can perform these actions yourself.

Starting the OSS Monitor as a Normal Process

1. Log in as the super ID.
2. At a TACL prompt, enter:

```
OSSMON / NAME $ZPMON, NOWAIT, CPU nn, TERM name, PRI pri / &
       [ AUTOSTART { AUTO | MANUAL } ]
```

   `CPU nn`

   is the processor number of the processor you want the OSS Monitor to run on. The processor number you specify is arbitrary and should be chosen based on the system workload for all the processors.

   `TERM name`

   is the name of the home terminal for the OSS Monitor. Make sure that this terminal is always available and that it is not a Telserv or Multilan session. The $ZHOME process can be specified for *name*.

   `PRI pri`

   is the priority to run the OSS Monitor at. The value used should be lower than the priority used by $ZSMP (198).

   `AUTOSTART { AUTO | MANUAL }`

   specifies whether the automatic startup service should start servers or filesets that are configured for automatic startup. If this parameter is specified on the command line, it overrides any previous specification of an AUTOSTART PARAM. This parameter also overrides the current value for the AUTOSTART attribute of the subsystem; see the SCF ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34 for more information.

3. Use the TACL STATUS command to verify that the OSS Monitor process $ZPMON is running:

```
STATUS $ZPMON
```

4. If the OSS Monitor process is not running, check the Event Management Service (EMS) log for related event messages. See Appendix A, Messages, for explanation of any messages received and possible corrective actions.

Starting the OSS Monitor from a TACL prompt with an AUTOSTART value of AUTO does not take full advantage of the automatic startup service. To fully use the automatic startup service (so that the OSS environment is brought up when the system is loaded), you must start the OSS Monitor as a persistent process.

Starting the OSS Monitor as a Persistent Process

1. At a TACL prompt, enter:

   ```
   SCF
   ASSUME PROCESS $ZZKRN
   ```

2. Start the $ZSMP process as a generic process so that the OSS Monitor can communicate with it without a delay during restart.

   a. At an SCF prompt, enter the following to add $ZSMP to the NonStop Kernel subsystem configuration as a generic process:

      ```
      ADD PROCESS $ZZKRN.#ZSMP, &
          PROGRAM $SYSTEM.SYSTEM.OSMP, &
          NAME $ZSMP, &
          PRIMARYCPU 0, &
          BACKUPCPU 1 , &
          AUTORESTART 0, &
          STARTMODE KERNEL, &
          HOMETERM $ZHOME, &
          DEFAULTVOL $SYSTEM.SYSTEM, &
          STARTUPMSG "<BCKP-CPU>"
      ```

      where STARTUPMSG contains a literal that causes the use of a backup copy regardless of which specified processor comes up first. The AUTORESTART value of 0 is used because $ZSMP cannot be stopped by SAFECOM if it is configured as a persistent process. The STARTMODE value of KERNEL is used so that $ZSMP starts before $ZPMON (which uses STARTMODE SYSTEM).

   b. At the SCF prompt, enter the following to start $ZSMP:

      ```
      START PROCESS $ZZKRN.#ZSMP
      ```

   c. At the SCF prompt, enter the following to verify that $ZSMP is running:

      ```
      STATUS PROCESS $ZZKRN.#ZSMP
      ```

   d. If the $ZSMP process is not running, check the Event Management Service (EMS) log for related event messages. See the *Operator Messages Manual* for explanation of any console messages received and possible corrective actions.

3. To add the OSS Monitor to the NonStop Kernel subsystem configuration as a persistent process, at an SCF prompt, enter:

   ```
   ADD PROCESS $ZZKRN.#ZPMON, &
       PROGRAM $SYSTEM.SYSTEM.OSSMON, &
       NAME $ZPMON, &
       CPU FIRST, &
       AUTORESTART 5, &
       STARTMODE SYSTEM, &
       HOMETERM $ZHOME, &
       DEFAULTVOL $SYSTEM.SYSTEM
   ```

4. At an SCF prompt, to start the OSS Monitor, enter:

   ```
   START PROCESS $ZZKRN.#ZPMON
   ```

5. At an SCF prompt, to verify that the OSS Monitor is running, enter:

   ```
   STATUS PROCESS $ZZKRN.#ZPMON
   ```

6. If the OSS Monitor process is not running, check the Event Management Service (EMS) log for related event messages. See Appendix A, Messages, for explanation of any console messages received and possible corrective actions.

Running the OSS Monitor as a persistent process does not automatically enable the automatic startup service or select OSS servers or filesets for automatic restart. See the SCF ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34 for information on enabling the automatic startup service, and see the SCF ADD and ALTER commands for information on configuring automatic restart of filesets and OSS servers.

For more information about managing generic and persistent processes, see the *SCF Reference Manual for the Kernel Subsystem*.

PARAMs Used by the OSS Monitor

In G-series RVUs preceding G04.00, you could alter the behavior of the OSS Monitor by setting TACL PARAMs before starting the OSS Monitor. The actions controlled by these PARAMs could be changed only while the OSS Monitor was running and in those instances where an SCF command allowed you to specify a different value.

Beginning with the G04.00 RVU, the OSS Monitor ignores most TACL PARAMs and uses values configured in one of the following files:

- The ZOSSPARM file
- The ZOSSSERV file

as appropriate for the related OSS facility. The tasks required to configure and check these files are described later in this section for the ZOSSPARM file and in Configuration Files on page 4-7 for the ZOSSPARM file and the ZOSSSERV file.

The TACL PARAMs listed in Table 2-1 on page 2-11 are intended to be used during the first execution of the OSS Monitor. If this is a new installation with no OSS Monitor database, then the OSS Monitor looks for these PARAMs. It stores the values of these PARAMs in the global subsystem settings, an action equivalent to using the SCF ALTER SUBYS, IOTIMEOUT *value*, FSCKCPU *value*, AUTOSTART *value* command described in ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34.

Table 2-1. Currently Used TACL PARAMs for the OSS Monitor (page 1 of 2)

PARAM	Description
AUTOSTART { AUTO | MANUAL }	Specifies whether the automatic startup service should be used for any stopped servers or filesets that are configured to use that service. This PARAM can provide the initial configuration value for the AUTOSTART attribute of the subsystem. If this PARAM is supplied after the first execution of the OSS Monitor, it replaces the current value for the AUTOSTART attribute of the subsystem. This value is kept in the ZOSSPARM file and controlled through the SCF ALTER SUBSYS command. See ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34 for more information about the options that can be specified.
OSS^FSCK^CPU *processor*	Specifies which processor to use when the FSCK integrity-checker program performs fileset recovery. This PARAM can provide the initial configuration value for the FSCKCPU attribute of the subsystem. If this PARAM is supplied after the first execution of the OSS Monitor, it replaces the current value for the FSCKCPU attribute of the subsystem. This value is kept in the ZOSSPARM file and controlled through the SCF ALTER SUBSYS command. See ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34 for more information about the values that can be specified.

Table 2-1. Currently Used TACL PARAMs for the OSS Monitor (page 2 of 2)

PARAM	Description
OSS^NAMESERVER^TIMEOUT *seconds*	Defines the amount of time that the OSS Monitor waits for a response from an OSS name server. This PARAM can provide the initial configuration value for the IOTIMEOUT attribute of the subsystem. If this PARAM is supplied after the first execution of the OSS Monitor, it replaces the current value for the IOTIMEOUT attribute of the subsystem. This value is kept in the ZOSSPARM file and controlled through the SCF ALTER SUBSYS command. See ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34 for more information about the value that can be specified.

The TACL PARAMs listed in Table 2-2 were used in G-series RVUs preceding G04.00 but no longer affect OSS Monitor or OSS environment behavior. These PARAMs can be safely deleted from TACL scripts.

Table 2-2. Obsolete TACL PARAMs for the OSS Monitor

PARAM Name	Description
OSS^FSCK^SWAPVOL	Defines the volume and subvolume used for FSCK swap files. This value is now automatically provided by the NonStop Kernel subsystem.
OSS^NAMESERVER^CPU	Defines the processor number of the processor on which the primary OSS name server process is started. This value is now kept in the ZOSSSERV file and controlled through the SCF ALTER SERVER command.
POSIX^CONFIG^LOC	Defines the volume and subvolume of the ZPCONFIG file. This value is no longer meaningful, because the location of configuration files is permanently set to $SYSTEM.ZXOSSMON.

Specifying a Home Terminal

If you start the OSS Monitor from a terminal connected through Telserv (that is, a TELNET pseudo-terminal), you must specify a different device as your home terminal in the RUN command that starts the OSS Monitor. If you do not specify a different device and if your terminal session exits after you start the OSS Monitor, the OSS

Monitor can no longer start an OSS name server or the FSCK utility. A valid terminal device must exist when the OSS Monitor starts these programs.

One possible solution to the problem is to select a permanent terminal such as an asynchronous terminal or a terminal-simulator process (such as $ZHOME) and then specify that device as the OSS Monitor home terminal. For example, to specify the $ZHOME process as the home terminal for the OSS Monitor, enter:

```
OSSMON /NAME $ZPMON, NOWAIT, TERM $ZHOME, PRI 180/
```

Naming the OSS Monitor Process

The OSS Monitor must be given the process name $ZPMON. An attempt to run the OSS Monitor terminates immediately if $ZPMON is already running or if the OSS Monitor is given a process name other than $ZPMON.

If the wrong process name is specified, the OSS Monitor abends with completion code 3 (Abnormal, voluntary, but premature termination) and with the Subsystem Programmatic Interface (SPI) error-number token ZSPI^ERR^BAD^NAME.

Using OSS Monitor Commands

OSS Monitor management operations are performed using the Subsystem Control Facility (SCF) to enter OSS Monitor commands. When the OSS Monitor is running, anyone with the proper permission can enter SCF commands to the OSS Monitor.

To start SCF from a TACL prompt, issue the following command:

```
SCF
```

The SCF prompt (->) appears. Issue the following SCF command:

```
ASSUME PROCESS $ZPMON
```

to begin to communicate with the OSS subsystem. ($ZPMON is the name of the OSS Monitor process.) You can now enter OSS Monitor commands; however, some commands are restricted for use only by members of the super group and others are restricted for use only by the super ID.

Using Wildcard Characters in OSS Monitor Commands:

- You can use standard UNIX file and directory wildcard characters when specifying an entity name given as a parameter in certain SCF commands (as indicated in Section 12, Open System Services Monitor). An entity name is the name of a fileset, an OSS name server, the OSS sockets local server, or the OSS message-queue server.
- The wildcard characters are the asterisk (*), question mark (?), and square brackets ([ ]). The use of these characters is described in Table 2-3 on page 2-14. Characters used in wildcard matches are not case-sensitive.

Table 2-3. Wildcard Characters in OSS Monitor Commands

Characters	Uses and Examples
*	An asterisk represents any number of characters (including zero) in an entity name. Use this character to save keystrokes when: ● Entering a single name. For example, you can enter RO* for ROOT. ● Naming many entities at once. For example, `/us*` matches `/user1`, `/us`, and `/usr`.
?	A question mark represents any single character. For example, h?p matches hop and hip, but not help.
[ ]	Square brackets enclose a choice of characters you want to match. You can specify a range of characters by separating them with a hyphen, with the lower ASCII value to the left of the hyphen. Uppercase A has a lower ASCII value than lowercase a, and there are characters between Z and a. For example: ● [Cc]hapter matches both Chapter and chapter. ● [ch]apter matches both capter and hapter (but not chapter). ● chapter[1-3] matches chapter1, chapter2, and chapter3.

- For example:
  - The SCF command:

    ```
    INFO /OUT $S.#srv.info/ SERVER $ZPMON.*
    ```

    returns descriptions of all the servers in the ZOSSSERV file.
  - The SCF command:

    ```
    STATUS /OUT $S.#file.stat/ FILESET $ZPMON.*user*
    ```

    and the SCF command:

    ```
    STATUS /OUT $S.#file.stat/ FILESET $ZPMON.*USER*
    ```

    both return the status of every fileset in the ZOSSFSET file that contains the string “USER” in its name.

Creating Command Aliases:

- You can use the SCF ALIAS command to create aliases for OSS Monitor commands. Such aliases can be similar to UNIX commands. For detailed information about the SCF ALIAS command, see the *SCF Reference Manual for G-Series RVUs* or the *SCF Reference Manual for H-Series RVUs*.

- For example, you can create the alias DF for the STATUS FILESET command to simulate the UNIX `df` command by doing the following at SCF prompts:

  1. Use the SCF ASSUME command to access the OSS Monitor:

     ```
     ASSUME PROCESS $ZPMON
     ```

  2. Issue the SCF ALIAS command for the desired alias:

     ```
     ALIAS DF STATUS FILESET
     ```

  3. Display the status of the fileset USER1 by entering this command:

     ```
     DF USER1
     ```

- Other aliases you could define that are similar to UNIX commands include:

Alias for UNIX Command	Corresponding SCF Command
`fsck`	DIAGNOSE FILESET
`mkfs`	ADD FILESET
`mount`	START FILESET
`umount`	STOP FILESET

Stopping the OSS Monitor

Stop the OSS Monitor process by doing one of the following:

- If the OSS Monitor is running as a standard process, then, as the super ID, enter at a TACL prompt:

  ```
  STOP $ZPMON
  ```

- If the OSS Monitor is running as a persistent process, then, as a member of the super group, enter at an SCF prompt:

  ```
  ABORT PROCESS $ZZKRN.#ZPMON
  ```

Obtaining Information About the OSS Subsystem

You can use SCF commands to obtain information as described in the following subsections:

- Checking Whether a Subsystem Process Is Running on page 2-15
- Listing the Objects Managed by the OSS Monitor on page 2-17
- Checking the Configuration of the OSS Monitor on page 2-17
- Checking the Version of the OSS Monitor on page 2-18

Checking Whether a Subsystem Process Is Running

There is no OSS Monitor command that provides the state of an OSS subsystem process. To determine the state of an OSS subsystem process, you must use the basic SCF LISTDEV command.

For example, if you enter LISTDEV at an SCF prompt, a display similar to that shown in Figure 2-3 would appear. In the figure, the OSS Monitor process $ZPMON is shown as:

- Running, with:
  - Its primary process in processor 1 at high PIN 372
  - No backup process, indicated by processor 0 with PIN 0
- Having a:
  - Logical device number of 581
  - Device type of 24
  - Device subtype of 0
  - Scheduling priority of 180
- Using the object file \NODE1.$SYSTEM.SYS01.OSSMON

Other servers used by OSS processes (for example, the $ZTC*n* server mentioned under OSS Sockets on page 1-15) and by the OSS subsystem also appear in the figure.

Note. A copy of the TCP/IP process is not necessarily given the default name of $ZTC0. However, as long as either the DEFINE =TCPIP^PROCESS^NAME matches the running TCP/IP process or the OSS application calls the `socket_transport_name_set( )` function to select the TCP/IP process name, OSS `AF_INET` and `AF_INET6` sockets function properly.

Figure 2-3. Sample SCF LISTDEV Command Display

```
LDev Name      PPID    BPID   Type    RSize Pri Program
  .
  .
  .
 65  $ZZKRN    0,15    1,30    (66,0 )   132 180 \NODE1.$SYSTEM.SYS01.OZKRN
 66  $ZZWAN    0,274   1,287   (50,3 )   132 180 \NODE1.$SYSTEM.SYS01.WANMGR
 67  $ZZW05    5,262           (50,0 )     0 199 \NODE1.$SYSTEM.SYS01.CONMGR
 68  $ZZW04    4,262           (50,0 )     0 199 \NODE1.$SYSTEM.SYS01.CONMGR
 69  $ZZSTO    0,275   1,324   (65,0 )  4096 180 \NODE1.$SYSTEM.SYS01.TZSTO
 70  $ZZLAN    0,14    1,20    (43,0 )   132 180 \NODE1.$SYSTEM.SYS01.LANMAN
 71  $ZZFOX    7,282   6,275   (27,0 )   132 199 \NODE1.$SYSTEM.SYS01.FOXMON
 78  $ZSNET    0,15    1,30    (66,0 )   132 180 \NODE1.$SYSTEM.SYS01.OZKRN
 79  $ZSMS     5,30    4,39    (52,0 )  4096 180 \NODE1.$SYSTEM.SYS01.OMP
 86  $ZQ09     9,9             (45,0 )   132 201 \NODE1.$SYSTEM.SYS01.QIOMON
  .
  .
  .
438  $ZTNP0    0,357   1,289   (46,0 )  6144 170 \NODE1.$SYSTEM.SYS01.TELSERV
  .
  .
  .
580  $ZTC0     4,324   5,319   (48,0 ) 32000 170 \NODE1.$SYSTEM.SYS01.TCPIP
581  $ZPMON    1,372           (24,0 )  4096 180 \NODE1.$SYSTEM.SYS01.OSSMON
587  $ZTC1     5,321   4,319   (48,0 ) 32000 170 \NODE1.$SYSTEM.SYS01.TCPIP
  .
  .
  .
```

Listing the Objects Managed by the OSS Monitor

When the OSS Monitor is running, you can use the SCF NAMES command to identify the objects managed by the OSS Monitor. If you enter the following SCF commands:

```
ASSUME PROCESS $ZPMON
NAMES
```

you can determine:

- All currently configured objects being managed
- All currently configured objects being managed that are of a specific subtype
- All currently configured objects being managed that are in, or not in, a specific state

See the SCF NAMES Command for the command syntax and an example.

Checking the Configuration of the OSS Monitor

When the OSS Monitor is running, you can use the SCF INFO SUBSYS command to determine the current configuration settings for the OSS Monitor in the ZOSSPARM database file. To determine:

- The timeout value in seconds for requests to an OSS name server
- The current default spooler location assigned when no REPORT attribute is configured for a fileset
- The current default processor in which the FSCK utility starts when the subsystem needs to run it and either no processor is configured for the fileset or the processor configured for the fileset is unavailable
- The name of the Guardian disk volume that contains the program files to be used for the CVT utility, the OSS Monitor, and other OSS environment management software
- The setting for the automatic startup service

enter the following SCF commands:

```
ASSUME PROCESS $ZPMON
INFO SUBSYS
```

The information displayed is the configuration currently used by the OSS Monitor.

The syntax of the SCF INFO SUBSYS command and an example appear under INFO SUBSYS, INFO MON, and INFO PROCESS Commands on page 12-57.

Checking the Version of the OSS Monitor

When the OSS Monitor is running, you can use the SCF VERSION command to determine the product-version information for the running copy of the OSS Monitor. For example, if you enter the following SCF commands:

```
ASSUME PROCESS $ZPMON
VERSION
```

the system displays basic product-version information for the running OSS subsystem.

The use of product-version information is described in Gathering Version Information About OSS Files on page 11-1.

The syntax of the SCF VERSION command and an example appear under VERSION SUBSYS, VERSION MON, and VERSION PROCESS Commands on page 12-82.

Changing the OSS Subsystem Configuration

If the OSS Monitor is currently running, you can change the configuration of the OSS Monitor in the ZOSSPARM database file by using the SCF ALTER SUBSYS command. You can change:

- The timeout value used for requests to an OSS name server
- The default spooler location assigned when no REPORT attribute is configured for a fileset
- The default processor in which the FSCK utility is started when the subsystem needs to run it and either no processor is configured for the fileset or the processor configured for the fileset is unavailable
- The name of the Guardian disk volume that contains the program files to be used for the CVT utility, the OSS Monitor, and other OSS environment management software
- The setting for the automatic startup service

The syntax of the SCF ALTER SUBSYS command and an example appear under ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34.

The changed values are effective immediately.

Enabling the Automatic Startup Service

To enable the automatic startup service:

1. Start the OSS Monitor without using the AUTOSTART option. See Starting the OSS Monitor on page 2-7.
2. Use the SCF ALTER command to set the DESIREDSTATE attribute to STARTED for all filesets and servers that are to be automatically started. See ALTER

FILESET Command on page 12-20 and ALTER SERVER Command on page 12-28.

3. Use the SCF ALTER SUBSYS, AUTOSTART AUTO command to enable the automatic startup service and initiate automatic startup. See ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34.

Removing the OSS File System

This action can be performed in one of these ways:

- Using the procedure described in this subsection.
- Using the OSSREMOV utility. See OSSREMOV Utility on page C-17 for more information.

To remove the entire OSS file system (all filesets) from your local NonStop S-series or NonStop NS-series node:

1. Notify all users that the OSS file system is shutting down. Follow the procedure described under Sending Warnings to Users on page 2-5.
2. Use the OSS shell `ps` command to identify all running OSS processes and then stop those processes with the OSS shell `kill` command.
3. Use the SCF INFO FILESET * command to locate the mount point of each fileset if you do not maintain a diagram of mount points.
4. Use the SCF STOP FILESET * command. You must stop filesets in a specific order; a fileset cannot be stopped until every fileset mounted on it is stopped. The * specification stops all filesets in the correct order. The last fileset to stop is the root fileset.
5. Use the SCF DIAGNOSE FILESET command with the REPAIR ALL option on all filesets.
6. Use the SCF START FILESET command on all filesets.
7. Start an OSS shell with the TACL OSH command.
8. Back up all OSS filesets. Use the procedure under Backing Up User Files on page 6-15 for all files accessible from your root directory (`/`). Remember to avoid including files in `/dev`, `/G`, and `/E`.
9. Use the OSS shell `exit` command to stop your OSS shell.
10. Use the SCF STOP FILESET * command to unmount all filesets.
11. Use the SCF DELETE FILESET command to delete each fileset. This action deletes all files in the fileset, the catalog files for the fileset, and the fileset record in the configuration database.

12. Remove all previously saved catalogs by entering the following TACL command:

    RUN $*tsvvol*.ZOSS.CVT PURGE SERIAL *mmmm* IN $*old_vol*.ZX0*nnnnn*

    $*tsvvol*

    specifies the disk volume of the installation target subvolume (TSV) containing the CVT utility. See Managing and Repairing Fileset Catalog Files on page 5-40 for more information on the CVT utility.

    mmmm

    specifies the FSCK serial number of a previous FSCK diagnose or repair operation. See Generated Catalog Files on page 5-33 for more information on FSCK serial numbers.

    $*old_vol*

    specifies the disk volume on which you previously saved catalog volume files; such files have names of the form:

    PX*ffmmmm*

    ff

    specifies one of the following:

IN	An old inode file for the catalog
LI	An old link file for the catalog
LO	An old FSCK log file for the catalog

    mmmm

    is the FSCK serial number.

    nnnnn

    specifies the device identifier for an OSS fileset.

When this procedure is completed, no one can use an OSS file until you reconfigure all filesets and restore or reinstall OSS files. See the *Open System Services Installation Guide* for information on installing the complete OSS product set.

Monitoring OSS Processes

You can monitor OSS process execution using:

- OSS shell commands with special HP extensions
- Guardian TACL commands

Monitoring OSS Processes From the OSS Environment

The OSS shell `ps` command has a `-W` flag with many options for monitoring processes from the OSS environment. You can monitor processes:

- In a given processor
- By Guardian process name
- By Guardian filename
- By priority
- By user
- By NonStop S-series or NonStop NS-series node
- By using terminal

See the `ps(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about that facility.

Monitoring OSS Processes From the Guardian Environment

The TACL STATUS command in the Guardian environment reports information about processes running in the OSS environment:

- A STATUS command for an OSS process returns information about the pathname of the executing program file and indicates that the process has an OSS process ID. For example, the TACL command:

  ```
  STATUS 4,354
  ```

  would return a display similar to Figure 2-4 when that process is the OSS shell. In the output, the `X` to the left of the Guardian *`cpu`*,*`pin`* under “Process” indicates that the process has an OSS process ID.

Figure 2-4. TACL STATUS Display for an OSS Process

```
System \NODE1

Process            Pri PFR %WT Userid  Program file           Hometerm
        X 4,354    130 000     103,225 /bin/sh                $ZTN0.#PTVNNP7
                   Swap File Name: $OSS001.#0
```

- A STATUS, DETAIL command provides more information in a different format. For example, a STATUS *, DETAIL command could return information about a copy of an iTP WebServer utility process among the rest of its output, as shown in Figure 2-5 on page 2-22.

Figure 2-5. TACL STATUS, DETAIL Display for an OSS Process

```
.
.
.
System: \NODE1                              June 4, 2001  12:09
Pid: 9,326    ($Z0S5)    Primary
Priority: 160
Wait State: %001    (LREQ)
Userid: 255,44    (SUPER.WEBMASTR)
Myterm: $ZTN1.#PT74DAB
Program File Name: $ROOT.ZYQ00000.Z00116NL
Swap File Name: $ROOT.#0
Process Time: 0:0:0.008
Process Creation Time: February 17, 2001 16:53:25.940213
Process States: RUNNABLE
GMOMJOBID:
OSS Pathname: /usr/tandem/webserver/applications/sis/_oss/bin/vtopic.pway
OSS Arguments: -configpath
/usr/tandem/webserver/applications/sis/_oss/bin/inetsrch.ini
OSS PID: 24379404
.
.
.
```

The SCF PROCESS INFO and PROCESS STATUS commands for the NonStop Kernel subsystem can provide information about persistent OSS processes. These commands work for:

- The OSS Monitor (see Starting the OSS Monitor as a Persistent Process on page 2-9)
- Any OSS server configured under the NonStop Kernel subsystem by OSS EasySetup (see EasySetup Utilities on page C-7)
- Any OSS application configured within the NonStop Kernel subsystem (see Making OSS Application Processes Persistent with the Kernel Subsystem on page 2-23)

Managing OSS Processes

You can manage the availability of OSS processes in several ways:

- Using the Kernel Subsystem, as described in Making OSS Application Processes Persistent with the Kernel Subsystem on page 2-23
- Using OSS tools that schedule periodic tasks, as described in Using the cron Process on page 2-35
- Using Guardian tools that schedule periodic tasks, as described in Using the NetBatch Product on page 2-38

You can manage workload-related behaviors of OSS processes best from the OSS environment using OSS shell `run` command options that correspond to TACL RUN command options. See Managing OSS Process Scheduling on page 2-30 and Managing OSS Process Processor Use on page 2-33.

Making OSS Application Processes Persistent with the Kernel Subsystem

Just as the OSS Monitor can be made a persistent process using the SCF interface to the NonStop Kernel subsystem, the same SCF interface provides commands to configure, start, and stop OSS application processes that are persistent. A persistent process is automatically started when certain criteria are met and then automatically restarted if events make the original copy of the process unavailable. Such processes are called generic processes in SCF documentation. This subsection provides suggestions for possible uses of this facility; for a complete description of the commands, process attributes, and parameters, see the *SCF Reference Manual for the Kernel Subsystem*.

Generic OSS processes have the following characteristics:

- Both shell scripts and compiled programs can be made persistent.
- Each OSS process is started by configuring a copy of the Guardian OSH utility to start it.
- Each process has the following configurable attributes under the persistence manager (attributes are the same as for a Guardian generic process):
  - AUTORESTART
  - CPU – Applies to names used for both NAME and ASSOCPROC attributes
  - HOMETERM – Always the terminal used by the TACL for the ADD command
  - MEMPAGES
  - NAME – Always identifies a copy of OSH
  - PRIMARYCPU
  - PRIORITY – Always use the default priority
  - PROGRAM – Always specifies $SYSTEM.SYSTEM.OSH
  - SAVEABEND
  - STARTMODE – Always must be MANUAL
  - STOPMODE
  - TYPE
  - STARTUPMSG – Specifies a string of up to 128 characters to pass to OSH
  - USERID
- As an alternative to specifying the HOMETERM attribute, you can use OSS shell input/output redirection at the end of the STARTUPMSG value. For example:

  ```
  STARTUPMSG "... <- >>out 2>>err"
  ```

  The appended information is interpreted by the OSS shell as follows:

  ```
  <-
  ```

  closes standard input

`>>out`

redirects standard output to the file `out` in the current working directory of the OSS process

`2>>err`

redirects standard error to the file `err` in the current working directory of the OSS process

See the `sh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about OSS shell redirection.

- Do not use the OSS shell `nice` command in the STARTUPMSG value or in a script started by the STARTUPMSG value. Similarly, programs run as persistent processes should not use the `nice()` function. Persistent processes with nondefault priorities can cause problems.
- OSS generic processes also have the attribute ASSOCPROC, which provides a name for the process itself, separate from the name of the OSH process that launches it:
  - The value used for ASSOCPROC must follow Guardian file-system naming conventions for processes.
  - You must ensure that this name is unique among all those configured or running on your node.
  - If you intend to run the process in more than one processor (for example, by using the CPU attribute), then the value used for ASSOCPROC cannot contain more than three characters in addition to the dollar sign ($) character.
  - The value configured for ASSOCPROC must be used in the `-name` flag of the command used to launch the process (must be part of the value in the STARTUPMSG attribute for OSH or embedded in a script called by part of the value in the STARTUPMSG attribute). If an intermediary process is created by OSH such that the OSH `-name` flag would apply to the wrong child process, then the OSS process must be started by a copy of the OSS shell `run` command that includes the `-name` flag to name the correct child process.
  - The OSS environment variable $ZCPU contains the number of the processor used for each child process launched. $ZCPU must be used to append the processor number to the base process name when a `-name` flag is specified. This environment variable ensures that process names remain unique within the node and allows you to synchronize the ASSOCPROC name generated for each processor when the CPU attribute is used with the process name used by the child process within the STARTUPMSG specification.

    $ZCPU is defined only when the CPU ALL attribute is used or the CPU attribute is specified as a list of processor numbers.

    $ZCPU cannot be defined unless an OSS shell is running, so it can only be used in an OSS shell script started by the STARTUPMSG specification. This

consideration means that an OSS persistent process that runs in more than one processor (an OSS persistent process with a valid CPU attribute) must be started from within a script that uses an OSS shell `run` command with a `-name` flag of the form:

`run -name` *`base_name`*`$ZCPU...`

See the `osh(1)` and `run(1)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the `-name` flags.

- An OSS persistent process does not need to run in the same processor as the OSH process that starts it. The OSH command `-cpu` flag can be used in the STARTUPMSG attribute to start the OSS persistent process in a specific, currently available processor. For example:

  `STARTUPMSG "-cpu` *n* `-p /bin/sh ..."`

  starts an OSS shell (`-p /bin/sh`) in processor *n*.

  The `-cpu` flag is not compatible with use of the CPU attribute and the $ZCPU environment variable. Do not use the `-cpu` flag when the same OSS persistent process will be run in more than one processor.

  See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the `-cpu` flag.

- All features available to the process when it is started from an OSS shell command line are available when it is started by the persistence manager:

  - The `/etc/profile` file that provides the initial environment for all processes when an OSS shell is started can provide the initial environment for the process; to use this feature, you must use the `-ls` flag in the OSH command that launches the process (must be part of the STARTUPMSG attribute).

    See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the `-ls` flag.

  - The `.profile` file associated with the user ID during logon to OSS can provide the initial environment for the process; the USERID attribute provides the user ID that determines the `.profile` file used.

  - Any argument can be passed to the process that would be entered in a command line; the STARTUPMSG attribute contains these arguments.

  - Environment variables (`env` and `environ`) can be retrieved by the process.

  - The standard input, output, and error files are supported, with redirection governed by the rules that apply to the OSH command. If the process requires that tty devices provide or use these files, then redirection to appropriate terminal simulation devices must be done in the STARTUPMSG attribute.

- When more than 128 characters are required to correctly start an application or script, a script requiring less than 128 characters in its STARTUPMSG attribute can be run to start the command or script that needs to be made persistent.

The following SCF commands for the NonStop Kernel subsystem support OSS generic processes:

- ABORT – Stops a persistent process
- ADD – Initially configures a persistent process
- ALTER – Changes the configuration of a persistent process
- DELETE – Removes the configuration for a persistent process
- INFO – Provides information about the configuration of a persistent process
- STATUS – Provides information about the current state of a persistent process
- START – Starts a persistent process when it is not started automatically

Examples

1. Suppose your site needs to monitor an application log and routinely runs the command

   ```
   tail -f log
   ```

   from an OSS shell prompt at a dedicated operator console. To make this process persistent, enter the following commands:

   a. From a TACL prompt:

      ```
      WHO
      ```

      which produces output that includes the HOMETERM value you need to use, such as:

      ```
      Home terminal: $ZTN0A.#PT4KH30
      ...
      ```

   b. Then, at an SCF prompt for the NonStop Kernel subsystem:

      ```
      ADD PROCESS OSSAPP,
          NAME $OSH1,
          AUTORESTART 10,
          HOMETERM $ZTN0A.#PT4KH30,
          PRIMARYCPU 1,
          STARTMODE MANUAL,
          USERID OSS.APPS,
          PROGRAM $SYSTEM.SYSTEM.OSH,
          ASSOCPROC $OSS1,
          STARTUPMSG "-ls -name /G/oss1 -p /bin/tail -f log"
      ```

   c. Because the configuration uses a STARTMODE of MANUAL, you must enter the following command at an SCF prompt to start the process:

      ```
      START PROCESS $ZZKRN.#OSSAPP
      ```

   These commands configure and start the persistent process object OSSAPP in processor 1, using the named process $OSH1 to launch the OSH utility. Then OSH

launches the named process $OSS1 that runs the program (`-p`) `/bin/tail` as if it were a login shell (`-ls`); the `tail` utility uses the file (`-f`) named `log` in the home directory of the user with the user ID OSS.APPS. Because of the `-ls` flag, the `tail` command inherits any environment variables defined in either `/etc/profile` or `$HOME/.profile` for the user ID OSS.APPS.

Once the SCF START command completes, the persistence manager monitors both the $OSH1 and $OSS1 processes. If either process fails, the persistence manager aborts the remaining process and makes up to 10 attempts to restart $OSH1.

To stop the persistent process object, enter the following command at an SCF prompt:

```
ABORT PROCESS $ZZKRN.#OSSAPP
```

This command stops both the $OSH1 and $OSS1 processes. $OSS1 is stopped such that it does not create a zombie process.

2. Suppose your site has a shell script with the filename `startmyapp`. One of the tasks performed by `startmyapp` (starting an application program named `myapp`) needs to be done by a persistent process. `startmyapp` contains the following:

```
#! /bin/sh
#
PATH=.:/usr/ucb:/usr/bin:/bin;
export PATH;
CONFIGURATION = $1;
#
run -name=/G/oss2 myapp $CONFIGURATION
```

where the value passed to `myapp` through the variable `CONFIGURATION` is supplied as a command line parameter (`$1`) each time `startmyapp` is invoked.

To configure and run `myapp` as a persistent process that uses the `CONFIGURATION` value of `reload`, enter the following commands:

a. From a TACL prompt:

```
WHO
```

which produces output that includes the HOMETERM value you need to use, such as:

```
Home terminal: $ZTN0A.#PT4KH30
...
```

b. Then, at an SCF prompt, enter:

```
ADD PROCESS OSSAPP,
    NAME $OSH2,
    HOMETERM $ZTN0A.#PT4KH30,
    AUTORESTART 5,
    PRIMARYCPU 2,
    STARTMODE MANUAL,
    USERID OSS.APPS,
```

```
PROGRAM $SYSTEM.SYSTEM.OSH
ASSOCPROC $OSS2,
STARTUPMSG "-ls -p /bin/sh startmyapp reload"
```

c. Because the configuration uses a STARTMODE of MANUAL, you must enter the following command at an SCF prompt to start the persistent process:

```
START PROCESS $ZZKRN.#OSSAPP
```

These commands configure and start the persistent process object OSSAPP in processor 2, using the named process $OSH2 to launch the OSH utility. Then OSH starts a login (`-ls`) shell (`-p /bin/sh`) that executes the script `startmyapp` and passes the value `reload` to it.

`startmyapp` uses the OSS `run` command to start the program (`-p`) `myapp` as the named process (`-name`) $OSS2. Because of the `-ls` flag, `myapp` inherits any environment variables defined in either `/etc/profile` or `$HOME/.profile` for the user ID OSS.APPS.

Once the SCF START command completes, the persistence manager monitors both the $OSH2 and $OSS2 processes. If either process fails, the persistence manager aborts the remaining process and makes up to five attempts to restart $OSH2.

To stop the persistent process object, enter the following command at an SCF prompt:

```
ABORT PROCESS $ZZKRN.#OSSAPP
```

This command stops both the $OSH2 and $OSS2 processes. $OSS2 is stopped such that it does not create a zombie process.

3. Suppose the shell script with the filename `launchmyapp` must be run constantly in each available processor because one of the tasks performed by `launchmyapp` (starting an application program named `myapp`) needs to be done by a persistent process. `launchmyapp` contains the following:

```
#! /bin/sh
#
PATH=.:/usr/ucb:/usr/bin:/bin;
export PATH;
CONFGURATION = $1;
#
run -name=/G/oss$ZCPU myapp $CONFIGURATION
```

where:

- The value passed to `myapp` through the variable `CONFIGURATION` is supplied as a command line parameter (`$1`) each time `launchmyapp` is invoked
- The unique value used for the `myapp` process name consists of the base character string "oss" and the number of the processor running the copy of the child process, derived by appending the OSS environment variable `$ZCPU` to the base string.

To configure and run myapp in all processors as a persistent process that uses the CONFIGURATION value of reload, enter the following commands:

a. From a TACL prompt:

```
WHO
```

which produces output that includes the HOMETERM value you need to use, such as:

```
Home terminal: $ZTN0A.#PT4KH30
...
```

b. Then, at an SCF prompt, enter:

```
ADD PROCESS OSSAPP,
    NAME $OSH,
    HOMETERM $ZTN0A.#PT4KH30,
    AUTORESTART 5,
    CPU ALL,
    STARTMODE MANUAL,
    USERID OSS.APPS,
    PROGRAM $SYSTEM.SYSTEM.OSH
    ASSOCPROC $OSS,
    STARTUPMSG "-ls -p /bin/sh launchmyapp reload"
```

c. Because the configuration uses a STARTMODE of MANUAL, you must enter the following command at an SCF prompt to start the persistent process:

```
START PROCESS $ZZKRN.#OSSAPP
```

These commands configure and start the persistent process object OSSAPP in all processors (00 through 15, assuming a 16-processor system), using the named processes $OSH00 through $OSH15 to launch the OSH utility. Then each copy of OSH starts a login (-ls) shell (-p /bin/sh) that executes the script launchmyapp and passes the value reload to it.

Each copy of launchmyapp uses the OSS run command to start the program (-p) myapp as the named process (-name) $OSS*nn*, where *nn* is the value of $ZCPU for a specific processor. Because of the -ls flag, myapp inherits any environment variables defined in either /etc/profile or $HOME/.profile for the user ID OSS.APPS.

Once the SCF START command completes, the persistence manager monitors all of the $OSH*nn* and $OSS*nn* processes. If any of these processes fails, the persistence manager aborts the remaining process and makes up to five attempts to restart the corresponding $OSH*nn*.

To stop the persistent process object, enter the following command at an SCF prompt:

```
ABORT PROCESS $ZZKRN.#OSSAPP
```

This command stops all the $OSH*nn* and $OSS*nn* processes. Each copy of $OSS*nn* is stopped such that it does not create a zombie process.

Managing OSS Process Scheduling

A frequently used process might not obtain adequate processor time when it runs with a default priority. If that happens, you can assign a nondefault priority to the process.

In the Guardian environment, you would write a CMON process to control the scheduling of other processes. In the OSS environment, you can start processes with nondefault priorities using the OSS `nice` command. You can also use a shell script or a shell alias to achieve this result.

Normal Guardian environment mechanisms for changing process priorities also can be used on OSS processes.

An OSS program can modify its own scheduling priority by changing its `nice` value. Note that the priority value assigned using the `nice()` function in an OSS program or using the `nice` command is not the same value as that used for Guardian environment commands. Whether an increased value indicates an increased or decreased priority depends on the environment in which you use a command. The Guardian environment convention for relative scheduling priority is the opposite of the UNIX convention used in the OSS environment.

Using the nice Command

The `nice` command lets you start an OSS process at a modified priority. All users can lower the execution priority of a process, but only the super ID can increase the execution priority of a process.

Priority values in a traditional UNIX system have a significance opposite that of the values used on a NonStop S-series or NonStop NS-series server. For a description of the relationship among the `nice` value of a process, the OSS scheduling priority for the process, and the Guardian priority for the process, see the `run(1)`, `nice(1)`, and `nice(2)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual* and *Open System Services System Calls Reference Manual*.

You can use the `nice` command to adjust the response time of servers that run in the OSS environment. For example, to start the `inetd` process at a priority slightly higher than the default priority, enter:

```
nice -n -1 /usr/ucb/inetd
```

To specify a lower priority for the `inetd` process, enter:

```
nice -n 15 /usr/ucb/inetd
```

This command executes the `inetd` process and decreases the priority by 15, which is a priority even lower than the default priority set by `nice`.

Using an OSS Shell Script

Unlike a shell alias, the shell script method works for calls to a program from within another shell script. To use an OSS shell script that changes the default priority of a program, follow these steps:

Note. Using shell scripts increases system overhead. Be sure that the performance tradeoff is worthwhile before you use this technique.

---

1. Create a script file that has the same name as the program for which you want to change the default priority. Put the following information in the script file:

   ```
   #! /bin/sh
   nice -n nn program_name
   ```

   nn

   is the priority adjustment for the process.

   program_name

   is the name of the program to be executed.

2. Place the script file in a special directory you control. Secure the file so that it can be executed by everyone but not altered by anyone.

3. Add your special directory to the PATH environment variable used to find program files. You need to do this only the first time you create such a script, provided you always use the same OSS directory.

   Most users use the PATH definition in `/etc/profile`, so change that file such that your special directory is searched first.

4. Notify those users that have their own `.profile` file or otherwise alter the PATH variable about what you are doing so that they can make comparable changes to their definitions of PATH.

For example, to run a program called `logout` that is normally stored in `/usr/bin` with a priority decreased by 5, do the following:

1. Put the following lines into a file named `/usr/local/script/logout`:

   ```
   #! /bin/sh
   nice -n 5 logout
   ```

2. Secure the file for only read and execute access by entering:

   ```
   chmod a=rx /usr/local/script/logout
   ```

3. Change the PATH environment variable in `/etc/profile` to something similar to the following:

   ```
   export PATH=/usr/local/script:/bin:/bin/unsupported:
   /usr/ucb:/usr/bin
   ```

4. Post a broadcast message to users, using the technique described in Manually Stopping the OSS File System and the OSS Environment on page 2-3.

This procedure can be used to alter the scheduling priority of any process, including those released by HP as part of the OSS environment. For example, the following steps decrease the scheduling priority for the c89 compiler by 15:

1. Put the following lines into a file named `/usr/local/script/c89`:

   ```
   #! /bin/sh
   nice -n 15 c89 -c *.c
   ```

2. Secure the file for only read and execute access by entering:

   ```
   chmod a=rx /usr/local/script/c89
   ```

3. Change the `PATH` environment variable in `/etc/profile` to something similar to the following:

   ```
   export PATH=/usr/local/script:/bin:/bin/unsupported:/usr/ucb:
   /usr/bin
   ```

4. Post a broadcast message to users.

Using an OSS Shell Alias

To use a shell alias that changes the default priority of a program, follow these steps:

1. Add an alias in `/etc/profile`. This alias should have the form:

   `alias` *`program`*`=nice -n` *`nn object_file_path`*

   `program`

   is the program name usually entered by the user as an OSS shell command.

   `nn`

   is the priority adjustment for the process.

   `object_file_path`

   is the OSS pathname for the program object file to be executed as the process.

   When users without their own `.profile` file attempt to run the program, your alias will be found instead of the program file.

2. Notify those users that have their own `.profile` file about what you are doing so that they can make comparable changes to their file.

Managing OSS Process Processor Use

You can also use OSS shell scripts or aliases to force specific processes to execute in specific processors to distribute the work load within your NonStop S-series or NonStop NS-series node.

Note. Using shell scripts increases system overhead. Be sure that the performance tradeoff is worthwhile before you use this technique.

To change the default processor for a specific process:

1. Create a script file that has the same name as the program for which you want to change the default processor. Put the following information in the script file:

   ```
   #! /bin/sh
   run -cpu=nn object_file_path "$@"
   ```

   nn

   is the processor number of the processor you want the process to run in.

   object_file_path

   is the OSS pathname for the program object file to be executed as the process.

   `"$@"`

   causes any parameters entered for the process to be sent to the process.

2. Place the script file in a special directory you control, secured so that it can be executed by everyone but not altered by anyone.

3. Add your special directory to the `PATH` environment variable used to find program files. You need to do this only the first time you create such a script, provided you always use the same OSS directory.

   Most users use the `PATH` definition in `/etc/profile`, so change that file such that your special directory is searched first.

4. Notify those users that have their own `.profile` file or otherwise alter the `PATH` variable about what you are doing so that they can make comparable changes to their definitions of `PATH`.

For example, the following steps change the system to run all C language program compilations in processor 4:

1. Put the following lines into a file called `/usr/local/script/c89`:

   ```
   #! /bin/sh
   run -cpu=4 /usr/bin/c89 "$@"
   ```

2. Secure the file for only read and execute access by entering:

   ```
   chmod a=rx /usr/local/script/c89
   ```

3. Change the PATH environment variable in /etc/profile to something similar to the following:

   ```
   export PATH=/usr/local/script:/bin:/bin/unsupported:
   /usr/ucb:/usr/bin
   ```

4. Post a broadcast message to users.

Managing OSS Interprocess Communication Facilities

The OSS interprocess communication (IPC) facilities require little management. However, applications can fail and leave no-longer-used message queues, semaphore IDs, or shared memory segment IDs in the system. You can detect such wasted resources by regularly monitoring IPC facility use and correcting the situation when necessary.

OSS IPC facilities such as message queues, semaphores, and shared memory segments can be monitored and controlled with OSS shell commands. The `ipcs` command displays information for those IPC mechanisms. The `ipcrm` command removes facilities associated with failed processes. See the `ipcs(1)` and `ipcrm(1)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual*.

OSS named pipes (FIFOs) can be monitored using the OSS shell `ls -l -F` command on the directory containing the FIFO. The output from the command lists FIFOs with a p as the first character in the mode field.

OSS AF_UNIX (local) sockets have filenames and can also be watched using the `ls -l` command. The output from the command lists AF_UNIX sockets with an s as the first character in the mode field. OSS AF_UNIX sockets are administered by configuring, starting, and stopping their servers. See Section 4, Managing Servers, for information about OSS AF_UNIX sockets local servers.

OSS IPC facilities such as unnamed pipes and OSS AF_INET or AF_INET6 sockets are more difficult to monitor.

The `inetd` command provides OSS sockets with specific services, sometimes by starting the corresponding server. See the `inetd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. More information about the `inetd` command can be found in Starting a Network Services Server on page 4-38.

Scheduling Periodic Tasks

You can schedule programs or multiple-task OSS shell scripts to run in the OSS environment at predetermined times and intervals. The OSS environment offers two ways to do this: the `cron` utility with related OSS shell commands, and the NetBatch

product with the TACL OSH command. The following subsections discuss these options:

Using the cron Process

The `cron` process runs shell commands at specified dates and times. The commands that are to be run and the schedule are placed in entries within files in a specific directory. Commands that are to run only once are placed within files in the `at` queue. Commands that can be run at system-determined times are placed within files in the `batch` queue.

The `cron` process supports the following OSS shell commands:

Because the `cron` process exits only when killed or when the system stops, only one `cron` process should exist on the system at any given time. You can ensure that only one process runs by using the `CRON_NAMED` environment variable before starting any copy of `cron`; always use the same process name.

For more information about starting and managing the `cron` process, see the `cron(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Configuring the cron Process

You must create the following files to configure the `cron` process. The `cron` process uses these files to determine whether and when to run shell commands:

`/usr/lib/cron/at.deny`	Lists the user IDs that are denied access to the `at` queue
`/var/spool/cron/crontabs/`*`crontab`*	Lists the commands to be run at specified times.
`/var/adm/cron/cron.allow`	Lists the user IDs that are allowed to change the `crontab` file. (This file is optional; see the `crontab(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information.)

`/var/adm/cron/cron.deny`	Lists the user IDs that are denied access to the `crontab` file.
`/var/adm/cron/.proto`	Contains shell commands required to provide the correct shell environment for `at` and `batch` jobs.
`/var/adm/cron/queuedefs`	Configures the task queues to be used for the `at`, `batch`, and `cron` commands.

HP provides sample files for the `at.deny`, `cron.deny, .proto`, and `queuedefs` files. You can create the working copies of these files by entering the following at an OSS shell prompt before you use the `cron` process for the first time:

```
cp /usr/lib/cron/at.deny.sample    /usr/lib/cron/at.deny
cp /var/adm/cron/cron.deny.sample  /var/adm/cron/cron.deny
cp /var/adm/cron/.proto.sample     /var/adm/cron/.proto
cp /var/adm/cron/queuedefs.sample     /var/adm/cron/queuedefs
```

Do not repeat this action after updates to the OSS shell product (T8626) or you will overwrite your site-specific modifications with HP's sample data.

To add commands to the `crontab` file, use the `crontab` command as described in The crontab Command on page 2-36. To add commands to an `at` queue file, use the `at` command as described in The at Command on page 2-37.

To monitor the task queues, use the `atq` command as described in The atq Command on page 2-37. To remove unwanted entries, use the `atrm` command as described in The atrm Command on page 2-38. To add processes that the system determines when to run, use the `batch` command as described in The batch Command on page 2-38.

During process initialization and when `cron` detects a change, it examines the files in the `/var/spool/cron/crontabs` directory and the `at` queue files. This strategy reduces the overhead of checking for new or changed files at scheduled intervals.

The `cron` command creates a log of its activities. The `cron` process starts each job with the following process attributes stored with the job by the invoking process:

- Effective and real user IDs
- Effective and real group IDs
- Supplementary groups

The crontab Command

The `crontab` utility adds files that you specify to the `/var/spool/cron/crontabs` directory. You can use the `crontab` command to read from the standard input file or accept as arguments the names of commands to be run and when the commands are to be run.

For more information about the crontab command, see the crontab(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

The at Command

The at command runs OSS shell commands at a time you specify. You can use the at command to read from the standard input file or accept as arguments the names of commands to be run and when the commands are to be run. If a file specified in an at command is executable (that is, has the x permission for the user executing the command), at treats the file as a command and the job consists only of this command. If the file is not executable, at uses the file's contents as the instructions for the job. If at cannot find the file, the specification is passed to the date parser. If the date parser does not recognize the specification, you receive an error message.

Variables in the shell environment, the current directory, umask, and ulimit are retained when the commands are run; for more information about the mechanism used to do this, see the .proto(4) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. Open file descriptors, interrupts, and priority are lost.

The at command and the cron process use these files to determine whether and what commands to run:

/var/spool/cron/atjobs/*at*	Lists the commands to be run once.
/usr/lib/cron/at.allow	Lists the user IDs that are allowed to change the at file.
/usr/lib/cron/at.deny	Lists the user IDs that are denied access to the at file.

You can use the at command if your login name appears in the /usr/lib/cron/at.allow file. If that file does not exist, the at command checks the /usr/lib/cron/at.deny file to determine whether your login name is denied access to at. If neither file exists, only a user who has appropriate privileges can submit a job. If the at.allow file does exist, it must include the login name of a user who has appropriate privileges to use the at command.

The at command also allows you to:

- Specify a file to be used as input instead of the standard input file
- Find out information about jobs that are queued

For more information about the at command, see the at(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

The atq Command

The atq command prints the queue of jobs that were created using the at command and are waiting to be run at a later time. If you have appropriate privileges and you

specify one or more user names, only jobs belonging to those users are displayed. If you do not specify any user names, a list of all jobs submitted is displayed.

The `atq` command allows you to sort the output in chronological order based on the time that the `at` command was issued. You can also specify which queue you want to have printed.

For more information about the `atq` command, see the `atq(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

The atrm Command

The `atrm` command removes jobs queued by the `at` command. If you specify job numbers, `atrm` attempts to remove only those jobs. If you specify user names, `atrm` removes all jobs belonging to those users.

For more information about the `atrm` command, see the `atrm(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

The batch Command

The `batch` command runs commands at a system-determined later time and when the system workload level permits. The `batch` command is equivalent to the following `at` command:

```
at -q b now
```

where queue `b` is an `at` queue for batch jobs.

For more information about the `batch` command, see the `batch(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Using the NetBatch Product

To use the NetBatch or NetBatch Plus product to schedule programs successfully, you need to know:

- OSS programs that use the OSS file system can open local terminal windows and local disk files when started using this method. Remote devices, processes such as the spooler, and terminals cannot be opened unless OSSTTY is also used. OSSTTY can be used to redirect one or more OSS standard files to Guardian EDIT files or Guardian processes. See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about starting OSSTTY with the OSH command and OSSTTY on page C-1 for more information about starting OSSTTY as a server.
- To the NetBatch product, the TACL OSH command is an executor program.
- Information generated by any periodic task should be appended to output files, so that each execution of the task does not overwrite output from the prior execution.

- The general form of the TACL OSH command for batch execution of a program in the OSS environment is:

  `OSH <- >>`*`out_file`* `2>>`*`err_file`* `-p` *`program_path program_args`*

  `<-`

  indicates that the OSS shell started by the command ignores its standard input (`stdin`) file.

  `>>`

  indicates that the OSS shell appends normal output to the identified file instead of the standard output (`stdout`) file.

  `out_file`

  is the OSS pathname of the file to receive normal output.

  If the initial working directory is not the current working directory, relative pathnames are resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and normal output is saved in a file-code-180 file in that subvolume.

  `2>>`

  indicates that the OSS shell appends exception output to the identified file instead of the standard error (`stderr`) file.

  `err_file`

  is the OSS pathname of the file to receive exception output.

  If the initial working directory is not the current working directory, relative pathnames are resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and exception output is saved in a file-code-180 file in that subvolume.

  `-p`

  explicitly runs a program file in the OSS environment. Note that `-P` (uppercase P) is not a valid command option.

  `program_path`

  specifies the OSS pathname for the program to be run.

  If the initial working directory is not the current working directory, relative pathnames are resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is

specified, the specified pathname must be a Guardian file identifier and the program file must reside in that subvolume.

program_args

specifies any arguments expected on a command line by the *program_path* program. Such arguments would include the name of an input file.

- The general form of the TACL OSH command for batch execution of an OSS shell script with output routed to OSS files is:

  OSH [-c] *script_path* <- >>*script_out_file* 2>>*err_file*

  -c

  implicitly runs the OSS shell to execute the script as a command. Note that -C (uppercase C) is not a valid command option.

  script_path

  specifies the OSS pathname for the OSS shell script file to be run.

  If the initial working directory is not the current working directory, relative pathnames are resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and the script file must reside in that subvolume.

  <-

  indicates that the OSS shell started by the command ignores its standard input (stdin) file.

  >>

  indicates that the OSS shell appends normal output to the identified file instead of the standard output (stdout) file.

  script_out_file

  is the OSS pathname of the file to receive normal output.

  If the pathname is not fully qualified, it is resolved from the current working directory.

  2>>

  indicates that the OSS shell appends error output to the identified file instead of the standard error (stderr) file.

  err_file

  is the OSS pathname of the file to receive exception output.

If the initial working directory is not the current working directory, relative pathnames are resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and exception output is saved in a file-code-180 file in that subvolume.

Site-written shell programs can also be used instead of the default OSS shell. See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for the syntax to use for nondefault shell programs.

- An alternate form of the TACL OSH command for batch execution of an OSS shell script allows creation of output files in an initial working directory (sometimes called a login directory) that differs from the current working directory. This form of the command is:

  `OSH -p sh <- >>`*`script_out_file`* `2>>`*`err_file script_path`*

  `-p sh`

  explicitly runs the OSS shell. Note that `-P` (uppercase P) is not a valid command option.

  `<-`

  indicates that the OSS shell started by the command ignores its standard input (`stdin`) file.

  `>>`

  indicates that the OSS shell appends normal output to the identified file instead of the standard output (`stdout`) file.

  `script_out_file`

  is the OSS pathname of the file to receive normal output.

  If the initial working directory is not the current working directory, a relative pathname is resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and normal output is saved in a file-code-180 file in that subvolume.

  `2>>`

  indicates that the OSS shell appends exception output to the identified file instead of the standard error (`stderr`) file.

  `err_file`

  is the OSS pathname of the file to receive exception output.

  If the initial working directory is not the current working directory, a relative pathname is resolved from the initial working directory. When the initial working

directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and exception output is saved in a file-code-180 file in that subvolume.

script_path

specifies the OSS pathname for the OSS shell script file to be run.

If the initial working directory is not the current working directory, a relative pathname is resolved from the initial working directory. When the initial working directory is a Guardian subvolume and only a relative pathname is specified, the specified pathname must be a Guardian file identifier and the script file must be a file-code-180 or file-code-101 file in that subvolume.

Site-written shell programs can also be used instead of the default OSS shell. See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for the syntax to use for nondefault shell programs.

- The general form of the NetBatch command for OSS use is:

```
BATCHCOM;SUBMIT JOB, IN filename, EVERY period * * * * when
```

filename

is the Guardian filename of a file containing the job statements.

period

indicates the time period between attempts to execute the OSS program or OSS shell script.

when

indicates the time span over which the schedule applies.

As an example of running an OSS program periodically, suppose:

- You want to run a Perl script every 60 minutes only on weekdays.
- Your site has Perl installed at `/bin/usr/perl` (Perl is not supplied with the OSS environment).
- You have created the script at `/script/hourly.pl`.
- You want to record normal output in the OSS file `/script/hourly.log` and error output in the OSS file `/script/hourly.err`.

You would:

1. Create a valid job file for the NetBatch product, named \NODE.$SYSTEM.SYSTEM.HOURLY and containing the following TACL command statement line:

```
OSH <- >>/script/hourly.log 2>>/script/hourly.err &
-p /bin/usr/perl /script/hourly.pl
```

2. Enter the following at a TACL prompt:

```
BATCHCOM;SUBMIT JOB, IN \NODE.$SYSTEM.SYSTEM.HOURLY, &
EVERY 60 * * * * mon-fri
```

As an example of running an OSS shell script periodically, suppose:

- You want to record OSS process activity every 10 minutes only on weekdays.
- You have created an OSS shell script file containing the `ps` command at `/script/processes.sh`.
- You want to record normal output in the OSS file `/script/processes.log` and error output in the OSS file `/script/processes.err`.

You would:

1. Create a valid job file for the NetBatch product, named \NODE.$SYSTEM.SYSTEM.PROCESS and containing the following TACL command statement line:

```
OSH /script/processes.sh <- >>/script/processes.log &
2>>/script/processes.err
```

2. Enter the following at a TACL prompt:

```
BATCHCOM;SUBMIT JOB, IN \NODE.$SYSTEM.SYSTEM.PROCESS, &
EVERY 10 * * * * mon-fri
```

For an example of using the TACL OSH command to execute a single OSS shell command through the NetBatch product, see Controlling the Growth of Directories on page 9-8. For more information on setting up and scheduling batch jobs, see the *NetBatch Manual* or the *NetBatch Plus Reference Manual*.

3

Understanding the OSS File System

To manage the Open System Services (OSS) environment effectively, you must understand the OSS file system.

The OSS file system works in the same way as a UNIX file system from the point of view of the user. Files in the OSS environment are organized in a hierarchical tree structure. For further information about using the tree structure, see the *Open System Services User's Guide*. For information about the content of the tree structure as released by HP, see the `hier(5)` reference page either online or in the *Open System Services System Calls Reference Manual*.

The OSS file system consists of one or more filesets. Each fileset is a hierarchy of files: a set of directories, subdirectories, and files themselves.

A fileset can have other filesets mounted on directories in it. In fact, the collection of directories and files under the root directory is part of one fileset. Every file belongs to a fileset. You control the operation of filesets as described in Section 5, Managing Filesets.

Every file has an OSS pathname. The OSS pathname consists of one or more OSS filenames and helps locate the file within the tree structure.

The following subsections provide a brief overview of the relationships between OSS pathnames and Guardian filenames and an overview of the tree structure. Later subsections describe how the OSS name servers maintain and use these relationships for access to the filesets they manage.

OSS Pathnames

An OSS pathname describes a path through the OSS directory tree to a file. The length of OSS pathnames is limited:

- Each OSS directory name or OSS filename in an OSS pathname can contain up to 248 characters.
- OSS pathnames can contain up to 1024 characters.

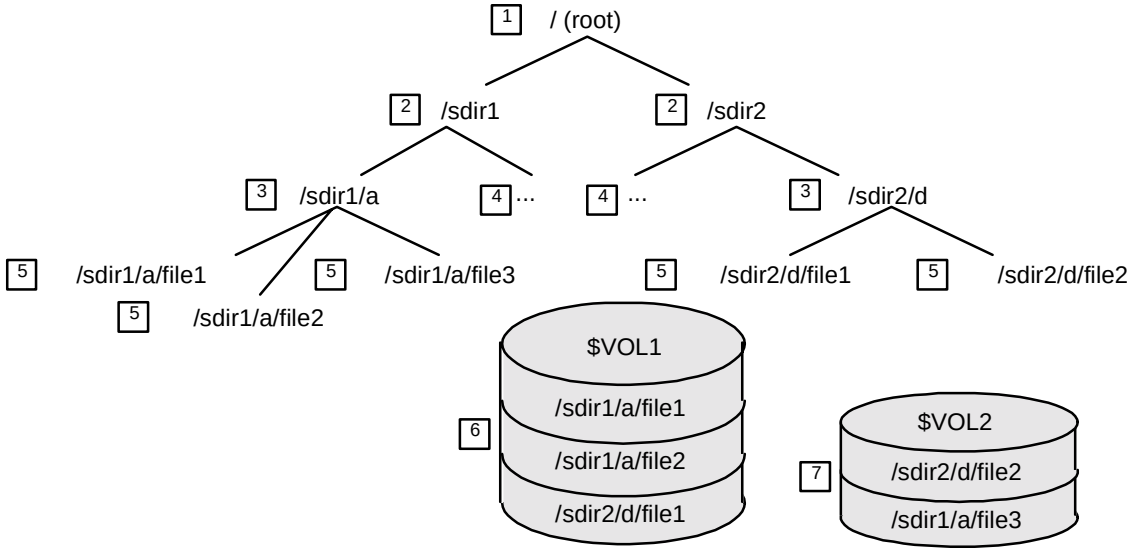
An OSS file can have more than one pathname, using either hard or symbolic links:

- An OSS file can have as many as 128 links (128 OSS pathnames can point to the same file).
- As many as 20 symbolic links can be followed when resolving an OSS pathname.

For further information about OSS files and pathnames, see the *Open System Services User's Guide* and to the `filename(5)` reference page either online or in the *Open System Services System Calls Reference Manual*.

OSS pathnames are logical names; they have no connection to storage devices. The relationship of OSS files to disk volumes is illustrated in Figure 3-1.

Figure 3-1. OSS Files and Disk Volumes



Legend

1. Root directory.
2. Directories in the root directory (subdirectories).
3. Directories in a subdirectory (also called subdirectories).
4. Any number of directories and files can be in a subdirectory.
5. Files.
6. Disk volume $VOL1 contains files from two directories.
7. Disk volume $VOL2 contains files from two directories.

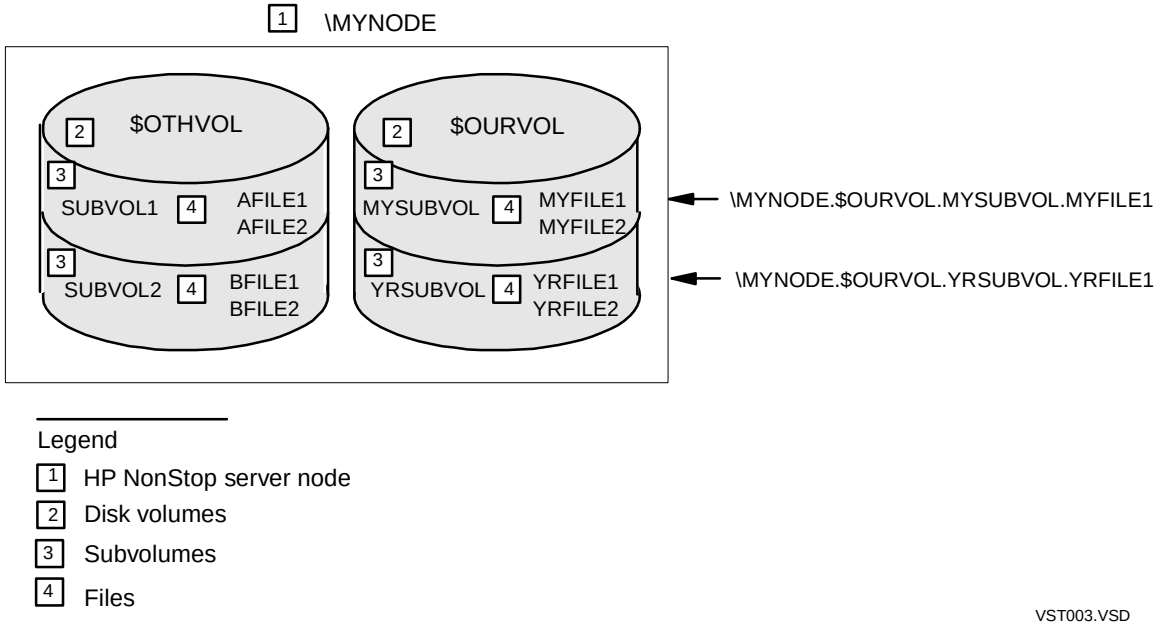
VST002.VSD

In contrast, Guardian filenames that are not administered through the NonStop Storage Management Foundation (SMF) are physical names: they mention physical storage devices. A fully qualified Guardian filename includes the name of a system node, a disk volume name, a subvolume name, and a name for the file itself. The relationship of Guardian files to disk volumes is illustrated in Figure 3-2 on page 3-3.

The OSS file system allows a high degree of nesting: you can have subdirectories, subsubdirectories, subsubsubdirectories, and so on. The Guardian environment allows only three levels: volume, subvolume, and file ID.

In the Guardian environment, you cannot have a volume within a volume. In the OSS environment, directories that are within directories are common.

Figure 3-2. Guardian Files and Disk Volumes



Each pathname for an OSS regular file has an underlying Guardian filename. The OSS name servers map such pathnames onto corresponding Guardian filenames. This mapping, known as name resolution, is necessary because OSS pathnames are different from Guardian filenames.

Each OSS file therefore has two names:

- An OSS pathname, such as `/usr/henry/workfile`
- A Guardian filename, such as $VOL2.ZYQ00001.Z0000034

The parts of a Guardian filename and an OSS pathname for an OSS file are:

Environment	Volume	Fileset	Inode
Guardian	$VOL2	ZYQ00001	.Z0000034
OSS	--	`/usr`	`/henry/workfile`

Directory files, terminal device files, `AF_UNIX` sockets, and FIFOs (named pipes) do not have underlying Guardian files or Guardian filenames. These OSS special files are managed entirely by OSS server processes. Such files do not require any configuration action by the system manager; their operation cannot be controlled through SCF FILESET commands.

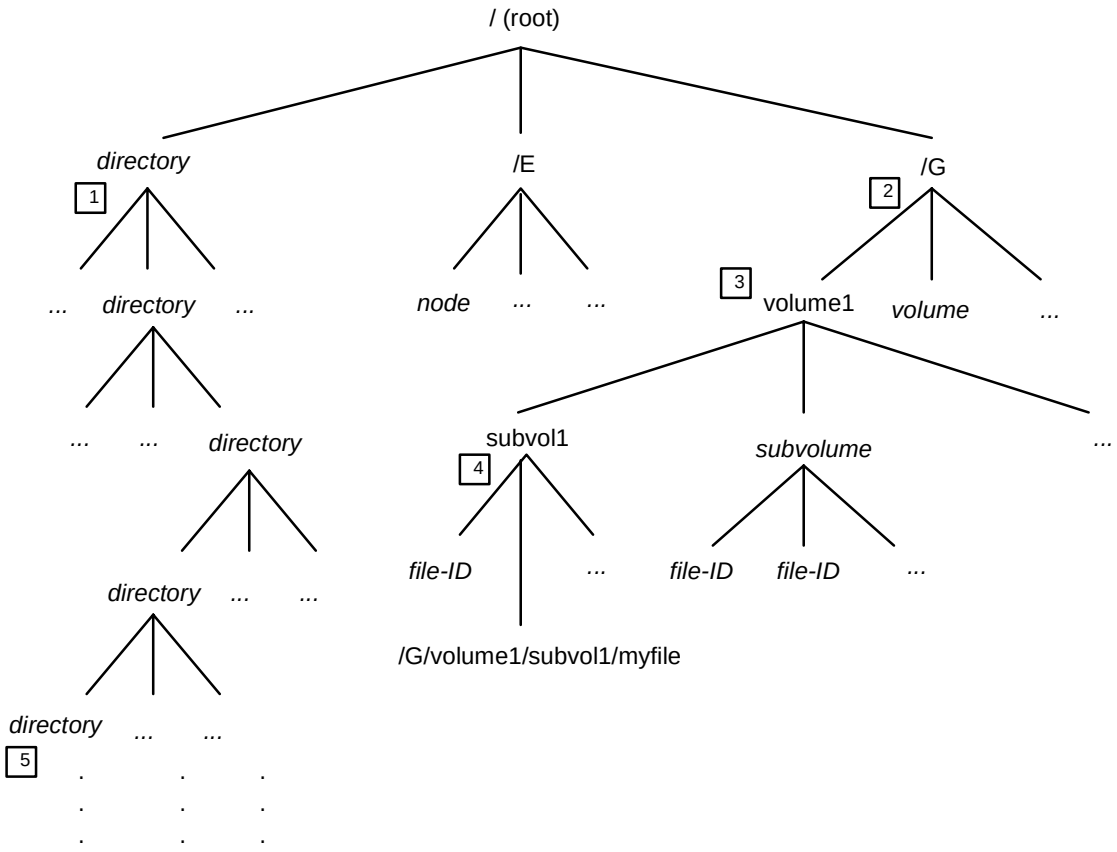
Guardian files also exist in the OSS environment. All accessible Guardian files on the local node are located in the `/G` directory of the OSS file system.

From a programmer's or end user's perspective, the `/G` directory is its own fileset in the OSS file system; each disk volume within the Guardian file system and each terminal process also are separate filesets. An OSS administrator does not need to

define or manage these filesets through the SCF interface and they do not appear in the fileset configuration database.

The OSS pathnames of Guardian files can have no more than four elements including the /G, as illustrated in Figure 3-3. These elements, although technically OSS filenames, are subject to the length restrictions for Guardian filenames.

Figure 3-3. Guardian Files in the OSS File System



Legend

1 A directory can contain files and other directories.

2 The local node (/G directory) must contain only volumes.

3 A volume must contain only nonempty subvolumes.

4 A subvolume can contain only files.

5 There is no theoretical limit to the number of directories and files in a directory.

VST004.VSD

Using Pathnames for Remote Files

The rules described in the preceding subsection can be extended to the files accessible through the /E directory. A file on a remote node that is connected to your node through the Expand network appears in /E when all of the following are true:

- The remote Expand node has a TOSVERSION of D40 or later.
- The remote node has an OSS name server running with a device subtype of 5.
- The file is either:
  - A Guardian disk file
  - An OSS file in a started fileset with a catalog that uses a format newer than a D3x version

Remote Guardian files appear in the /G directory of the remote node. For example, if the remote node is named NODE1, the Guardian files on its $SYSTEM disk in subvolume ZOSSUTL would appear to your system as files in /E/node1/G/system/zossutl/.

Remote OSS files can be found from the root (/) directory of the remote node. For example, the files in /usr/share/man on the remote node named NODE1 would appear to your system as /E/node1/usr/share/man.

Files visible through /E on the remote node are not visible to your system. That is, if you are on NODE0, you cannot access files on NODE0 by looking in /E/node1/E/node0.

Your local node also has an entry in its own /E directory. That entry consists of a symbolic link to its own root directory; for example, /E/node0 is a symbolic link to the / directory if your local node is NODE0.

Some software permits you to specify files on your local node by prefixing the OSS pathname with /E and the local node name (for example, on NODE0 you can enter /E/node0/usr/share/man/cat5/filename.5 or /E/node0/G/system/zossutl/T8620MAN). This form of pathname is intended for programming use; it is not a good practice when using the OSS shell and should be avoided.

Using the Local Root Directory as a Pathname

Because /G and /E both appear in your local root directory, you should be very careful when using OSS shell commands on or from the root directory. OSS shell commands that perform recursive actions make no distinction between Guardian and OSS files or between local and remote files.

For example, a simple search with the find command that is intended to look for an OSS file also searches all Guardian files on your local node and all files on all connected Expand nodes if you enter just:

```
find / -name log*
```

Such a search would be extremely time-consuming and would return undesired information on /E and /G files.

OSS shell commands that perform recursive operations include:

```
chgrp    cp        ls        rm
chmod    diff      mv        rmdir
chown    find      pax
```

Other products running in the OSS environment might also have commands with recursive behavior. For example, the grep, egrep, fgrep, tar, and cpio utilities should not be used on the / directory because of the scope of the files involved.

Avoid specifying the / directory or using the wildcard character * on any object in an OSS shell command. Using these commands when you are logged on with the super ID can have far-reaching consequences.

Recursive OSS shell commands allow you to avoid unwanted behavior when the / directory is specified in a recursive command. You use the -W flag in the command with one or both of the following options to specify the behavior you want from that invocation of the command:

Option	Effect
NOE	The /E directory is skipped during recursive operations.
NOG	The /G directory is skipped during recursive operations.

For example, the following command performs a recursive search of the / directory without searching the /G and /E directories:

```
find / -W NOG -W NOE -name log*
```

See the *Open System Services Shell and Utilities Reference Manual* for more information on the use of the -W flag with a specific command.

The OSS shell also supports an environment variable, UTILSGE, that allows you to specify recursive command behavior for an entire terminal session (or until the value of UTILSGE is changed). The UTILSGE variable accepts the same values as the -W command flag. To use UTILSGE, you either enter the appropriate one of the following at a shell prompt or add it to your .profile file:

```
export UTILSGE=NOE

export UTILSGE=NOG

export UTILSGE=NOG:NOE
```

Note. Do not enter NOE:NOG. This value for the variable UTILSGE is not valid.

OSS File Components

Each OSS data file, or regular file, consists of two components:

- The catalog portion, which is recorded in an OSS name server catalog file. This portion supplies structural information about the file.
- The data portion, which is stored in a file that is identified by a Guardian filename. This portion contains the data seen by end users of the file.

The Guardian filename is mapped from the OSS pathname using an OSS name server catalog so that the NonStop operating system disk process can find the data file.

Directories, terminal device files, `AF_UNIX` sockets, and FIFOs do not have underlying data files. All information about these files is contained in the catalog.

OSS Catalog Files

The OSS name servers use catalog files to maintain and manage fileset information. The catalog files maintained by an OSS name server for each fileset contain:

- Information about the hierarchical directory structure of the fileset.
- Unique identifiers for the files, called inode numbers in OSS and UNIX terminology or file serial numbers in POSIX terminology. Each file in the OSS file system has such an identifier.

The OSS catalog files enable the hierarchical OSS file system to be mapped into the flat Guardian file system. The catalog files include PXINODE, PXLINK, PXLOG, and any files saved as described in Generated Catalog Files on page 5-33. All catalog files:

- Reside in the Guardian environment
- Are always stored in Guardian subvolumes whose names begin with the letters ZX.
- Have a Guardian file code of 444 and Guardian file access read, write, execute, and purge permissions that display as `----`.

OSS Data Files

The name of each data file in the OSS file system is mapped into a Guardian filename with the form *node*.$*volume*.*subvol*.*file-id*. The file identifier is the Guardian representation of the inode for the actual file that contains data usable in the OSS environment—the OSS regular file.

All OSS regular files:

- Are always stored in Guardian subvolumes whose names begin with the letters ZYQ.
- Have a Guardian file code of 100 and Guardian file access read, write, execute, and purge permissions that display as ####.

Guardian disk files are referenced using the OSS filenames in the `/G` directory. Only Guardian disk files that are not on disk volumes administered through SMF are visible in the `/G` directory.

Guardian disk files on optical disks are visible in the `/G` directory but cannot be read from or written to through the OSS file system.

For information on accessing files in the `/G` directory, see the *Open System Services User's Guide.*

Relating OSS Files, Filesets, and Disk Volumes

You perform OSS file-system configuration and administration tasks through the Guardian environment. To do these tasks, you need to consider:

- The sizes of the files your site might use
- The sizes of the filesets your site might use
- How your OSS configuration files are used

OSS File Size Considerations

An individual OSS file cannot span multiple volumes. In the OSS environment, a single file is always stored on just one disk. While a user cannot extend an existing file for which no more room is available, the user can readily open a new file in the same directory on another disk.

Prior to the H06.06 RVU, OSS files had a size limit of approximately 2 gigabytes, and an underlying Guardian file of Format 1 (the file format was normally not visible to customer applications). For H06.06 and later RVUs, OSS files are either small files or large files.

Small Files

These OSS files:

- Have an underlying Guardian file format of either Format 1 or Format 2. This underlying file format is normally not visible to customer applications.
- Have a size limit of approximately 2 gigabytes when opened or created using a 32-bit API. For information about the exact size limits, see Appendix E, Environment Limits.
- Can be accessed using the existing 32-bit application programmatic interfaces (APIs) such as `creat( )` and `open( )` in addition to the 64-bit APIs such as `creat64( )` and `open64( )`. If the file is accessed using a 64-bit API:
  - The file is automatically converted to use an underlying Guardian file format of Format 2.
  - The file no longer has a size limit of approximately 2 gigabytes. It can grow to the size limit for large files.

- Can be moved, copied, or restored to systems running RVUs that do not support OSS files larger than 2 gigabytes. In such cases, if the file has an underlying Guardian file format of Format 2 but a size of less than approximately 2 gigabytes:
  - The file is automatically converted to use an underlying Guardian format of Format 1.
  - The file becomes subject to the file size limit of approximately 2 gigabytes.

Large Files

These OSS files:

- Are larger than approximately 2 gigabytes.
- Have a size limit of approximately 1 terabyte, constrained by the space available on the disk volume. For information about the exact size limits, see Appendix E, Environment Limits.

  Note. A `pax` archive file is limited to 8 gigabytes.

- Have an underlying Guardian file format of Format 2. This underlying file format is normally not visible to customer applications.
- Can be accessed using the 64-bit APIs such as `creat64( )` and `open64( )` only. Depending on the compilation environment, the 32-bit APIs can be automatically mapped to the 64-bit APIs. The utilities operators use have been enhanced to support large files. Only application programmers must be aware of which APIs to use for large files.
- Cannot be restored to systems running RVUs that do not support OSS files larger than approximately 2 gigabytes.
- If OSS APIs are used, large files cannot be accessed from systems running RVUs that do not support OSS files larger than 2 gigabytes. However, Guardian Enscribe APIs with 64-bit elections can access these files.

Fileset Size Considerations

OSS files reside in filesets. A fileset can contain up to 500,000 files as a practical limit.

Filesets reside on disk volumes that are grouped into storage pools. Each fileset has its own storage pool of one or more disk volumes. A fileset can span multiple disk volumes.

As a storage pool's disk volumes are filled, you can add more volumes to the pool to accommodate the files in that fileset. Up to 20 disk volumes can be specified as current members of a storage pool by including them in the storage-pool file for the fileset; the current members of a storage pool are sometimes called the creation pool, to contrast them with all of the disk volumes used by the fileset as a pool for file storage.

Fileset size depends mainly on the type and complexity of the application mix running on your system. If you are porting an application from another system, you would have some idea of the application's requirements and could use that as a basis for estimating the application's disk-space requirements on the NonStop system.

OSS Configuration Files

The OSS file system gets configuration information from the ZOSSFSET file and storage-pool files. You configure the OSS file system by updating these files according to directions in Section 5, Managing Filesets.

The ZOSSFSET File

The names of all filesets defined on the system and information about the disk storage they use resides in a file named ZOSSFSET, defined in the Guardian environment. The ZOSSFSET file is completely described under Configuration Files on page 4-7.

When a fileset is mounted, its OSS name server accesses catalog files in the catalog volume that you specify for the fileset in the ZOSSFSET configuration file.

Within that catalog volume, the OSS name server for that fileset uses a Guardian subvolume whose name begins with ZX0. This name is a reserved subvolume name used only by an OSS name server.

In this subvolume, the OSS name server for the fileset accesses (and creates if necessary) the catalog files PXINODE, PXLINK, and PXLOG. Thereafter, whenever someone mounts or remounts a fileset, the OSS name server that manages that fileset uses these catalog files. Each fileset has a volume, called a catalog volume, that contains these catalog files and other information about the fileset.

Files in subvolumes whose names begin with ZYQ are subject to special access restrictions. You cannot access these files from the Guardian environment, and you cannot create new files in these subvolumes from the Guardian environment.

Storage Pools and Storage-Pool Files

A storage pool is the set of disk volumes on which the OSS data files of a fileset reside. The storage-pool file is a Guardian EDIT file that determines which disk volumes of the storage pool can be used for creating new OSS data files that are being added to the fileset. The disk volumes listed in the storage-pool file can be viewed as the creation pool, a subset of the entire storage pool used by the fileset. Figure 3-4 on page 3-12 shows the difference between an OSS storage pool and the contents of the storage-pool file for the fileset DATA5; the creation pool is enclosed in a rectangle to indicate that it is the set of disk volumes identified in the storage-pool file.

The OSS name server for a fileset uses the storage-pool file for that fileset to determine where to create each new OSS data file. When that OSS name server receives a file-creation request, the server reads the storage-pool file and creates the file on the disk volume whose name appears in the storage-pool file following the volume name used for the last request.

As each new file is created, the fileset's OSS name server continues along the list of volume names, selecting a new volume with each request. The OSS name server ultimately wraps around to the beginning of the list in a round-robin fashion.

Thus, if users write only small files in one disk volume in the list and only large files in another, then one volume can fill up before the others. By allocating a large enough fileset, you can help avoid the problems produced by this unlikely file distribution.

An OSS name server cannot allocate files on more than 20 disk volumes for one fileset. However, each time a fileset is mounted, you can specify either:

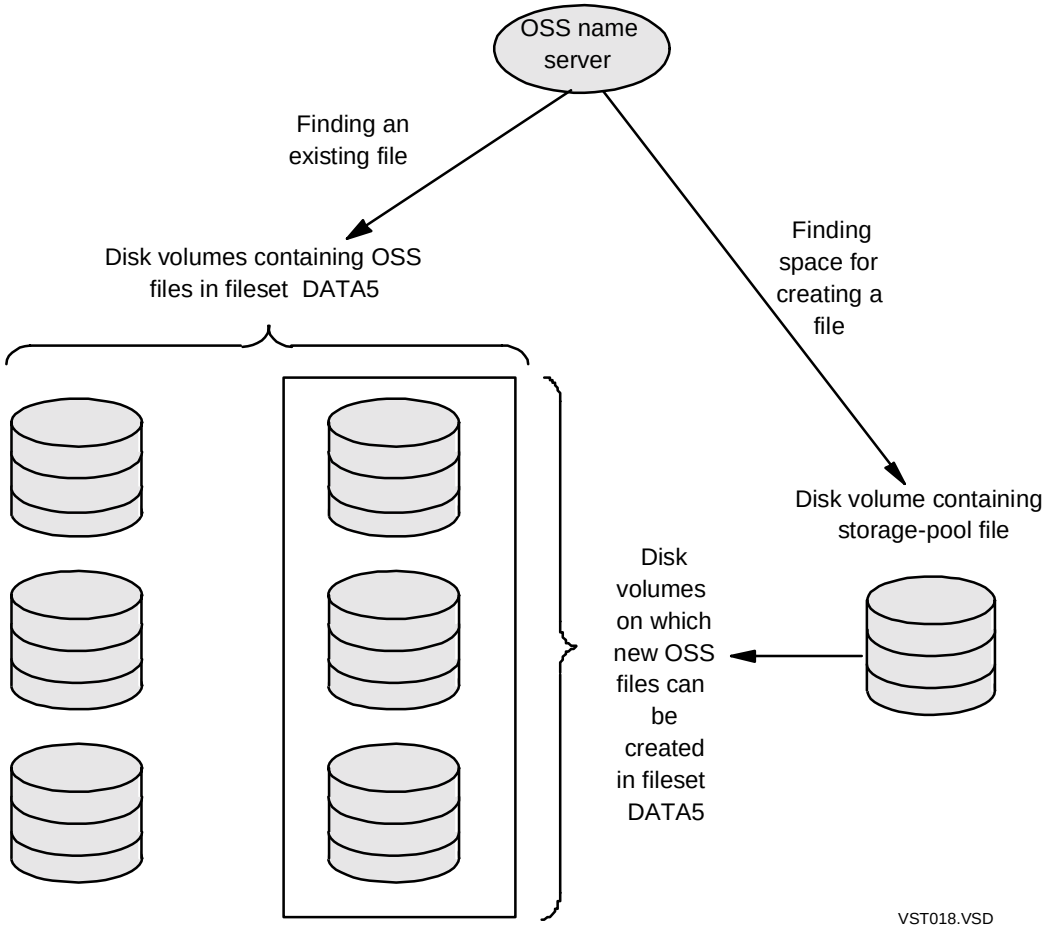
- A different set of disk volumes for the creation pool of the fileset (different content of the same storage-pool file)
- A different storage-pool file for the fileset, containing a different set of disk volumes

As a result, a fileset can span many disk volumes. OSS files can exist on disk volumes that are part of the fileset even though they are not in any active storage-pool file. The storage pool for the fileset can be much larger than the creation pool defined by the content of the storage-pool file.

For this reason, a storage-pool file cannot contain more than 20 active volume names (the creation pool) but the volume list maintained by an OSS name server for the fileset (the entire storage pool) can contain up to 256 disk volumes. See FSCK Log File on page 5-25 for more information about the volume list.

You should not set up a fileset to use a disk volume that is not always in the storage-pool file for that fileset. Normal operating procedures for a fileset can have unplanned side effects on the OSS files on such disk volumes. See Changing the Physical Makeup of a Fileset on page 5-21 for more information.

Figure 3-4. Storage Pools and Disk Volumes



While OSS filesets can span multiple physical disk volumes, individual files cannot. Thus if you allocate for a fileset a disk volume that is running out of space, you might not be able to extend existing files or write new large files on that volume, even if all other available volumes are nearly empty.

After a fileset is created, you can check the individual disk volumes in the fileset's storage-pool file to monitor the disk space that the fileset is using. If one or more volumes are almost full, you can make more volumes available for the fileset. To do this, unmount the fileset, add more volumes to the storage-pool file, and then remount the fileset. From this point on, as that OSS name server for that fileset continues to create new files, it uses the new disk volumes as well as remaining space on the other specified volumes.

A sample storage-pool file is shown under Creating a Storage Pool on page 5-6.

4 Managing Servers

This section describes how to manage the servers used to provide the Open System Services (OSS) environment. Not all servers are managed through the same interface; read the subsections Introducing the OSS Servers on page 4-1 and Configuration Files on page 4-7 before attempting any operation on an OSS server.

The introductory information in this section helps you perform the following operations:

- Adding a Server on page 4-28
- Configuring a Server on page 4-29
- Starting a Server on page 4-36
- Obtaining Information About a Server on page 4-39
- Stopping a Server on page 4-43
- Reconfiguring a Server on page 4-46
- Removing a Server on page 4-49
- Troubleshooting a Server on page 4-50

Procedures in this section are representative. Normal practices at your site might require additional steps or use of additional command parameters for logging purposes.

Introducing the OSS Servers

A UNIX system depends on the operation of many demon processes that provide services as servers. Similarly, the OSS environment is created by servers, depends on other servers running in the Guardian environment, and runs servers within itself.

The servers that create the OSS environment were introduced in Components to Be Managed on page 1-9. These servers are further described in the following subsections:

- The OSS Name Servers on page 4-2
- The OSS Message-Queue Server on page 4-2
- The OSS Sockets Local Server on page 4-4
- The OSS Transport Agent Servers on page 4-4

These servers are all managed using OSS Monitor SCF commands. All SCF commands for servers, except INFO SERVER and STATUS SERVER, can be run only by a member of the super group (255,*nnn*).

Servers that run in the OSS environment are not managed through OSS Monitor SCF commands. Such servers include network services servers such as `inetd`, `rshd`, `rexecd`, and `named`, as discussed in The Network Services Servers and Tools on page 4-5.

Like the OSS transport agent servers, the OSS terminal helper server processes start automatically, as described in The Terminal Helper Servers on page 4-4. However, the OSS terminal helper server processes have no management interface.

Use of the OSSTTY utility as a server is application-dependent. The OSSTTY server is managed using TACL commands, as described in OSSTTY on page C-1.

Servers in other subsystems use OSS name servers for OSS pathname resolution. Such servers and subsystems include the following:

- The iTP WebServer `httpd` process
- The Network File System (NFS)
- HP NonStop TS/MP
- The SQLCAT process used with SQL/MP

Communication with such servers can be affected by the configuration of OSS servers and of the OSS subsystem. These other subsystem servers are beyond the scope of this guide; see the appropriate manual set for more information about the configuration and operation of a specific server.

Figure 4-1 on page 4-3 shows relationships among servers using the OSS environment. An OSS application program can use any server shown in the figure.

The OSS Name Servers

Open System Services supports multiple OSS name server processes. This feature can improve performance by allowing multiple processes to share the task of resolving OSS pathnames.

You can run as many OSS name servers as you need on a system simultaneously. Each fileset is managed by only one OSS name server; however, one OSS name server can manage many filesets. You can use OSS Monitor SCF commands to change the configuration of filesets and OSS name servers on your system as the demands of your system vary.

An OSS name server can run as a single process or as a fault-tolerant process pair. The OSS name server for the root fileset uses the process name $ZPNS; OSS name servers for all other filesets can use any process name with a length of up to 5 characters plus the dollar sign.

If there is a backup server process, it preserves mounted fileset data as well as fileset access if the primary server process fails. You can control the processor in which the backup server process runs.

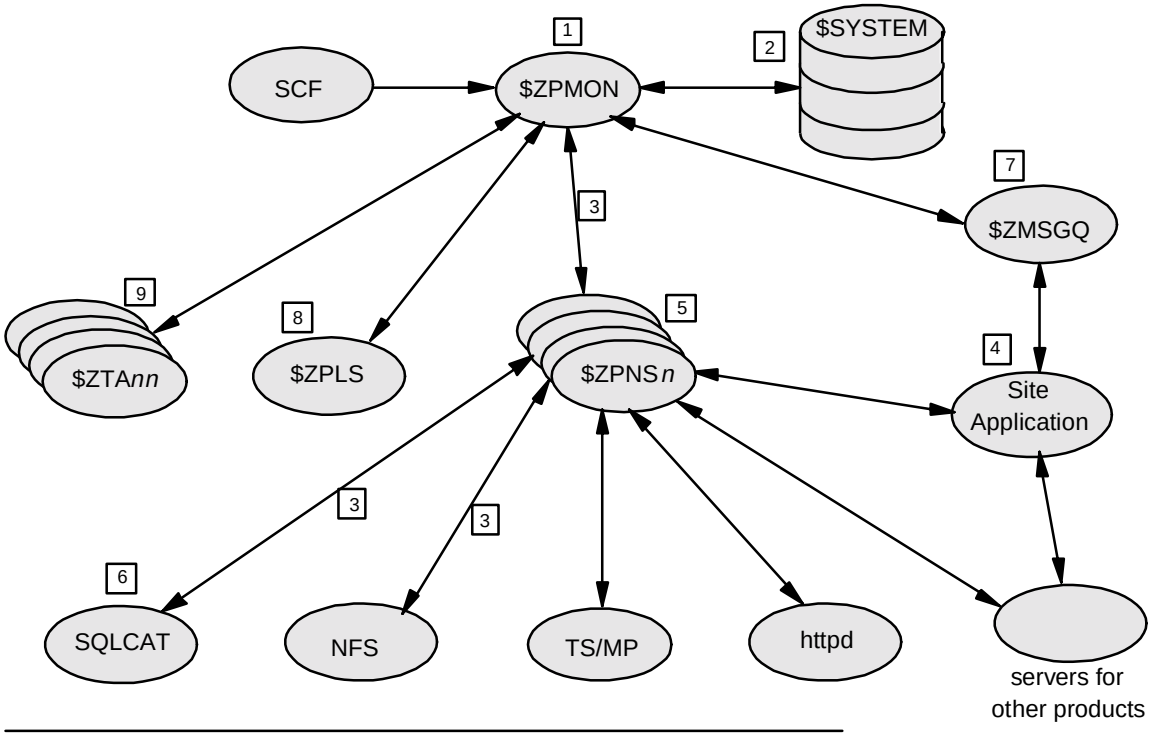
The OSS name servers are configured and controlled through OSS Monitor SCF commands. See Section 12, Open System Services Monitor, for more information about those SCF commands.

The OSS Message-Queue Server

The OSS message-queue server runs as a fault-tolerant process pair. The server uses the default process name $ZMSGQ.

The backup server process preserves message-queue data as well as the queues themselves if the primary server process fails. You can control the processor in which the backup server process runs.

Figure 4-1. OSS Environment Servers



Legend

1. OSS Monitor process
2. Configuration files for the OSS environment
3. Request timeout
4. Site application in the OSS environment
5. OSS name servers
6. NonStop SQLCAT process
7. OSS message-queue server
8. OSS sockets local server
9. OSS transport agent servers

VST017.VSD

You can start and stop the OSS message-queue server using the OSS Monitor SCF commands START SERVER and STOP SERVER (see START SERVER Command on page 12-65 and STOP SERVER Command on page 12-81 for detailed information about these commands).

If the OSS message-queue server fails completely, you can restart it. Server failure can be detected from the Event Management Service (EMS) messages issued to your system logs.

The OSS Sockets Local Server

The OSS sockets local server can run as a single process or as a fault-tolerant process pair. The server uses the default process name $ZPLS.

The backup server process preserves socket access if the primary server process fails. You can control the processor in which the backup server process runs.

You can start and stop the OSS sockets local server using the OSS Monitor SCF commands START SERVER and STOP SERVER (see START SERVER Command on page 12-65 and STOP SERVER Command on page 12-81 for detailed information about these commands).

If the OSS sockets local server fails completely, you can restart it. Server failure can be detected from the Event Management Service (EMS) messages issued to your system logs.

The OSS Transport Agent Servers

There is one OSS transport agent server (used for OSS sockets communication) for each processor in your system. The process name is $ZTA*nn*, where *nn* is the processor number. Each OSS transport agent server is automatically started when the processor it runs on is started.

You can start and stop the OSS transport agent servers using the OSS Monitor SCF START SERVER Command and STOP SERVER Command. You cannot add, modify, or remove an OSS transport agent server.

If an OSS transport agent server fails completely, you can restart it. Server failure can be detected from the Event Management Service (EMS) messages issued to your system logs.

The Terminal Helper Servers

There is one terminal helper server process (used for OSS nonblocking terminal input and output) for each processor in your system. The terminal helper server provides support for application use of the `select()` function and the FILE_COMPLETE_ family of procedure calls on terminal device files to provide nonblocking terminal input/output.

The process name is $ZTT*nn*, where *nn* is the processor number. Each terminal helper process server is automatically started when the processor it runs on is started.

You cannot start and stop the OSS terminal helper servers using the OSS Monitor SCF START SERVER Command and STOP SERVER Command. You cannot add, modify, or remove an OSS terminal helper server.

The Network Services Servers and Tools

The following subsections briefly discuss:

All but RPCINFO are usually demon processes or processes started by demon processes on UNIX systems.

inetd

The `inetd` process is the UNIX and OSS equivalent of the Guardian LISTNER process for `AF_INET` and `AF_INET6` OSS sockets applications. In the OSS environment, `inetd` is the server process that listens for network activity.

`inetd` is started from an OSS shell command line or script and listens for connections on certain Internet ports. When a connection request is received, `inetd` decides which service the request corresponds to and invokes a server program to service the request. After the program completes the request, `inetd` continues to listen. The `inetd` process allows one process to invoke several others, reducing load on the system.

`inetd` simplifies the interface of a server program that it starts, because it duplicates its socket descriptors for an incoming request as file descriptors 0, 1, and 2 before the server application is processed by an `exec` function call. This action allows the server application to use the `stdin`, `stdout`, and `stderr` files in function calls to perform the requested service.

If the `inetd` server fails, you can restart it. Server failure can be detected from the Event Management Service (EMS) messages issued to your system logs.

For more information about the `inetd` server, see the `inetd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

rshd

The `rshd` process is the server process for the `rsh` utility. It is started by the `inetd` process, which must be running when you use the `rsh` utility.

For information about the behavior of the `rshd` process, see the `rshd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

rexecd

The OSS remote execution server, `rexecd`, is used for remote NonStop SQL/MX compiler (`mxcmp`) invocation from the native C/C++ and NMCOBOL cross-compilers,

running by themselves, or under the Enterprise Toolkit—NonStop Edition. `rexecd` is started by the `inetd` process, which must be running for remote SQL/MX compilations.

For information about the behavior of the `rexecd` process, see the `rexecd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

portmap and RPCINFO

The `portmap` process is a Guardian server process or process pair that converts host port numbers to Remote Procedure Call (RPC) program numbers. `portmap` is started from a TACL prompt, runs as a process named $ZPM*n*, and is required by products such as the OSS Network File System (NFS) that use the RPC interface.

The Guardian `portmap` process corresponds to `/etc/portmap` on UNIX systems. For more information about the `portmap` process, see the `portmap(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

The RPCINFO process is started from a TACL prompt and reports the status of certain servers, including `portmap`. RPCINFO is a Guardian process that reports RPC program numbers and can be used to modify the status of RPC servers available from your node. RPCINFO provides a means to monitor and change `portmap` behavior; RPCINFO is required by the same products that require `portmap`.

For more information about the RPCINFO process, see the `rpcinfo(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

BIND 9 Domain Name Server and Tools

The Internet domain name server (DNS) runs in the OSS environment as the `named` process. It is started from an OSS shell.

The `named` process provides services comparable to the older Guardian-based T6021 DNS product but implements part of the Berkeley Internet Name Domain (BIND) 9 distribution from the Internet Software Consortium (ISC). Two versions of `named` are available:

- Product T0685, a version based upon BIND 9.2.3, without security features
- Product T0708, a secure version based upon BIND 9.3

Both versions can be run on an HP NonStop node at the same time if they have been started on different IP addresses and configured to maintain their own sets of data files.

The secure version of `named` can be used with the DNS security extensions. The DNS security extensions (DNSSEC) are a collection of resource records and protocol modifications that add data origin authentication and data integrity to the DNS. Domain name servers that employ DNSSEC add digital signatures to their zone files. By

checking the signature, other security-aware domain name servers can verify the integrity and authenticity of DNS data. For more information on this implementation of DNS, see RFCs 1033, 1034, and 1035, the *BIND 9 Administrator Reference Manual*, and the *DNS Configuration and Management Manual*.

The lightweight resolver utility, `lwresd`, is available for use with specific application program interface (API) functions. The `lwresd` server provides certain services for applications when the T0709 APIs in product T0709 are used.

The `rndc` utility provides a control interface for `named` and also starts from an OSS shell. The `nsupdate` dynamic DNS update utility submits dynamic DNS update requests (as defined in RFC 2136) to `named`. This utility allows resource records to be added or removed from a zone without manually editing the zone file. A single update request can contain requests to add or remove more than one resource record.

See the `named(8)`, `lwresd(8)`, `rndc(8)`, and `nsupdate(8)` reference pages online for more information about the nonsecure version of these BIND 9 programs. See the `dnssec_named(8)`, `lwresd(8)`, `dnssec_rndc(8)`, and `dnssec_nsupdate(8)` reference pages online for more information about the secure version of these BIND 9 programs.

Configuration Files

Each OSS server has its own configuration file requirements. Some OSS servers share database files. Other servers use text files created in the OSS file system or maintained in the Guardian environment.

The following subsections describe the configuration files used by each OSS server. All these files should be secured according to site security procedures so that only a system administrator can alter or remove them; see the recommendations in Preventing Security Problems on page 8-28.

△ **Caution.** If any of these configuration files are accidentally deleted, the current configuration of the OSS environment is lost. You should make frequent backups for these files.

Configuration Files Used for the OSS Name Servers

The OSS Monitor uses the following configuration files to manage OSS name server operation and OSS file access:

- The ZOSSFSET File on page 4-8
- The ZOSSPARM File on page 4-13
- The ZOSSSERV File on page 4-14
- The Storage-Pool Files on page 4-17

These files must be in the subvolume $SYSTEM.ZXOSSMON. Beginning with the G05.00 release version update (RVU), all these files except the storage-pool files are created automatically if they are missing when the OSS Monitor is started.

The ZOSSFSET File

The ZOSSFSET file is an Enscribe alternate-key file that contains the fileset configuration database. The ZOSSFSET file contains entries that identify the storage-pool file and operating characteristics for each fileset in the OSS file system. Entries in this file can be displayed with the SCF INFO FILESET command.

This file must be maintained by the system manager using the SCF ADD FILESET, ALTER FILESET, DELETE FILESET, or RENAME FILESET command. It must be present before the root fileset is first mounted.

Entries in the ZOSSFSET file must follow these rules:

- The name of a fileset:
  - Can consist of 1 to 32 uppercase letters and digits (A through Z and 0 through 9). The first character must be a letter.
  - Must be unique.
- The ZOSSFSET file must contain an entry for the root fileset.

  An entry for the root fileset is automatically created in the ZOSSFSET file.
- The device label of a fileset is a six-character value in the range 000000 through 0ZZZZZ. The characters can be 0 through 9 and A through Z (but not E, I, O, or U).
- Fileset catalog volumes:
  - Cannot be on optical disks.
  - Cannot be on disks administered through the NonStop Storage Management Foundation (SMF).
  - Should not be on the $SYSTEM disk, to avoid file security conflicts.
- The pathname of the OSS file-system directory that the fileset is mounted on:
  - Must be the pathname of a directory that exists.
  - Is case-sensitive and must start with `/`.
  - Cannot begin with `/G` or`/E`.
  - Should be secured as “`drwxrwxrwx`” (0777, recommended for `/` and `/home`) or “`drwxrwxrwt`” (1777, recommended for `/tmp`) in the OSS environment.

  The root fileset mount-point directory must be `/`. The mount point of the root fileset cannot be changed.
- The only valid name for the OSS name server of the root fileset is #ZPNS. The root fileset name cannot be changed.

- When the BUFFERED CREATE (fast-create) attribute is set for a fileset:
  - The fileset cannot use more than one volume.
  - The fileset catalog must reside on that volume.
  - The storage-pool file specification is ignored and the storage pool used for file creation (the creation pool) is the catalog disk volume.

  The BUFFERED CREATE attribute controls buffering of new file labels by the disk process. When this attribute is set, the labels for new files created on this fileset are kept in memory until the disk process has no other work to perform; then the labels are written to disk. When this attribute is not set, labels are written to disk immediately.

  This attribute increases the risk of losing file labels if the entire disk-process pair for the volume fails. This option is usually used only for a volume that will contain temporary files.

When a default ZOSSFSET file is created by the OSS Monitor on a system that was not previously configured, it contains only an entry for a prototype root fileset with the device label 000000. The prototype root fileset does not contain a valid catalog disk volume name or a valid storage-pool file filename. You must use the SCF ALTER FILESET command to provide that information before the root fileset can be mounted and before you can use the TACL PINSTALL utility or COPYOSS macro successfully.

Beginning with RVU G06.10, when HP shipped a system with the OSS environment preconfigured, ZOSSFSET contained a complete root fileset entry. This entry differed slightly from the default attribute values, as follows:

Attribute	Default Values When ZOSSFSET Is Created by the OSS Monitor	Initial Values When ZOSSFSET Was Delivered on a Preconfigured System
Fileset name	ROOT	ROOT
Device label	000000	000000
Storage-pool file	None	ZINSPOOL
Catalog volume	None	$OSS
Mount point	/	/
OSS name server name	#ZPNS	#ZPNS
AUDITENABLED	OFF	OFF
BUFFERED	NONE	LOG
READ-ONLY	FALSE	FALSE

In the following table, *disk1*, *disk2*, and *disk3* refer to the first three disk volumes from the set of volumes available for use in storage-pool files.

When HP ships a system with the OSS environment preconfigured or you use the OSSSETUP utility to configure your system and accept all defaults, ZOSSFSET

currently contains the following listed initial fileset entries. The root fileset entry differs slightly from the default attribute values provided by the OSS Monitor:

Attribute	**Fileset**	**Fileset**	**Fileset** (page 1 of 2)
Fileset name	ROOT	HOME [1]	TEMP [1]
Device label	000000	000001 [1]	000002 [1]
Storage-pool file	ROOTPOOL	HOMEPOOL [1]	TEMPPOOL [1]
Catalog volume: [1]			
● Preconfigured or all defaults accepted	$OSS	$OSS	$OSS
● Single pool volume specified	*disk1*	*disk1*	*disk1*
● Two pool volumes specified	*disk1*	*disk2*	*disk2*
● Three or more pool volumes specified	*disk1*	*disk2*	*disk3*
Mount point	/	/home [1]	/tmp [1]
OSS name server name:			
● Single OSS name server option	#ZPNS	#ZPNS	#ZPNS
● Multiple OSS name server option [1]	#ZPNS	#ZPNH	#ZPNS
AUDITENABLED [1]	OFF	OFF	OFF
BUFFERED [1]	LOG	LOG	CREATE
DESIREDSTATE [2]	STOPPED	STOPPED	STOPPED
FSCKCPU [3]	-1 (same processor as DP2 for fileset catalog)	-1 (same processor as DP2 for fileset catalog)	-1 (same processor as DP2 for fileset catalog)
FTIOMODE [5]	UNBUFFEREDCP	UNBUFFEREDCP	UNBUFFEREDCP
MAXDIRTYINODETIME [2]	30	30	30
MAXINODES [4]	500000	500000	500000
NORMALIOMODE [5]	OSSBUFFEREDCP	OSSBUFFEREDCP	OSSBUFFEREDCP
READ-ONLY [1]	FALSE	FALSE	FALSE

Attribute	Fileset	Fileset	Fileset (page 2 of 2)
REPORT [2]	SUBSYS REPORT specification	SUBSYS REPORT specification	SUBSYS REPORT specification

1 Beginning with RVU G06.15.

2 Beginning with RVU G06.17.

3 Beginning with RVU G06.17 and until RVU G06.18, the default was -1 (same processor as OSS Monitor).

4 Beginning with RVU G06.24. However, if the fileset was created using an earlier RVU, 0 is used as the initial default to force use of a value adequate for the existing number of inodes when the fileset is next started:

- If 110% of the in-use inodes is less than or equal to 500,000, the MAXINODES value is reset to 500,000 when the fileset is restarted.
- If 110% of the in-use inodes is greater than 500,000, the MAXINODES value is reset to the minimum of 2,200,000 and 110% of the number of in-use inodes rounded up to the nearest thousand when the fileset is restarted.

5 Beginning with RVU G06.27. However, if the fileset was created using an earlier RVU, the default behavior depends on the setting of the disks' OSSCACHING attribute in the storage management subsystem.

If your system has been upgraded from a G05.00 or later RVU, the OSS Monitor automatically creates an initial ZOSSFSET file containing the information from the older system's ZPOSFSET file.

If your system has been upgraded from an RVU preceding G05.00, the OSS Monitor automatically creates an initial ZOSSFSET file containing the information from the older system's ZPCONFIG and ZPMNTTAB files.

If your system is upgraded to a G06.24 or later RVU, the initial MAXINODES value is zero; the OSS Monitor changes that value to a more appropriate one as soon as the fileset is started.

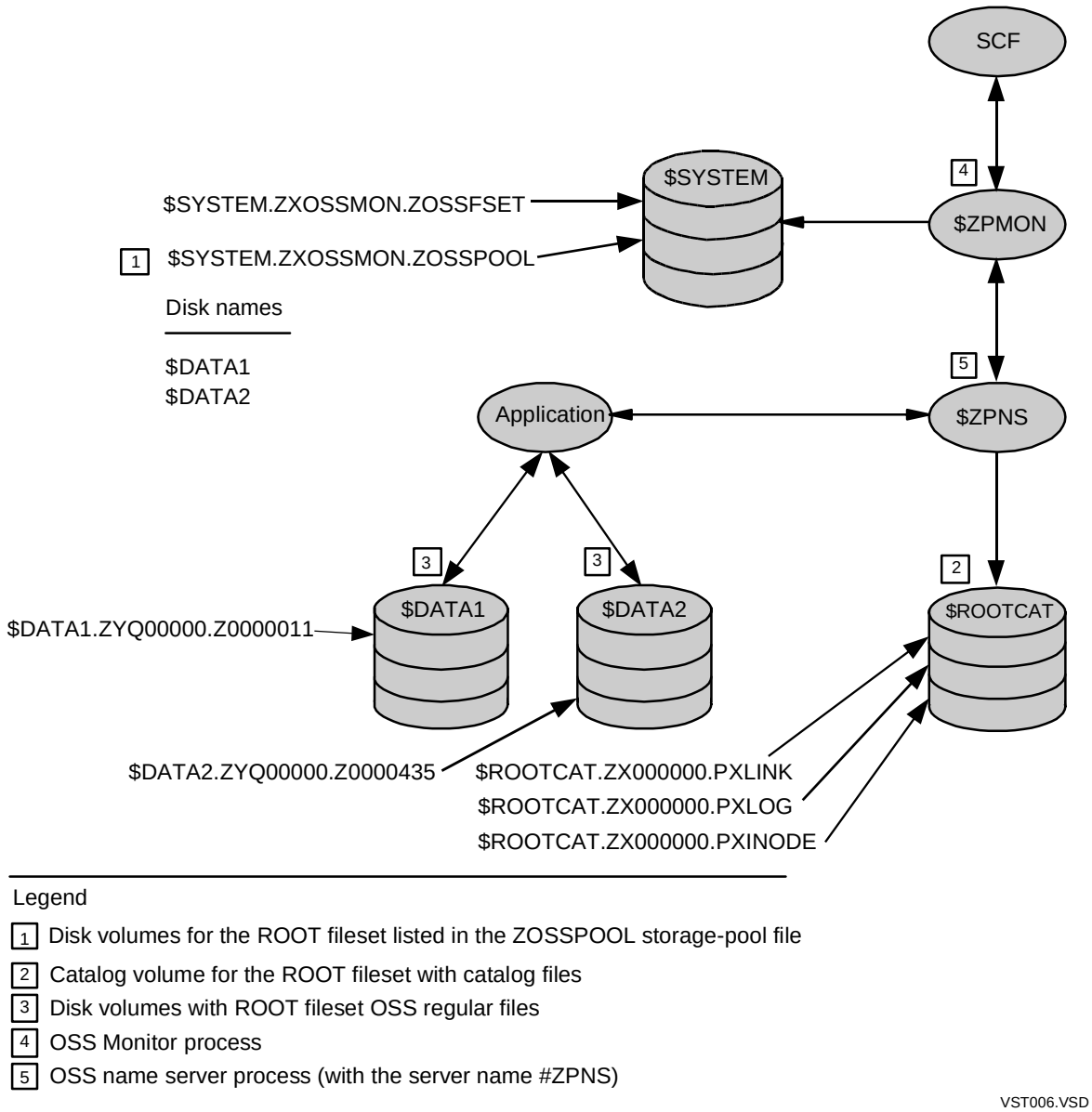
If your system is upgraded to a G06.17 or later RVU, the OSS Monitor automatically upgrades an existing ZOSSFSET file and creates a backup copy of your original file in $SYSTEM.ZXOSSMON.ZOLDFSET.

Figure 4-2 on page 4-12 shows how the entries in the ZOSSFSET file correspond to disk files, disk volumes, and other attributes in the following list:

Attribute	Value in Figure 4-2
Fileset name	ROOT
Device label	000000
Storage-pool file	ZOSSPOOL
Catalog volume	$ROOTCAT
Mount point	/
OSS name server name	#ZPNS
AUDITENABLED	OFF (default)
BUFFERED	NONE (default)
READ-ONLY	FALSE

The files $DATA2.ZYQ00000.Z0000435 and $DATA1.ZYQ00000.Z0000011 represent OSS data files located in the example root fileset (ROOT) illustrated in the figure.

Figure 4-2. Relationship Among OSS Configuration Files, Processes, and Disk Volumes



HP strongly discourages changes in node (system) numbers. However, in a few situations, such as during the first startup of a newly delivered system, you must change node numbers.

When an Enscribe alternate-key database file such as ZOSSFSET is created, the current node number might be embedded in its ALTFILE attributes. If that happens, the attributes become invalid after the node number is changed.

If your system was ordered with the OSS environment preconfigured, then the OSS Monitor was started at least once before you received the system. That action created

ZOSSFSET and might have embedded the factory-default node number in the ALTFILE attributes.

After you change the node number, you must check that the OSS Monitor can open the OSS fileset configuration database the next time the OSS Monitor is started. Enter the following command at a TACL prompt after changing the node number:

```
FUP INFO ZOSSFSET,DETAIL
```

If output includes an ALTFILE name that looks something like:

```
\??.$SYSTEM.ZXOSSMON.ZOSSFS00
```

you need to correct the ALTFILE attributes. The question marks indicate that the Guardian filename cannot be resolved because an unknown node number is embedded. Because the filename cannot be resolved, the database cannot be opened by the OSS Monitor at startup and the OSS file system cannot be restarted.

To fix the situation, enter the following commands at a TACL prompt:

```
VOLUME $SYSTEM.ZXOSSMON
FUP
ALTER ZOSSFSET, ALTFILE (0,ZOSSFS00)
ALTER ZOSSFSET, ALTFILE (1,ZOSSFS01)
EXIT
```

After these commands are executed, FUP INFO, DETAIL output does not contain question marks because the Guardian file system is able to resolve the filename correctly. Subsequent attempts to open ZOSSFSET succeed.

The ZOSSPARM File

The ZOSSPARM file is an Enscribe file that contains the subsystem configuration database for the OSS Monitor. Entries in this file can be displayed by using the OSS Monitor SCF INFO SUBSYS command.

This file must be maintained by the system manager using the OSS Monitor SCF ALTER SUBSYS command. It must be present and must contain valid data before the root fileset is first mounted.

Entries in the ZOSSPARM file must follow these rules:

- Timeout values must be positive multiples of 1 second.
- Any processor number specified must either be:
  - In the range 0 through 15
  - -1 (which specifies a process-dependent option)

Changes to the ZOSSPARM file take effect immediately.

When a default ZOSSPARM file is created by the OSS Monitor, it contains entries that have the default values described in ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34.

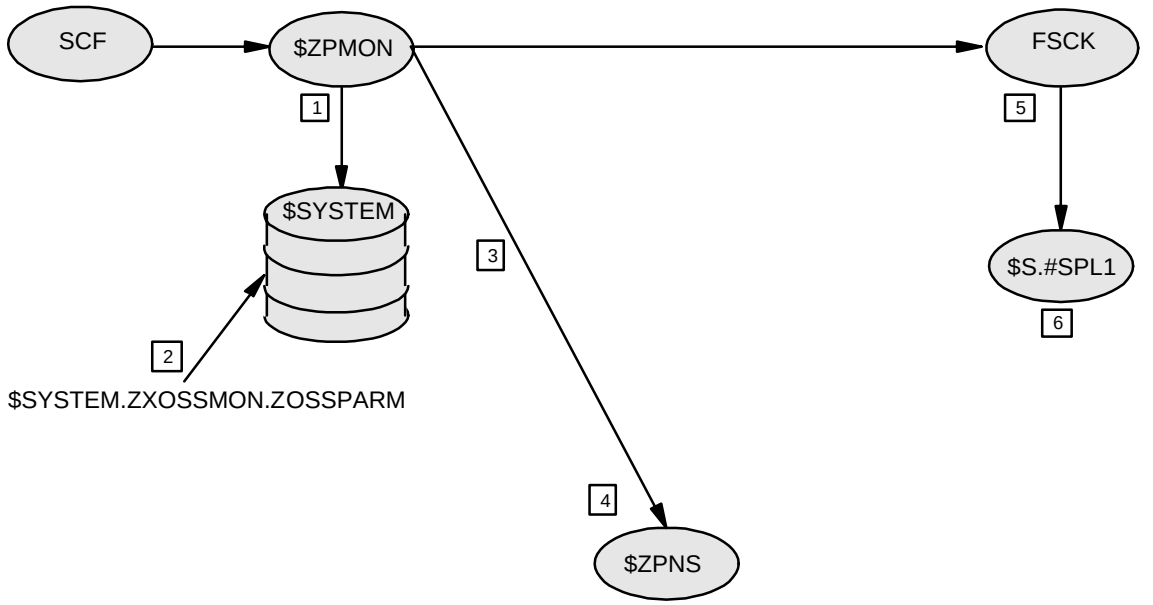
If your system has been upgraded from a G05.00 or later G-series RVU, the OSS Monitor automatically creates an initial ZOSSPARM file containing the information from the old system's ZPOSPARM.

If your system has been upgraded from a G-series RVU preceding G05.00, the OSS Monitor automatically creates an initial ZOSSPARM file containing the information from the old system's TACL PARAMs. See PARAMs Used by the OSS Monitor on page 2-10.

If your system is upgraded to a G06.17 or later G-series RVU, the OSS Monitor automatically upgrades an existing ZOSSPARM file and creates a backup copy of your original file in $SYSTEM.ZXOSSMON.ZOLDPARM.

Figure 4-3 shows how the entries in the ZOSSPARM file affect communications and diagnostic file logging. An entry also controls the processor in which the FSCK utility usually runs.

Figure 4-3. OSS Configuration Files, Processes, and Disk Volumes Affected by Changing ZOSSPARM



Legend

1. OSS Monitor process
2. Name of the subsystem configuration file for the OSS Monitor
3. Request timeout
4. OSS name server process
5. Fileset integrity-checker process
6. Log file default destination

VST016.VSD

The ZOSSSERV File

The ZOSSSERV file is an Enscribe file containing the OSS Monitor configuration database that stores information about the characteristics of each OSS name server. Entries in this file can be displayed with the SCF INFO SERVER command.

This file must be maintained by the system manager using the SCF ADD SERVER, ALTER SERVER, and DELETE SERVER commands. ZOSSSERV must be present before the root fileset is first mounted and before the first OSS name server is started.

Entries in the ZOSSSERV file must follow these rules:

- There should always be a #ZPNS entry for the root OSS name server process, $ZPNS. OSS name servers that manage other filesets must also have entries in this file; use the SCF ADD SERVER Command to add these entries.
- Primary and backup processor numbers should be assigned such that no single processor failure affects more than one server process pair.
- HP recommends that you specify processors not used by either the OSS Monitor or the FSCK utility (as specified by the ZOSSPARM file FSCKCPU value).

Certain changes to the ZOSSSERV file take effect only when the corresponding server is stopped and restarted.

When a default ZOSSSERV file is created by the OSS Monitor, it contains an entry for the OSS name server process of the root fileset, $ZPNS.

Beginning with RVU G06.10, when HP shipped a system with the OSS environment preconfigured, the root fileset OSS name server entry differed from the default, as follows:

Attribute	Default Values When ZOSSSERV Is Created by the OSS Monitor	Initial Values When ZOSSSERV Was Delivered on a Preconfigured System
BACKUPCPU	1	0
CPU	0	1
INODECACHE	4096	4096
LINKCACHE	4096	4096
SERVER	#ZPNS	#ZPNS
SQLTIMEOUT	60	60
TYPE	NAME	NAME

Beginning with RVU G06.15, when HP shipped a system with the OSS environment preconfigured or you used the OSSSETUP utility to configure your system, ZOSSSERV contained these initial OSS name server entries:

Attribute	Preconfigured or Single-Enclosure System	Multiple-Enclosure System
BACKUPCPU	1	1
CPU	0	0
INODECACHE	4096	4096
LINKCACHE	4096	4096

SERVER:		
• For the ROOT fileset	#ZPNS	#ZPNS
• For the HOME fileset	#ZPNS	#ZPNH
• For the TEMP fileset	#ZPNS	#ZPNS
SQLTIMEOUT	60	60
TYPE	NAME	NAME

Beginning with RVU G06.17, when HP ships a system with the OSS environment preconfigured or you use the OSSSETUP utility to configure your system, ZOSSSERV contains these initial OSS name server entries:

Attribute	Preconfigured or Single-Enclosure System	Multiple-Enclosure System
AUTORESTART	0	0
BACKUPCPU	1	1
BACKUPCPUOK	TRUE	TRUE
CPU	0	0
DESIREDSTATE	STOPPED	STOPPED
INODECACHE	4096	4096
LINKCACHE	4096	4096
MAXWAITTIME	0	0
SERVER:		
• For the ROOT fileset	#ZPNS	#ZPNS
• For the HOME fileset	#ZPNS	#ZPNH
• For the TEMP fileset	#ZPNS	#ZPNS
SQLTIMEOUT	60	60
TYPE	NAME	NAME

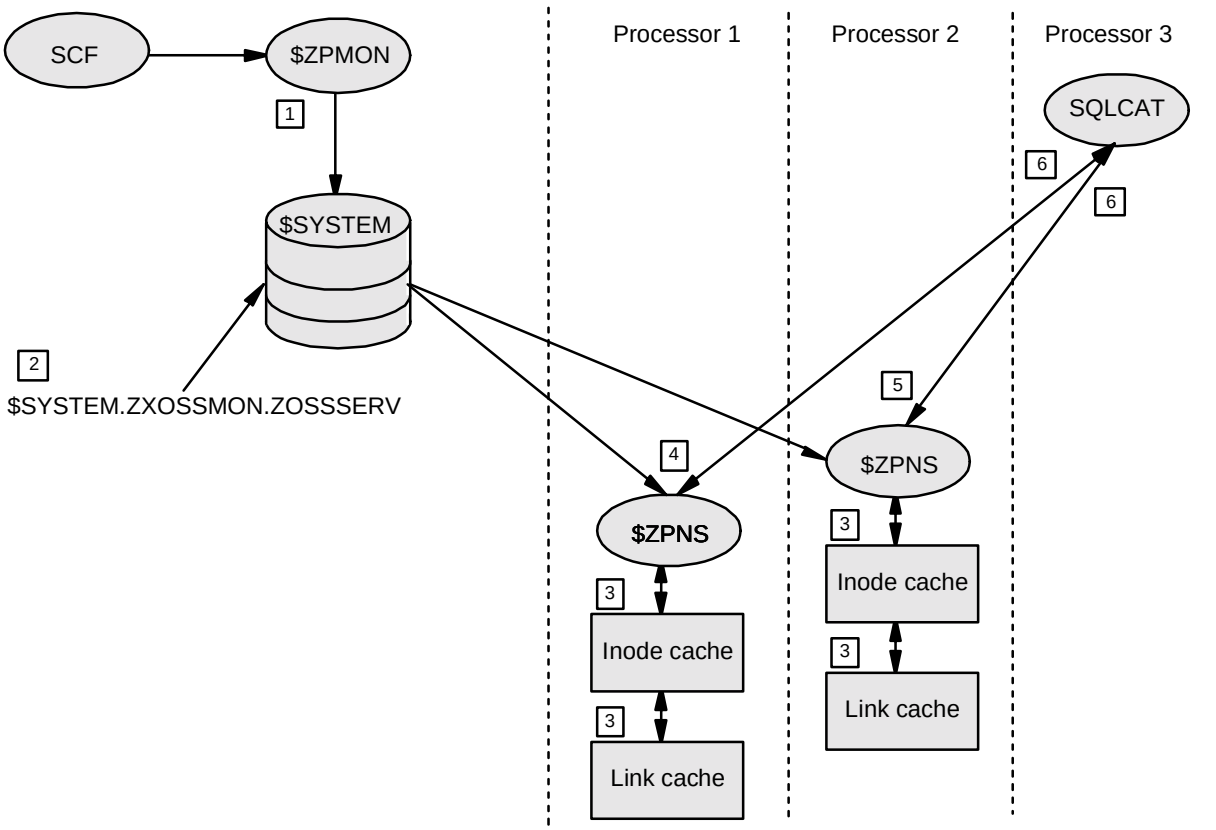
If your system has been upgraded from a G05.00 or later G-series RVU, the OSS Monitor automatically creates an initial ZOSSSERV file containing the information from the older system's ZPOSSERV file.

If your system has been upgraded from a G-series RVU preceding G05.00, the OSS Monitor automatically creates an initial ZOSSSERV file containing the information from the older system's ZPCONFIG and ZPMNTTAB files.

If your system is upgraded to a G06.17 or later G-series RVU, the OSS Monitor automatically upgrades an existing ZOSSSERV file and creates a backup copy of your original file in $SYSTEM.ZXOSSMON.ZOLDSERV.

Figure 4-4 shows how the OSS name server entries in the ZOSSSERV file correspond to processes, disk files, and disk volumes. The entries control OSS name server caching, request timeout between an OSS name server and an SQLCAT, and the processor in which each member of the process pair normally runs.

Figure 4-4. OSS Configuration Files, Processes, and Disk Volumes Affected by Altering an OSS Name Server Entry in ZOSSSERV



Legend

1 OSS Monitor process
2 Name of the configuration database file for the OSS name server
3 OSS name server caches
4 Primary OSS name server process
5 Backup OSS name server process
6 Request timeout value

VST014.VSD

The Storage-Pool Files

The storage-pool files define the disk volumes that each fileset is currently allowed to use when creating new files. The storage-pool files must be created or edited by the system manager using a Guardian text editor. They must be present and must contain valid data before the OSS Monitor is started.

HP recommends:

- The Guardian file identifier for a storage-pool file should either start with Z or have the form *mnemonic*POOL, where *mnemonic* is a four-character identifier that associates that storage-pool file with its corresponding fileset. For example, in a system with only one fileset named ROOT, the storage-pool file should be named ROOTPOOL.

- The file identifier ZOSSPOOL should not be used, to avoid conflict with an unreleased HP feature.
- The file identifier OSSPOOL should not be used, to avoid conflict with the file identifier for the sample storage-pool file installed with the OSS product set.

Entries in a storage-pool file must follow these rules:

- The name of a disk volume can consist of a dollar sign ($) followed by 1 to 6 uppercase letters and digits (A through Z and 0 through 9). The character after the $ must be a letter. (Avoid 8-character disk volume names to avoid problems with HP NonStop SQL/MP program objects that might reside in the OSS environment.)
- One disk volume name can be specified on each line.
- Up to 20 entries are allowed in the file.
- The disk volumes specified:
  - Cannot be optical disks.
  - Cannot be administered through the NonStop Storage Management Foundation (SMF).
  - Should not be in any other storage pool. Disk volumes can be shared among storage pools, but that practice makes it very difficult to monitor and control free space within a fileset.
- HP recommends that:
  - A fileset should not include the disk volumes $SYSTEM or $DSMSCM in its storage pool.
  - The root fileset should have more than one disk volume in its storage pool to allow for future expansion.
  - A small fileset should have only one disk volume in its storage pool for best performance.
- Comments are allowed; see the sample file shown in Figure 5-3 on page 5-8.

Additions to or deletions from a storage-pool file take effect only after the fileset is restarted (remounted).

Beginning with RVU G06.10, when HP shipped a system with the OSS environment preconfigured, a storage-pool file named ZINSPOOL was defined for the single predefined fileset and specified the disk volume $OSS. Beginning with RVU G06.15, when HP ships a system with the OSS environment preconfigured or you use the OSSSETUP utility to configure your system, the initial storage-pool files can be:

Attribute	Fileset	Fileset	Fileset
Fileset name	ROOT	HOME	TEMP
Storage-pool file name	ROOTPOOL	HOMEPOOL	TEMPPOOL

Storage-pool file content:

● Preconfigured or all defaults accepted	$OSS	$OSS	$OSS
● Single pool volume specified	*disk1*	*disk1*	*disk1*
● Two pool volumes specified	*disk1*, *disk2*	*disk1*, *disk2*	*disk2*
● Three or more pool volumes specified	all specified disks	all specified disks	*disk3*

where *disk1*, *disk2*, and *disk3* are volume names specified to OSSSETUP.

Figure 4-2 on page 4-12 shows how a storage-pool file (ZOSSPOOL in the figure) corresponds to the information in other database files in an OSS environment configuration.

Configuration Database Files Used for the OSS Message-Queue Server

To start the OSS message-queue server, the OSS Monitor requires a ZOSSSERV file in the subvolume $SYSTEM.ZXOSSMON.

The ZOSSSERV file is the OSS message-queue server configuration database, and it contains an automatically created entry that identifies the characteristics of the OSS message-queue server process. This file must be edited by the system manager using the SCF ALTER SERVER command.

When a default ZOSSSERV file is created by the OSS Monitor, it contains a #ZMSGQ entry for the OSS message-queue server process, $ZMSGQ.

Beginning with RVU G06.10, when HP shipped a system with the OSS environment preconfigured, the #ZMSGQ entry matched the default, as follows:

Attribute	Default Values When ZOSSSERV Is Created by the OSS Monitor	Initial Values When ZOSSSERV Is Delivered on a Preconfigured System
BACKUPCPU	0	0
CPU	1	1
MSGMQB	65535	65535
MAXMQID	32	32
MAXMSG	32 * MAXMQID = 1024	32 * MAXMQID = 1024
MSGMSIZE	32000	32000
SERVER	#ZMSGQ	#ZMSGQ
TYPE	MSGQ	MSGQ

Beginning with RVU G06.15, when HP shipped a system with the OSS environment preconfigured or you used the OSSSETUP utility to configure your system and

accepted all defaults, ZOSSSERV contained the following initial OSS message-queue server information:

Attribute	Single-Enclosure System	Multiple-Enclosure System
BACKUPCPU	0	3
CPU	1	2
MSGMQB	65535	65535
MAXMQID	32	32
MAXMSG	32 * MAXMQID = 1024	32 * MAXMQID = 1024
MSGMSIZE	32000	32000
SERVER	#ZPLS	#ZPLS
TYPE	LOCAL	LOCAL

Beginning with RVU G06.17, when HP ships a system with the OSS environment preconfigured or you use the OSSSETUP utility to configure your system and accept all defaults, ZOSSSERV contains the following initial OSS message-queue server information:

Attribute	Single-Enclosure System	Multiple-Enclosure System
AUTORESTART	3	3
BACKUPCPU	0	3
BACKUPCPUOK	TRUE	TRUE
CPU	1	2
DESIREDSTATE	STOPPED	STOPPED
MSGMQB	65535	65535
MAXMQID	32	32
MAXMSG	32 * MAXMQID = 1024	32 * MAXMQID = 1024
MAXWAITTIME	0	0
MSGMSIZE	32000	32000
SERVER	#ZPLS	#ZPLS
TYPE	LOCAL	LOCAL

If your system is upgraded to a G06.17 or later G-series RVU, the OSS Monitor automatically upgrades an existing ZOSSSERV file and creates a backup copy of your original file in $SYSTEM.ZXOSSMON.ZOLDSERV.

Configuration Database Files Used for the OSS Sockets Local Server

To start an OSS sockets local server, the OSS Monitor requires a ZOSSSERV file to be in the subvolume $SYSTEM.ZXOSSMON.

The ZOSSSERV file is the OSS sockets local server configuration database, and it contains an automatically created entry that identifies the characteristics of the OSS sockets local server process.

This file must be edited by the system manager using the SCF ALTER SERVER command. ZOSSSERV need not be configured before the root fileset is first mounted or before the first OSS sockets local server is started. The default configuration for ZOSSSERV that is created by OSSMON is complete.

Entries in the ZOSSSERV file must follow these rules:

- There must always be a #ZPLS entry for the OSS sockets local server process, $ZPLS.
- Primary and backup processor numbers must be assigned such that no single processor failure affects more than one server process pair.

HP recommends that you specify processors in the ZOSSSERV file that are not used by either the OSS Monitor or the FSCK utility (as specified by the ZOSSPARM file FSCKCPU value).

Changes to the ZOSSSERV file take effect only when the corresponding server is stopped and restarted.

When a default ZOSSSERV file is created by the OSS Monitor, it contains an entry for the default OSS sockets local server process, $ZPLS.

Beginning with RVU G06.10, when HP shipped a system with the OSS environment preconfigured, the #ZPLS entry matched the default, as follows:

Attribute	Default Values When ZOSSSERV Is Created by the OSS Monitor	Initial Values When ZOSSSERV Is Delivered on a Preconfigured System
BACKUPCPU	0	0
CPU	1	1
SERVER	#ZPLS	#ZPLS
TYPE	LOCAL	LOCAL

Beginning with RVU G06.15, when HP shipped a system with the OSS environment preconfigured or you used the OSSSETUP utility to configure your system and accepted all defaults, ZOSSSERV contained the following initial OSS sockets local server information:

Attribute	Single-Enclosure System	Multiple-Enclosure System
BACKUPCPU	0	2
CPU	1	1
SERVER	#ZPLS	#ZPLS
TYPE	LOCAL	LOCAL

Beginning with RVU G06.17, when HP ships a system with the OSS environment preconfigured or you use the OSSSETUP utility to configure your system and accept all defaults, ZOSSSERV contains the following initial OSS sockets local server information:

Attribute	Single-Enclosure System	Multiple-Enclosure System
AUTORESTART	3	3
BACKUPCPU	0	2
BACKUPCPUOK	TRUE	TRUE
CPU	1	1
DESIREDSTATE	STOPPED	STOPPED
MAXWAITTIME	0	0
SERVER	#ZPLS	#ZPLS
TYPE	LOCAL	LOCAL

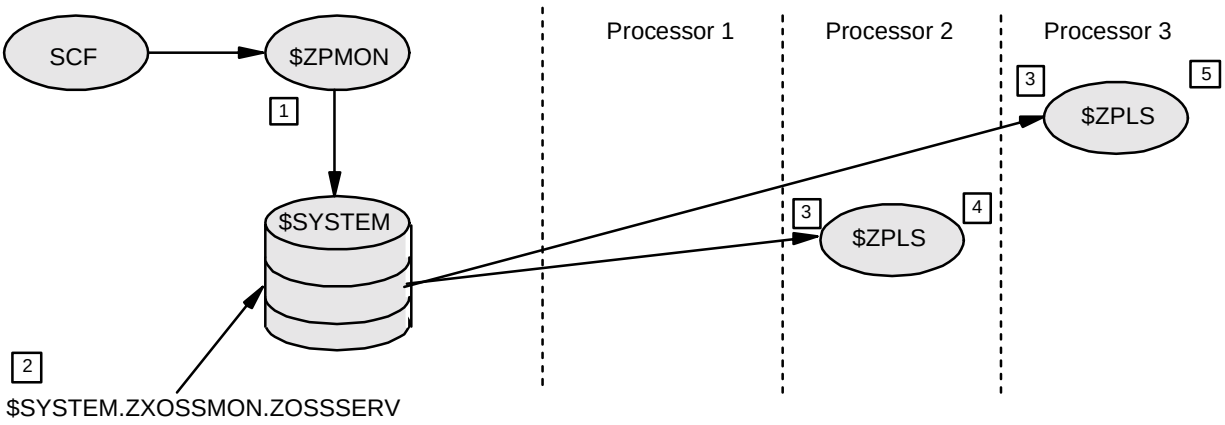
If your system has been upgraded from a G06.00 or later G-series RVU, the OSS Monitor automatically creates an initial ZOSSSERV file containing the information from the older system's ZPOSSERV file.

If your system has been upgraded from a G-series RVU preceding G06.00, the OSS Monitor automatically creates an initial ZOSSSERV file containing the information from the older system's ZPCONFIG and ZPMNTTAB files and adds the entry for $ZPLS to the ZOSSSERV file.

If your system is upgraded to a G06.17 or later G-series RVU, the OSS Monitor automatically upgrades an existing ZOSSSERV file and creates a backup copy of your original file in $SYSTEM.ZXOSSMON.ZOLDSERV.

Figure 4-5 on page 4-23 shows how the OSS sockets local server entries in the ZOSSSERV file correspond to processes, disk files, and disk volumes. The entries control the processor in which each member of the process pair normally runs.

Figure 4-5. OSS Configuration Files, Processes, and Disk Volumes Affected by Altering an OSS Sockets Local Server Entry in ZOSSSERV



Legend

1 OSS Monitor process
2 Name of the configuration file for the OSS sockets local server
3 OSS sockets local server process
4 Primary processor copy of server process
5 Backup processor copy of server process

VST015.VSD

Configuration Database Files Used for the OSS Transport Agent Servers

The ZOSSSERV file is the OSS transport agent server configuration database, and it contains an automatically created entry that identifies the characteristics of each OSS transport agent server process. This entry cannot be changed.

When a default ZOSSSERV file is created by the OSS Monitor, it contains a #ZTA*nn* entry for each copy of the OSS transport agent server process, $ZTA*nn*. These entries have the following default values:

Attribute	Default Value
BACKUPCPU	-1 (does not run as a process pair)
CPU	*nn*
TYPE	TAGENT

The number of entries and the value of *nn* for each entry depends on the number of processors on the node that are up when the OSS Monitor starts.

The OSS Monitor updates the OSS transport agent server entries in the ZOSSSERV file each time the OSS Monitor starts. For example, if processors 0 and 1 of a four-processor system are up when the OSS Monitor starts, the ZOSSSERV file contains a record for #ZTA00 and a record for #ZTA01.

Configuration Files for the Network Services Servers and Tools

The following subsections briefly discuss the configuration files for the following programs:

The remote execution server, `rexecd`, does not have a configuration file.

inetd

The `inetd` process reads its configuration file when it is started or interrupted. That file defines the servers to be invoked when a request comes in from the network. The file contains a table with port-to-service assignments, used to service requests that start servers. Using the information in its configuration file, `inetd` invokes the appropriate server for the connection request.

The configuration file used by `inetd` is either a default file or a file specified in the command that starts `inetd`. The default configuration file is `/etc/inetd.conf`. HP provides a default version of `/etc/inetd.conf`; see inetd on page 4-31 for a description of how to modify `/etc/inetd.conf`. The $SYSTEM.ZTCPIP.PORTCONF file provides analogous information for the Guardian LISTNER process, so you should coordinate content between PORTCONF and the `inetd` configuration file.

△ **Caution.** Changes to `/etc/inetd.conf` might be overwritten during any future installation or reinstallation of the corresponding T9660 product files. See You configure the necessary network services by making AF_INET or AF_INET6 sockets configuration files available in the OSS file system. To prevent confusion and conflicts between servers, use and maintain the Guardian version of the AF_INET or AF_INET6 sockets configuration files for both environments. Set up the Guardian files for use from the OSS environment by creating symbolic links between the Guardian files and the /etc directory. Check that the files or links do not already exist in the /etc directory, then create them if necessary: on page 4-35 for ways to avoid this problem.

See the `inetd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for the format of entries in the `inetd` configuration file.

`inetd` and OSS sockets provide functions that depend on the content of the following files in the `/etc` directory:

```
hosts
networks
protocols
resolv.conf
services
```

These OSS files contain the same kind of information as is used by Guardian sockets programs in the Guardian environment. Guardian sockets services are configured using the following files:

$SYSTEM.ZTCPIP.HOSTS
$SYSTEM.ZTCPIP.IPNODES (for NonStop TCP/IPv6 only)
$SYSTEM.ZTCPIP.NETWORKS
$SYSTEM.ZTCPIP.PROTOCOL
$SYSTEM.ZTCPIP.RESCONF
$SYSTEM.ZTCPIP.SERVICES

There are two versions of the resolver that you can use on a node. Each version has its own rules for the content of its resolver configuration file:

- The BIND 4 version is described in the `resolv.conf(4)` reference page online and in the *Open System Services Shell and Utilities Reference Manual*
- The BIND 9 version is described in the `resolv.conf(5)` reference page online and in the *Open System Services Shell and Utilities Reference Manual*

The Guardian resolver configuration file can be changed from $SYSTEM.ZTCPIP.RESCONF by using the DEFINE =TCPIP^RESOLVER^NAME. The equivalent ability for the OSS environment is provided by using the OSS shell `export` command or the `putenv()` function to set the `TCPIP_RESOLVER_NAME` environment variable to a value other than `/etc/resolv.conf`.

HP provides a default version of each of these files. See the *TCP/IP Configuration and Management Manual* or the *TCP/IPv6 Configuration and Management Manual* for more information about these files; see inetd on page 4-31 and OSS Sockets Applications on page 4-34 for a description of how to provide these files in the OSS environment.

rshd

The `rshd` process does not use a configuration file. However, the behavior of the `rsh` commands that it services can be affected by the content of files on targeted systems:

- For a specific targeted UNIX system, see the `rsh` or `rshd` documentation of that system for more information.
- For users of an `rsh` command on a remote UNIX system who want to gain access to the OSS environment, you need to provide and properly secure the following configuration files in the OSS file system:
  - `/etc/hosts.equiv`
  - `.rhosts`

The `hosts.equiv` file in the `/etc` directory of the OSS file system describes which hosts and which users of each host are allowed to start remote shells on an OSS system. The `.rhosts` file resides in the home directory within the OSS file system for each authorized remote user of the OSS environment. The most important copy of `.rhosts` is the one that resides in the home directory of the super ID because

`/etc/hosts.equiv` is bypassed when a remote user attempts to use an `rsh` command as the super ID.

These files are sometimes the target of UNIX system intruders. Take standard precautions for their use on a UNIX system when setting them up for the OSS environment.

See the `hosts.equiv(4)` and `.rhosts(4)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the content and use of these files.

portmap

The `portmap` process uses the Guardian files configured for TCP/IP processes running in the Guardian environment:

$SYSTEM.ZTCPIP.HOSTS
$SYSTEM.ZTCPIP.RESCONF

HP provides a default version of each of these files. See the *TCP/IP Configuration and Management Manual* or the *TCP/IPv6 Configuration and Management Manual* for more information about these files. For information on changing the configuration of portmap, see portmap on page 4-32.

RPCINFO

The RPCINFO process uses the same Guardian files as the `portmap` process, plus an additional one:

$SYSTEM.ZTCPIP.HOSTS
$SYSTEM.ZTCPIP.RESCONF
$SYSTEM.ZRPC.RPC

HP provides a default version of the HOSTS and RESCONF files. See the *TCP/IP Configuration and Management Manual* or the *TCP/IPv6 Configuration and Management Manual* for more information about those files.

$SYSTEM.ZRPC.RPC is the program definition file that identifies all programs that RPCINFO should report information for. The program definition file is an EDIT file and can be created using any Guardian text editor.

Each line in the RPC file is either a comment line beginning with a pound sign (#) or an entry for one program. Program entries contain the program name to be reported by RPCINFO, the program number, and possible aliases for the program name.

For example, the following sample RPC file defines only one program for RPCINFO to report data about, `portmapper`. This file defines `portmap` and `sunrpc` as being possible aliases for the `portmapper` program (PORTMAP is the name used in the

Guardian environment and in OSS documentation). The portmapper program has RPC program number 100000.

```
#
#   rpc   version 1.16   2001/04/27
#
portmapper      100000  portmap sunrpc
```

The fields of each program-entry line are separated by blanks.

Any entry other than the portmapper entry would be a requirement unique to another product requiring the use of RPCINFO. For the information to include in such entries, see the documentation for the specific product.

BIND 9 Domain Name Server and Tools

The named, lwresd, rndc, and nsupdate programs use the following files, which you need to configure:

File	Purpose
/etc/named.conf	Default DNS name server configuration file; identifies the location of the zone files. A different file can be specified using an environment variable or a command line flag when named is started. The directory used for named.conf can be separately changed from /etc.
/etc/dns923/named.conf	Sample of /etc/named.conf provided for the nonsecure version of named.
/etc/dns_secure/named.conf	Sample of /etc/named.conf provided for the secure version of named.
/etc/resolv.conf	Provides the domain name and IP address of the domain name server to be used when running other BIND utilities such as nsupdate.
/etc/rndc.conf	Default rndc configuration file. A different file can be specified when rndc is started.
/etc/rndc.key	Default file used to authenticate commands sent to named when an /etc/rndc.conf configuration file does not exist.
/var/run/named.pid	The default process-id file used by named.

You can copy and modify the sample files provided:

File	Purpose
/etc/dns923/named.conf	Sample of /etc/named.conf provided for the nonsecure version of named.

File	Purpose (continued)
/etc/dns_secure/named.conf	Sample of /etc/named.conf provided for the secure version of named.
/etc/dns923/rndc.conf	Sample of /etc/rndc.conf provided for the nonsecure version of rndc.
/etc/dns_secure/rndc.conf	Sample of /etc/rndc.conf provided for the secure version of rndc.

See the named(8), lwresd(8), rndc(8), and nsupdate(8) reference pages online or the *DNS Configuration and Management Manual* for more information about the use of these files with the nonsecure BIND 9 server. See the dnssec_named(8), lwresd(8), dnssec_rndc(8), and dnssec_nsupdate(8) reference pages online or the *DNS Configuration and Management Manual* for more information about the use of these files with the secure BIND 9 server.

The DNS security extension utilities (dnssec-keygen and dnssec-signzone) generate the following additional files:

File	Purpose
basename.key	Public key for a signed zone. This file is created by dnssec-keygen.
basename.private	Private key for a signed zone. This file is created by dnssec-keygen.
zone-filename.signed	Signed zone file. This file is created by dnssec-signzone.

See the *BIND 9 Administrator Reference Manual* and the *DNS Configuration and Management Manual* for more information about these files.

Adding a Server

Some kinds of OSS servers can be added to the OSS configuration database. In the current RVU, you can add only an OSS name server for a fileset other than the root fileset.

To add an OSS name server to the OSS configuration database:

1. Determine the appropriate configuration for the new server. For help in defining an appropriate configuration, see the rules described under The ZOSSSERV File on page 4-14.
2. Use the SCF ADD SERVER Command to add the new server to the configuration database.
3. If the new OSS name server is assuming some of the workload of an existing OSS name server:
   a. Use the ALTER FILESET Command to change the OSS name server assigned to the NAMESERVER attribute of a started fileset.

b. Use the STOP FILESET Command to stop the started fileset.

4. Use the START FILESET Command to start (mount) the new or stopped fileset assigned to the new OSS name server. This action automatically starts the new OSS name server.
5. If your site uses the STARTOSS utility and the new OSS name server services a new fileset, you should also add the new fileset name to the OSSINFIL file. See OSSINFIL File on page C-19 for more information.

Other servers, such as the network services servers, do not need to be added to an OSS configuration database. Such servers can be added to the OSS environment by starting them. See Starting a Network Services Server on page 4-38.

Configuring a Server

How and when you configure a server depends on the type of server.

Other servers used by OSS applications require separate procedures. For more information, see the manual appropriate for a specific server.

Configuring an OSS Name Server

You configure an OSS name server by adding or deleting an entry for it in the Enscribe database ZOSSSERV file. Remember that a fileset cannot be managed by more than one OSS name server; however, an OSS name server can manage more than one fileset.

To add a new OSS name server to a configuration, follow the procedure described in Adding a Server on page 4-28. To remove an OSS name server from a configuration, follow the procedure described in Removing an OSS Name Server on page 4-49.

If the initial or default values for the attributes of a specific OSS name server are not optimal, see the procedure described in Reconfiguring an OSS Name Server on page 4-46. The attribute values appropriate to the best performance on a specific system depend on factors unique to each site’s configuration and mix of applications. Use nondefault values to tune your system only after gathering performance data, analyzing process memory requirements, and considering the effects of a default configuration on system overhead.

Configuring the OSS Message-Queue Server

You configure the OSS message-queue server by changing its entries in the Enscribe database ZOSSSERV file. The following procedure assumes that no applications that use message queues have been started yet:

1. To stop the OSS message-queue server, enter this OSS Monitor SCF command:

   ```
   STOP SERVER $ZPMON.#ZMSGQ
   ```

2. Use the OSS Monitor SCF ALTER SERVER Command to change the ZOSSSERV entry for the OSS message-queue server.

   a. To reassign the process to a new primary or backup processor, change the corresponding processor entry. If you do not explicitly assign a backup processor, the backup server process is started in the next higher-numbered processor after the processor that runs the primary server process.

   b. To change behavior associated with use of the automatic startup service, change the corresponding BACKUPCPUOK, MAXWAITTIME, DESIREDSTATE, or AUTORESTART entry.

   c. To change the message-queue configuration, change the corresponding MSGMQB, MAXMQID, MAXMSG, or MSGMSIZE entry. The attribute values appropriate to the best performance on a specific system depend on factors unique to each site's configuration and mix of applications.

      Use nondefault values to tune message-queue performance on your system only after gathering performance data, analyzing process memory requirements, and considering the effects of a default configuration on system overhead because of the volume of interprocess communication involved.

3. To restart the server, enter this OSS Monitor SCF command:

   ```
   START SERVER $ZPMON.#ZMSGQ
   ```

If you use the STARTOSS utility at your site, you can also configure the OSS message-queue server to be started by that utility. To do so, add the server name to the OSSINFIL file. See OSSINFIL File on page C-19 for more information.

Configuring the OSS Sockets Local Server

You configure an OSS sockets local server by changing its entries in the Enscribe database ZOSSSERV file. The following procedure assumes that no applications that use `AF_UNIX` sockets have been started yet:

1. To stop the OSS sockets local server, enter this OSS Monitor SCF command:

   ```
   STOP SERVER $ZPMON.#ZPLS
   ```

2. Use the ALTER SERVER Command to change the ZOSSSERV entry for the OSS sockets local server:

- To reassign the process to a new primary or backup processor, change the corresponding processor entry.
- To change behavior associated with use of the automatic startup service, change the corresponding BACKUPCPUOK, MAXWAITTIME, DESIREDSTATE, or AUTORESTART entry.

3. To restart the server, enter this OSS Monitor SCF command:

```
START SERVER $ZPMON.#ZPLS
```

If you use the STARTOSS utility at your site, you can also configure the OSS sockets local server to be started by that utility. To do so, add the server name to the OSSINFIL file. See OSSINFIL File on page C-19 for more information.

Configuring the OSS Transport Agent Servers

You cannot configure an OSS transport agent server. The OSS Monitor SCF ALTER SERVER command is not valid for OSS transport agent servers.

If processors are added to your node or brought up after the OSS Monitor starts, the OSS Monitor configuration database will not contain entries for the OSS transport agent servers in those processors. To force the OSS Monitor to update its database and add entries for those servers, you must stop and restart the OSS Monitor.

Configuring Network Services Servers, Tools, and Applications

The following subsections briefly discuss configuring:

The RPCINFO program is configured using an EDIT file in the Guardian environment; see Configuration Files for the Network Services Servers and Tools on page 4-24 for more information.

Information about configuring the BIND 9 domain name server `named` and the lightweight resolver server can be found in the *DNS Configuration and Management Manual* and the `lwresd(8)`, `dnssec_ named(8)`, `named(8)`, `dnssec_nsupdate(8)`, `nsupdate(8)`, `dnssec_rndc(8)`, and `rndc(8)` reference pages online.

inetd

To prevent confusion and conflicts between servers, you should use and maintain the Guardian version of the `inetd` configuration files for both environments when Guardian versions exist. The Guardian files can be set up for use from the OSS

environment by creating symbolic links between the Guardian files and the /etc directory:

1. Check that the files or links do not already exist in the /etc directory by entering the following command at an OSS shell prompt:

   ```
   cd /etc
   ls -al
   ```

   If resolv.conf, hosts, and inetd.conf appear, ignore the rest of this procedure and see portmap on page 4-32 to configure the portmap process.

   Note. inetd can use any configuration file identified to it during its startup. The default configuration file is /etc/inetd.conf. If you use the /etc/inetd.conf file and your system does not initially have a smplinetd.conf file, your entries in /etc/inetd.conf can be overwritten during a product update; you should make a backup of the configuration file whenever you change that file. You can make a backup by entering the following at an OSS shell prompt:

   ```
   cp /etc/inetd.conf /etc/inetd.conf.bak
   ```

2. If inetd.conf does not appear but smplnetd.conf appears, at the OSS shell prompt, enter:

   ```
   cp smplinetd.conf inetd.conf
   ```

   However, see the note above; in this guide, the filename is presumed to be inetd.conf, but your site might use alternative names.

3. If neither inetd.conf or smplnetd.conf appears, at the OSS shell prompt, enter:

   ```
   ln -s /G/system/ztcpip/inetconf inetd.conf
   ```

4. If one of the other files does not appear, create a symbolic link to its Guardian equivalent by entering one or more of these commands at the OSS shell prompt:

   ```
   ln -s /G/system/ztcpip/resconf resolv.conf
   ln -s /G/system/ztcpip/hosts hosts
   ```

portmap

The portmap process is configured either by creating NonStop operating system DEFINEs for it in the TACL session used to start it or by passing parameter values to it in the command that starts it.

You can change the following by specifying an ADD DEFINE command at a TACL prompt before starting portmap:

- The TCP/IP (transport-provider) process used by portmap. For example:

  ```
  ADD DEFINE =TCPIP^PROCESS^NAME, FILE $ZTC1
  ```

- The TCP/IP domain name resolution (resolver configuration) file used by `portmap`. For example:

  ```
  ADD DEFINE =TCPIP^RESOLVER^NAME, FILE ALTRES
  ```

- The TCP/IP host definition file used by `portmap`. For example:

  ```
  ADD DEFINE =TCPIP^HOST^FILE, FILE ALTHOST
  ```

Some declarations are not valid in certain combinations. See the `portmap(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the command to start `portmap` and the configuration information that can be specified for it.

rshd

You configure the `rshd` process by:

1. Adding the following entry to the configuration file used for the `inetd` process:

   ```
   shell stream tcp nowait root /bin/rshd
   ```

2. Stopping and restarting `inetd`. Alternatively, you can force `inetd` to reread its configuration file, as described in the `inetd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

You must also configure the OSS environment for users of the `rsh` command on workstations or other NonStop servers. Each `rsh` user must have either a NonStop operating system user ID and login name or an alias configured through Safeguard. If an alias is used, the user must also have an initial working directory (specified by the Safeguard INITIAL-DIRECTORY attribute) defined for it in the OSS file system.

You can also set up an `/etc/hosts.equiv` file or `.rhosts` files for remote `rsh` command users.

△ **Caution.** These files can be used by intruders to compromise your system's security. Create and secure them carefully. The `/etc/hosts.equiv` file must be owned by the super ID and only the super ID must have writer permission for it. An `.rhosts` file must be owned by the NonStop operating system user ID or alias that owns the initial working directory in which it resides and it must be secured such that only the owner has write permission for it.

See the `.rhosts(4)` and `hosts.equiv(4)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about these files.

See the `rshd(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for information about setting up the `rshd` process. See Section 8, Managing Security, for more information about setting up aliases and initial working directories.

rexecd

To configure the rexecd process:

1. Ensure that the /etc/services file is accessible by entering:

   ```
   cd /etc
   ls -al
   ```

   If the /etc/services file is not listed, follow the directions on page 4-35 to make it accessible, then continue with Step 2.

2. Add a port specification for the exec service to the /etc/services file.

3. Add the following entry to the configuration file used for the inetd process:

   ```
   exec stream tcp nowait super.super /bin/rexecd
   ```

4. Stop and restart inetd. Alternatively, you can force inetd to reread its configuration file, as described in the inetd(8) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

You must also configure the OSS environment for remote users of the rexecd server. Each user of the rexecd server must have either a NonStop operating system user ID and login name or an alias configured through Safeguard. If an alias is used, the user must also have an initial working directory (specified by the Safeguard INITIAL-DIRECTORY attribute) defined for it in the OSS file system.

See the rexecd(8) reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about the rexecd process.

OSS Sockets Applications

If your site runs applications that use OSS sockets:

- You should confirm the existence of a $ZTC0 TCP/IP process or define a substitute for that process. Unless an AF_INET or AF_INET6 sockets application program is coded to select its own transport-provider process, it attempts to use the $ZTC0 default process and fails if there is no $ZTC0 transport-provider process; defining a substitute for the default avoids this situation.

  To set up an alternative to $ZTC0 if your node does not run a $ZTC0 process:

  a. Determine the name of the TCP/IP process that is used instead and create a NonStop operating system DEFINE for it. For example:

     ```
     DEFINE =TCPIP^PROCESS^NAME, FILE $B018
     ```

  b. Add this DEFINE to the TACLLOCL file that is executed whenever a TACL session is started.

     If your users do not always login through TACL, follow the equivalent procedure for the /etc/profile file to add the DEFINE to all OSS shells when they are started.

- You must configure network services that such applications might use.

  You configure the necessary network services by making AF_INET or AF_INET6 sockets configuration files available in the OSS file system. To prevent confusion and conflicts between servers, use and maintain the Guardian version of the AF_INET or AF_INET6 sockets configuration files for both environments. Set up the Guardian files for use from the OSS environment by creating symbolic links between the Guardian files and the /etc directory. Check that the files or links do not already exist in the /etc directory, then create them if necessary:

  ```
  cd /etc
  ls -al
  ...
  ln -s /G/system/ztcpip/networks networks
  ln -s /G/system/ztcpip/protocol protocols
  ln -s /G/system/ztcpip/services services
  ln -s /G/system/ztcpip/hosts hosts
  ```

  To use the full NonStop TCP/IPv6 addressing capabilities instead of the /etc/hosts file, add the following NonStop operating system DEFINE to the TACLLOCL file that is executed whenever a TACL session is started:

  ```
  DEFINE =TCPIP^HOST^FILE, FILE $SYSTEM.ZTCPIP.IPNODES
  ```

  If your users do not always login through TACL, follow the equivalent procedure for the /etc/profile file to add the DEFINE to all OSS shells when they are started.

  Note that OSS AF_INET or AF_INET6 sockets and the inetd process use almost the same set of configuration files. NonStop TCP/IPv6 searches for host information can be controlled by the environment variable TCPIP_RESOLVER_ORDER, as described in the environ(5) reference page, either online or in the *Open System Services System Calls Reference Manual*. This environment variable can be set in the .profile file on a user-by-user basis or in /etc/profile for all shell processes that launch OSS sockets programs.

- You can change the transport-provider process used for OSS AF_INET or AF_INET6 sockets by specifying a NonStop operating system DEFINE. This ability is useful when your node runs several copies of its TCP/IP processes or when you run more than one NonStop TCP/IP product at the same time and want to distribute workload between them.

  For example, the following DEFINE allows the $ZTC1 process to be used as an OSS AF_INET transport-provider process:

  ```
  add_define =TCPIP^PROCESS^NAME, FILE \$ZTC1
  ```

  If the DEFINE is declared in the /etc/profile file, then all OSS AF_INET or AF_INET6 sockets applications that are started from an OSS shell prompt use the specified transport provider process. If the DEFINE is declared in the .profile file of a specific user, you can control workload distribution on the basis of user.

- You must keep track of the process names or OSS process IDs (PIDs) that OSS sockets applications use so that those processes running in a specific processor (and possibly all processes used by the same application) can be stopped before you stop an OSS transport agent server.
- You might want to run multiple TCP/IP processes for scalability or load-leveling. Then you can assign OSS `AF_INET` or `AF_INET6` sockets applications to a specific transport-provider process so that the default transport-provider process for your node is not overloaded.

Starting a Server

How and when you start a server depends on the type of server:

- Starting an OSS Name Server on page 4-36
- Starting the OSS Message-Queue Server on page 4-37
- Starting the OSS Sockets Local Server on page 4-37
- Starting an OSS Transport Agent Server on page 4-38
- Starting a Network Services Server on page 4-38

Other servers used by OSS applications require separate procedures. For more information, see the manual appropriate for a specific server.

Starting an OSS Name Server

You do not start an OSS name server directly. Instead, you start at least one of the filesets that it services.

You can start an OSS name server:

- As part of bringing up the OSS environment by using the STARTOSS Utility
- Automatically by having one of its filesets started using the automatic startup service (see the ADD FILESET Command on page 12-7 for more information about that alternative)
- Manually using the following procedure

1. Make sure that you are a member of the super group (255,*nnn*).
2. If the OSS name server you want to start is not the OSS name server for the root fileset:
   a. Use the OSS Monitor SCF INFO FILESET $ZPMON.*, DETAIL command to select the fileset the OSS name server manages that has a mount point closest in the file system directory hierarchy to the `/` directory. See the INFO FILESET Command on page 12-47 for a description of the output.
   b. Use the OSS shell `ls -l` command to make sure that the mount point is defined as a directory in the OSS file system. If it is not, use the OSS shell `mkdir` command to define it.

c. Use the OSS Monitor SCF STATUS FILESET $ZPMON.*, DETAIL command to make sure that all filesets with mount points between / and the mount point directory for the fileset selected in Step 2a are started. See the STATUS FILESET Command on page 12-66 for a description of the output.

d. Issue the OSS Monitor SCF START FILESET Command for any unstarted filesets identified in Step 2c.

3. Issue the OSS Monitor SCF START FILESET command for the OSS name server's fileset and for any unstarted filesets that are mounted on it.

The OSS name server starts automatically when the first of its filesets starts.

Starting the OSS Message-Queue Server

You can start an OSS message-queue server:

- As part of bringing up the OSS environment by using the STARTOSS Utility
- Automatically using the automatic startup service (see the ALTER SERVER Command on page 12-28 for more information about that alternative)
- Manually using the following procedure

To start this server manually when the system is brought up or to restart this server if it fails:

1. Make sure that you are a member of the super group (255,*nnn*).
2. Enter the OSS Monitor SCF command:

```
START SERVER $ZPMON.#ZMSGQ
```

Starting the OSS Sockets Local Server

You can start an OSS sockets local server:

- As part of bringing up the OSS environment by using the STARTOSS Utility
- Automatically using the automatic startup service (see the ALTER SERVER Command on page 12-28 for more information about that alternative)
- Manually using the following procedure

To start this server manually when the system is brought up or to restart this server if it fails:

1. Make sure that you are a member of the super group (255,*nnn*)
2. Enter the OSS Monitor SCF command:

```
START SERVER $ZPMON.#ZPLS
```

Starting an OSS Transport Agent Server

An OSS transport agent is started automatically when its processor starts. You do not need to start an OSS transport agent server to use an OSS AF_UNIX, OSS AF_INET, or OSS AF_INET6 sockets application in a given processor.

To restart a stopped OSS transport agent server:

1. Make sure that you are a member of the super group (255,*nnn*).
2. Enter an OSS Monitor SCF START SERVER Command. For example:

   ```
   START SERVER $ZPMON.#ZTA15
   ```

Starting a Network Services Server

You start network services such as rshd and rexecd by starting the inetd process. To start the inetd process, enter a command similar to the following from an OSS shell prompt:

```
/usr/ucb/inetd -R 10 -W /G/INETD -L /etc/inetd.conf &
```

The maximum number of times a service can be invoked in one minute is controlled by the -R flag to this command. The -W flag allows you to assign a process name to the inetd process so that you can more easily identify and manage it using TACL commands; in the example, the process is named $INETD.

If you start the inetd process using the -L flag, you can use a field within the /etc/inetd.conf file to assign network service server processes to specific processors or otherwise perform load-balancing for your node. You can also change the file that inetd uses from /etc/inetd.conf to another file of your choice. See the inetd(8) reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for a description of the format and flags for the inetd command and a description of the fields within an inetd configuration file.

By default, inetd and all processes that it starts use $ZTC0 as their transport-provider process. However, you can change that use. You make that change by doing either of the following:

- Specifying the PARAM SOCKET^TRANSPORT^NAME at a TACL prompt before starting your OSS shell
- Defining the OSS shell environment variable SOCKET_TRANSPORT_NAME before starting inetd.

The ability to change transport-provider processes is useful when your node runs several copies of its TCP/IP processes and you want to do load-balancing among them. For example:

- The following PARAM allows the $ZTC2 process to be used as the inetd transport-provider process:

  ```
  PARAM SOCKET^TRANSPORT^NAME $ZTC2
  ```

- The following environment variable allows the $ZTC2 process to be used as the `inetd` transport-provider process:

  ```
  export SOCKET_TRANSPORT_NAME=\$ZTC2
  ```

You start the `portmap` program from a Guardian TACL prompt or using the OSS shell `gtacl` utility.

Information about starting the BIND 9 domain name server `named` and the lightweight resolver server `lwresd` can be found in the *DNS Configuration and Management Manual* and the `dnssec_named(8)`, `named(8)`and `lwresd(8)` reference pages online.

Obtaining Information About a Server

You can obtain the following information:

- Whether a server is running in the Guardian environment, as described in Determining Whether a Server Is Running on page 4-39.
- The current configuration of an OSS name server, OSS sockets local server, OSS message-queue server, or OSS transport agent server, as described in Determining the Current Configuration of a Server on page 4-41.
- Usage and configuration information for certain network services servers, as described in Determining Usage and Configuration of Network Services Servers on page 4-42.

Determining Whether a Server Is Running

You can determine whether the following servers are running:

- A server used by the OSS environment but not administered through the OSS Monitor, as described in Checking Servers in the Guardian Environment That Are Not Administered Through the OSS Monitor on page 4-39.
- A server used by the OSS environment and administered through the OSS Monitor, as described in Checking Servers That Are Administered Through the OSS Monitor on page 4-41.
- A server running in the OSS environment and not administered through an SCF interface, as described in Checking Servers That Run in the OSS Environment on page 4-41.

Checking Servers in the Guardian Environment That Are Not Administered Through the OSS Monitor

To determine whether such a server is running, you use the basic SCF LISTDEV command. (There is no OSS Monitor command to provide this information.)

Here is an example: Suppose you want to determine whether a transport-provider process for OSS AF_INET sockets is running. The transport provider is the $ZTC*n* server, described under OSS Sockets on page 1-15.

If you enter the LISTDEV command at an SCF prompt, a display similar to that shown in Figure 4-6 on page 4-40 appears. In the figure, the transport-provider process $ZTC0 is shown as:

- Running, with:
  - Its primary process in processor 1 at PIN 47
  - Its backup process in processor 0 at PIN 63
- Having a:
  - Logical device number of 228
  - Device type of 48
  - Device subtype of 0
  - Scheduling priority of 180
- Using the object file \NODE1.$SYSTEM.SYS00.TCPIP

Figure 4-6. Sample SCF LISTDEV Command Display

```
 LDev Name      PPID    BPID   Type    RSize Pri Program
   .
   .
   .
 142  $XMIOP    0,31           ( 6,4 )    80 190 \NODE1.$SYSTEM.SYS00.XMIOP
 162  $Z003     0,48           (50,63)  3900 150 \NODE1.$SYSTEM.SYS00.SCP
 191  $ZLMG     1,36    0,34   (56,63)   132 149 \NODE1.$SYSTEM.SYS00.MLMAN
 192  $ZEXP     1,37    0,55   (63,30)   132 149 \NODE1.$SYSTEM.SYS00.OZEXP
 193  $ZNET     1,38    0,57   (50,63)  3900 149 \NODE1.$SYSTEM.SYS00.SCP
 215  $ZPMON    0,275          (24,0 )  1024 149 \NODE1.$SYSTEM.SYS00.OSSMON
 222  $ZM00     0,65           (45,0 )   132 201 \NODE1.$SYSTEM.ZQIOLIB.QIOMON
   .
   .
   .
 228  $ZTC0     1,47    0,63   (48,0 ) 32000 180 \NODE1.$SYSTEM.SYS00.TCPIP
More text? ([Y],N) y
 229  $ZTNT     1,49           (46,0 )  6144 155 \NODE1.$SYSTEM.ZTCPIP.TELSERV
```

$ZTC0 is the only transport provider running, because it is the only process displayed with a name of that form. Other servers used by OSS processes and by the OSS subsystem also appear in Figure 4-6.

You can also check configuration information for many servers that are not administered by the OSS Monitor if they appear in the LISTDEV output. Use the following SCF command:

```
INFO PROCESS process-name, DETAIL
```

where *process-name* is the name of the server process as it appears in the LISTDEV output.

Checking Servers That Are Administered Through the OSS Monitor

You can use the OSS Monitor SCF STATUS SERVER command to determine the current status for a server that is administered through the OSS Monitor. For example, if you enter the following command at an SCF prompt:

```
STATUS SERVER $ZPMON.*
```

you can determine the state of all servers administered through the OSS Monitor. The information displayed is the state of current processes.

Additional information about recent server errors is available using the DETAIL option of the OSS Monitor SCF STATUS SERVER command. See the STATUS SERVER Command on page 12-75 for the command syntax and an example.

Checking Servers That Run in the OSS Environment

You can use the OSS shell `ps` command to check the status of any servers started under your current user ID in the OSS environment. If you start servers only under the super ID, then the basic form of this command returns a list of all such servers.

The `ps` command has many HP extensions available through its `-W` flag that you can use to obtain detailed status information about any running process. See the `ps(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information.

Determining the Current Configuration of a Server

You can use the OSS Monitor SCF INFO SERVER command to determine the current configuration settings in the ZOSSSERV database file for a server administered through the OSS Monitor. For example, if you enter the following command at an SCF prompt:

```
INFO SERVER $ZPMON.*
```

you can determine the following information for all servers administered through the OSS Monitor:

- The type of server (OSS name server, OSS message-queue server, and so on)
- The processor used by the primary server process
- The processor used by the backup server process

The following additional information is displayed when you specify the DETAIL option in the INFO SERVER command:

- The user who created the server configuration, when it was created, the user who last modified the configuration, and when it was last modified
- The automatic startup settings for the server (AUTORESTART, DESIREDSTATE, and PERSISTENCECOUNT attributes)
- If the server is an OSS name server:
  - The size of the inode cache
  - The size of the link cache
  - The timeout value used for input or output with the SQL catalog server
- If the server is the OSS message-queue server:
  - The maximum number of bytes allowed in a message queue
  - The maximum number of message queue IDs allowed at any time
  - The maximum number of messages allowed on all message queues on a node
  - The maximum size of a message in bytes

The information displayed is the configuration to be used the next time the server is started. The currently running server process might be using different values.

Additional information about recent server errors is available using the DETAIL option of the OSS Monitor SCF STATUS SERVER Command.

For the OSS message-queue server, you can also obtain information about current usage by entering the `ipcs` command from an OSS shell prompt. For more information about the `ipcs` command, see the `ipcs(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Determining Usage and Configuration of Network Services Servers

To determine information about a running `inetd`, `rshd`, `rexecd`, or `named` process, use the `ps` command. To check the most recent configuration information for `inetd`, use a text editor in the environment in which the configuration file actually resides to view the content of the file.

To determine information about a running `portmap` process, use the RPCINFO command. Examples of RPCINFO use can be found in relevant product manuals, such as the *Open System Services NFS Management and Operations Guide*.

To check the most recent configuration for `portmap`, you need to know how the process was started; the OSS shell `show_define` command or the TACL SHOW DEFINE command might be useful if you know that DEFINEs were used when the process was started.

Stopping a Server

How and when you stop a server depends on the type of server.

Other servers used by OSS applications require separate procedures. For more information, see the manual appropriate for a specific server.

If your site uses the STOPOSS utility, it stops all OSS name servers but does not affect other servers listed here; those servers need not be stopped to shut down a node. See STOPOSS Utility on page C-16 for more information about that alternative.

Stopping a Specific OSS Name Server

1. To identify all filesets managed by the OSS name server you intend to stop, enter the OSS Monitor SCF command:

   ```
   INFO FILESET $ZPMON.*, DETAIL
   ```

   See the INFO FILESET Command on page 12-47 for a description of the command output.

2. Make sure that you are a member of the super group (255,*nnn*).

3. Warn your users to make sure that all their files in affected filesets are closed and all OSS shell sessions using those filesets are terminated. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

4. Do one of the following:

   - If the OSS name server is the server for the root fileset, stop all filesets by entering the following OSS Monitor SCF command:

     ```
     STOP FILESET $ZPMON.*
     ```

   - If the OSS name server is not the server for the root fileset, reassign all filesets it manages to another running OSS name server. Use the OSS Monitor SCF ALTER FILESET Command on each fileset to change the NAMESERVER attribute for the fileset.

   The OSS name server should stop as soon as the last of its filesets stops.

When an OSS name server first starts, it might become “unstoppable” if one of the following is true:

- The OSS name server manages only one fileset and that fileset cannot be started.
- The OSS name server manages multiple filesets and none of them can be started.

If the OSS name server becomes unstoppable, do one of the following:

- Repair all filesets involved using the OSS Monitor SCF DIAGNOSE FILESET Command, restart them, then stop them again.
- Add a new fileset to be managed by the unstoppable OSS name server (see the ADD FILESET Command on page 12-7 for command syntax). Start and stop that fileset; for example, if the new fileset is named DUMMY, enter:

  ```
  START FILESET $ZPMON.DUMMY
  STOP FILESET $ZPMON.DUMMY
  ```

  The fileset can be deleted as soon as it stops.
- Reassign an existing fileset to the unstoppable OSS name server; use the OSS Monitor SCF ALTER FILESET Command on the fileset to change the NAMESERVER attribute for the fileset. Stop that fileset; for example, if you decide to use the TEMP fileset to fix this problem, enter:

  ```
  STOP FILESET $ZPMON.TEMP
  ```

  Remember to reassign the fileset to its proper OSS name server and restart it again.

The unstoppable OSS name server should stop as soon as the last of its filesets stops.

Stopping the OSS Message-Queue Server

1. Make sure that you are a member of the super group (255, *nnn*).
2. Enter the OSS shell `ipcs` command to determine whether any message queues are in use. If any message queues are in use, notify OSS users of the pending shutdown of the OSS message-queue server. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

   See the `ipcs(1)`reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about this utility.
3. Issue the following OSS Monitor SCF command:

   ```
   STOP SERVER $ZPMON.#ZMSGQ
   ```

   As soon as the server stops, any OSS application that was still using an OSS message queue should halt itself. All stale message queues are automatically removed.

Stopping the OSS Sockets Local Server

When you stop the OSS sockets local server, the server notifies all applications using `AF_UNIX` sockets by closing their open sockets. Depending on how an application has been coded, this action might cause the application to fail.

1. Make sure that you are a member of the super group (255,*nnn*).

2. Warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.
3. Issue the following OSS Monitor SCF command:

   ```
   STOP SERVER $ZPMON.#ZPLS
   ```

Stopping an OSS Transport Agent Server

If you need to shut down the QIO subsystem, you must first stop the OSS transport agent server. For more information about the QIO subsystem, see the *QIO Configuration and Management Manual*.

To stop an OSS transport agent server:

1. Make sure you are a member of the super group (255, *nnn*).
2. Warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

Identify and stop all OSS applications that use OSS `AF_UNIX` or OSS `AF_INET` sockets in the processor served by the OSS transport agent that you want to stop. An OSS transport agent server will not shut down if it is servicing an open socket.

Use an OSS Monitor SCF STOP SERVER Command to stop the OSS transport agent server. For example:

```
STOP SERVER $ZPMON.#ZTA15
```

Stopping a Network Services Server

Network services servers do not shut down if they are in use. You must:

1. Warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.
2. Stop each process separately by identifying its OSS process ID (PID) using the `ps` command from an OSS shell prompt and then issuing a `kill` command.

   For example, to stop the `inetd` process, enter commands similar to the following:

   ```
   ps
   .
   .
   .
   kill -s KILL 4291
   ```

   where 4291 is the OSS process ID of the `inetd` process. This form of the `kill` command sends the nondefault `KILL` signal to the `inetd` process.

Reconfiguring a Server

How and when you reconfigure a server depends on the type of server.

You cannot reconfigure an OSS transport agent server.

Other servers used by OSS applications require separate procedures. For more information, see the manual appropriate for a specific server.

Reconfiguring an OSS Name Server

You reconfigure an OSS name server by changing its entry in the Enscribe database ZOSSSERV file.

1. Make sure that you are a member of the super group (255,*nnn*).
2. Use the OSS Monitor SCF ALTER SERVER Command for the OSS name server. Change one or more of the following attributes: AUTORESTART, BACKUPCPU, BACKUPCPUOK, CPU, DESIREDSTATE, INODECACHE, LINKCACHE, MAXWAITTIME, and SQLTIMEOUT.
3. If you changed only one or more of these attributes:

   AUTORESTART, BACKUPCPUOK, or MAXWAITTIME, you have completed the task. The changes take effect immediately.
4. If you changed only one or more of these attributes:

   BACKUPCPU, CPU, INODECACHE, LINKCACHE, and SQLTIMEOUT, use the SCF CONTROL SERVER Command with the SYNC option for the OSS name server. You have completed the task. The changes take effect immediately.
5. Otherwise, use the following OSS Monitor SCF command to identify all filesets managed by that server:

   ```
   INFO FILESET $ZPMON.*, DETAIL
   ```

   See the INFO FILESET Command on page 12-47 for a description of the command output.
6. Warn your users to make sure that all their files in affected filesets are closed and all OSS shell sessions using those filesets are terminated. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

7. Do one of the following:
   - If the OSS name server is the server for the root fileset, stop all filesets by entering the following OSS Monitor SCF command:

     ```
     STOP FILESET $ZPMON.*
     ```
   - If the OSS name server is not the server for the root fileset, reassign all filesets it manages to another running OSS name server. Use the OSS Monitor SCF ALTER FILESET Command on each fileset to change the NAMESERVER attribute for the fileset.

   The OSS name server should stop as soon as the last of its filesets stops.
8. If the OSS name server you want to restart with changed attributes is not the OSS name server for the root fileset:
   a. Use the following OSS Monitor SCF command to identify the fileset that OSS name server manages that has a mount point closest in the file system directory hierarchy to the `/` directory:

      ```
      INFO FILESET $ZPMON.*, DETAIL
      ```

      See the INFO FILESET Command on page 12-47 for a description of the command output.
   b. Use the following OSS Monitor SCF command to make sure that all filesets with mount points between `/` and its mount point directory are started:

      ```
      STATUS FILESET $ZPMON.*, DETAIL
      ```

      See the STATUS FILESET Command on page 12-66 for a description of the command output.
9. Issue the OSS Monitor SCF START FILESET Command for the OSS name server's fileset.

Restarting any one of its filesets restarts the reconfigured OSS name server. Repeat Step 9 if necessary to restart all the filesets managed by the restarted OSS name server.

Reconfiguring the OSS Message-Queue Server

You reconfigure the OSS message-queue server by changing its entries in the Enscribe database ZOSSSERV file and restarting the server.

1. Make sure that you are a member of the super group (255,*nnn*).
2. Use the OSS Monitor SCF ALTER SERVER Command to change the appropriate field in the ZOSSSERV entry for the OSS message-queue server. See Configuring the OSS Message-Queue Server on page 4-30 for information about the fields that you can change.

3. If you changed only one or more of these attributes:

   AUTORESTART, BACKUPCPUOK, or MAXWAITTIME, you have completed the task. The changes take effect immediately.

4. Otherwise, warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

5. Use the following OSS Monitor SCF command to stop the server:

   ```
   STOP SERVER $ZPMON.#ZMSGQ
   ```

6. Use the following SCF command to restart the reconfigured server:

   ```
   START SERVER $ZPMON.#ZMSGQ
   ```

Reconfiguring the OSS Sockets Local Server

You reconfigure the OSS sockets local server by changing its entry in the Enscribe database ZOSSSERV file and restarting the server.

1. Make sure that you are a member of the super group (255,*nnn*).

2. Use the OSS Monitor SCF ALTER SERVER Command for the local server to change the appropriate attribute.

3. If you changed only one or more of these attributes:

   AUTORESTART, BACKUPCPUOK, or MAXWAITTIME, you have completed the task. The changes take effect immediately.

4. Otherwise, warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

5. Locate and stop all OSS applications that use `AF_UNIX` sockets.

   When you stop the OSS sockets local server to restart it using the new configuration, the server notifies all applications using `AF_UNIX` sockets by closing their open sockets. Depending on how an application has been coded, this action might cause the application to fail.

6. Use the following OSS Monitor SCF command to stop the server:

   ```
   STOP SERVER $ZPMON.#ZPLS
   ```

7. Use the following OSS Monitor SCF command to restart the reconfigured server:

   ```
   START SERVER $ZPMON.#ZPLS
   ```

8. If feasible, restart all applications using OSS `AF_UNIX` sockets.

Reconfiguring a Network Services Server

Most network services servers ignore changes to configuration files while they are running. The BIND 9 domain name server `named` can be reconfigured using the `rndc` or `nsupdate` utility, as described in the *DNS Configuration and Management Manual* and the `nsupdate(8)` and `rndc(8)` reference pages online.

Configuration files such as `/etc/inetd.conf` can be edited while the servers are running. However, to make such configuration changes take effect, you must:

1. Warn your users. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.
2. Stop or interrupt each process separately by identifying its OSS process ID (PID) using the `ps` command from an OSS shell prompt and then issuing a `kill` command.

   For example, to interrupt the `inetd` process, enter commands similar to the following:

   ```
   ps
   .
   .
   .
   kill -s SIGHUP 4291
   ```

   where 4291 is the OSS process ID of the `inetd` process. This form of the `kill` command sends the `SIGHUP` signal to the `inetd` process; `inetd` rereads its current configuration file in response to this signal and continues to run using the new configuration.

Removing a Server

How and when you remove a server depends on the type of server. You cannot remove the OSS sockets local server, the OSS message-queue server, or an OSS transport agent server.

- Removing an OSS Name Server on page 4-49
- Removing a Network Services Server on page 4-50

Other servers used by OSS applications require separate procedures. For more information, see the manual appropriate for a specific server.

Removing an OSS Name Server

You remove an OSS name server by removing its entry in the Enscribe database ZOSSSERV file.

Note. You cannot remove the OSS name server for the root fileset, $ZPNS.

---

1. Make sure that you are a member of the super group (255,*nnn*).

2. Use the following OSS Monitor SCF command to determine which filesets are managed by the OSS name server that you want to remove:

   ```
   INFO FILESET $ZPMON.*, DETAIL
   ```

   See the INFO FILESET Command on page 12-47 for a description of the output.

3. Use the OSS Monitor SCF ALTER FILESET Command on each fileset managed by the OSS name server you want to remove to change its NAMESERVER entry to specify another OSS name server.

4. Warn your users to make sure that all their files in affected filesets are closed and all OSS shell sessions using those filesets are terminated. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.

5. Use the OSS Monitor SCF STOP FILESET Command on each fileset managed by the OSS name server you want to remove.

   The OSS name server stops itself as soon as the last fileset it was managing stops.

6. Use the OSS Monitor SCF START FILESET Command on each fileset formerly managed by the OSS name server you want to remove.

7. Use the OSS Monitor SCF DELETE SERVER Command to remove the stopped OSS name server from your configuration.

8. If your site uses the STARTOSS utility and the deleted OSS name server serviced a deleted fileset, you should also delete the fileset name from the OSSINFIL file. See OSSINFIL File on page C-19 for more information.

Removing a Network Services Server

Stopping the `inetd`, `lwresd`, or `named` server effectively removes that server from the OSS environment. See Stopping a Network Services Server on page 4-45 for more information on stopping the `inetd`, `lwresd`, or `named` process.

To remove the remote shell server `rshd` or the remote execution server `rexecd`, edit the `/etc/inetd.conf` file to delete or comment out the entry for `rshd` or `rexecd`.

Although `portmap` and `rpcinfo` are used by products in the OSS environment, they are not OSS processes and cannot be removed from the OSS environment. Stopping them in the Guardian environment effectively removes then from the node.

Troubleshooting a Server

When you have problems managing a server, follow this general procedure:

1. Check the messages from the OSS Monitor that are sent to your terminal. If you redirect such messages to a log file, check the log file for its most recent entries. Look up the cause, effect, and recovery information for a message either in OSS

Monitor Messages on page A-27 or by using the SCF HELP facility described in Online Help Facility on page 12-2.

2. When you are unsure of the outcome of a command entry, check the EMS log for the most recent status messages. Look up the status messages in the *Operator Messages Manual*. For a list of the subsystem IDs to look for in the EMS log, see Appendix A, Messages.
3. When you are unsure of the effect of an action on a server, use the STATUS SERVER command DETAIL option to obtain the last internally reported error information for it. Look up that error in Numbered Messages on page A-35.

5 Managing Filesets

This section describes how to perform operations on Open System Services (OSS) filesets.

You use OSS Monitor SCF commands for all these tasks except managing and repairing fileset catalogs. You use the Guardian environment CVT utility to perform operations on fileset catalogs.

Creating a Fileset

You can create a new fileset when:

- You install a unique set of related files, such as a product or utilities.
- You need to establish access permissions that apply to a set of files as a whole but are not appropriate for all files (for example, a fileset can be made read-only).

Creating a Unique Fileset

To create a unique fileset:

1. Make sure that the mount-point directory exists; if it does not, create it and give it universal access permission. HP recommends against the use of symbolic links as mount-point names or within mount-point names; such use can cause problems when filesets are restarted.

   Use either the OSS shell directly or the TACL OSH command to create the mount-point directory, as shown in the following examples:

   - To create the directory `/home/henrysp` from within the OSS environment, enter the following OSS shell commands:

     ```
     /bin/mkdir /home/henrysp
     /bin/chmod 0777 /home/henrysp
     ```

- To create the directory `/home/henrysp` from within the Guardian environment, enter the following commands at a TACL prompt:

```
OSH -p /bin/mkdir /home/henrysp
OSH -p /bin/chmod 0777 /home/henrysp
```

Note that the OSS shell `chmod` command gives read, write, and search (execute) permission to all users of OSS files within the mount-point directory. The UNIX sticky bit can also be set so that only the super ID can delete files from the fileset; to set the sticky bit, specify `chmod 1777` instead of `chmod 0777`.

2. Create a storage-pool file with the name specified in the new ZOSSFSET entry. This action is described in Creating a Storage Pool on page 5-6.

3. Add a record for the fileset to the ZOSSFSET file by using the SCF ADD FILESET command. Select the appropriate settings for the fileset:

   - HP suggests that you select a consistent name for the fileset, directory mount point, and report file filename to make administration of the fileset easier.
   - If the fileset is frequently used, consider having it started or restarted automatically after a system load or processor failure. An automatically started fileset cannot start until all filesets above it in the OSS directory hierarchy are started, so a fileset with a DESIREDSTATE of STARTED cannot start if it uses a mount point on a fileset with a DESIREDSTATE of STOPPED.
   - Read the information under Changing Fileset Input/Output Fault Tolerance on page 5-15 and choose a fault-tolerance option for file input/output within the fileset.
   - Read the information under Changing Fileset Catalog Buffering on page 5-17 and choose a buffering option. These options allow you to increase the relative speed of file creation, deletion, or opens within the fileset at the expense of increasing the probability of needing to perform a recovery (repair) for the fileset after a failure. Using a memory cache to buffer open and close catalog file information for the fileset instead of recording each transaction in a disk file speeds up transaction processing slightly; however, if the cache is lost before it can be flushed to disk, the true state of the fileset cannot be determined without performing a recovery on it:
     - A fileset using the BUFFERED NONE option directly accesses the fileset catalog on disk without caching any data in memory. This direct access provides the highest assurance of data integrity for the fileset catalog and is the least likely to require a recovery. However, BUFFERED NONE provides the slowest access time to the catalog. A fileset in which file creation, deletion, and opens are infrequent is a good candidate for the BUFFERED NONE option.
     - A fileset using the default BUFFERED LOG option provides slightly faster access time by using memory cache for some open and close catalog file information; the tradeoff between performance and recovery for such filesets can be adjusted using the MAXDIRTYINODETIME option. A fileset

in which file creation, deletion, or opens are a relatively common occurrence is a a good candidate for the BUFFERED LOG option.

- A fileset using the BUFFERED CREATE (fast-create) option uses memory cache for as much file creation, open, and close information as possible between updates of the actual fileset catalog file on disk. This indirect access to the catalog file provides the fastest access time for the fileset catalog but with reduced assurance of data integrity for the catalog and the highest probability of requiring a recovery after a failure. A fileset in which files are constantly created, deleted, and opened is a good candidate for the BUFFERED CREATE option.

  Although using the BUFFERED CREATE option for a fileset provides better performance than not using it, the following disadvantages exist:

  - Fast-create filesets can contain only one disk volume, and the catalog must reside on that volume.
  - If there is a double failure of the disk process serving a fast-create fileset, recently created files might be permanently lost.

    The fileset becomes inaccessible until it is stopped, repaired using the SCF DIAGNOSE FILESET command (see Checking and Repairing Fileset Integrity on page 5-24), and restarted.
  - Fileset recovery information is not recorded in the PXLOG catalog file (see FSCK Log File on page 5-25).
  - After a double failure of an OSS name server, when the OSS Monitor attempts to remount a fast-create fileset, the remount request is rejected by the OSS name server. However, the OSS Monitor automatically runs the FSCK utility against the fileset and then retries the remount.

  HP recommends that you have a fast-create fileset named TEMP mounted on `/tmp`. `/tmp` should be secured so that all users have read, write, and execute or search permission, but only the file owner or the super ID can delete directories or files from the fileset; use this command at an OSS shell prompt:

  ```
  /bin/chmod 1777 /tmp
  ```

- Specify a processor for the FSCK utility. This action allows you to spread fileset recovery after a system failure and reload across many processors so that filesets can be repaired in parallel. That practice allows for faster recovery and improves the availability of the OSS file system.

Figure 5-1 on page 5-4 shows the relationships created among the configuration files, processes, and disk volumes by using the SCF ADD FILESET command. The figure illustrates the effects of this command:

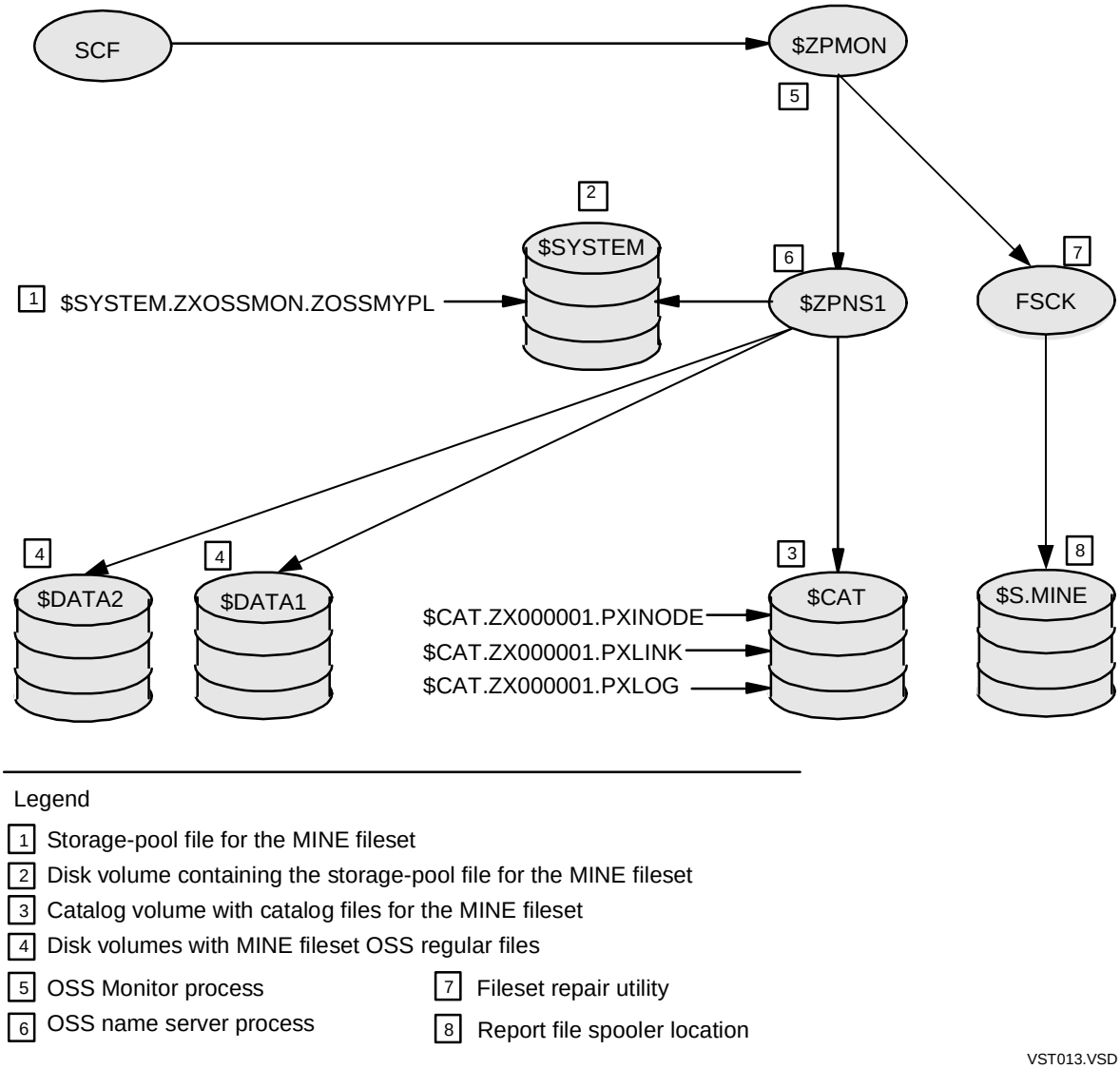
```
ADD FILESET $ZPMON.MINE, CATALOG $CAT, POOL ZOSSMYPL, &
MNTPOINT "/mine", BUFFERED LOG, NAMESERVER #ZPNS1,    &
```

```
NFSTIMEOUT 70, DESIREDSTATE STARTED, FSCKCPU -1,      &
MAXDIRTYINODETIME 10, REPORT $S.#MINE
```

This command:

- Adds the fileset named MINE at the mount point `/mine` of the local node and sends repair information to the report file spooler location $S.#MINE
- Keeps the fileset catalog on disk volume $CAT

Figure 5-1. OSS Configuration Files, Processes, and Disk Volumes Involved in Adding a Fileset



- Only checkpoints what would be PXLOG file entries
- Allows up to 500,000 files (the default MAXINODES value) in the fileset

- Decreases the likelihood of needing fileset recovery by decreasing the value used for MAXDIRTYINODETIME from its default value to 10 seconds
- Makes the fileset automatically restart when necessary if the automatic startup service is used, and uses the processor (specified as -1) used by the disk process for the fileset to perform any fileset repairs needed
- Specifies that the OSS name server for the fileset retains the results of nonretryable Network File System (NFS) operations for 70 seconds

- Assigns the fileset MINE to the storage pool defined by the storage-pool file with the name of ZOSSMYPL
- Allows read and write access to the fileset by default
- Accepts the FTIOMODE and NORMALIOMODE default settings for file opens

4. Start the fileset with the SCF START FILESET command.

   Figure 5-2 on page 5-6 shows how a fileset is mounted at a mount point when the fileset is started. The figure illustrates the effects of the following command:

   ```
   START FILESET $ZPMON.USER1
   ```

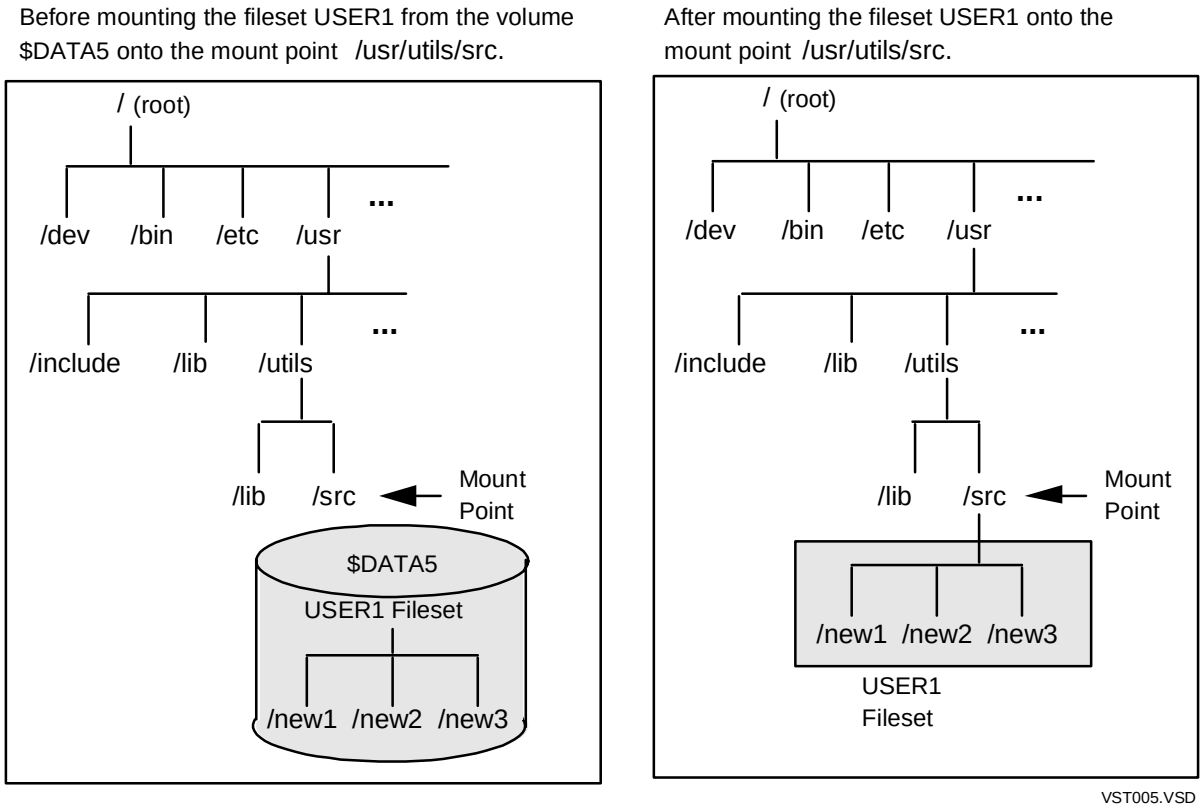
   The files new1, new2, and new3 in the fileset USER1 are not available to users until this command is entered, although they are in the system on the disk volume $DATA5. Previous commands, such as:

   ```
   ADD FILESET $ZPMON.USER1, CATALOG $USR1CAT, POOL ZOSSU1PL, &
   MNTPOINT "/usr/utils/src"
   START FILESET $ZPMON.USER1
   .
   .
   .
   STOP FILESET $ZPMON.USER1
   ```

   would have allowed these files to be created in the storage-pool file ZOSSU1PL (not shown) on the catalog volume $USR1CAT (not shown).

   After the fileset USER1 is mounted, the files new1, new2, and new3 are available as /usr/utils/src/new1, /usr/utils/src/new2, and /usr/utils/src/new3.

Figure 5-2. Starting (Mounting) a Fileset



Creating a Storage Pool

You create a storage pool by creating a storage-pool file.

You can use any valid Guardian file identifier for the name of a storage-pool file. However, you should not name a storage-pool file OSSPOOL so that your storage pool definition is not overwritten by the sample file in a reinstallation or an upgrade of the OSS environment.

To define the storage-pool file that is associated with a fileset record:

1. Use a Guardian text editor to create the storage-pool file.

   The storage-pool file must be in the same subvolume ($SYSTEM.ZXOSSMON) as the ZOSSFSET file.

2. Enter the names of disk volumes that will contain the OSS files in this fileset, one disk volume name on each line. See The Storage-Pool Files on page 4-17 for the rules about specifying disk volumes in storage-pool files.

3. Exit the editor.

The name of the storage-pool file used for each fileset appears in that fileset's database record. The OSS Monitor uses this information to tell the OSS name servers where to put OSS data files.

Figure 5-3 on page 5-8 shows the contents of a storage-pool file. This storage-pool file was created during installation by copying the sample EDIT file OSSPOOL from the ZXOSSMON subvolume of the installation volume and modifying it appropriately.

Starting (Mounting) or Restarting Filesets

An OSS fileset is not available to users until it is started or restarted. This action is also known as mounting a fileset.

You start a fileset:

- At least once to create the catalog files for it
- After it has been deliberately stopped for any reason
- When it is not automatically restarted

If your site uses the STARTOSS utility, that utility starts all filesets named in the OSSINFIL file. See STARTOSS Utility on page C-14 for more information.

Alternatively, you can use the SCF START FILESET Command to manually start or restart an existing fileset, as described on page 12-64. Only super-group users (255,*nnn*) can use the START FILESET command.

Certain failure conditions cause filesets to be automatically restarted. The following subsections describe:

- Automatic Restart of Filesets During OSS Monitor Startup on page 5-8
- Automatic Restart of Filesets by the Automatic Startup Service on page 5-9
- Automatic Restart of Filesets After OSS Name Server Failure on page 5-10
- Automatic Restart of OSS Name Servers After Processor Failure on page 5-10
- Potential Problems During Automatic Restart of Filesets on page 5-10

Figure 5-3. Example of a Storage-Pool File

```
==
== File:  $SYSTEM.ZXOSSMON.OSSPOOL
==
== This file is a sample POOL file.  A POOL file defines the disk volumes
== where OSS files of an OSS FILESET can be created.  OSS FILESETs, which
== are managed by the OSS Monitor ($ZPMON), have an attribute named POOL,
== which is the name of a Guardian EDIT file that resides in the subvolume
== $SYSTEM.ZXOSSMON.  Multiple OSS FILESET objects may share a common
== POOL file.

== The contents of this file are a list of disk volumes, one per line.
== Up to 20 disk volumes may be specified in a POOL file. Leading whitespace
== is not allowed before a filename, but trailing whitespace is allowed.
== Comment lines are allowed and start with the characters ==
==
== This sample POOL file specifies 3 disk volumes where OSS files may be
== created.
==
$OSS1
$OSS2
$OSS3
==
== ****************************************************************************
== * WARNING  WARNING  WARNING  WARNING  WARNING  WARNING  WARNING  WARNING *
== ****************************************************************************
== *                                                                          *
== * 1) This file, $SYSTEM.ZXOSSMON.OSSPOOL, is a sample file that is        *
== * replaced each time the T8622 product is installed.  Do not configure    *
== * OSS FILESETs to use this file as a POOL file, since this file will be   *
== * overwritten during each T8622 installation.                             *
== *                                                                          *
== * 2) Do not use the volume $SYSTEM in POOL file, as OSS files should not  *
== * be placed on $SYSTEM.                                                   *
== *                                                                          *
== * 3) The contents of a POOL file may be changed at any time, however,     *
== * changes only take affect when the associated FILESET is started.        *
== *                                                                          *
== * 4) For installations that use SQL/MP and have OSS programs with         *
== * embedded SQL statements in them, do not use volumes that have 8         *
== * character names (including the $).  SQL/MP has a restriction that       *
== * it cannot access programs that reside on volumes with 8 character       *
== * names.                                                                  *
== *                                                                          *
== ****************************************************************************
```

Automatic Restart of Filesets During OSS Monitor Startup

The OSS Monitor uses the ZOSSFSET file to record the state of each fileset (for example, STARTED, STOPPED, or DIAGNOSING). Whenever a fileset is started, an

entry is made indicating that the fileset is in the STARTED state. When the fileset is stopped, the state of the fileset is changed to STOPPED.

When the OSS Monitor is started for the first time after a system load, it checks the desired-state configuration for all filesets to determine which filesets it must start or restart. Refer to Automatic Restart of Filesets by the Automatic Startup Service on page 5-9 for more information.

When the OSS Monitor is restarted at any other time, it performs a restart sequence. The OSS Monitor checks the recorded state of each fileset.

For each fileset that was left in the STARTED state, the OSS Monitor checks whether the OSS name server for that fileset is still running. If that OSS name server is still running and the fileset was left in the STARTED state, the OSS Monitor assumes that the fileset is still in the STARTED state and is not corrupt.

If the OSS Monitor finds that a fileset was left in the STARTED state but its OSS name server process is not running, one of the following might have occurred:

- A serious problem has occurred.
- A fileset was not properly stopped before the OSS Monitor last stopped.
- An OSS name server failed while the OSS Monitor was not running.

That fileset might need repair.

If the OSS Monitor finds that a fileset was left in the UNKNOWN state, its OSS name server might have failed while the OSS Monitor was not running. Again, that fileset might need repair.

The OSS Monitor then attempts to restart each fileset suspected of needing repair, in the order of the fileset mount points within the OSS file system directory structure, beginning with the root fileset. If the OSS name server for a fileset reports that the fileset cannot be restarted, the OSS Monitor runs the Guardian FSCK utility, then tries again to restart the fileset.

The OSS Monitor restart sequence does not wait for the FSCK repair operation on a fileset to finish; the restart sequences continues with other filesets. The OSS Monitor does wait indefinitely for a repair operation to finish before attempting to restart the fileset. If the attempt to restart the fileset fails, the OSS Monitor marks the fileset state as UNKNOWN.

If the OSS Monitor is restarted while an OSS name server is running, the new instance of the OSS Monitor continues to monitor the OSS name server and can recover from any future failures of the OSS name server.

Automatic Restart of Filesets by the Automatic Startup Service

A fileset can be configured so that the OSS Monitor automatically starts that fileset after a system load, regardless of whether the fileset was in the STARTED state. Restarted filesets are automatically repaired if necessary.

The automatic startup service can also restart the fileset a maximum number of times during a 10-minute period. See the ADD FILESET Command on page 12-7 or the ALTER FILESET Command on page 12-20 for more information about this service.

Automatic Restart of Filesets After OSS Name Server Failure

If an OSS name server fails, the OSS Monitor initiates a recovery procedure similar to that performed during OSS Monitor startup. All filesets that were left in the STARTED state and were managed by that OSS name server (as the OSS name server for either the fileset or the mount-point) are repaired and restarted. The OSS Monitor also restarts that OSS name server.

Automatic Restart of OSS Name Servers After Processor Failure

If failure of a processor causes failure of an OSS name server process running without a backup or causes termination of a fault-tolerant OSS name server process pair, the OSS Monitor initiates a recovery procedure similar to that performed upon OSS Monitor startup. No recovery is attempted if the processor failure only affects one process of the OSS name server process pair.

During the recovery, all filesets that were left in the STARTED state and were managed by the failed OSS name server (as the OSS name server for either the fileset or the mount-point) are repaired and restarted. The OSS Monitor also restarts that OSS name server.

Potential Problems During Automatic Restart of Filesets

The OSS Monitor might be unable to successfully restart all filesets that were left in the STARTED state when their OSS name servers failed. If a failure occurs, you can attempt to manually recover from that failure.

Note. When the OSS Monitor attempts automatic restart of filesets, it does not retry if certain failures occur.

Both the primary and backup OSS name server processors for a fileset can fail during the restart. If both the primary and backup OSS name server processors for a fileset fail, the OSS Monitor checks processor messages until one of the OSS name server processors is reloaded, then initiates the recovery sequence. You do not need to take any action.

If a restart operation fails for a fileset, that fileset, which was in the STARTED state, is changed to the UNKNOWN state by the OSS Monitor. You can use the SCF STATUS FILESET command to determine which filesets remain in the UNKNOWN state after the failure of an automatic fileset restart sequence.

The FSCK integrity checker also might fail during a restart. For example, if the FSCKCPU value for the OSS Monitor specifies a different processor than the processor that the OSS Monitor is running on, the specified processor could have

failed. The OSS Monitor reinitiates the fileset restart sequence when the specified FSCK processor is reloaded.

You might not want to wait for the FSCK processor to be reloaded. You can correct this situation manually by changing the FSCKCPU value (See ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands on page 12-34).

If the recovery sequence does not begin automatically, you can perform it manually by issuing commands with the following format at an SCF prompt:

```
DIAGNOSE FILESET filesetname, CPU nn, REPAIR SERIOUS
```

filesetname

is the name of a stopped fileset. For greatest efficiency, you should specify filesets in the correct order for mounting, beginning with the root fileset.

nn

is the processor number of the processor used by the OSS Monitor. You know that this processor is not reloading. By specifying a processor in the command, you override the processor number specified in the subsystem configuration.

This command runs the FSCK integrity checker. When FSCK finishes successfully, the fileset is placed in the STOPPED state. If FSCK fails, the fileset is put back in the UNKNOWN state.

If FSCK fails:

1. Check the EMS log for messages related to the DIAGNOSE FILESET command.
2. If necessary, purge the FSCK log file. (See FSCK Log File on page 5-25 to help locate the FSCK log file.)
3. Reissue the DIAGNOSE FILESET command.

If the DIAGNOSE FILESET command finishes successfully:

1. Verify that at least one of the OSS name server processors for that fileset is running.
2. Issue the following SCF command:

   ```
   START FILESET $ZPMON.ROOT
   ```

3. Restart every fileset that was in the STARTED state.

A quicker alternative to recovering from a problem during an automatic fileset restart sequence is:

1. Verify that at least one of the OSS name server processors is running.
2. Stop the OSS Monitor and restart it.

The OSS Monitor invokes the automatic fileset restart sequence again, and the restart should succeed this time.

Auditing a Fileset

An important component of a secure file system is the ability to trace the history of security-related operations on objects in the system. OSS security auditing allows you to collect a history of audited operations—that is, an audit trail—on a specified set of auditable objects in the system.

OSS security auditing allows you to audit access to objects in the OSS filename space. Audit commands for OSS objects and operations are provided by Safeguard, and SAFEART allows you to search for audit records of operations on OSS files.

Using the AUDITENABLED Attribute

The OSS fileset AUDITENABLED attribute determines whether audit records are generated on objects within the fileset. When another fileset is mounted on an audited fileset, whether the mounted fileset is audited depends on its own AUDITENABLED attribute.

The AUDITENABLED attribute is either ON or OFF (the default value). In addition, the Safeguard global audit setting AUDIT-CLIENT-SERVICE must be ON for fileset auditing to be in effect (for more information, see the *Safeguard Audit Service Manual*).

When the AUDITENABLED attribute is ON, audit records are created whenever an access-control decision is made on an object in the fileset. The AUDITENABLED attribute can be assigned a value during fileset creation and can be changed at any time through the OSS Monitor SCF command ALTER FILESET. However, the change takes effect only when the fileset is next started.

Auditing cannot be controlled directly at the OSS file or directory level. However, the AUDITENABLED attribute applies to all objects named within the fileset and generates an audit record at the fileset level. Therefore, if you want to audit a particular file, you must enable auditing of the fileset that contains that file.

Note. Guardian files (those under `/G`) and OSS filesets on other nodes (those accessed through `/E`) cannot be assigned the audit-enabled attribute by using OSS Monitor SCF commands.

Audited SCF Operations

The following SCF fileset operations are audited:

SCF Commands Used	Actions taken
START FILESET and STOP FILESET	When an audited fileset is started or stopped, the OSS Monitor generates a mount/unmount record. The mount point pathname is present only in the record generated by use of the START FILESET command.
ADD FILESET and DELETE FILESET	When a member of the super group (255, *nnn*) attempts to add or delete an audited fileset, an audit record is generated.

SCF Commands Used	Actions taken (continued)
ALTER FILESET	When a member of the super group attempts to alter the value of a fileset's AUDITENABLED attribute, an audit record is generated. The before and after values of the AUDITENABLED attribute are included.

Obtaining Information About a Fileset

The following subsections describe:

- Checking the Current Configuration of a Fileset on page 5-13
- Checking the Current State of a Fileset on page 5-13

Checking the Current Configuration of a Fileset

You can obtain information about the current configuration of a specific fileset by entering the SCF INFO FILESET command and specifying the name of that fileset. You can obtain configuration information about all filesets by entering the SCF INFO FILESET * command.

The SCF INFO FILESET command displays only information about the fileset as it currently is defined in the ZOSSFSET file; this information applies only to the next time that fileset is restarted. Information about past configurations is not retained.

To identify all disk volumes on which a given fileset has OSS files, search all disk volumes on your local node using the SCF STATUS FILESET command with the DETAIL parameter. To get the complete list of storage-pool disk volumes used by the fileset, the fileset must be started; otherwise, the command returns only the disk volumes currently listed in the storage-pool file (the creation pool).

Checking the Current State of a Fileset

To display the state of a fileset, including whether the fileset is currently started (mounted), you use the SCF STATUS FILESET command. See STATUS FILESET Command on page 12-66 for the command syntax and a description of the display information.

Stopping (Unmounting) a Fileset

You stop an existing, mounted fileset to make it unavailable to users. This action is also known as unmounting a fileset.

Note. A fileset cannot be stopped until every fileset mounted on it is stopped.

You stop a fileset under any of the following conditions:

- Before diagnosing the fileset.

- To make the files in the fileset unavailable to users. If a fileset is stopped while users have files open in that fileset, the open files can be used normally. However, once a file is closed, the file cannot be reopened until the fileset is mounted again.
- To free memory in the OSS name server for that fileset.
- To delete the fileset from the OSS file system.

If your site uses the STOPOSS utility, that utility stops all filesets. See STOPOSS Utility on page C-16 for more information.

Alternatively, you can use the SCF STOP FILESET command to manually stop a fileset:

1. Warn your users to make sure that all their files in the fileset are closed and all OSS shell sessions using that fileset are terminated. You can use a method similar to the one described under Manually Stopping the OSS File System and the OSS Environment on page 2-3.
2. Do one of the following:
   - If the fileset is the root fileset, stop all filesets by entering the following OSS Monitor SCF command:

     ```
     STOP FILESET $ZPMON.*
     ```

   - If the fileset is not the root fileset, use the SCF STOP FILESET command.

   For example, to stop (unmount) the fileset USER1 and send informational messages to the file CMDLOG, enter the following SCF command:

   ```
   STOP /OUT CMDLOG/ FILESET $ZPMON.USER1
   ```

If a user remains logged in with a current working directory in the stopped fileset, that user might need to use the `cd` command again to return to the directory after the fileset is restarted; otherwise, files in the restarted fileset might not be accessible to the user.

Reconfiguring a Fileset

You can reconfigure a fileset by:

- Changing the Operating Parameters of a Fileset on page 5-14
- Changing OSS File Caching for the Disks of a Fileset on page 5-18
- Changing the Physical Makeup of a Fileset on page 5-21

Changing the Operating Parameters of a Fileset

You change a fileset configuration when you enter new values for:

- Automatic restart (DESIREDSTATE attribute)
- Fileset recovery utility processor choice (FSCKCPU attribute)
- Fileset recovery utility log file destination (REPORT attribute)
- Fileset auditing (AUDITENABLED attribute)

- Fileset input/output fault-tolerance (FTIOMODE and NORMALIOMODE attributes)
- Fileset catalog buffering (the BUFFERED attribute)
- Fileset storage-pool storage-pool file (POOL attribute)
- Maximum number of files and directories (MAXINODES attribute)
- User access restriction (the READONLY attribute)
- Network File System (NFS) request timeout or cache (pool) size
- Catalog file volume (CATALOG attribute)
- OSS file system mount point (MNTPOINT attribute)
- OSS name server identity (NAMESERVER attribute)

To change any of these attributes, use the following procedure:

1. Change the appropriate value in the ZOSSFSET file by using the SCF ALTER FILESET command.

Changes made to the DESIREDSTATE, FSCKCPU, and REPORT attributes take effect immediately. Changes made to any other attribute require the following additional steps before they can take effect.

- To change any of the AUDITENABLED, BUFFERED, FTIOMODE, MAXINODES, NORMALIOMODE, POOL, READONLY, or NFS attributes:

  2. Apply the change to the fileset using the SCF CONTROL FILESET Command with the SYNC option.

  Changes to FTIOMODE or NORMALIOMODE only apply to files opened after the new attribute value takes effect; already opened files are not affected.

  Changes to any attribute made by the ALTER FILESET command in Step 1 take effect when the fileset restarts, regardless of whether CONTROL FILESET, SYNC is used.

- To change any of the CATALOG, MNTPOINT, or NAMESERVER attributes:

  2. If the fileset is started, stop it by using the SCF STOP FILESET command, as described under Stopping (Unmounting) a Fileset on page 5-13.

  3. Restart the fileset using the SCF START FILESET Command.

  All changes to these attributes made by the ALTER FILESET command in Step 1 take effect when the fileset restarts.

Changing Fileset Input/Output Fault Tolerance

The value used for the FTIOMODE and NORMALIOMODE options affect application performance and fault tolerance. Fault tolerance for files opened without using the O_SYNC bit is controlled by the NORMALIOMODE option setting; fault tolerance for files opened using the O_SYNC bit is controlled by the FTIOMODE option setting.

The FTIOMODE settings are a subset of the NORMALIOMODE settings. The setting used for the FTIOMODE attribute of a specific fileset must have a fault-tolerance level at least as high as that of the NORMALIOMODE attribute setting for the fileset. The

attribute settings are, from highest to lowest fault tolerance and from lowest to highest performance:

UNBUFFEREDCP
DP2BUFFEREDCP
OSSBUFFEREDCP
DP2BUFFERED
OSSBUFFERED

The behaviors associated with these levels of fault tolerance are shown in Table 5-1.

Table 5-1. Effects of File I/O Fault-Tolerance Attribute Settings (page 1 of 2)

Setting	Scenarios	Results
UNBUFFEREDCP	Single DP2 processor failure	Transparent recovery
	System failure (or double DP2 processor failure)	Application fails with possible loss of a single write request if failure occurs during the write request
	Application processor failure	Application fails with possible loss of a single write request if failure occurs during the write request
DP2BUFFEREDCP	Single DP2 processor failure	Transparent recovery
	System failure (or double DP2 processor failure)	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	Application processor failure	Application fails with possible loss of a single write request if failure occurs during the write request
OSSBUFFEREDCP	Single DP2 processor failure	Transparent recovery
	System failure (or double DP2 processor failure)	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	Application processor failure	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk

Table 5-1. Effects of File I/O Fault-Tolerance Attribute Settings (page 2 of 2)

Setting	Scenarios	Results
DP2BUFFERED (the same behavior as legacy OSS file caching turned off [OSSCACHING OFF])	Single DP2 processor failure	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	System failure (or double DP2 processor failure)	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	Application processor failure	Application fails with possible loss of a single write request if failure occurs during the write request
OSSBUFFERED (the same behavior as legacy OSS file caching turned on [OSSCACHING ON])	Single DP2 processor failure	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	System failure (or double DP2 processor failure)	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk
	Application processor failure	Application fails with possible loss of multiple buffered write requests if failure occurs before buffered data is written to disk

Changing Fileset Catalog Buffering

The value used for the BUFFERED option can have a dramatic effect on application performance when there are a lot of calls to OSS file system functions that cause fileset catalog updates; such calls include: `creat()`, `unlink()`, `rename()`, `chmod()`, `chown()`, `mkdir()`, and so on. The possible values of the BUFFERED option are, in order of increasing buffer usage:

Value	When to use it
`NONE`	An OSS name server writes a record in the PXLOG file as part of each fileset catalog operation. When both processes of a OSS name server process pair involved in the operation fail, recovering from a partially completed operation is fast because it is based on the PXLOG file entry. This option is provides the slowest performance for fileset catalog updates but provides the fastest recovery from complete OSS name server failure.

Value	When to use it (continued)
LOG	An OSS name server does not write PXLOG records; it checkpoints these records to its backup process. The backup process of an OSS name server keeps these records in its memory and uses them to recover partially completed operations in the event of a failure of an OSS name server primary process. If total failure of an OSS name server occurs, FSCK is automatically run against the fileset during the fileset remounting operation. This option provides better performance for fileset catalog updatesat the expense of slower recovery in the event of complete OSS name server failure. You can control the relative likelihood of needing fileset repair, and therefore the relative speed of recovery, by adjusting the MAXDIRTYINODETIME attribute for the fileset.
CREATE	The OSS name server does not write PXLOG records but rather checkpoints these records to its backup process. The backup process keeps these records in its memory and uses them to recover partially completed operations in the event of a failure of the OSS name server primary process. If total failure of the OSS name server occurs, FSCK is automatically run against the fileset during the fileset remounting operation. The disk process does not write new file labels immediately but rather defers these label writes until it has nothing else to do. This option provides the best performance for fileset catalog updatesbut limits the fileset to one disk volume and has the potential for lost files in the event of a double-disk process failure. See Creating a Unique Fileset on page 5-1 for additional considerations when you use BUFFERED CREATE.

Changing the OSS File System Mount Point

Changing the mount point can affect the behavior of programs that use OSS files, particularly programs that use the symbolic-link feature. You should notify users well in advance of changing an existing mount point.

Changing OSS File Caching for the Disks of a Fileset

Note. HP recommends that you not modify OSS file caching at the disk level. The FTIOMODE and NORMALIOMODE fileset attributes now provide better control over fault-tolerance and performance for file input or output. See Table 5-1 on page 5-16 to map OSSCACHING ON and OFF settings to the use of the new attributes.

You should decide whether to use OSS file caching when you first configure a fileset, because changing the use of OSS file caching requires you to stop the fileset. For more information about OSS file caching, see OSS File-System Components on page 1-10 and OSS File Caching Overview on page 5-20.

If you want to enable OSS file caching initially for all disk volumes that contain OSS files, you need do nothing. OSS file caching is enabled by default when disk volumes are configured.

If you disable OSS file caching, HP strongly recommends that you disable OSS file caching on all the disk volumes in a fileset. To disable OSS file caching for a specific disk volume, you must disable all filesets mounted on the affected fileset:

1. At a TACL prompt, enter:

   ```
   SCF
   ```

2. At the SCF prompt, enter the following SCF command to stop each fileset involved:

   ```
   STOP FILESET $ZPMON.filesetname
   ```

   Enter this command beginning with the last fileset mounted on the affected fileset. Stop the filesets in the reverse order in which they were last started. Stop the affected fileset last. If the root fileset is the affected fileset, you can enter:

   ```
   STOP FILESET $ZPMON.*
   ```

   This command will stop all filesets in the correct order.

3. At an SCF prompt, enter the following set of commands once for each disk volume in the fileset:

   ```
   ALTER DISK diskname, OSSCACHING OFFdiskname
   ```

4. Restart the affected portion of the OSS file system by entering the following SCF command one or more times:

   ```
   START FILESET $ZPMON.filesetname
   ```

   filesetname

   is the name of each fileset that you previously stopped, specified in the order in which mount points occur.

Similarly, to enable OSS file caching for a specific disk volume:

1. At a TACL prompt, enter:

   ```
   SCF
   ```

2. At the SCF prompt, enter the following SCF command to stop each fileset involved:

   ```
   STOP FILESET $ZPMON.filesetname
   ```

   Enter this command beginning with the last fileset mounted on the affected fileset. Stop the filesets in the reverse order in which they were last started. Stop the affected fileset last. If the root fileset is the affected fileset, you can enter:

   ```
   STOP FILESET $ZPMON.*
   ```

   This command will stop all filesets in the correct order. This command begins with the last fileset mounted and stops the filesets in the reverse order in which they were last started.

3. At an SCF prompt, enter the following set of commands once for each disk volume in the fileset:

   ```
   STOP DISK diskname
   ALTER DISK diskname, OSSCACHING ON
   START DISK diskname
   ```

   diskname

   is the name of a disk volume that contains OSS files.

4. Restart the affected portion of the OSS file system by entering the following SCF command one or more times:

   ```
   START FILESET $ZPMON.filesetname
   ```

   filesetname

   is the name of each fileset that you previously stopped, specified in the order in which mount points occur.

If you want to add disks to a storage pool for a fileset that has OSS file caching disabled:

1. Use the Subsystem Control Facility (SCF) storage subsystem to add the disks to the system.
2. Modify the storage-pool file for the fileset.
3. Do one of the following:
   a. Stop and start the fileset as described under Starting (Mounting) or Restarting Filesets on page 5-7
   b. Apply the change to the started fileset using the SCF CONTROL FILESET Command with the SYNC option.

OSS File Caching Overview

By default, the OSS environment provides a file cache for regular files in each processor that does input or output with a disk volume that contains OSS files. HP strongly recommends that you leave OSS file caching enabled. This cache is necessary for the fault tolerant behavior controlled by the fileset FTIOMODE or NORMALIOMODE attributes.

Enabling or disabling this feature does not affect access from the Guardian environment to Guardian files (including SQL files) on a volume that contains OSS regular files.

If you disable OSS file caching on a disk volume that is in a fileset, you must disable OSS file caching on all disk volumes that you want to use for that OSS fileset. You cannot predict which disk volume in a fileset will be used for a given file; if you have OSS file caching enabled on one disk volume in a given fileset but disabled for another disk volume in that fileset, you cannot predict whether a particular file might be cached.

Disabling OSS file caching changes the fault tolerant behavior of the fileset. Turning it off converts OSSBUFFERED behavior to DP2BUFFERED behavior or OSSBUFFEREDCP behavior to DP2BUFFEREDCP behavior.

The caching status of a file can change as opens, closes, and other events occur on the file. The data integrity of a file and the access (data transfer) speed for the file are affected by the following:

- Whether OSS file caching is enabled
- Where data is buffered and when it is checkpointed, as controlled by the settings for the fileset FTIOMODE fault-tolerance attribute and the NORMALIOMODE attribute
- Which program access options are used, such as how the file is opened

When OSS file caching is enabled, behavior comparable to that experienced on a node running an RVU prior to G06.27 occurs when the NORMALIOMODE fileset attribute is OSSBUFFERED and the FTIOMODE fileset attribute is UNBUFFEREDCP.

Note. HP recommends that the `S_NONSTOP` extension not be used in new applications. The `S_NONSTOP` extension is ignored on nodes running an RVU more recent than G06.26 or H06.03; the standard `O_SYNC` feature used with the FTIOMODE attribute provides better data integrity and improved performance.

Changing the Physical Makeup of a Fileset

You can change the physical makeup of a fileset by:

- Changing the Fileset Catalog on page 5-21
- Adding a Disk Volume to a Fileset on page 5-22
- Removing a Disk Volume From a Storage-Pool File on page 5-22
- Removing a Disk Volume From a Fileset on page 5-23
- Moving a Disk Volume to Another Fileset on page 5-24
- Controlling the Maximum Number of Files on page 6-30

Remember that a fileset can contain files on disk volumes that are not currently in the fileset's storage-pool file. When you perform operations that depend upon or potentially change the device label for a fileset, you need to manage the files as well as the fileset.

Changing the Fileset Catalog

To assign a new fileset catalog to a fileset:

1. Stop the fileset with the SCF STOP FILESET command, as described in Stopping (Unmounting) a Fileset on page 5-13.
2. Use the SCF INFO FILESET Command with the DETAIL option to determine whether the fileset uses the BUFFERED CREATE option.

   If the fileset uses the BUFFERED CREATE option but not the READONLY TRUE option, the creation pool for the fileset will be the new catalog disk volume of the

fileset; existing OSS files will not be moved to the new volume, but new OSS files will be created on the new disk volume. Make sure that the new catalog disk volume has adequate space for the fileset.

3. Use the SCF ALTER FILESET Command to enter a new name for the catalog disk volume of the fileset.

   This change causes current catalog files to be moved to the specified new disk volume; previously saved catalog files (with file identifiers of the form PXIN*nnnn*, PXLI*nnnn*, and PXLO*nnnn*) are not moved. The subvolume name of the catalog files is not changed.

4. Use the CVT command described in Managing and Repairing Fileset Catalog Files on page 5-40 to purge the saved catalog files (with file identifiers of the form PXIN*nnnn* PXLI*nnnn*, and PXLO*nnnn*) from the old catalog disk volume.

5. Use the SCF START FILESET Command to restart the fileset.

To modify the limits in an existing fileset catalog, see Controlling the Maximum Number of Files on page 6-30.

Adding a Disk Volume to a Fileset

To add a disk volume to a fileset (perhaps to make more storage space available):

1. Use a Guardian text editor to insert the disk volume name in the storage-pool file.

   To avoid problems in managing fileset space, do not put the same volume in more than one storage pool.

2. Do one of the following:

   - Use the SCF CONTROL FILESET Command with the SYNC option to make the change take effect immediately.
   - Stop the fileset with the SCF STOP FILESET command, as described under Stopping (Unmounting) a Fileset on page 5-13, then restart the fileset using the SCF START FILESET Command.

     The change takes effect when the fileset is restarted.

Removing a Disk Volume From a Storage-Pool File

You remove a disk volume from a storage-pool file when the disk volume becomes too full to safely accommodate new files. Removing the disk volume from the storage-pool file leaves it in the storage pool for the fileset and leaves its existing files available for use.

To remove a disk volume from a storage-pool file:

1. Stop the fileset with the SCF STOP FILESET command, as described in Stopping (Unmounting) a Fileset on page 5-13.

2. Use the SCF INFO FILESET Command to determine the name of the storage-pool file for the fileset.
3. Use a Guardian text editor to convert the entry for the disk volume to a comment within the storage-pool file.

   Note. You should never delete the entry for a volume name from a storage-pool file once the disk volume contains OSS files; converting such entries to comments provides an easy way to document the volume list for the fileset.

4. Restart the fileset with the SCF START FILESET Command.

New files created within the fileset are not added to the disk volume.

Removing a Disk Volume From a Fileset

It is sometimes necessary to remove a disk volume that is already in use as part of a fileset. Removing a disk volume from a fileset means removing it from both its storage-pool file and its storage pool.

To remove a disk volume from a fileset:

1. Use the SCF INFO FILESET command to determine the mount point for the fileset that uses the disk volume you want to remove.
2. Use the OSS shell `cd` command to reach the mount point. For example:

   ```
   cd /data1
   ```

3. Create a `pax` archive of the entire fileset, as follows:

   ```
   pax -wvf ../oss_files.pax ./
   ```

4. Stop the fileset with the SCF STOP FILESET command, as described under Stopping (Unmounting) a Fileset on page 5-13.
5. Use a Guardian text editor to delete the entry for the disk volume from the storage-pool file for the fileset.
6. Restart the fileset with the SCF START FILESET command.
7. Delete all the files beneath the mount point with the OSS shell `rm -r` command. This step:
   - Prevents the OSS file system from using the old catalog entries when one of these files is accessed.
   - Empties the corresponding ZYQ subvolume so that subsequent use of the SCF DIAGNOSE FILESET command does not inappropriately restore the files to the `lost+found` directory of the fileset.
8. Restore the previously archived files to the mount point using the OSS shell `pax` command. For example:

   ```
   pax -rvf oss_files.pax
   ```

9. If the catalog disk volume for the fileset is on the removed disk volume, use the SCF ALTER FILESET command to enter a new name for the catalog disk volume of the fileset.

   This change causes current catalog files to be moved to the specified new disk volume; previously saved catalog files (with file identifiers of the form PXIN*nnnn*, PXLI*nnnn*, and PXLO*nnnn*) are not moved. The subvolume name of the catalog files is not changed.

10. Use the Guardian CVT utility described under Managing and Repairing Fileset Catalog Files on page 5-40 to purge the saved catalog files (with file identifiers of the form PXIN*nnnn*, PXLI*nnnn*, and PXLO*nnnn*) from the old catalog disk volume.

You can now physically remove the disk volume.

Moving a Disk Volume to Another Fileset

To move a disk volume from one fileset to another:

1. Remove the disk volume from the storage-pool file for its current fileset, using the procedure described under Removing a Disk Volume From a Storage-Pool File on page 5-22.
2. Remove the disk volume from its current fileset, using the procedure described in Removing a Disk Volume From a Fileset on page 5-23.
3. Add the disk volume to the storage-pool file of the appropriate fileset, using the procedure described under Changing the Fileset Catalog on page 5-21.

Checking and Repairing Fileset Integrity

This subsection describes use of the SCF DIAGNOSE FILESET command, which runs the Guardian FSCK utility. The FSCK utility checks and optionally repairs the integrity of an OSS fileset. FSCK issues EMS events at the start of each diagnostic operation, every 5 minutes during the operation, and at the end of the operation; see FSCK Messages on page A-6.

The DIAGNOSE FILESET command corresponds to the `fsck` command of many UNIX systems. Only super-group users (255,*nnn*) can use the DIAGNOSE FILESET command.

When Do You Need to Check Fileset Integrity?

You diagnose a fileset to make sure that you can read and write files in it.

The Guardian FSCK utility looks at every regular file to verify that it has both a directory entry and data. FSCK makes sure that the superblock, inodes, and catalog files are consistent. You can choose to repair all inconsistencies or only the serious ones.

Open System Services was designed to allow customers to configure systems where FSCK should never need to run, even in most multiple-failure situations. Traditional UNIX systems are different: those systems have relied on running `fsck` whenever there is a system failure.

FSCK is provided as part of Open System Services primarily to be used in the event of catastrophic failures such as the following:

- The complete loss of a disk volume containing OSS files or an OSS catalog (loss of both primary and mirror drives)
- The failure of the built-in recovery mechanisms of an OSS name server

In most of these cases, FSCK is run automatically by the OSS Monitor. Cases where you must run FSCK manually are signaled by Event Management Service (EMS) OSS subsystem message 3. For more information about EMS operator messages, see the *Operator Messages Manual*.

Filesets that are configured with the BUFFERED CREATE (fast-create) option require more frequent use of FSCK. For such filesets, you must run FSCK when a double failure results in the loss of either the OSS name server process (both primary and backup) or a disk volume containing a fast-create fileset.

Note that even with the fast-create option enabled, FSCK is needed only in cases of multiple failures and is run automatically by the OSS Monitor in most of these cases.

Note. Before you use the DIAGNOSE FILESET command, make sure that the fileset you are about to check is stopped (unmounted). The OSS Monitor displays an error message if you attempt to diagnose a fileset that is not stopped. See Appendix A, Messages, for information about OSS Monitor messages.

The DIAGNOSE FILESET command can take a long time to finish. You should therefore execute the command during a time when having a fileset unmounted for a long period does not disrupt normal user activity.

While a fileset is being diagnosed, that fileset is put into the DIAGNOSING state. When the diagnosis operation is finished, the fileset reverts to the STOPPED state and you can mount it with the SCF START FILESET command.

If the FSCK utility fails, the fileset is put into the UNKNOWN state instead of the STOPPED state.

If the DIAGNOSE FILESET command is issued with the OPTION STOP option and the fileset being diagnosed has a problem that has not been corrected, a subsequent mount of the fileset might fail.

FSCK Log File

The FSCK utility writes its output to a Guardian log file. Figure 5-4 on page 5-27 shows two examples of FSCK log files.

The FSCK log file is created with the access permissions for the user ID of the OSS Monitor process when that process was started.

You can specify a Guardian filename for the log file using the REPORT option of the ADD FILESET Command, ALTER FILESET Command, or DIAGNOSE FILESET Command. If you do not specify a filename for the log file, the log file is sent to the spooler location specified for the REPORT attribute of the SUBSYS object. If no value is specified in the ZOSSPARM file or if the specified spooler locations are unavailable, the log file is put in the same volume and subvolume as the OSS Monitor code file, OSSMON (normally a $SYS*nn* subvolume).

The Guardian file identifier of the default log file consists of the characters ZX0 followed by the rightmost portion of the device identifier of the fileset. (The device identifier of a fileset is a unique, sequentially assigned set of letters and digits used in system internals.) For example:

- For the root fileset named ROOT (with device identifier 000000), the file ID of the default log file is ZX000000.
- For the temporary fileset named TEMP (with device identifier 00007Z), the file ID of the default log file is ZX00007Z.

The device identifier is also a field of the INFO FILESET Command display.

A default log file has file code 180 in the Guardian file system and is suitable for use as an OSS text file with an OSS text editor such as `vi`. Therefore you can read a default log file with an OSS text editor.

You can also read a log file with a Guardian text editor after converting the file to an EDIT file (Guardian file code 101) with the Guardian CTOEDIT utility. However, the EDIT form of a log file can contain a maximum of 99,999 lines of entries using a line number increment of 1. To avoid exceeding the limit on EDIT files, you should either periodically delete the log file after it has been analyzed or use a spooler location for the log file. When a file-code-180 log file exceeds the maximum size of an EDIT file, the log file should not be converted but should be read using only an OSS shell utility such as `vi`.

Figure 5-4. FSCK Log File Examples (page 1 of 2)

```
FSCK - T8621G09 - (15DEC2001) OSS Fileset Validate/Repair Utility
Copyright Compaq Computer Corporation 1994, 1995, 1996, 2001

August 31,2001 01:14:53

Options Selected are: CATALOG $OSS.ZX000000, REPAIR ALL, OPEN

FSCK Serial Number = 0005

Catalog Subvolume is \NODE1.$OSS.ZX000044

PHASE I -- Verify Super Block

  Catalog Version 2

  The Name Server has detected no inconsistencies in this fileset

  Volume List:

    $OSS2
    $OSS3

PHASE II  -- Build Directory Tree

PHASE III -- Check Inodes

  Scanning PXINODE File...
  Checking Parent Lists...
  Checking for Directory Loops...

PHASE IV  -- Verify Free Lists

  *** SERIOUS 201 *** Broken Free List, Inode=5465

PHASE V    -- Create New PXLINK File

PHASE VI   -- Look For Orphan ZYQ Files

PHASE VII   -- Create New PXINODE File

PHASE VIII -- Rename Catalog Files

FSCK Completed

  Minor Inconsistencies:   0
  Serious Inconsistencies: 1
  Warnings:                0
  Errors:                  0

August 31,2001 01:14:54
```

Figure 5-4. FSCK Log File Examples (page 2 of 2)

```
FSCK - T8621G09 - (15DEC2001) OSS Fileset Validate/Repair Utility
Copyright Compaq Computer Corporation 1994, 1995, 1996, 2001

August 31,2001 01:14:53

Options Selected are: CATALOG $OSS.ZX000000, STATUS, ROOT

Catalog Subvolume is \NODE1.$OSS.ZX000000

PHASE I -- Verify Super Block

  Catalog Version 2

  The fileset is marked as Mounted with BUFFERED LOG by a down-rev Name Server

  The Name Server has detected no inconsistencies in this fileset

  Volume List:

    $OSS1

FSCK Completed

  Minor Inconsistencies:   0
  Serious Inconsistencies: 0
  Warnings:                0
  Errors:                  0

August 31,2001 01:14:53
```

The volume list shown in the Phase I output (see Figure 5-4 on page 5-27) is a list of all disk volumes associated with the fileset. This list is automatically added to the superblock or updated when FSCK is run. The volume list is used by the OSS name server for that fileset to refresh information about file opens within that fileset.

When you use the DIAGNOSE FILESET command with the STATUS option, FSCK writes a status report to its log file that describes:

- The current state of the fileset as recorded in the mounted field in the superblock for the fileset
- Whether the OSS name server used for that fileset is a current version of the software

If the DETAIL option is also used, the mounted field contents are displayed in hexadecimal. Possible values are:

Value	Mounted state
0x0000	The fileset is unmounted.
0x0001	The fileset is being recovered.
0x0002	The fileset is mounted in BUFFERED LOG mode by an OSS name server that is not running the current code version.
0x0006	The fileset is mounted in BUFFERED CREATE mode by an OSS name server that is not running the current code version.
0x2004	The fileset is mounted in BUFFERED CREATE mode.
0x2010	The fileset is mounted in READONLY TRUE mode.

Value	Mounted state (continued)
0x6002	The fileset is mounted in BUFFERED LOG mode and is marked clean.
0x8001	The fileset is being recovered.
0x8002	The fileset is mounted in BUFFERED LOG mode.
0xA000	The fileset is mounted in BUFFERED NONE mode.
0xFFFF	The fileset is mounted in BUFFERED NONE mode by an OSS name server that is not running the current code version.

When the mounted field does not correspond to a known state, FSCK reports its hexadecimal value as follows:

```
The fileset is in Mount State = 0xvalue
```

For information about messages such as the following, see Appendix A, Messages:

```
*** SERIOUS 201 *** Broken Free List, Inode=5465
```

Inconsistencies Checked by FSCK

The FSCK utility checks for the inconsistencies listed in Table 5-2. The inconsistencies are listed in the order in which FSCK checks them. The corresponding FSCK log file messages are described in Appendix A, Messages.

Table 5-2. Inconsistencies Checked by FSCK (page 1 of 4)

Inconsistency	Type	Explanation
Missing or corrupt superblock	Serious	If the superblock (node #1) is missing from the PXINODE file, the fileset is unusable. A new superblock must be constructed. FSCK creates the new superblock; you do not need to do anything.
Nonzero mounted flag	Minor	Each time a fileset is mounted for read/write access, its mounted flag is set to a nonzero value. When the fileset is successfully unmounted, the mounted flag is set to zero. A nonzero mounted flag means that the fileset has not been cleanly unmounted because one of the following is true: ● The OSS name server on which the fileset was mounted failed. ● The OSS name server detected serious inconsistencies within the catalog. Although a nonzero mounted flag is a minor inconsistency, it is corrected regardless of whether FSCK has been directed to correct minor inconsistencies. If only minor inconsistencies are discovered and only major inconsistencies are being corrected, FSCK does not create a

Table 5-2. Inconsistencies Checked by FSCK (page 2 of 4)

Inconsistency	Type	Explanation
Bad free-inode list	Serious	An OSS name server maintains two free-inode lists in each catalog. One is used to list the inodes that can be immediately reused. The other is used to list those inodes that can be reused only after a successful unmount and mount sequence. The FSCK utility detects and corrects the following inconsistencies: ● An inode other than a free-inode block appears on a free-inode list. ● A loop occurs in a free-inode list. ● An in-use inode also appears in a free-inode block. ● The same free-inode block appears on both free-inode lists. If any free-inode list inconsistencies are found, the free-inode lists are rebuilt. If FSCK is run using the SCF DIAGNOSE FILESET command, FSCK places all free inodes in the list of inodes that can be reused immediately. If FSCK is run automatically by the OSS Monitor as a result of a failure of an OSS name server, FSCK places all free inodes in the list of inodes that cannot be reused until after an unmount and remount sequence.
Missing inode	Serious	There are references to an inode but the inode itself does not exist. FSCK corrects this inconsistency as follows: ● If the missing inode is `/`, `/E`, `/G`, `/lost+found`, or `/dev`, it is added as a directory. ● If the missing inode is `/dev/tty`, that special file is added. ● If the missing inode is `/dev/null`, that special file is added. ● If the missing inode appears as a parent in one or more PXLINK records, it is added as a directory in the new PXINODE file. ● If there is a nonempty ZYQ file corresponding to the missing inode and if the disk process link count for that file is nonzero, the missing inode is added as a regular file that refers to the ZYQ file. ● If none of the above apply, any references to the missing inode are omitted from the new catalog.
Too many links	Serious	An inode has more links than are allowed (1 for directories, 20 for other inodes). Those links in excess of the maximum are omitted from the new PXLINK file.

Table 5-2. Inconsistencies Checked by FSCK (page 3 of 4)

Inconsistency	Type	Explanation
Loop in directory graph	Serious	If FSCK detects a loop in the directory graph, it breaks the loop by deleting a PXLINK record in the loop. Such a loop occurs when an inode is linked through other inodes back to itself; this means that the directory referred to by the inode number is its own parent.
Lost inode number	Minor	There is no PXINODE record with this number as the key, but the inode number does not appear in any free-inode block. FSCK includes the inode in the appropriate free-inode list in the new PXINODE file. Although this is a minor inconsistency, it is also corrected as part of resetting the mounted flag. For more information, see Nonzero mounted flag on page 5-29.
Orphan inode	Serious	An inode has no links. The inode is given the name `#`*`inode_number`* and is placed in the fileset's `/lost+found` directory.
Orphan ZYQ file	Minor	There is a ZYQ file for which there is no inode. If the ZYQ file is empty, FSCK issues a warning message and purges the file. If the file is nonempty, FSCK obtains the current number of links for the file from the disk process.
Orphan ZYQ file (*continued*)		If the number of links is zero and the file is not currently open, FSCK purges the file. If the link count is nonzero, FSCK does the following: 1. An inode is allocated for the file. 2. The disk process copy of the link count is set to one. 3. The file is placed in `/lost+found` under a synthesized OSS filename. FSCK attempts to use the inode whose number is encoded in the name of the ZYQ file. If that inode is unavailable, FSCK allocates an unused inode and renames the ZYQ file so that its name reflects this new inode number. If the ZYQ file cannot be renamed, a warning is issued and the new inode is not added.
Corrupt record	Serious	Records containing undefined record types or with record lengths inappropriate for the type of record are deleted. This applies to records in both the PXLINK and PXINODE files.
Bad parent list	Serious	Each inode contains a list of its parent inode numbers. A parent list is missing inode numbers or contains incorrect inode numbers. Any errors in the list are corrected so that the parent list accurately reflects the actual links to the inode.

Table 5-2. Inconsistencies Checked by FSCK (page 4 of 4)

Inconsistency	Type	Explanation
Missing ZYQ file	Minor	The ZYQ file corresponding to a regular inode does not exist. Because this inconsistency is usually the result of a failure during file creation or during the removal of the last link to a file, the inode and any links to it are deleted.
Wrong fileset type	Serious	The root fileset on a system is unique in that it contains special files not required in other filesets (such as `/E` and `/G`). Normally the root fileset is assigned fileset identifier (device number) 0 and its catalog resides in a subvolume named ZX000000. Recorded in the first record (superblock) of the PXINODE catalog is the fileset type (root or nonroot). If the catalog subvolume is ZX000000, FSCK assumes that it is operating with a root fileset. When dealing with a root fileset, FSCK makes additional checks regarding the existence and integrity of special files (`/E`, `/G`, `/dev`, `/dev/tty`, and `/dev/null`). If the subvolume name or the keyword indicates a root fileset but the catalog indicates otherwise, FSCK reports numerous inconsistencies and converts the catalog into a root catalog. If the subvolume name is other than ZX000000 but the superblock indicates that it corresponds to a root fileset, FSCK issues a warning message and does not perform those checks or repairs that are unique to root filesets.
Invalid inode number	Serious or minor	A record in the PXINODE file has an inode number that is one of the following cases: ● Less than or equal to 0 (zero) ● Greater than or equal to 2^{31} ● In the reserved inode range 8 through 31 If the inode number is <=0 or >= 2^{31}, the inode is discarded along with any links to the inode. These cases are serious inconsistencies. If the inode number is in the reserved inode range, the inode is relocated to a currently unused inode number and the PXLINK records for the inode are adjusted accordingly. If the inode represents a regular file, the ZYQ file is renamed to reflect the new inode number. This case is a minor inconsistency. If FSCK fails after renaming a ZYQ file but before the new catalog is completely created, then the next time FSCK is run against the old catalog, the ZYQ file is restored to its original name and the PXINODE file is updated to reflect the new CRVSN of the file.

Generated Catalog Files

The FSCK utility saves a copy of the existing catalog and creates a new one. Each time that FSCK is run to repair a fileset catalog, a unique four-digit FSCK serial number (FSN) is assigned for that run.

The FSN is used to encode the names of the saved files as follows:

PXINODE: PXIN*nnnn*

PXLINK: PXLI*nnnn*

PXLOG: PXLO*nnnn*

where *nnnn* is the FSN.

During the final phase of FSCK, the catalog files are renamed. During this phase, FSCK might fail or be otherwise stopped when there are no files in the catalog subvolume with the names PXINODE, PXLINK, or PXLOG. To allow for the orderly restart of FSCK in this case, FSCK creates a file named PXCKSTAT in the catalog subvolume. The PXCKSTAT file records the current FSN so that FSCK can locate the most recent catalog files.

After the catalog has been completely repaired (rebuilt), the PXCKSTAT file is purged from the catalog subvolume. The renamed catalog files remain in the subvolume until you purge or rename them with the Guardian Catalog Volume Tool (CVT).

For information about the CVT utility, see Managing and Repairing Fileset Catalog Files on page 5-40.

What Happens When Diagnosis Appears to Fail?

Errors occurring on ZYQ files cause FSCK to issue a warning and continue. In these cases, the new catalog might still contain minor inconsistencies.

Any other errors occurring during writes to the new catalog files cause FSCK to issue an error message and terminate.

When FSCK restarts a failed run that has renamed ZYQ files, it restores these files and attempts to update the PXINODE file accordingly. If an I/O error occurs during this update, FSCK issues a warning and attempts to continue.

The FSCK run might fail if there is a file input or output error. If you place fileset catalogs on mirrored volumes, you eliminate most I/O errors on catalog files.

If FSCK encounters:

- Any read error on the old catalog other than Guardian File Management Error 1 (reaching an end of file), it issues an error message and terminates abnormally.
- A Guardian File Management Error 1 (reaching an end of file) on the new PXLINK file, FSCK issues a warning message and continues.

Any other errors occurring during writes to the new catalog files cause FSCK to issue an error message and terminate.

- Any error other than Guardian File Management Error 1 (reaching an end of file) on FSCK's inode swap file, FSCK issues an error message and terminates.
- An inconsistency in its internal data structures, it issues an error message and terminates.

If the FSCK run fails, the fileset that FSCK is checking enters the UNKNOWN state rather than the STOPPED state. If this happens, check the FSCK log file (described under FSCK Log File on page 5-25). If FSCK failed before writing anything to this file, the file is probably full. Either rename the file or copy it to another volume, then purge the original file.

Deleting a Fileset

If your site uses the OSSREMOV utility, that utility deletes all filesets. See OSSREMOV Utility on page C-17 for more information.

To delete a single fileset:

1. Determine whether the fileset is mounted by using the SCF STATUS FILESET Command.
2. If the fileset is mounted, stop the fileset by using the procedure described under Stopping (Unmounting) a Fileset on page 5-13.
3. Determine the device identifier of the fileset with the SCF INFO FILESET Command.
4. Determine whether you have a current backup of the files in the fileset. If not, create a backup of those files using a procedure described under Backing Up and Restoring OSS Files on page 6-11.
5. Delete the fileset with the SCF DELETE FILESET Command.

All files in the fileset and the fileset catalog are deleted by this procedure. Any OSS files on a disk volume that was once part of this fileset are also deleted, even though the disk volume no longer appears in the storage-pool file for the fileset and is not part of the creation pool.

Renaming a Fileset

A super-group user (255, *nnn*) can rename a running fileset, and the new name takes effect immediately. To rename a fileset, use the SCF RENAME FILESET Command. The new fileset name must begin with a letter and must consist of letters and numeric characters.

OSS fileset names can be 1 to 32 characters long, and they are case-sensitive. You are not allowed to use the name of a fileset that already exists.

Updating Existing Fileset Configurations

A fileset created while your site used a previous version of Open System Services can be used unmodified with the current software. However, the contents of the ZPCONFIG and ZPMNTTAB configuration files from release version updates (RVUs) preceding G05.00 no longer affect the current configuration of your filesets.

If you had a G-series RVU preceding G05.00 installed, then the first time you start the OSS file system after installing a G05.00 or later G-series RVU, new configuration files are created from your existing ZPCONFIG and ZPMNTTAB files, as shown in Table 5-3 on page 5-35. The state of all your existing filesets are included in the new configuration files, and the old configuration files are renamed to ZXCONFIG and ZXMNTTAB, respectively, so that you have them if you need to fall back to an earlier version of Open System Services.

If you had a G05.00 or later G-series RVU installed, then the first time you start the OSS file system after installing a more current RVU, upgraded configuration files are created from your existing configuration files, as shown in Table 5-3 on page 5-35. The old configuration database files are not destroyed, in case you need to fall back to an earlier version of Open System Services.

Table 5-3. Configuration File Upgrades

Old Configuration File (D40 Version of OSS Monitor)	Old Configuration File (D46 Version of OSS Monitor)	New Configuration File (G09 or Newer Version of OSS Monitor)
ZPCONFIG and ZPMNTTAB	ZPOSFSET	ZOSSFSET
	ZPOSFS00	ZOSSFS00
	ZPOSFS01	ZOSSFS01
	ZPOSSERV	ZOSSSERV
	ZPOSPARM	ZOSSPARM

Removing Older Configuration Files

If you have upgraded from an earlier RVU, you might be able to save some disk space by deleting old database files. Whether you should delete files depends on whether you are:

- Upgrading From a G05.00 or Subsequent G-Series RVU on page 5-35
- Upgrading From an RVU Preceding G05.00 on page 5-36

Upgrading From a G05.00 or Subsequent G-Series RVU

Do not delete the older OSS Monitor database files (ZPOSFSET, ZPOSPARM, ZPOSSERV, ZPOSFS00, and ZPOSFS01). If you need to fall back to the RVU you upgraded from, you need those database files.

Upgrading From an RVU Preceding G05.00

Older configuration files can be deleted. Once the G05.00 RVU is installed and the OSS file system is started for the first time, the content of the ZXMNTTAB file is no longer accurate. That file should be deleted as soon as you are sure that you need not fall back to a previous RVU.

The ZXCONFIG file can also be deleted. However, you might want to maintain the contents of ZXCONFIG in parallel with the ZOSSFSET file, because if you need to fall back to the RVU you upgraded from, an accurate ZXCONFIG file is essential. You would need to rename ZXCONFIG to ZPCONFIG before you could bring up the previous RVU.

Moving a Directory Hierarchy to Its Own Fileset

Before moving a directory hierarchy, you should back up the entire fileset using the `pax` utility. For information on how to back up OSS files to the Guardian file system, see Creating a pax Backup of OSS Files in the Guardian File System on page 6-21. HP recommends that you make multiple copies of the backup.

The following example shows how to move the `/home` directory from the ROOT fileset to a newly-defined HOME fileset. To move the `/home` directory, you need to be logged in as the super ID.

1. Using the OSS shell command `mv`, change the name of the `/home` directory to a file or directory name that does not already exist in the ROOT directory, for example, `/homex`. This action removes the `/home` directory from the ROOT fileset's namespace so another fileset named `/home` can be created:

   ```
   mv /home /homex
   ```

2. Create a new directory named `/home`, using the shell command `mkdir`:

   ```
   mkdir /home
   ```

3. Record the existing security permissions for the `/homex` directory.

4. Change the security permissions for `/home` to 777 (all read, write, and execute). When the new fileset is mounted on this directory, you can reset the permissions to those used for `/homex`.

   ```
   chmod 777 /home
   ```

5. Create a fileset named HOME, using a unique OSS name server, that has a DEVICELABEL of 000001. The use of multiple OSS name server processes can improve overall performance if the default name server, $ZPNS, is very busy.

   a. Start SCF.

b. Add a new OSS name server for the new fileset:

```
ADD SERVER $ZPMON.#ZPN1, CPU 1, BACKUPCPU 0
```

c. Add the new fileset:

```
ADD FILESET $ZPMON.HOME, DEVICELABEL 000001, &
CATALOG $OSS, BUFFERED LOG, POOL POOL, &
NAMESERVER #ZPN1, MNTPOINT /home
```

d. Start the new fileset:

```
START FILESET $ZPMON.HOME
```

6. Copy all files and directories from the old `/home` directory into the new fileset using the OSS shell `cp` command:

   ```
   cp -pR /homex/* /home
   ```

   The `-p` flag preserves the original permissions and ownership of the files. If you do not specify this flag, the copied files are owned by the super ID.

   Note. If you have the `SUID` or `SGID` bits set for any file in this fileset, the `cp` operation does not preserve those settings and you must set those bits again manually.

   You can verify the results of the copy by using the following OSS shell commands:

   ```
   cd /homex
   ls -lR * > /tmp/homex.list
   cd /home
   ls -lR * > /tmp/home.list
   diff /tmp/home.list /tmp/homex.list
   ```

   If the copy was executed correctly, the `diff` command produces no output, meaning that the two directories are identical.

7. Remove the `/homex` directory and all files and directories underneath it:

   ```
   rm -r /homex
   ```

8. Restore the security permissions that you recorded from the `/homex` directory in Step 3 to the new `/home` directory:

   ```
   chmod 744 /home
   ```

Cleaning Up a Fileset

No OSS Monitor command exists to clean up unused inode entries in a fileset. The procedure to move files into their own filesets does not reduce the number of inode entries in the original (source) fileset. Rearranging filesets alone might not improve the performance of the SCF DIAGNOSE FILESET command on an affected fileset. To clean up the inode entries for a fileset, the fileset must be removed and reinstalled.

The following example, which uses a fileset called HOME, shows how to eliminate extraneous inode entries in the ROOT fileset. You must follow this same procedure for every fileset within the ROOT hierarchy except the ROOT fileset itself:

1. In SCF, stop the HOME fileset by using the following commands:

   ```
   STOP FILESET $ZPMON.HOME
   ```

2. Use pax from an OSS shell prompt to back up the ROOT fileset. The HOME fileset is stopped, so it is not included in this backup. HP recommends that you make multiple copies of this backup. For more information, see Creating a pax Backup of OSS Files in the Guardian File System on page 6-21.

3. In SCF, stop the ROOT fileset by using the following commands:

   ```
   STOP FILESET $ZPMON.ROOT
   ```

4. Display and record all the configuration information for the ROOT fileset:

   ```
   INFO FILESET $ZPMON.ROOT, DETAIL
   ```

5. Delete the ROOT fileset and then redefine it using the recorded information from Step 4:

   a. At a TACL prompt, go to the volume that is listed as CATALOG for the ROOT fileset:

      ```
      VOLUME $OSS.ZX000000
      ```

   b. Delete all the files in the ROOT fileset by using the FSCK utility:

      ```
      FSCK PURGE
      ```

6. Go back to SCF and start the ROOT fileset again:

   ```
   START FILESET $ZPMON.ROOT
   ```

7. Restore the ROOT fileset from the backup.

   If the pax backup was done to the Guardian file system, you can restore the ROOT fileset directly. See Restoring a pax Archive of OSS Files Directly From the Guardian Environment on page 6-25 for more information.

   If the pax backup was done to tape, you must first use COPYOSS to reinstall a basic configuration, so that you have a copy of the OSS shell you can start and a copy of pax to do the restoration with.

8. In SCF, start the HOME fileset:

   ```
   START FILESET $ZPMON.HOME
   ```

Troubleshooting Filesets

When you have problems managing a fileset, follow this general procedure:

1. Check the messages from the OSS Monitor that are sent to your terminal. If you redirect such messages to a log file, check the log file for its most recent entries. Look up the cause, effect, and recovery information for a message either in OSS Monitor Messages on page A-27 or by using the SCF HELP facility described in Online Help Facility on page 12-2.

   For example, this dialog illustrates an attempt to mount a new fileset on another fileset's mount point:

   ```
   ASSUME PROCESS $ZPMON
   START FILESET FILESET01
   OSS E00009 Failed to start fileset FILESET01
   ```

   The possible cause (two filesets on the same mount point) is suggested in the message description in OSS Monitor Messages on page A-27.

2. When you are unsure of the outcome of a command entry, check the EMS log for the most recent status messages. Look up the status messages in the *Operator Messages Manual*.

   For example, the following messages result from the START FILESET example in Step 1 when the OSS Monitor attempts to start the new fileset but must back out the operation because the command fails:

   ```
   2001-07-26 13:41:50 \NODE1.$ZPMON TANDEM.OSS.D40 -00003 $A ,
   STATE changed from Stopped to Started because of Operator
   Request.

   2001-07-26 13:41:50 \NODE1.$ZPMON TANDEM.OSS.D40 -00003 $A ,
   STATE changed from Started to Stopped because of Automatic
   Unmount by OSS Monitor.
   ```

3. When you are unsure of the effect of an action on a fileset, use the STATUS FILESET command DETAIL option to obtain the last internally reported error information for it. Look up that error in Numbered Messages on page A-35.

   For example, this dialog illustrates the condition of the fileset from Step 1:

   ```
   STATUS FILESET FILESET01, DETAIL

   OSS Detailed Status FILESET \PIMA.FILESET01

    State.................. STOPPED
    MountTime..............
    LastError..............  9
    ErrorDetail............  0
    ErrorTime.............. 26 Jul 2001, 13:41:50.392
    FsckName...............
    NumVols................  2
    Volumes:
    $DATA01  $DATA02
   ```

The LastError value reported (9) indicates that confusion exists over the state of the mount point. In this example, the LastError value is the same as the OSS Monitor error message number (E0009).

Managing and Repairing Fileset Catalog Files

An OSS name server maintains a catalog file for each fileset it manages. These catalog files require maintenance when:

- New OSS features require changes to the format of entries in the catalog. This maintenance is described in Upgrading OSS Catalog Files on page 5-40.
- Old catalog files need to be moved or removed as part of fileset management. This maintenance is described in Moving and Removing OSS Catalog Files on page 5-41.

Upgrading OSS Catalog Files

A fileset created while your site used a previous version of Open System Services can be used unmodified with the current software. However, filesets with catalog files that are not upgraded to the current RVU cannot support OSS features available in the current RVU but not available in the RVU used at the time the fileset was created or last upgraded. For example, a fileset created on a system running a D3x RVU cannot contain symbolic links until its catalog file is upgraded.

Your fileset catalog files should have been automatically upgraded when you installed a new product version of the OSS Monitor and started it for the first time. If your fileset catalog files were not upgraded at that time, you should consider upgrading them as soon as possible.

To upgrade a fileset catalog file:

1. Determine whether the fileset is mounted by using the SCF STATUS FILESET Command.
2. If the fileset is mounted, stop the fileset by using the procedure described under Stopping (Unmounting) a Fileset on page 5-13.
3. Enter the following SCF command for the fileset:

   ```
   DIAGNOSE FILESET filesetname, UPGRADE
   ```

   filesetname

   specifies the fileset to have its catalog file upgraded so that it can support current OSS features.
4. As soon as the upgrade operation is complete on the fileset catalog file, use the SCF START FILESET Command for the fileset. Note that the root fileset must be started before any other fileset can be started.

All mounted filesets can be stopped in one operation by using the wildcard character * in place of a fileset name in the SCF STOP command. To run the SCF DIAGNOSE FILESET Command on multiple filesets, use the CPU option to begin the process for each fileset on a different processor.

Moving and Removing OSS Catalog Files

OSS catalog files cannot be moved or purged with standard OSS or Guardian file-system commands; you must use the Guardian Catalog Volume Tool (CVT) utility.

CVT is useful in cases such as:

- You can purge old catalog files once you have decided that an FSCK repair operation was successful and the old files are no longer needed.
- In the rare case of a double media failure, you can still perform a partial catalog recovery by restoring the latest backup copy of the catalog file and running the SCF DIAGNOSE FILESET command with the REPAIR ALL option.

  The Guardian RESTORE utility cannot open files on OSS catalog subvolumes, but you can restore the catalog to another subvolume and then use CVT to move the restored catalog to the OSS catalog subvolume.

CVT also enables you to:

- Move catalog files between subvolumes on the same disk volume
- Move catalog files onto and off of ZX0 subvolumes
- Save a current catalog in the same manner that FSCK does when it saves an existing catalog and creates a new one
- Make a saved catalog the current catalog for a fileset
- Purge catalog files saved by FSCK

CVT is an unlicensed privileged program and, therefore, can be run only by the super ID (which is 255,255 in the Guardian environment, 65535 in the OSS environment). CVT is restartable; if it fails while moving or purging a catalog file, you can restart it using the same command.

Running CVT

Run CVT using the following command at a TACL prompt.

```
RUN $vol.ZOSS.CVT  [ HELP | ?                                    ]
                   [ PURGE SERIAL serialno [ IN subvolume ] ]
                   [ RENAME files1  [ TO ]  files2               ]
```

$vol

is the name of the disk volume where the ZOSS subvolume is located.

HELP | ?

is the HELP command, which provides an overview of the CVT command syntax. This is the default action; that is, if this RUN command is entered without any options, the command is interpreted as if the option HELP had been entered.

PURGE SERIAL *serialno* [IN *subvolume*]

is the PURGE command, which purges a catalog saved by FSCK.

serialno

is the FSCK serial number (FSN) of the saved catalog to be purged.

IN *subvolume*

is the name of the subvolume where the saved catalog is located. If this name is not supplied, the user's default subvolume is assumed.

RENAME *files1* [TO] *files2*

is the RENAME command, which moves (renames) catalog files between and within subvolumes.

files1

specifies the files you want to move (rename).

files2

specifies the new locations (names) for the files being moved (renamed).

Both *files1* and *files2* have the following syntax:

{ CURRENT | SERIAL *serialno* } [IN *subvolume*]

CURRENT

for *files1*, specifies that the current catalog files PXINODE, PXLINK, and PXLOG are to be renamed.

For *files2*, indicates that the

- New location for the catalog files is the current catalog.
- New names for the catalog files are PXINODE, PXLINK, and PXLOG.

SERIAL *serialno*

for *files1*, indicates that a saved catalog with the FSN *serialno* (consisting of the files PXIN*serialno*, PXLI*serialno*, and PXLO*serialno*) is to be renamed.

For *files2*, indicates that the

- New location for the catalog files is the saved catalog with the FSN *serialno*.
- Files are to be renamed PXIN*serialno*, PXLI*serialno*, and PXLO*serialno*.

IN *subvolume*

indicates the name of the subvolume for the corresponding *files1* or *files2* saved catalog files. If this name is not supplied, the user's default subvolume is assumed.

Using the CVT HELP Command

To get help information for CVT, enter one of the following commands at a TACL prompt:

```
RUN $vol.ZOSS.CVT
RUN $vol.ZOSS.CVT HELP
RUN $vol.ZOSS.CVT ?
```

$*vol*

is the name of the disk volume where the ZOSS subvolume is located.

The CVT HELP command produces the following output.

```
CVT - T8621G09 - (01FEB01) OSS Catalog Volume Tool
Copyright Compaq Computer Corporation 1994, 1995, 1996, 2001

CVT { RENAME <files1> [ TO ] <files2>       }
    { PURGE SERIAL <serial#> [IN <subvol> ] }

where <files1> and <files2> are:

     { CURRENT         } [ IN <subvol> ]
     {SERIAL <serial#>  [ IN <subvol> ] }
```

CVT Examples

Here are some examples of CVT commands:

- To purge the files PXIN0004, PXLI0004, and PXLO0004 on subvolume $VOL.ZX000003, enter the following command at a TACL prompt:

  ```
  RUN $SYSTEM.ZOSS.CVT PURGE SERIAL 4 IN $VOL.ZX000003
  ```

- To save the current catalog on $VOL.ZX000003 as PXIN0004, PXLI0004, and PXLO0004 on the same subvolume, enter the following command at a TACL prompt:

  ```
  RUN $SYSTEM.ZOSS.CVT RENAME CURRENT IN $VOL.ZX000003 TO &
     SERIAL 4 IN $VOL.ZX000003
  ```

  This command is equivalent to the following command sequence:

  ```
  VOLUME $VOL.ZX000003
  RUN $SYSTEM.ZOSS.CVT RENAME CURRENT TO SERIAL 4
  ```

- To make the current catalog on $VOL.TEMP the current catalog on $VOL.ZX000003, enter the following commands at a TACL prompt:

  ```
  VOLUME $VOL.ZX000003
  RUN $SYSTEM.ZOSS.CVT RENAME CURRENT IN TEMP TO CURRENT
  ```

- To restore from a backup tape a catalog that has been destroyed due to a double media failure:

  1. Use the Guardian RESTORE utility VOL keyword to restore the catalog to a subvolume whose name does not begin with ZX0.
  2. Move the catalog onto the ZX0 subvolume with a TACL command of the following form:

     ```
     RUN $SYSTEM.ZOSS.CVT RENAME CURRENT IN nonZX0subvol  &
        TO CURRENT IN ZX0subvol
     ```

  3. After you restore the catalog, run the integrity checker with the following SCF command:

     ```
     DIAGNOSE FILESET fileset, REPAIR ALL
     ```

- When FSCK detects catalog corruption, it does not repair the catalog in place; instead, it creates a new catalog and saves the old one. The old catalog is saved under the Guardian file identifiers PXIN*nnnn*, PXLI*nnnn*, and PXLO*nnnn* (where *nnnn* is an FSN).

  To remove an old catalog retained by FSCK, enter:

  ```
  RUN $SYSTEM.ZOSS.CVT PURGE SERIAL fsn
  ```

  where *fsn* is the FSN of the old catalog.

6 Managing OSS Files

To manage files in the OSS file system, you need to:

- Obtain specific information about their size, location, and fileset membership, as described in Obtaining Information About OSS Files.
- Install new or updated product files, as described in Installing New Product Files on page 6-4.
- Remove obsolete files, as described in Removing Obsolete OSS Files and Directories on page 6-9.
- Update OSS database files that are not maintained through SCF commands, such as the `whatis` database files used by the `man` command. This action is described in Updating the whatis Database Files on page 6-10.
- Back up and restore the OSS environment and user files, as described in Backing Up and Restoring OSS Files on page 6-11.
- Redirect data when Guardian files or processes need to be used instead of a Telserv terminal session. This action is described in Redirecting OSS Standard Files on page 6-27.
- Control the number of files allowed in a fileset by controlling the maximum number of inodes allowed in the catalog for the fileset. This action is described in Controlling the Maximum Number of Files on page 6-30.

Section 9, Managing With the Shell, describes other file-oriented management tasks, beginning under the topic of Overuse of Resources on page 9-8.

Obtaining Information About OSS Files

In the OSS environment, you use the OSS shell `ls` command and its optional flags to obtain size, access, and other information about OSS files. In the Guardian environment, you can use the TACL FILEINFO command or the File Utility Program (FUP) INFO command for the same purpose.

The FILEINFO command provides only information that is appropriate for a file in the Guardian environment; however, the FUP INFO command can provide information that is appropriate for the file in the OSS environment. See Using FUP INFO on OSS Regular Files on page 6-3 for more information.

Sometimes you need to determine the Guardian filename used for an OSS file. For example, portions of the Guardian filename can provide you with information useful when you perform the procedures in Section 5, Managing Filesets. See Interpreting Guardian Filenames for OSS Files on page 6-2 for more information.

The OSS shell `gname` and `pname` utilities allow you to provide a Guardian filename and obtain the OSS pathname for a specific file and vice versa. See the following subsections for more information:

- Using the OSS gname Command on page 6-2
- Using the OSS pname Command on page 6-3

You can also use the Guardian VPROC utility or the OSS shell `vproc` command to determine the product-version information for an OSS file supplied by HP. The procedure to use either VPROC or `vproc` is described under Gathering Version Information About OSS Files on page 11-1.

Interpreting Guardian Filenames for OSS Files

OSS files have a distinctive form of Guardian filename in the Guardian environment. In addition to the usual node name and volume name information, the Guardian filename for an OSS file has the following form:

- Subvolume names begin with ZYQ. These subvolume names correspond to OSS filesets.

  The digits that follow ZYQ are the device identifier for the fileset within the ZOSSFSET database file used by the node. See The ZOSSFSET File on page 4-8 for more information about OSS device identifiers.

- File identifiers begin with Z0. These file identifiers correspond to OSS file system inodes.

  The rightmost digits in the file identifier correspond to the OSS inode number for the file. (You can also have files other than OSS files whose file identifiers begin with Z0.) See Inconsistencies Checked by FSCK on page 5-29 for more information on the use of inode numbers.

Using the OSS gname Command

To display the Guardian equivalent of an OSS pathname, enter the following command from the OSS shell:

```
gname [-s] filename
```

where the optional `-s` flag displays only the Guardian filename and *`filename`* is an OSS pathname. You can use shell wildcard characters in the OSS pathname; however, you should not use wildcard characters in the node-name portion of a pathname that includes the `/E` directory because of the potentially large number of files involved.

Figure 6-1 shows examples of using `gname`.

Figure 6-1. OSS gname Command Examples

```
$ gname test
gname: test --> \NODE1.$VOL.ZYQ00000.Z0000DV3
$ gname tes*
gname: test2 --> \NODE1.$VOL.ZYQ00000.Z0000KHP
$ gname -s test
\NODE1.$VOL.ZYQ00000.Z0000DV3
$ gname /E/node2/usr/test3
gname:  /E/node2/usr/test3 --> \NODE2.$DATA.ZYQ00001.Z0000DV2
```

When more than one OSS pathname matches the possible wildcard expansion, only the last file with a matching pathname is listed. Figure 6-1 illustrates this, where both `test` and `test2` are in the directory `/usr` on the NonStop node \NODE1 but only `test2` is listed for the command `gname tes*`.

Using the OSS pname Command

To display the OSS equivalent of a Guardian filename, enter the following command from the OSS shell:

`pname [-s]` *`filename`*

where the optional `-s` flag displays only the OSS pathname and *`filename`* is a Guardian system-qualified filename.

Note. In *`filename`*, you must put another backslash character before the backslash (\) and dollar sign ($) characters or else the shell interprets these characters with their special shell meanings rather than as plain characters.

Figure 6-2 shows examples of using `pname`. When the file is on another NonStop node, the pathname of the file relative to the `/` directory on that node is displayed after the prefix of `/E/` and the node name.

Figure 6-2. OSS pname Command Examples

```
$ pname \\NODE1.\$VOL.ZYQ00000.Z0000DV3
pname: \NODE1.$VOL.ZYQ00000.Z0000DV3 --> /home/henrysp/test
$ pname -s \\NODE1.\$VOL.ZYQ00000.Z0000DV3
/home/henrysp/test
$ pname \\NODE2.\$DATA.ZYQ00001.Z0000DV2
pname:  \NODE2.$DATA.ZYQ00001.Z0000DV2 --> /E/node2/usr/test3
```

Using FUP INFO on OSS Regular Files

The FUP INFO display for an OSS regular file (you must use the Guardian equivalent of the OSS pathname in this command) shows OSS file access permissions rather than Guardian security. Examples of displays from the FUP INFO and FUP INFO, DETAIL commands for OSS regular files are shown in Figure 6-3 on page 6-4.

The FUP INFO, DETAIL display shows the OSS pathname for the file next to the PATH heading. The OSS permissions appear under the RWEP heading in the FUP INFO

display and next to the SECURITY heading in the FUP INFO, DETAIL display. For information about interpreting the OSS permissions, see the *Open System Services User's Guide*.

Figure 6-3. FUP INFO Displays for OSS Files

```
3>   fup info $VOL.ZYQ00000.Z0000DV4
                CODE           EOF    LAST MODIF   OWNER RWEP   TYPE   REC BL
$VOL.ZYQ00000
Z0000DV4        OSS            142         10:55   254,254 -rw-rw-rw-

4> fup info $VOL.ZYQ00000.Z0000DV4, detail

$VOL.ZYQ00000.Z0000DV4         26 Jul 1994, 13:24
    OSS
    PATH: /usr/henrysp/stuff
    OWNER 254,254
    SECURITY: -rw-rw-rw-
    CREATION DATE:   2 Jun 1994, 10:38
    ACCESS TIME:  19 Jul 1994, 19:52
    EOF 142

5> fup info \NODE1.$DATA.ZYQ00001.Z0000DV2, detail

\NODE1.$DATA.ZYQ00001.Z0000DV2         26 Jul 1996, 14:22
    OSS
    PATH: /E/node1/usr/test3
    OWNER 254,254
    SECURITY: -rw-rw-rw-
    CREATION DATE:   2 Jun 1996, 12:48
    ACCESS TIME:  19 Jul 1996, 19:56
    EOF 152
```

Installing New Product Files

Beginning with the G06.18 release version update (RVU), you have two possible methods of installing new product files into the OSS file system:

- You can use a D46 or newer software product revision (SPR) of the Distributed Systems Management/Software Configuration Manager (DSM/SCM). This is the recommended method for all DSM/SCM-enabled products (products that have an A7CINFO file associated with them).
- You can use the two utilities that come with the basic OSS products:
  - The COPYOSS TACL macro
  - The PINSTALL utility

  Both of these utilities provide the equivalent of a UNIX `pax` utility in the Guardian environment. (COPYOSS uses PINSTALL.)

Most OSS product files are distributed in `pax` archive form (`ustar` format) on the $*`tsvvol`*.ZOSSUTL subvolume, where *`tsvvol`* is the disk volume where DSM/SCM puts your target subvolumes (TSVs). Each product archive contains one or more files with predetermined OSS pathnames.

This set of archive files is large—on the order of tens of megabytes. It is, however, a temporary group of files. Unless DSM/SCM maintains the OSS files on your node, this set of files can be deleted after OSS installation is complete; deleting them when DSM/SCM maintains OSS files can slow down subsequent installations or upgrades because DSM/SCM will replace them.

The contents of these files must be copied to their proper places in the filesets on your system by using one of the following:

- DSM/SCM with the **Manage OSS Files** maintenance check box selected. This alternative allows DSM/SCM to maintain product files in your OSS file system and to perform its normal backout functions for them when necessary. If you do not select **Manage OSS Files**, DSM/SCM just installs the files needed to set up the OSS environment into the TSV.
- The TACL COPYOSS macro, which automatically runs the Guardian PINSTALL utility on all the archive files in the subvolume.
- The PINSTALL utility, which you can run manually for individual archive files.

The use of either COPYOSS or PINSTALL also requires you to use the OSS `Pcleanup` utility to remove obsolete files. DSM/SCM does not require `Pcleanup`. See Removing Obsolete OSS Files and Directories on page 6-9 for more information.

The considerations for these options differ slightly, as shown in Table 6-1.

Table 6-1. Comparing the Installation Tools (page 1 of 2)

Consideration	DSM/SCM	COPYOSS	PINSTALL	Pcleanup
Permanent OSS file system disk space	Up to twice the total size of installed product files (includes backout configuration)	Space required by files for the set of products installed	Space required by files for the set of products installed	Reduces usage slightly if option to remove obsolete files is used
File mode/security settings for previously installed files being updated	Preserved	Overwritten by HP-supplied default values for T8626, T8627, and T8628	Controlled by user	Not applicable
Use on subvolumes other than ZOSSUTL	Not currently supported	Supported as a nondefault option	Supported	Not applicable

Table 6-1. Comparing the Installation Tools (page 2 of 2)

Consideration	DSM/SCM	COPYOSS	PINSTALL	Pcleanup
Can have multiple versions of the same file in different directories (for example, installing `/bin/ver4/file` to replace `/bin/ver3/file` does not remove `/bin/ver3/file`)	No; obsolete files are removed automatically	Yes; has no effect on older files other than the ones in a directory specified in a `pax` archive	Yes	Yes; unless the option to remove the older files is used
Can use on `pax` archives that are not DSM/SCM enabled	No	Yes	Yes	Not applicable
Directories emptied of obsolete files are retained unless manually deleted	No	Yes; has no effect on obsolete files or directories	Yes; has no effect on obsolete files or directories	Yes

Using COPYOSS

COPYOSS can be used to load the contents of a single `pax` archive file into the OSS file system or to load the contents of all the `pax` archive files in a subvolume. When COPYOSS is used on the entire contents of a subvolume, it processes all files in the subvolume with file codes of 0 or 180 unless the files have file identifiers beginning with ZFB or ZPG; files with file identifiers beginning with ZFB or ZPG are assumed to be old files renamed by DSM/SCM and are ignored.

COPYOSS is used when:

- A system is initially set up manually
- The OSSSETUP utility invokes it
- You do not use DSM/SCM to install and maintain your OSS product files but a major upgrade requires you to load multiple new or revised `pax` archives into the OSS file system

For example:

- To load all the OSS product files using the COPYOSS macro, enter these commands from a TACL prompt:

```
VOLUME $tsvvol.ZOSSUTL
RUN COPYOSS ZOSSUTL
```

tsvvol

is the disk volume where DSM/SCM puts your TSVs.

From the archive files, COPYOSS copies the OSS product files into the OSS file system, placing them in locations where you would find them on a typical UNIX system—for example, into directories such as `/bin`, `/usr/ucb`, and `/usr/include`.

This command completely installs the basic OSS product set and all other products that use the same installation subvolume.

- To load only the Java servlet product files, enter these commands from a TACL prompt:

  `$`*`tsvvol`*`.ZOSSUTL.COPYOSS `*`$tsv1`*`.T0094PAX`

  `tsvvol`

  is the disk volume where DSM/SCM put your basic OSS product set TSVs.

  `$tsv1`

  is the disk volume and subvolume where DSM/SCM put the NonStop Java Server TSVs.

  `T0094PAX`

  is the `pax` archive containing the Java servlet files.

See the `copyoss(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for more information about command options.

Considerations

- COPYOSS exists only in the target subvolume used for installation of the basic OSS product set. If that subvolume is deleted after installation, your system will not have the COPYOSS file. You can save the COPYOSS file to the $SYSTEM disk volume if you need to use it again.
- Do not use COPYOSS unless explicitly told to when DSM/SCM installs and maintains OSS product files in your OSS file system (when the DSM/SCM **Manage OSS Files** check box is selected).
- Files installed by COPYOSS are not always secured in conformance with the best practices at your site. You should always resecure files installed by COPYOSS. For example, after installing SQL/MX files, you might enter commands such as the following to secure the installed software and the directories it uses:

```
find / -WNOE -WNOG \( -type d -o -type f \) -perm -o+w |
     xargs chmod o-w
chmod a=rwxt /tmp /usr/tmp /var/tmp /var/preserve
     /usr/tandem/sqlmx/USERMODULES
```

- Under rare conditions, the PINSTALL command used by COPYOSS can return an error message that indicates a disk file could not be found; this message can be ignored when the named file is an empty directory in the corresponding `pax`

archive file. You can use the `-cvf` flags of the PINSTALL command to display the table of contents for the `pax` archive to determine if the named file is actually an empty directory.

Using PINSTALL

PINSTALL is used when:

- An SPR is installed manually
- Ported or third-party software requires you to install `ustar`-format files into the OSS file system
- The COPYOSS utility invokes it
- You do not use DSM/SCM to install and maintain your OSS product files but a major upgrade requires you to load multiple new or revised `pax` archives into the OSS file system

You can speed up the process of loading OSS product files by loading individual files in parallel. You can do this in either of the following ways:

- Run the PINSTALL utility in each of multiple terminal windows
- Repeatedly run the PINSTALL utility with the NOWAIT option on your home terminal and send the output of each command to the spooler

For example, to load the contents of the basic OSS product set files individually, enter a PINSTALL command at a TACL prompt in the following form for each file on the ZOSSUTL subvolume that has a file code of 0 or 180:

```
PINSTALL -rvf /G/tsvvol/zossutl/archfile
```

`tsvvol`

is the disk volume where DSM/SCM puts your TSVs.

`archfile`

is the Guardian file identifier for a `pax` archive file.

The options (such as `-rvf`) to the PINSTALL utility are case-sensitive. For more information about the PINSTALL utility, see the `pinstall(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Considerations

- Do not use PINSTALL on the files in an SPR if DSM/SCM installs the SPR and its **Manage OSS Files** check box is selected.
- Do not use PINSTALL on the entire contents of a subvolume. Files that have file identifiers beginning with ZFB or ZPG are probably old files renamed by DSM/SCM

and should be ignored. Using PINSTALL on such files would overwrite current files with obsolete ones.

- PINSTALL is installed as a part of the basic OSS product set and is available even if the subvolume containing COPYOSS has been deleted from your system.
- Under rare conditions, the PINSTALL command can return an error message that indicates a disk file could not be found; this message can be ignored when the named file is an empty directory in the corresponding `pax` archive file. You can use the `-cvf` flags of the PINSTALL command to display the table of contents for the `pax` archive to determine if the named file is actually an empty directory.
- Files installed by PINSTALL are not always secured in conformance with the best practices at your site. You should always resecure files installed by PINSTALL. For example, after installing SQL/MX files, you might enter commands such as the following to secure the installed software and the directories it uses:

```
find / -WNOE -WNOG \( -type d -o -type f \) -perm -o+w |
     xargs chmod o-w
chmod a=rwxt /tmp /usr/tmp /var/tmp /var/preserve
     /usr/tandem/sqlmx/USERMODULES
```

- Beginning with SPR T8626AAY and the G06.08 RVU, the PINSTALL utility should not be used to install individual `pax` archives for these products:
  - T8626
  - T8627
  - T8628

  Use the COPYOSS utility instead. COPYOSS performs special processing for those products.

  When circumstances force the use of PINSTALL instead of COPYOSS for these products, the operator must manually run the utility `/bin/replace` immediately after installing the T8626 `pax` archive RELUTILS, then remove the `/bin/replace` utility from the OSS file system.

  This consideration was removed beginning with SPR T8626ABH and the G06.15 RVU.

Removing Obsolete OSS Files and Directories

When you update OSS products, you might need to remove files from previous RVUs.

The installation process places files containing lists of obsolete files in the directory `/etc/install_obsolete`. If:

- DSM/SCM installed and maintains the OSS product files in your OSS file system, you should do nothing with these files. Attempting to use the files for maintenance will invalidate DSM/SCM database information about OSS product files on your system.

- DSM/SCM does not install and maintain OSS product files in your OSS file system, you must manually remove obsolete files after installing any new release version, RVU, or software product revision (SPR) and before using the `merge_whatis` command.

You manually remove obsolete files by entering the following OSS shell commands:

```
cd /etc/install_obsolete
Pcleanup -r source
```

Note that the command `Pcleanup` starts with an uppercase letter.

You can also use the `Pcleanup` utility to:

- Display all obsolete files, by using the `-i` flag
- Move all obsolete files to `/etc/install_obsolete`, by using the `-m` flag
- Remove all files in the obsolete files source directory, by using the `-r target` flag

For additional information, see the `Pcleanup(8)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

DSM/SCM automatically deletes directories left empty after obsolete files are removed but the `Pcleanup` utility does not. To remove unneeded directories after using `Pcleanup`, use the OSS shell `rmdir` command.

△ **Caution.** Invisible OSS files with names containing `.dsmscm` are used by DSM/SCM for OSS file/configuration management and should never be deleted. Such files can be found in otherwise empty directories, so you should use the OSS shell utility `/bin/ls -al` on any directory to check for hidden files before the directory is removed.

Updating the whatis Database Files

When you install or update products that include reference pages, you need to update the `whatis` database files in the OSS environment.

Note. Beginning with the G06.14 RVU, if your system was ordered preconfigured or your initial OSS configuration was performed by using the OSSSETUP utility, this action occurred automatically. However, you must perform this action manually after any subsequent update to the content provided by HP for the OSS file system.

A `whatis` database file contains a summary of each reference page (sometimes known as a man page) in the corresponding set of OSS directories.

The OSS shell supports the `MANPATH` environment variable. Each set of reference pages accessible through a single `MANPATH` environment variable entry has its own `whatis` database file in the directory specified for the `MANPATH` variable value.

These database files are accessed by the `man`, `apropos`, and `whatis` utilities, based upon either the `MANPATH` variable value in effect for the user or a `man` command flag that the user can specify. Although the commands can function without a database file, they do not return valid information.

You create or update a whatis database file by merging whatis database fragment files using the merge_whatis utility. Each whatis database fragment file has an OSS filename with the form whatis.*piece*, where *piece* varies according to the product containing the reference page files.

The whatis database fragment file for each product shipped as part of Open System Services is automatically installed into the appropriate one of the following directories:

/usr/share/man/whatis.frag
/nonnative/usr/share/man/whatis.frag (G-series only)

To merge these whatis.*piece* files into the corresponding whatis database after installing an update to Open System Services, enter the following commands from an OSS shell prompt:

merge_whatis
merge_whatis /nonnative/usr/share/man (G-series only)

The corresponding whatis databases are now available for use with the following MANPATH variable values:

/usr/share/man (OSS product reference pages only)
/nonnative/usr/share/man (G-series only)

You can verify the existence of the whatis databases by entering the OSS shell ls command for each of these MANPATH variable values.

Your site can acquire additional products to install in the OSS environment. If these additional products include reference pages, you need to add entries to an existing whatis database or create a new whatis database for a new MANPATH variable value.

For additional information about the merge_whatis utility, see the merge_whatis(8) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Backing Up and Restoring OSS Files

This subsection describes the tasks for backing up and restoring the OSS environment and user files on your system.

OSS files can be backed up with the OSS pax command or the BRCOM interface for Backup and Restore 2.0. An entire OSS fileset can be backed up by using its mount point as the current directory when running either utility.

System administrators familiar with Guardian tools can use Backup and Restore 2.0 for almost all archiving tasks. Administrators more familiar with UNIX can use pax; however, the pax utility has several restrictions:

- For H06.06 and later release version updates (RVUs), a pax archive has a file size limit of 8 gigabytes. For earlier RVUs, a pax archive has the same file size limit as other OSS files (approximately 2 gigabytes). These limits might prevent complete backup of a large fileset or directory tree.

- The pax utility does not support labeled tapes and imposes requirements for unlabeled tapes.
- The pax utility does not support filenames longer than 100 characters in archives using its default USTAR format.

The pax utility can be used when any of the following conditions exists:

- The system that will be restored (target system) might not have the same node name or node number as the system being backed up (source system).
- The operating system release version update (RVU) of the target system might not be the same as that of the source system.
- The disk volume names of the target system might not be the same as those on the source system. (The storage-pool files and any OBEY files used on the source system will still need to be modified to be used on the target system.)

Considerations

For all backup activities described in this guide, the pax utility must be run at a time when no updates or changes are being made to the OSS files that are being saved.

Multiple copies of pax output from a backup activity should be saved in case of media failure.

You cannot use the following media or devices for an OSS backup from the OSS environment:

- Storage Management Foundation (SMF) logical volumes
- Tape libraries

You cannot perform remote backups of OSS files, directories, or file systems directly. For information about remote backups, see Backing Up OSS Files to Other Expand Nodes on page 6-22.

Note. You cannot restore an OSS file that is larger than approximately 2 gigabytes to a system running an RVU that does not include support for OSS large files.

Guardian Files and the pax Utility

You should not use the pax utility to back up or restore files in /G. Such files should be backed up and restored using TMF or a version of Backup/Restore. See Backing Up User Files on page 6-15.

Unless otherwise noted, the pax utility makes no distinction between Guardian and OSS files.

If the pax utility cannot process a specific /G file, pax returns a diagnostic message and an error value from the underlying program interface. If possible, pax continues to process the other files.

Guardian Tape Devices and the pax Utility

The pax utility uses Guardian tape devices to read and write tape archives.

You cannot use pax on labeled tapes. If you need to backup OSS files to labeled tapes or restore OSS files from labeled tapes, use Backup and Restore 2.0.

You are also restricted when using pax to single write operations on unlabeled tape. You can use the pax -W norewind option only when you combine all write operations into one subshell; see the NOTES section of the pax(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*, or see Consolidate Changing OSS Files on page 6-15.

Guardian tape devices are controlled by the Guardian tape process executing in the Guardian environment and do not behave in the same way as UNIX devices. The interaction between the tape process and the tape device is transparent to the pax user.

To set the block size to its maximum when using tape, specify the pax command -b28k flag.

When the pax utility accesses a tape archive, the pax utility issues a mount request to the Guardian tape process. If you use the -W wait flag and no tape is correctly mounted on the specified drive, the following message appears on the originating terminal:

```
Device not ready or tape is not mounted?
```

The pax utility does not proceed with reading or writing a tape until an unlabeled tape has been correctly mounted on the specified tape drive.

If you do not use the -W wait flag and no tape is correctly mounted on the specified drive, pax exits.

To check for outstanding requests for tape drive status, use the Guardian utility MEDIACOM. Invoke this utility from a TACL prompt or from the OSS shell with the gtacl command. For information about MEDIACOM, see the *DSM/Tape Catalog Operator Interface (MEDIACOM) Manual*.

If there are errors related to the device or to the mounted tape during the tape-mount process, one or more of the following messages are sent to the originating terminal:

```
Tape is unloaded -- media is write protected
Tape mount error (Guardian file system error: n)
Tape read failed with Guardian file system error: n
Tape write failed with Guardian file system error: n
```

where *n* is a Guardian file-system error number. You can use the Guardian ERROR utility to find the meaning of the error number. Correct the error and remount the tape or cancel the tape-mount request using the MEDIACOM utility.

If the request is canceled or the archive cannot be opened, the pax utility issues the following message to the originating terminal:

```
filename cannot be opened, Guardian file system error: n
```

where *n* is a Guardian file-system error number.

A single archive can span more than one reel. The pax utility issues this message during reel switching:

```
Mount next tape to continue?
```

Backing Up the OSS Environment Using a Version of Backup/Restore

1. Record all configuration information for the OSS environment. You can do this in either of the following ways:
   - Use the OSS Monitor Subsystem Control Facility (SCF) INFO commands to gather information about all objects managed as part of the OSS environment (the OSS Monitor itself, all OSS servers, and all OSS filesets):

     ```
     SCF
     INFO / OUT SNAPSHOT / MON $ZPMON, DETAIL
     INFO / OUT SNAPSHOT / SERVER $ZPMON.*, DETAIL
     INFO / OUT SNAPSHOT / FILESET $ZPMON.*, DETAIL
     ```

     Build SCF OBEY files from the captured information such that the attributes of each object can be recreated if necessary.
   - Backup all OSS configuration files described in Section 4, Managing Servers, using the T9074 Backup and Restore utilities.
2. Backup all startup files using the T9074 Backup and Restore utilities:
   a. If your site uses any of the OSS EasySetup utilities described in Appendix C, OSS Management Utilities, and maintains the files used by those utilities, back those files up.
   b. If your site has manually configured the persistence monitor to manage the OSS Monitor, use the SCF INFO DETAIL command for the kernel subsystem to gather the information needed to build an OBEY file that can recreate the attributes of the OSS Monitor, if necessary.
3. Record the information in all storage-pool files. (You can back up and restore storage-pool files using the T9074 Backup and Restore utilities.)
4. Backup and Restore 2.0 preserves OSS file access information. However, if you are using the Backup and Restore utilities (T9074), you need to record all user ID

information relevant to OSS file access. You can do this in either of the following ways:

- Using Safeguard:

```
SAFECOM
DISPLAY AS COMMANDS ON
LOG OSSGRPS
INFO GROUP *
LOG OSSUSRS
INFO USERS *.*
LOG OSSALIAS
INFO ALIAS *
LOG
```

  Edit these log files to create OBEY files for Safeguard so that the attributes of each file-sharing group, user ID, and user ID alias can be recreated if necessary.

- Using the T9074 Backup and Restore utilities (if the node you are backing up uses the same RVU of Safeguard as the node you will be restoring). The relevant user ID information is kept in the following files:

```
$SYSTEM.SYSTEM.USERID
$SYSTEM.SYSTEM.USERIDAK
$SYSTEM.SAFE.LUSERID
$SYSTEM.SAFE.LUSERIDG
```

  Use the OPEN option when doing the BACKUP operation. Ensure that no Safeguard changes are made while BACKUP is running.

Backing Up User Files

You should back up and restore all Guardian files and SQL data tables used by OSS applications using normal procedures for Guardian data:

- Recover audited files and tables using TMF; nonaudited files and tables
- Backup and recover unaudited files and tables using a version of Backup/Restore

Use Backup and Restore 2.0 BRCOM commands to back up SQL/MX and OSS files. Use Backup and Restore utility (T9074) commands to backup SQL/MP and Enscribe files.

Although several OSS filesets can be simultaneously backed up—assuming that your system has more than one storage device—each fileset backup is a separate task.

Consolidate Changing OSS Files

To make backup easier, organize your filesets so that changing data files are on filesets that are regularly backed up while static files are on filesets that are backed up only occasionally. You can simplify backup by copying isolated, changing files from filesets that are not periodically backed up to those that are, just before you perform a backup. This action allows the dynamic files to be backed up without requiring an

entire file-system backup. You could write a shell script to do this. For information about shell scripts, see the *Open System Services User's Guide*.

You can use the `find` command to produce a list of files that must be backed up and then pipe this list to the backup command `pax`. For example, to back up all user files that were modified in the past week onto the tape using the tape device $TAPE, use a command such as:

```
find /user/* -mtime -7 | pax -w -f /G/TAPE
```

where `/user` contains all working directories.

Note. Do not begin `find` or `pax` operations at the root (`/`) directory. `find` and `pax` perform recursive operations within directories unless you use the `UTILSGE=NOG:NOE` environment variable or the `-W NOG` and `-W NOE` flags. The `/` directory contains `/G` and `/E`. If you begin an operation that recursively processes directories at `/`, you can unintentionally process an entire Guardian file system and the operation will take a very long time to finish.

See Using the Local Root Directory as a Pathname on page 3-5 for more information about `UTILSGE`, `-W NOG`, and `-W NOE`.

If you attempt to archive individual files, you must compensate for the fact that the `pax` utility cannot append a file to an unlabeled tape. Each successive write to such a tape begins at the beginning of the tape.

For example, if you issue the following commands from the shell:

```
find xlog -print | pax -wv -f /G/TAPE -W norewind
find xlog.bsm -print | pax -wv -f /G/TAPE -W norewind
```

then physically unload the tape, reload the tape, and enter:

```
pax -rv -f /G/TAPE -W norewind
```

then the tape contains only the last file archived by the two command lines entered. To archive more than one file on an unlabeled tape, you must enter all the commands within the same subshell. For example:

```
( find xlog -print; find xlog.bsm -print ) | pax -wv -f /G/tape
```

This command causes all the files printed by both `find` commands to be written to tape, because the `find` commands are executed in a single subshell.

For more information about the `pax` and `find` commands, see the `pax(1)` reference page and the `find(1)` reference page, either online or in the *Open System Services Shell and Utilities Reference Manual*. Additional information about both commands is in the *Open System Services User's Guide*.

Backing Up the OSS File Hierarchy of the Current Directory

To back up the file hierarchy of the current directory to the tape mounted on Guardian tape device $TAPE, using the blocking factor for 5120 bytes, enter the following OSS shell command:

```
pax -wv -f /G/tape -b 10b .
```

This command has the following form:

```
pax -w -v -f archive_name -b blocksize .
```

`-w`

writes files to the standard output file in the specified archive format (the default format is `ustar`).

`-v`

writes archive member pathnames to the standard error file.

`-f` *`archive_name`*

specifies the pathname of the output archive, overriding the default standard output file. Guardian tape devices can be specified with the `/G` naming convention (for example, `/G/tape`).

If the `-a` flag is also specified and a disk archive medium is used, files are appended to the end of the archive.

`-b` *`blocksize`*

records an archive as a series of fixed-size blocks to make physical output more efficient. Blocking is automatically determined on input.

The *`blocksize`* argument can have values no greater than 32,256 for disk archives and 28,672 for tape archives.

The *`blocksize`* argument can be specified as a series of digits (0 through 9) followed by a flag letter, “b” or “k”. If “b” is used, the *`blocksize`* value is multiplied by 512. If “k” is used, the *`blocksize`* value is multiplied by 1024. For example, “`10b`” translates to a *`blocksize`* value of 5120 bytes (10 * 512).

The default *`blocksize`* value for `cpio` archive format is `10b` (5120 bytes). The last group of blocks is always at the full size.

The default *`blocksize`* value for `ustar` archive format is `20b` (10240 bytes). You specify the *`blocksize`* argument as a multiple of 512 bytes.

`.`

specifies the current directory.

Backing Up OSS File Hierarchies Using Backup and Restore 2.0

You can specify part or all of multiple directory hierarchies and selectively backup files from within them using Backup and Restore 2.0. For example, if you enter the following at a Backup and Restore 2.0 BR> prompt:

```
BACKUP =MYTAPE, OSS ((/user/bin, /home/sv/myfile,
/usr/local/bin) WHERE (EOF > 20000 AND OWNER = dev.user2),
(/etc/rc, /var/x) WHERE MODTIME AFTER JAN 17 1999),
TAPEDISPOSITION NOREWIND, VERIFYTAPE ON;
```

Backup and Restore 2.0 writes the following OSS files to the tape identified by the DEFINE =MYTAPE:

- All files larger than 200 Kilobytes belonging to the user `dev.user2` in the directories:

  ```
  /user/bin
  /home/sv/myfile
  /usr/local/bin
  ```

- All files from the directories `/etc/rc` and `/var/x` modified since January 1, 1999.

You can also backup entire filesets using the fileset mount points. See the *Backup and Restore 2.0 Manual* for more information about backing up OSS files.

Backing Up an OSS Directory Hierarchy to a New Directory

To back up the `olddir` directory hierarchy to the directory `newdir`, enter the following OSS shell commands:

```
mkdir newdir
pax -rw olddir newdir
```

The `mkdir` command creates an empty directory to receive the copied files. The `pax` command has the following form:

`pax -r -w` *`old-pathname new-pathname`*

`-r`

reads an archive file from the standard input file.

`-w`

writes files to the standard output file in the specified archive format (none).

`old-pathname`

is the relative OSS pathname of the directory to be copied; this specification overrides use of the standard input file.

In the example, this directory is named `olddir` and is within the current working directory.

new-pathname

is the relative OSS pathname of the directory to contain the copied files; this specification overrides use of the standard output file.

In the example, this directory is named `newdir` and is the one previously created by the `mkdir` command within the current working directory.

Restoring OSS Files Using Backup and Restore 2.0

You can specify part or all of multiple directory hierarchies and selectively restore files from within them using Backup and Restore 2.0. For example, if you enter the following at a Backup and Restore 2.0 BR> prompt:

```
RESTORE =MYTAPE, OSS ((/user/bin TGT /newdir, /usr/local/bin,
( /home/sv/myfile, TGT /home/sv ), /etc/rc, /var/x ),
TAPEDISPOSITION NOREWIND, VERIFYTAPE ON;
```

you start a restore process that reads selected OSS files from the tape identified by the DEFINE =MYTAPE and restores them to the OSS file system as follows:

- All files from `/user/bin` are restored to the existing directory `/newdir.`
- All files from `/usr/local/bin` are restored to their original parent directory (which is `/usr/local/bin`).
- All files from `/home/sv/myfile` are restored to the directory `/home/sv`, which effectively moves them up a level in the directory tree.
- All files from `/etc/rc` and `/var/x` are restored to their original parent directories.

You must create the TGT directories before entering the command if they do not already exist. You can also restore entire filesets using the fileset mount points. For more information about restoring OSS files, see the *Backup and Restore 2.0 Manual*.

Note. You cannot restore an OSS file that is larger than approximately 2 gigabytes to a system running an RVU that does not include support for OSS large files.

Restoring OSS Files From a pax Archive

The following command restores the content of a `pax` archive to the current directory:

`pax -r -s` *`replstr`* `-f` *`archive-name`*

`-r`

reads an archive file from the standard input file.

`-s replstr`

modifies the file or archive member names named by *pattern* or *file* operands according to the substitution expression *replstr*, using the syntax of the ed utility. For information about the ed utility, see the ed(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. (The ed utility concepts of “address” and “line” are meaningless in the context of the pax utility and must not be used.)

The format of *replstr* is:

/old/new/[gp]

where (as in the ed command) *old* is a basic regular expression, *new* is the replacement string to be inserted in place of matches for the regular expression, and the g and p options specify forms of replacement.

`-f archive-name`

specifies the relative OSS pathname of the archive file to be read.

For example, to read the archive a.pax, with all files rooted in /usr within the archive and extracted relative to the current directory, enter the OSS shell command:

```
pax -r -s ',^//*usr//*,,' -f a.pax
```

In this example, the expression:

```
',^//*usr//*,,'
```

translates into an ed replacement string as follows:

'	The opening and closing single quotation marks (') enclose the substitution expression.
,	The first comma (,) indicates the end of the omitted line address argument. The second and third commas indicate the beginnings of the two omitted arguments.
^	The circumflex (^) means search from the beginning of the line.
/	The first slash (/) means search for the first occurrence of a slash.
/*	The second slash and the asterisk (*) mean search for any number of slashes (0 or more) following the required slash.
usr	usr is the name of the directory to be searched.
/	The third slash means search for the first occurrence of a slash after usr.
/*	The fourth slash and the asterisk mean search for any number of slashes (0 or more) following the required slash.

Note. You cannot restore an OSS file that is larger than approximately 2 gigabytes to a system running an RVU that does not include support for OSS large files.

Creating a pax Backup of OSS Files in the Guardian File System

If the output of a pax backup is less than 8 gigabytes (for H06.06 and later RVUs) or less than 2 gigabytes (for RVUs prior to H06.06), you can create the backup in the Guardian file system. If a fileset backup requires more than the size limit of a pax archive file, you can either:

- Back up sections of the fileset (each section must be smaller than the size limit of a pax archive file).
- Compress the pax archive file. However, a compressed pax archive file can only be restored from an OSS shell; the PINSTALL command cannot correctly process a compressed pax archive.

To determine the size of the backup for an OSS fileset, assume that the backup requires as much space as the fileset itself.

To determine the size of the HOME fileset, for example:

1. Determine the subvolume name of the fileset by obtaining its DEVICELABEL value from the fileset definition. The last five digits in the Guardian subvolume name (which begins with ZYQ) correspond to the DEVICELABEL value.

   For the HOME fileset in this example, the DEVICELABEL is 000001, so this fileset's subvolume name is ZYQ00001.

2. Use the following form of the TACL DSAP command to determine the size of the fileset:

   ```
   DSAP oss_volume_name, BYSUBVOL
   ```

   You need to do this for all volumes that contain files with the subvolume name for the fileset. To determine the list of volumes with subvolumes, you can use the command:

   ```
   FUP SUBVOLS $*.ZYQnnnnn
   ```

   For the HOME fileset example, the commands would be:

   ```
   FUP SUBVOLS $*.ZYQ00001
   ```

   If the output of this SUBVOLS command was:

   ```
   $OSS2
       ZYQ00001
   $OSS1
       ZYQ00001
   $OSS
       ZYQ00001
   ```

   then the corresponding DSAP commands would be:

   ```
   DSAP $OSS2, BYSUBVOL
   DSAP $OSS1, BYSUBVOL
   DSAP $OSS,  BYSUBVOL
   ```

Figure 6-4 on page 6-22 shows the output of one such DSAP command. The total number of pages in a fileset on one disk is shown in the Total Pages column for its subvolume. To determine the total number of bytes, multiply the total pages by 2048 (the number of bytes/page).

Figure 6-4. Output of DSAP command

```
Subvolume              Total    Unused     Dealloc    Large     Min Age    Num
Name          Files    Pages     Pages     Pages       File     Mod,Opn    Exp
...
...
ZYQ00000        2      514         5         0         472       7,  0      2
ZYQ00001     6155   485776     25411        10       34082       0,---     1K+
ZYQ00002     1206   166448      6912         0       39082     116,---     1K+
ZYQ00003      283    16804       851         0        4182     162,---     283
```

In Figure 6-4, the total number of pages for the HOME fileset is shown as 485776. Multiplied by 2048, the total size of the HOME fileset would be 994,869,248 bytes if it is completely contained on this one disk. Because this is smaller than the size limit for pax files, you can back up the HOME fileset in the Guardian file system.

For more information about the DSAP command, see the *Guardian Disk and Tape Utilities Reference Manual.*

To create the backup for the OSS files in the HOME fileset as a file in the Guardian file system, enter the following OSS shell command:

```
pax -X -wvf /G/oss/backup/paxhome /home
```

This command creates the pax archive as the Guardian file $OSS.BACKUP.PAXHOME on the local NonStop server node. The -X flag prevents the pax utility from descending into directories within /home when those directories reside in a different fileset (as indicated by the use of a different device ID from that used for files in /home that are part of the HOME fileset).

If the HOME fileset had been slightly larger than the pax file size limit, the following OSS shell command might also have succeeded:

```
pax -X -wv /home | compress > /G/oss/backup/paxhome
```

The amount of data beyond the limit that can be backed up in this manner depends on the contents of the files within the fileset.

Backing Up OSS Files to Other Expand Nodes

You can perform a remote backup of OSS files indirectly by using the pax utility and writing the backup archive to a directory within the Guardian file system (a directory within/G), where the archive file is a Guardian file. For more information on creating a backup archive in a /G directory, see Creating a pax Backup of OSS Files in the Guardian File System on page 6-21.

Note. You cannot restore an OSS file that is larger than approximately 2 gigabytes to a system running an RVU that does not include support for OSS large files.

Once the pax archive exists in the Guardian file system, you use the T9074 Guardian BACKUP command to copy the archive file to a remote tape. For information about the BACKUP command, see the *Guardian User's Guide*.

OSS Files and Backup/Restore Utilities (T9074)

OSS files should not be backed up or restored with the T9074 Backup/Restore utilities.

The T9074 Guardian BACKUP utility cannot back up OSS files in file mode, only in volume mode. Volume mode BACKUP and T9074 volume mode RESTORE work properly only when all of the following conditions are true:

- All disk volumes containing OSS files can be shut down to allow the backup.
- Related OSS catalog and data disk volumes can be down at the same time.
- The operating system RVU on the system to be restored (the target system) is the same as on the system being backed up (the source system).
- The disk volumes used by OSS on the system to be restored have the same names as the disk volumes to be used on the system to be restored.
- The size of the each disk volume to be restored on the target system is the same as the corresponding disk volume on the source system.

In volume mode, BACKUP backs up only what is on the volume being archived. If you use BACKUP for OSS files, you must make sure that all filesets and all the catalog files for those filesets you are backing up reside entirely on the single disk being archived.

To back up and restore OSS files using BACKUP and RESTORE:

1. Perform all steps described in Backing Up the OSS Environment Using a Version of Backup/Restore on page 6-14.
2. Back up user Guardian files and related information as described in Backing Up User Files on page 6-15.
3. Stop all filesets on the source system as described in Stopping (Unmounting) a Fileset on page 5-13.
4. Stop all disk volumes used for OSS catalogs and OSS data files on the source system.
5. Use the BACKUP VOLUME mode option to backup the OSS catalog disk volume from the source system.
6. Use the BACKUP VOLUME mode option to backup the OSS data file disk volumes from the source system.
7. Use the RESTORE VOLUME mode option to restore the OSS catalog disk volume to the target system.
8. Use the RESTORE VOLUME mode option to restore the OSS data file disk volumes to the target system.

9. Restore any other files backed up in Step 2.
10. Start the OSS Monitor. If necessary, use an OBEY file for the kernel subsystem SCF interface to restore its configuration in the persistence monitor and OBEY files or RESTORE to restore the configuration of the OSS environment.
11. Use RESTORE to restore the storage-pool files. The source and target system DEVICELABEL values must match.
12. Restore the remainder of the OSS environment as described in Restoring NonStop SQL/MP Programs Using Backup/Restore Utilities (T9074) on page 6-24 and Restoring Security Data Used For File Access on page 6-25.
13. Start all filesets.

Restoring User Files

The following subsections describe:

- Restoring NonStop SQL/MP Programs Using Backup/Restore Utilities (T9074) on page 6-24
- Restoring Security Data Used For File Access on page 6-25
- Restoring a pax Archive of OSS Files Directly From the Guardian Environment on page 6-25
- Restoring a Compressed pax Archive of OSS Files From the Guardian File System on page 6-25
- Restoring Files From a pax Archive to the Guardian File System on page 6-26
- Restoring a pax Archive of Guardian Files From a Tape on page 6-26
- Verifying a Restored OSS File Backup on page 6-27

To restore files using the pax utility, the fileset to be restored must first be configured and started using the OSS Monitor.

Restoring NonStop SQL/MP Programs Using Backup/Restore Utilities (T9074)

When an OSS fileset is lost, there are orphan entries in surviving SQL/MP PROGRAMS and USAGES tables. These entries are harmless.

If the OSS fileset is recovered using volume mode RESTORE and the original SQL catalog exists, the orphaned entries again match existing, valid SQL programs. If the OSS fileset is not recovered using volume mode RESTORE, then these SQL objects must be SQL-compiled to reregister them in the PROGRAMS and USAGES table; the orphaned entries remain in the table.

Restoring Security Data Used For File Access

The security database files contain user ID aliases, file-sharing groups, and initial-directory information. This user information can be reconstructed by one of the following methods:

- Use Safeguard OBEY files, previously created as described in Backing Up the OSS Environment Using a Version of Backup/Restore on page 6-14
- Use RESTORE on all of the Guardian files described in Backing Up the OSS Environment Using a Version of Backup/Restore on page 6-14. This approach recovers passwords but at the risk of using files corrupted by being changed during backup. To use RESTORE:

  1. Restore the security database files using the RESTORE MAP NAMES and OPEN options as new files in a temporary location (subvolume).
  2. Use the File Utility Program (FUP) to rename the old security database files and to move the restored copies to the correct subvolume.
  3. Stop and restart Safeguard and all Expand lines to close the old security database files and open the restored ones.

Restoring a pax Archive of OSS Files Directly From the Guardian Environment

The PINSTALL utility that runs in the Guardian environment can restore a `pax` archive backup made as described in Creating a pax Backup of OSS Files in the Guardian File System on page 6-21 directly into the OSS file system.

At a TACL prompt, use PINSTALL to restore the backup to the OSS file system. For example, to restore the $OSS.BACKUP.PAXHOME `pax` archive file, you would enter the following command:

```
PINSTALL -pe -rvf /G/oss/backup/paxhome
```

Restoring a Compressed pax Archive of OSS Files From the Guardian File System

The compressed `pax` archive made as described in Creating a pax Backup of OSS Files in the Guardian File System on page 6-21 can be restored directly into the OSS file system using the following OSS shell command:

```
zcat < /G/oss/backup/paxhome | pax -rv -pe
```

The `zcat` command uncompresses the file and sends the result to the `pax` command, where the `-pe` flags preserve the file permissions.

Restoring Files From a pax Archive to the Guardian File System

Because of the syntactic differences between the names of Guardian and OSS files, the following behaviors can occur when a `pax` archive member is restored to a Guardian file system:

- An OSS filename might contain characters that are illegal in Guardian filenames. As a result, the archive member cannot be created on the Guardian target and the restore operation fails. For example, if the archived file were named `ca$h/.profile`, then an attempt to restore it to `/G/oss` fails because of the embedded dollar sign character.
- In the name-transformation process, OSS names that are longer than eight characters are truncated to the first valid eight characters. For example, the OSS pathname `abcde.fghi` is transformed to the Guardian name ABCDEFGH. This can cause confusion and make identification of files difficult. Filenames that are similar in one environment might be transformed to the same filename in the other environment, which might result in the overwriting of previously restored files.

Guardian files can be specified as pathnames in a `/G` directory. Files can be restored within `/G` directories, but existing Guardian files are overwritten only if you use the `-W clobber` flag. The Guardian file attributes are not preserved.

Guardian files are restored as unstructured Guardian files with the file code 0. Only Guardian files with file codes that are supported by the OSS environment are processed.

Restoring a pax Archive of Guardian Files From a Tape

To restore files from the tape that is mounted on $TAPE to the Guardian target $VOL.SUBVOL, extracting only files whose names end with `.c` and overwriting any existing Guardian files with the same name, enter the following OSS shell commands:

```
cd /G/vol/subvol
pax -rv -f /G/tape -W clobber *.c
```

The `cd` command selects the Guardian subvolume that you want to restore. The `pax` command has the following form:

```
pax -r -v -f archive-name -W clobber match-pattern
```

`-r`

reads an archive file from the standard input file.

`-v`

writes archive member pathnames to the standard error file.

`-f` *`archive-name`*

specifies the pathname of the input archive, overriding the default standard input file. Guardian tape devices can be specified with the `/G` naming convention (for example, `/G/tape`).

`-W clobber` *`match-pattern`*

is a HP extension. This flag allows selected files from an archive to be stored on a Guardian target and to overwrite any preexisting Guardian target file with the same name. Users must be aware that the files are restored as unstructured files and that Guardian file attributes might not be preserved.

△ **Caution.** Understand the potential danger of destroying data files (described in Guardian Filename Transformation in the DESCRIPTION section of the `pax(1)` reference page) before using the `-W clobber` flag to restore files to a Guardian target.

The variable *`match-pattern`* is a normal wildcard matching pattern for an OSS filename. In the example, any file with a name ending in the characters `.c` is matched and therefore overwritten.

Note. An OSS file that is larger than approximately 2 gigabytes cannot be restored to a system running an RVU that does not include support for OSS large files.

Verifying a Restored OSS File Backup

The `dircmp` utility reads two directories, compares their contents, and writes the results to the standard output file. Use `dircmp` to determine whether the contents of two directories differ in any way, such as when you restore backed-up files to a new location and want to be sure the contents are copied correctly.

The `dircmp` utility compares the filenames in each directory. When the same filename appears in both, `dircmp` compares the contents of the two files. In the output, `dircmp` first lists the files unique to each directory, then lists the files that have identical names but different contents. By default, `dircmp` also lists files that have both identical names and identical contents.

For more information about the `dircmp` command, see the `dircmp(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Redirecting OSS Standard Files

OSS processes use a different set of default files and a different file format than Guardian processes do. An OSS process always has three unstructured standard files, usually referred to in UNIX documentation as `stdin`, `stdout`, and `stderr`.

In a UNIX environment, these three files are usually associated with the user's terminal: standard input is read from the terminal keyboard, standard output is sent to the terminal display, and standard error is an output logging mechanism that is usually

also written to the terminal display. Data read or written through these files can be redirected using shell redirection specifications to other processes or to regular (disk) files. On a UNIX system, the pseudo-TTY feature `ptty` can be used as a source or sink for such data redirection; although the OSS environment does not have a pseudo-TTY feature, OSSTTY can do similar data redirection.

A Guardian process has three standard files, comparable to the OSS files mentioned above, that normally possess the structure of an EDIT file:

- STDIN, usually called the IN file in commands entered from a TACL prompt
- STDOUT, usually called the OUT file in commands entered from a TACL prompt
- STDERR, usually called the TERM file in commands entered from a TACL prompt

By default, these files are associated with the user's home terminal (HOMETERM). However, the command that runs a Guardian process can redirect those files to Guardian processes or files.

An OSS user normally launches an OSS shell by logging in through Telserv, most commonly by starting a TACL session and entering the OSH command. That access method leaves the OSS standard files associated with the user's terminal but allows no way for those files to be redirected to Guardian processes or files.

The OSSTTY utility, described in OSSTTY on page C-1, removes that restriction. To an OSS process, OSSTTY appears to be a set of three terminal devices (one for each OSS standard file); to a Guardian process or the Guardian file system, the OSS process appears to be a Guardian file-system object. OSSTTY can be started by an individual user through the OSH command or directly by an OSS administrator.

OSSTTY can be used to redirect one or more OSS standard files to Guardian EDIT files or Guardian processes. Figure 6-5 on page 6-29 illustrates use of the OSS shell when only the OSS standard output and standard error files are redirected, but input is still accepted from the Telserv terminal that enters the following command; the process name $ZAAA is assigned by the NonStop operating system when OSSTTY is started:

```
OSH -osstty / OUT $DATA.OSSLOG.EDITFIL, TERM $VHS /
```

Figure 6-5. Redirecting Selected OSS Standard Files

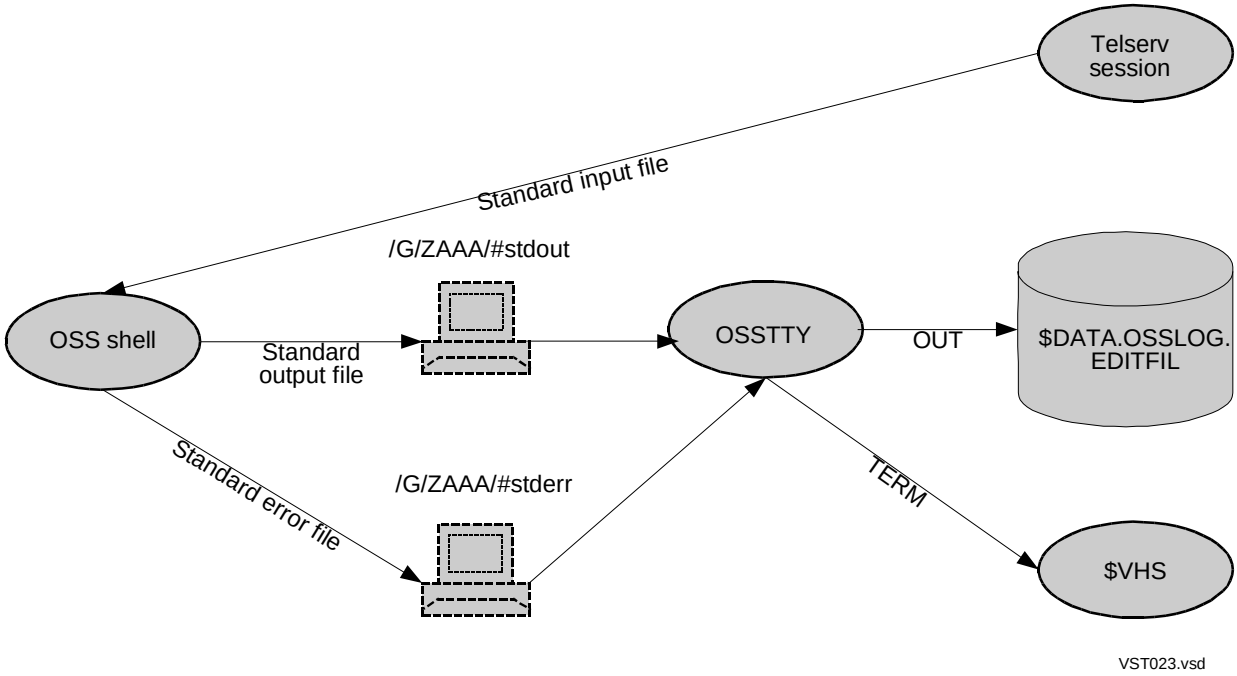


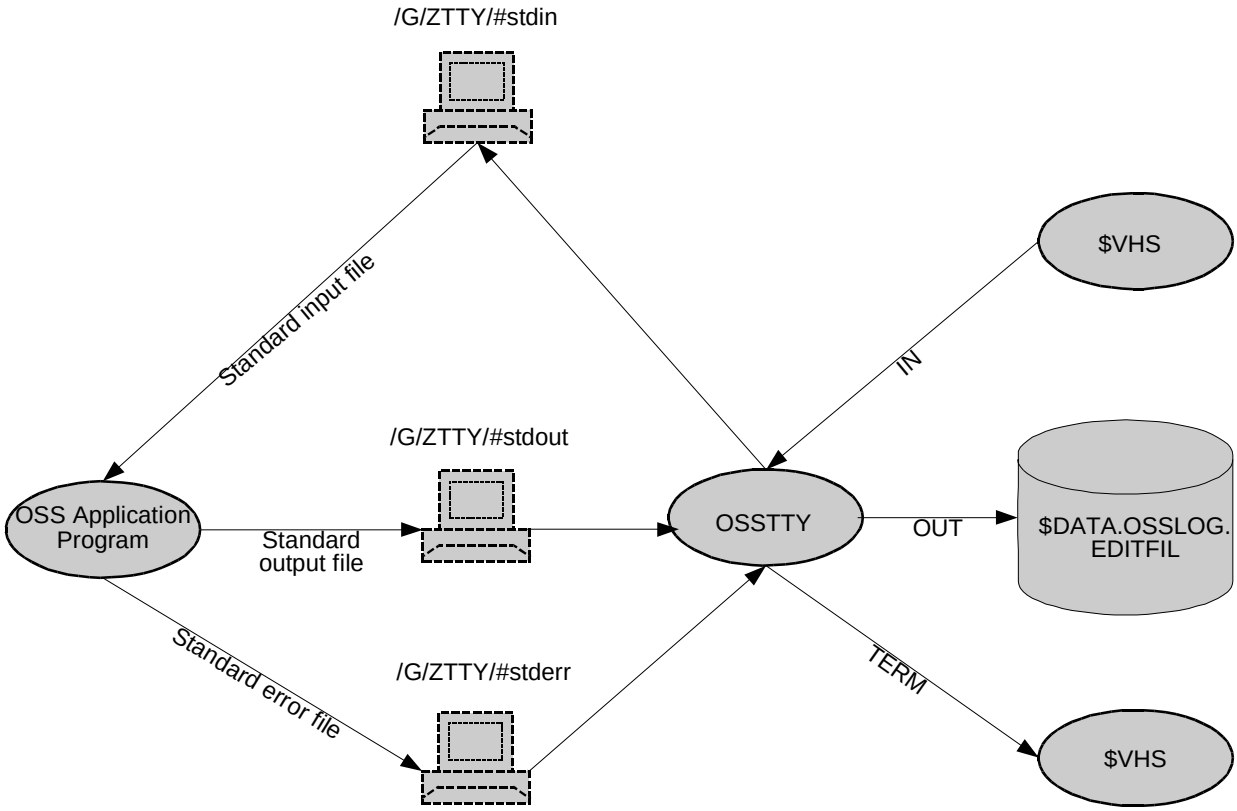
Figure 6-6 on page 6-30 illustrates the behavior when OSSTTY is run as a server named $ZTTY before starting an application program explicitly designed to use it, such as `mysample3` in the following command set:

```
RUN OSSTTY / NAME $ZTTY, IN $VHS,OUT $DATA.OSSLOG.EDITFIL,
            TERM $VHS, NOWAIT / -server

OSH -p "/usr/mysample3"
```

For information about using OSSTTY through the OSH command, see the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. For an example of a TACL script using OSSTTY, see the *Open System Services Programmer's Guide*.

Figure 6-6. Redirecting All OSS Standard Files



Controlling the Maximum Number of Files

The fileset catalog determines the number of regular files and directories (the number of inodes) a fileset can contain. The fileset catalog is kept in the PXINODE file described in OSS Catalog Files on page 3-7 and Changing the Fileset Catalog on page 5-21.

When a fileset catalog reaches its configured maximum number of inodes, application program errors might occur unless additional inode entries are allowed in the catalog. The fileset catalog can reach its maximum even when space remains for new file creation on disk volumes in the storage pool.

A catalog's approximate maximum number of inodes for files is determined by the fileset MAXINODES attribute. MAXINODE defaults to 500,000 inodes for a new fileset unless the SCF ADD FILESET Command was used to set it. For a fileset created under a version of the OSS Monitor prior to G11, the MAXINODES value is determined by applying a formula to the number of inodes currently in use; the formula usually allows at least a 10 percent increase in the number of inodes before the maximum is reached.

The number of inodes in a catalog also affects fileset recovery time. The more inodes in a fileset, the more time that fileset recovery takes when the SCF DIAGNOSE

FILESET Command is used. A fileset that has a only a few inodes (enforced by using a small MAXINODES value) can be recovered faster than a fileset that has a large number of inodes.

To change the approximate maximum number of inodes permissible in a fileset that is not started:

1. Use the SCF INFO FILESET Command with the DETAIL option to determine the configured value for the MAXINODES attribute.
2. Use the SCF ALTER FILESET Command MAXINODES option to change that value.

The change takes affect as soon as the fileset is started.

To change the approximate maximum number of inodes permissible in a fileset that is already started:

1. Use the STATUS FILESET Command with the DETAIL option to determine the number of inodes in use. If the value in use is not the value you want used, then:
   a. Use the SCF INFO FILESET Command to determine the configured value for the MAXINODES attribute.
   b. If the configured value is not the value you want used, change it using the SCF ALTER FILESET Command MAXINODES option.
2. Use the SCF CONTROL FILESET Command with the SYNC option to apply the change to the started fileset.

7 Managing Terminal Access

This section briefly describes the concepts and methods available for providing terminal users with access to the Open System Services (OSS) environment.

How Users Gain Access to the OSS Environment

A user gains access to the OSS environment through a server process. The most commonly used server subsystems are Telserv and the file transfer protocol (FTP) server. Other servers, such as the iTP WebServer `httpd` process, are beyond the scope of this guide.

Both Telserv and the FTP server authenticate the user's login information against the user definitions configured through either a third-party product or the Safeguard subsystem. (Section 8, Managing Security, describes configuration of user definitions.)

Telserv provides access in the following ways:

- Indirectly, when the user selects the TACL service, logs in to the Guardian environment from a TACL prompt, and then enters the OSH command. (See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual.*)
- Directly, when the user logs in to a direct service that invokes the OSH command to start an OSS shell.
- Directly, when the user logs in to a direct service that uses the OSH command to start any executable program correctly configured through the security subsystem, and through the Telserv SCF product module.

OSS users can optionally redirect one or more OSS standard files using either the OSH command and a private copy of the OSSTTY utility, or through an OSSTTY server. OSSTTY appears as a set of terminal devices to an OSS process and behaves as if it were a Telserv terminal session with three windows. OSS standard files that are not redirected remain associated with the terminal device provided by Telserv. See Redirecting OSS Standard Files on page 6-27 and OSSTTY on page C-1 for more information.

The FTP server provides access initially to either the Guardian environment or to the OSS environment. The choice of an initial environment depends on:

- Whether the user is anonymous and which anonymous login is used.

- The way the initial working directory is configured for a user definition that is not anonymous. (The FTP user can use an FTP client `quote` command to gain access to the opposite environment.)

See Section 8, Managing Security, for information about configuring initial working directories.

Configuring Telserv Access

Configuring Telserv access is fully described in the *Telserv Manual*. The following subsections provide a brief review of the options available to you and the effects of each option on user access.

Configuring the Telserv TACL Service

The default services for a Telserv session (window) provide the user with two choices. The user can either select “TACL” to obtain a TACL prompt in the Guardian environment or select “EXIT” and be disconnected from the NonStop node.

Using only default services, Figure 7-1 shows what a Telserv login usually looks like.

Figure 7-1. Telserv Login Using Default Telserv Services

```
Connecting..
.Connected to 123.456.789.111.
Escape character is '^]'.
WELCOME TO node1.subdom1.corporate.net [PORT $ZTCF0 #23
WINDOW $ZTNF0.#PTMNBVY]
TELSERV - T9553D40 - (29JUN2001) - (IPMADI)
Available Services:
TACL     EXIT
Enter Choice> TACL
TACL 1> logon guest.user1
Password:
Welcome to \NODE1.
The time is  8:37:07 am 08/21/2002.
Last Logon:  21 AUG 2002, 08:34
Last Unsuccessful Attempt: 29 APR 2002, 06:19  Total Failures: 13
TACL (T9205D46 - 20SEP2001), Operating System G06, Release G06.15
COPYRIGHT COMPAQ COMPUTER CORPORATION 1985,1987-2000
CPU 7, process has no backup
August 21, 2002  8:37:08
(Invoking $SYSTEM.SYSTEM.TACLLOCL)
(Invoking $GUEST.TACLCSTM)
Current volume is $GUEST.USER1
1> OSH
/home/user1:
```

As this example shows, the user can log in to OSS only by first logging in to the Guardian environment and then issuing the OSH command to access the OSS environment interactively.

Note. The OSS shell inherits any TACL DEFINEs that are in effect when the Telserv service is started. Telserv should be started with the appropriate DEFINEs for the OSS shell.

Configuring a Telserv Direct Service

A Telserv process can be configured to provide login either directly to the OSS shell or directly to the command interpreter code of any executable program accessible through the OSS file system.

Logging In to the Shell Directly

The services feature of Telserv provides OSS users with the ability to log in directly to the OSS shell, bypassing the TACL prompt.

The services feature is configured through the ADD SERVICE command in the Telserv SCF product module. This command is described in the *Telserv Manual*.

Figure 7-2 shows what a Telserv login looks like when you use the services feature to provide direct login to the OSS shell.

Figure 7-2. Telserv Login Using an OSS Shell Direct Login Service

```
Connecting...
Connected to node1.subdom1.corporate.net.
Escape character is '^]'.
WELCOME TO node1.subdom1.corporate.net [PORT $ZTCF0 #23
WINDOW $ZTNF0.#PTMNBVY]
TELSERV - T9553D40 - (29JUN2001) - (IPMADI)
Available Services:
OSS     TACL     EXIT
Enter Choice> OSS
login:  guest.user2
Password:
/G/SYSTEM/SYSTEM:
```

In this example, the OSS service allows the user to log in directly to the OSS environment. The following SCF command example shows how this service was configured:

```
ADD SERVICE $ZTNT.OSS, TYPE CONVERSATION, PRI 150, &
   SWAP $DATA1, CPU 2, PROGRAM $SYSTEM.SYSTEM.OSH, &
   SUBTYPE DYNAMIC, ACCESS SYSTEM
```

Considerations:

- Specify the SCF ADD SERVICE command with ACCESS set to SYSTEM. This access requires users to be prompted for login authorization, as shown in Figure 7-2.
- Do not specify a PARAM value or OWNER value in the ADD SERVICE command. The default value used by the OSH program when the PARAM option does not appear places the user in the OSS shell (`/bin/sh`).
- As shown in Figure 7-2 on page 7-3, the service name OSS is displayed on the screen. You can suppress this display by using the DISPLAY OFF option of the ADD SERVICE command.

  Use this option if you do not want availability of the service known to all users.

- HP recommends that you do not configure resilient windows for OSS logins. The default configuration for a service is to not have resilient windows, so simply omit the RESILIENT option when defining the service.
- Do not rely on the default value for the OSS priority in the SCF ADD SERVICE command; instead, set the priority to a value below 160.

Logging In to Another Program

The services feature of Telserv provides users with the ability to log in directly to any program accessible through the OSS file system, bypassing the TACL prompt and the OSS shell.

The services feature is configured through the ADD SERVICE command in the Telserv SCF product module. This command is described in the *Telserv Manual*. Additional information about this command is provided in the softdoc for the T9553D30ABX software product revision (SPR).

Figure 7-3 shows what a Telserv login looks like when you use the services feature to provide direct program login.

In this example, the OSH3 service allows the user to log in directly to a program that provides access to the OSS environment. The following SCF command example shows how this service was configured:

```
ADD SERVICE $ZTNT.OSH3, TYPE CONVERSATION, PRI 150, &
   SWAP $DATA1, CPU 2, PROGRAM $SYSTEM.SYSTEM.OSH, &
   PARAM "-p /bin/my_app", &
   SUBTYPE DYNAMIC, OWNER "SUPER.SUPER", ACCESS SYSTEM
```

Figure 7-3. Telserv Login Using a Site-Written Direct Login Service

```
Connecting...
Connected to node1.subdom1.corporate.net.
Escape character is '^]'.
WELCOME TO node1.subdom1.corporate.net [PORT $ZTCF0 #23
WINDOW $ZTNF0.#PTMNBVY]
TELSERV - T9553D40 - (29JUN2001) - (IPMADI)
Available Services:
OSH3     TACL     EXIT
Enter Choice> OSH3
login:  guest.any
Password:
Hello:
```

Considerations:

- Specify the SCF ADD SERVICE command with ACCESS set to SYSTEM. This access requires users to be prompted for login authorization, as shown in Figure 7-3 on page 7-4.
- The PARAM value specified in the ADD SERVICE command is passed to the program specified by the PROGRAM option. Telserv uses the PARAM value instead of the INITIAL-PROGRAM value from either a third-party product or

Safeguard user definition, as described in How Users Gain Access to the OSS Environment on page 8-10.

The default value used by the OSH program when the PARAM option does not appear places the user in the OSS shell (`/bin/sh`). Therefore, you must explicitly define a value for the PARAM option to start any program other than the OSS shell.

In the PARAM value definition, the `-p` flag indicates that a program file is to be executed; the `-p` flag is followed by the full OSS pathname of the application. (The quotation marks around the PARAM value are required.)

- The initial working directory of the user is set to that configured for the user name specified by the OWNER option. The INITIAL-DIRECTORY value configured for the user through either a third-party product or SAFECOM is not used.

  In the SCF ADD SERVICE command example on page 7-4, the super ID is configured without an initial working directory, so the user has the default subvolume for the super ID (`/G/SYSTEM/SYSTEM`) as an initial working directory. This configuration is probably not ideal.

  As an alternative, the following configuration could be used:

```
ADD SERVICE $ZTNT.OSH3, TYPE CONVERSATION, PRI 150, &
   SWAP $DATA1, CPU 2, PROGRAM $SYSTEM.SYSTEM.OSH, &
   PARAM "-p /bin/my_app", &
   SUBTYPE DYNAMIC, OWNER "GUEST.USER0", ACCESS SYSTEM
```

  where the user name GUEST.USER0 is assigned a more appropriate default volume and subvolume for an OSS initial working directory.

- As shown in Figure 7-3 on page 7-4, the service name OSH3 is displayed on the screen. You can suppress this display by using the DISPLAY OFF option of the ADD SERVICE command.

  Use this option if you do not want availability of the service known to all users.

- HP recommends that you do not configure resilient windows for direct program logins. The default configuration for a service is to not have resilient windows, so simply omit the RESILIENT option when defining the service.

Configuring FTP Access

Configuration of FTP server access is fully described in the *TCP/IP Configuration and Management Manual* and the *TCP/IPv6 Configuration and Management Manual*.

Whether an anonymous user of FTP initially enters the Guardian environment or the OSS environment depends on which anonymous login user name is used and whether an OSS initial directory is configured for that user name. Section 8, Managing Security, shows how to set up an OSS user for anonymous FTP access.

Additional information on defining the initial working directory for FTP server access can be found in Section 8, Managing Security.

OSS FTP client users should have the full OSS pathname (`/usr/ucb`) of the FTP client program added to the `PATH` environment variable in their `.profile` files. The default value for `PATH` does not include the FTP pathname.

Alternatively, you can add `/usr/ucb` to the `/etc/profile` file for all users (as is done in the `/etc/profile.sample` file supplied by HP). Section 8, Managing Security, describes how to set up the `/etc/profile` file.

8 Managing Security

This section covers:

Common and Unique Characteristics of OSS and UNIX Security

Basic file security is the same for the OSS environment as on a UNIX system. Files are accessed according to a file mode and access permissions, as described in the *Open System Services User's Guide*.

Certain differences might require you to code the security-management portions of a shell script in a manner specific to the OSS environment. If you are experienced in UNIX security administration, review the following topics before proceeding with the rest of this section:

Administrative Files and Directories

Most of the directories and files with security considerations on UNIX systems are absent from the OSS environment. For example:

- OSS user and group administration does not use any of the following files or directories in the `/etc` directory, which can be the target of UNIX security intruders:

  ```
  groups
  passwd
  security
  shadow
  ```

- C functions provide access to information needed from the security database. However, the database files themselves are not available in the OSS file system.

- OSS administration of device access does not use files in the `/dev` directory that are available on some UNIX systems, such as:

  ```
  console
  cua*
  fd
  kmem or mem
  ```

```
modem
ttyda or ttydfa
ttys0 through ttys9
```

These files also can be the target of UNIX security intruders.

- The OSS environment does not use the following files and directories sometimes found on UNIX systems:
  - In `/etc`:

    ```
    aliases
    dfs/dfstab
    exports
    ftpusers
    hosts.lpd
    mail/aliases or sendmail/aliases
    rc, rc*, or rc?.d
    shells
    syslog.conf
    system
    ttys
    tftpdaccess.cf
    ```

  - In any of various directories:

    ```
    .plan
    .project
    sendmail.cf
    ```

  - The `/home/quotas` file, used to establish user disk space quotas

  Some of these files and directories also provide mechanisms that intruders can use to compromise UNIX system security or integrity.

- The OSS file system does not provide the following UNIX features that are sometimes used to impose security on a system:
  - Immutable files (other than those secured read-only through normal permissions) or append-only files
  - Partitions within filesets
- The OSS implementation of object security does not conform to POSIX.6 draft 12, IEEE Standard 1003-1e, or IEEE Standard 1003-2c. In particular, the OSS environment does not provide access control list function calls.
- OSS file-auditing mechanisms and policies are implemented through the Guardian environment Safeguard product instead of through such UNIX commands or utilities as:

  ```
  /etc/reboot, /etc/shutdown, or /etc/syslog
  passwd
  ```

Access to OSS auditing logs occurs through the Safeguard audit reduction tool (SAFEART) program. UNIX directories and files such as the following are not provided and therefore do not require monitoring:

- In /var:

  aculog

  adm/acct, adm/lastlog, adm/loginlog, adm/messages, adm/pacct, adm/utmp or adm/utmpx, adm/wwtmp or adm/wtmpx

  spool/atrun or spool/ftp

  sulog, vold.log, or xferlog

- In /usr:

  adm/wtmp (FTP login log)
  etc/rpc.mountd (NFS access log)
  lib/aliases
  local/etc/http/logs/access_log

- The OSS environment uses the following files and directories that might be found on UNIX systems:

  - In /dev:

    null
    tty

  - In /etc:

    hosts and hosts.equiv
    inetd.conf and install_obsolete
    magic
    named.boot
    printcap and printcap.sample
    profile and profile.sample
    protocols
    resolv.conf
    services
    termcap

  - In /var:

    /adm/cron/.proto, /adm/cron/cron.allow, /adm/cron/cron.deny, and /adm/cron/queuedefs

    preserve

    spool/cron and spool/pcnfs

    tmp

- In the user's home directory:

  ```
  .netrc
  .rhosts
  ```

  Take normal security precautions with these files and directories.

Administrative Tools

On many UNIX systems, certain commands and utilities either exist for security administration or have security considerations. Most such commands and utilities are absent from the OSS environment. The OSS environment does not provide:

- The following shell commands and utilities (although some of these tools are available in versions ported to Open System Services):

  `acctcom`
  `accton`
  `chroot`
  `chsh`
  `crypt`
  `cu`
  `des`
  `/etc/reboot`, `/etc/shutdown`, or `/etc/syslog`
  `finger` (a TACL FINGER command exists for the Guardian environment)
  `ftpd`
  `fsck`
  `fsirand`
  `getstats`
  `identd`
  `in.named`
  `last`
  `lastcomm`
  `login`
  `lpd`
  `makekey`
  `mail`
  `md5`
  `mount`
  `netstat`
  `newaliases`
  `nfsd`
  `ntpd`
  `passwd`
  `ph`
  `pgp`
  `quot`
  `rcp`
  `rcs` or `sccs`

`rdist`
`rexd`
`rlogin` or `rlogind`
`routed` or `gated`
`sendmail`
`share`
`telnet` (a TACL TELNET command exists for the Guardian environment)
`tftpd`
`tip`
`ulimit`
`usermod`
`/usr/etc/exportfs` or `/usr/etc/showmount`
`vipw`

- The GNU suite of utilities
- The following environment variables:

  `maxuproc`
  `nfs_portmon`

- The `/bin/sh -r` restricted shell option
- For the `inetd` program:
  - `-t` tracing option
  - Service of `systat` for port 11

Some of these commands and utilities can provide mechanisms that intruders can use to compromise system security or integrity.

Guardian environment equivalents of `ftpd` and `nfsd` provide access to the OSS environment and security management of related activities occurs through Guardian environment tools. See the `ftpserver(7)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for information about the equivalent of the File Transfer Protocol (FTP) demon; information about the administration of anonymous FTP users can be found in Configuring FTP Access on page 7-5. See the manual set available for the Network File System (NFS) for Open System Services product.

The OSS environment provides the following tools and utilities:

`at`, `atq`, and `atrm`
`cron` and `crontab`
`df` and `du`
`logger`
`named` and `named-xfer`
`rsh`
`sum`
`wall`
`xargs`

The OSS shell supports the IFS environment variable but appropriately clears its settings.

The OSS inetd program provides the following default services:

```
time
echo
discard
daytime
chargen
```

Users and Groups

The OSS environment does not provide:

- The following predefined or generic user names (accounts):

```
agent
bin
daemon
demo
finger
ftp
games
guest
help
ingres
lp
mail
maint
manager
news
nobody
nuucp
.open
root
system
telnet
toor
uucp
visitor
who
```

- The following UNIX predefined or generic group names:

```
wheel
```

Many of these user and group names can provide mechanisms that intruders can use to compromise UNIX system security or integrity.

The OSS environment does not provide common UNIX default user names and user IDs unless they are explicitly created by a site administrator. However, equivalent OSS user names and user IDs do exist. For example, the privileges normally associated with the UNIX user name `root` and the user ID of 0 exist for the OSS user ID (UID) of 65535 (the super ID), which is usually the user SUPER.SUPER.

The OSS environment is incompatible with the following UNIX user and group conventions:

- The UNIX super ID has a UNIX UID of 0. The OSS user with an OSS user ID (scalar view of the NonStop operating system user ID) of 0 is NULL.NULL by default.
- The UNIX super group has a UNIX GID of 0. The OSS group with an OSS group ID (group number from the structured view of the NonStop operating system user ID) of 0 is NULL by default.
- Single UNIX user names such as `root` are always login names. The OSS user name is the complete NonStop operating system user name and group name pair (for example, USER.FREDA) unless an alias has been created for the underlying user ID (for example, when `freda` is an alias of the user ID for USER.FREDA).

The following OSS environment conventions are equivalent to UNIX user and group conventions:

- The super ID login name, with an OSS user ID (scalar view of the NonStop operating system user ID) of 65535, is the same as the UNIX user name `root` with a UNIX UID of 0.
- The super group, with an OSS group ID (group number from the structured view of the NonStop operating system user ID) of 255, is the same as the UNIX group name `wheel` with a UNIX GID of 0.
- Using `root` as an alias of the OSS user ID 65535 (which usually has the login name SUPER.SUPER) is the same as using `root` for the UNIX user name of the super ID.
- Using `wheel` as an alias for the OSS group ID 255 (the specially privileged super group, usually with the group name SUPER) is the same as using `wheel` for the UNIX group name of the trusted administrator group.

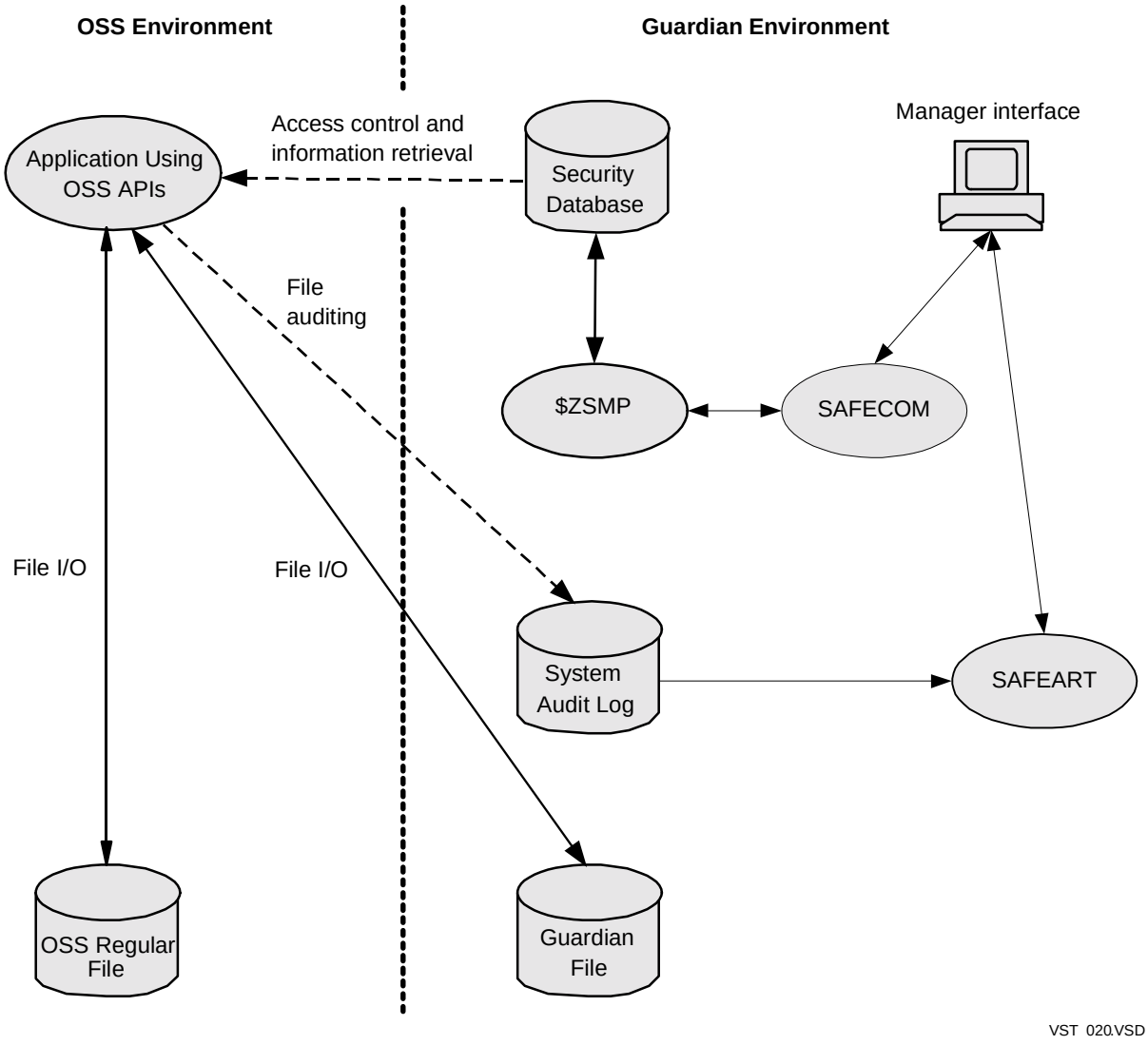
OSS user and group administration occurs through tools in the Guardian environment, such as the Safeguard command interpreter (SAFECOM) program, or through third-party software.

There is only one situation where a site should have more than one user name with the same user ID: when there are multiple administrators of the same group (for example, `news`). Each user name with the same user ID must have its own unique password so that login can be properly audited.

Components of OSS Security Management

Figure 8-1 on page 8-8 illustrates the major components and interfaces of OSS security management.

Figure 8-1. Major Components and Interfaces for OSS Security Management



See the *Security Management Guide* for an overview of both Guardian and OSS security. See the *Safeguard Audit Service Manual* for a description of the SAFEART program and the information logged for audited OSS files.

The commercial text *Practical UNIX & Internet Security* listed in UNIX Security on page xix contains many suggestions for securing file systems. Those suggestions include which actions to avoid in best-practice system administration.

Managing Users and Groups

This subsection discusses the following user and user-group management concepts that are relevant to the OSS environment:

Hints and suggestions for using these concepts are given in Hints and Suggestions on page 8-19.

Differences Between OSS and UNIX User and User-Group Configuration

You configure a NonStop operating system user for access to the OSS environment through a fully licensed version of the Safeguard product. Whether you license Safeguard determines the security and user configuration features available for managing the OSS environment. If you use the optional Safeguard product, you can:

- Configure a user name with an administrative group name and a member name
- Configure an alias for a user name, for Guardian or OSS environment login and use
- Configure an initial working directory OSS pathname for a user ID when the OSH command is used to start an OSS shell
- Configure the initial directory and interface when a user accesses the FTP server
- Configure an initial program to execute when access occurs for a specific user definition, if a server supports that feature
- Define file-sharing user groups for OSS access with group numbers above 255
- Perform auditing for OSS files and the OSS environment's filesets
- Control access to the `/E` directory using the REMOTEPASSWORD attribute
- Use the OSS EasySetup product

This guide illustrates the use of the Safeguard product in the Guardian environment.

You create a user definition with the SAFECOM ADD USER command, and you create a user group definition with the SAFECOM ADD GROUP command.

A NonStop operating system user can have alternate user names, called aliases. Most of the attributes of an alias can differ from those of its underlying user definition. You create an alias with the SAFECOM ADD ALIAS command.

A Safeguard user group is either an administrative group or a file-sharing group. An administrative group is used to manage user access; a file-sharing group is used to manage file access.

A NonStop operating system user belongs to a primary group and can belong to more than one file-sharing group. File-sharing groups other than the primary group are called supplementary groups in POSIX terminology, although that term does not appear in Safeguard manuals. All groups configured for the user make up the user's group list.

By default, the primary group for a new user is the administrative group of the user. The primary group should not be an administrative group and can be changed to any other group in the user's group list.

You should also configure an OSS user's initial working directory when you configure the user. You configure the user's initial working directory with the SAFECOM ADD USER, ALTER USER, ADD ALIAS, and ALTER ALIAS commands.

You cannot:

- Move user or group membership definitions directly from a UNIX system into the OSS environment. If you want to duplicate your UNIX system user and group definitions, you must recreate them through the Safeguard subsystem.
- Copy an `/etc/group` file to define user groups for the OSS environment. OSS security processing does not use an `/etc/group` file.
- Copy an `/etc/passwd` file to define users for the OSS environment. OSS security processing does not use an `/etc/passwd` file.
- Copy an `/etc/ftpusers` file to bar specific users from FTP access to the OSS and Guardian file systems.
- Use UNIX Network Information Service (NIS) "yellow pages" to define users for the OSS environment. OSS security processing does not currently support NIS.

How Users Gain Access to the OSS Environment

A user gains access to the OSS environment through a server process. The most commonly used server subsystems are Telserv and the file transfer protocol (FTP) server. Other servers, such as the iTP WebServer `httpd` process, are beyond the scope of this guide.

Both Telserv and the FTP server authenticate the user's login information against the user definitions configured through the Safeguard subsystem.

Note. The EDIT file $SYSTEM.ZTCPIP.FTPUSERS can be used to disallow access to FTP by valid users of other subsystems. When a user name appears in the FTPUSERS file, FTP rejects access without authenticating the user definition. This control mechanism is similar to that provided on a UNIX system by the `/etc/ftpusers` file.

When you configure a user, make sure that the FTPUSERS file does not conflict with your intent. For example, access by the FTP user `anonymous` is disallowed if the Guardian user NULL.FTP or the OSS user aliases `anonymous` or `ftp` are listed in the FTPUSERS file.

See the *TCP/IP Applications and Utilities User Guide* for more information about the use of FTPUSERS.

Telserv provides access in the following ways:

- Indirectly, when the user selects the TACL service, logs in to the Guardian environment from a TACL prompt, and then enters the OSH command. (See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual.*)
- Directly, when the user logs in to a direct service that invokes the OSH command to start an OSS shell.
- Directly, when the user logs in to a direct service that uses the OSH command to invoke any executable program correctly configured through the Safeguard subsystem and the Telserv SCF product module.

The initial working directory for the user is determined by the way the user definition is configured and by the Telserv service that the user selects.

The FTP server provides access in the following ways:

- Directly to the Guardian environment, when either of the following is true:
  - No initial working directory is configured for the user of the FTP client.
  - An initial working directory with a name that begins with the characters `/G/` is configured for the user of the FTP client.

  The FTP user must use an FTP client `quote OSS` command to gain access to the OSS environment. (See the `ftp(1)` and `ftpserver(7)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual*. The use of the `quote OSS` command is also discussed in the *TCP/IP Applications and Utilities User Guide*.)
- Directly to the OSS environment, when an initial working directory in the OSS file system is configured for the user of the FTP client. (The FTP user can use an FTP client `quote guardian` command to gain access to the Guardian environment.)
- Anonymously, providing direct access to either the OSS environment or the Guardian environment. An anonymous FTP user cannot use an FTP client

`quote OSS` or `quote GUARDIAN` command to gain access to the other environment.

User and User-Group Attributes

All NonStop operating system users can use either the Guardian environment or the OSS environment. However, additional attributes should be configured for OSS environment users.

In the Guardian environment as well as the OSS environment, users and user groups can have the following attributes:

- A user definition can belong to up to 32 user groups.
  - An administrative user group has a group number in the range 0 through 255.
  - A file-sharing user group has a group number in the range 0 through 65535.
  - A user definition can have a primary user group that is different from its administrative user group.
- Each user definition has both a user name and a user ID.
  - A user name (sometimes called a logon name) has the form *group-name.member-name*.
  - A user definition has a user ID that is unique within the NonStop node. The user ID is usually represented:
    - In the Guardian environment as the structured value pair *group-number, member-number*.

      The NonStop operating system predefines the user ID of the super ID as (255,255).
    - In the OSS environment as a unique scalar value in the range 0 through 65535 called the UID.

      The UID is equal to *member-number* + (256 * *group-number*).

      The NonStop operating system predefines the UID of the super ID as 65535.

      The UID value 65535 always has appropriate privileges in the OSS environment.
- A user definition can have aliases up to 32 characters long that can be used for login to the system. An alias is created using the SAFECOM ADD ALIAS command and can have different attributes (other than the UID) from those of the underlying user definition.
- User groups can exist independently of user definitions. Therefore:
  - A user group can be created before any users are added to it.

  - A user group can continue to exist after its last user has been removed from it.

- There can be up to 65,535 user groups, including 256 administrative groups.
- There is no limit on the number of file-sharing members (users) in a user group. An administrative group can have up to 256 users for user management and administration purposes; an administrative group can have additional file-sharing members administered through a different administrative group.
- User groups with group numbers above 255 are file-sharing user groups rather than administrative user groups.

In the OSS environment, users can have the following additional attributes (which are ignored in the Guardian environment):

- Initial working directory (if none is defined, the default value is null). Some servers do not use this attribute value; instead, they provide alternative attributes.
- Initial program (if none is defined, the default value is null). Some servers do not use this attribute value; instead, they provide alternative attributes.

For detailed information about user groups, user definitions, aliases, the Safeguard subsystem, and SAFECOM, see the *Safeguard Administrator's Manual* and the *Safeguard Reference Manual.* For detailed information about the equivalent concepts and facilities in a third-party product, see the appropriate manual.

Assigning an Initial Working Directory

An initial working directory is the location in the OSS file system where a user is placed upon entry to the OSS environment. An initial working directory is also known as a home directory.

If you log in, remain in the same directory, and refer to a file without specifically identifying any directory, Open System Services assumes that the file belongs to the initial working directory. This concept is similar to the concept of the default volume and subvolume in the Guardian environment.

You should provide each OSS user with an initial working directory in the OSS file system. The initial working directory can be assigned in several ways:

- You can use the Safeguard subsystem.
- For Telserv indirect users only, you can use a TACLCSTM file.

HP recommends that you use the Safeguard subsystem and its INITIAL-DIRECTORY attribute.

If you do not provide an initial working directory for a user, the effective default initial working directory for the user depends on how the user gained access to the OSS environment. For example, assume that a user logs in through a TACL prompt and uses the OSH command when the Safeguard null default value is in effect for the initial working directory. If OSH cannot find any other definition of an initial working directory, it uses `/G/volume/subvolume` as the user's effective initial working directory, where

volume is the user's default volume and *subvolume* is the user's default subvolume in the Guardian environment.

Initial working directories should not be in the Guardian file system (the /G directory), because the OSS environment creates history files in initial working directories. History files can grow indefinitely and cannot be truncated or deleted if the directory is in the /G directory. These history files could take up so much disk space that they could waste system resources. If the initial working directory is in the OSS file system, the history files are created in the OSS file system, where they can be truncated and deleted.

The best way to provide a user with an initial working directory is:

1. Create an initial working directory in the OSS environment.
2. Use SAFECOM to add the OSS pathname of the newly created initial working directory to the user definition as the value for the initial working directory attribute.

Be careful when assigning an initial working directory. Proofread the assignment for typographical errors and remember to create the initial working directory in the OSS file system. Some server processes do not give users access to the system if their user definition has an invalid initial working directory name or if the OSS file system is not running but an initial working directory is defined.

Creating an Initial Working Directory in the OSS Environment

Note. HP strongly recommends that you create a separate fileset for initial working directories.

To create an initial working directory in the OSS environment, use either the mkdir command from within the OSS environment or the OSH command from within the Guardian environment.

The following OSS shell command creates the OSS file-system directory /home/henrysp from within the OSS environment:

```
mkdir /home/henrysp
```

The following TACL command creates the OSS file-system directory /home/henrysp from within the Guardian environment:

```
OSH -p mkdir /home/henrysp
```

Assigning an Initial Working Directory Using Safeguard

You can assign an initial working directory to an existing user definition by using the SAFECOM ALTER USER command to change the value of the INITIAL-DIRECTORY attribute. You use a SAFECOM command such as:

```
ALTER USER group-number,member-number,
   INITIAL-DIRECTORY /home/dir
```

or

```
ALTER USER group-name.member-name,
   INITIAL-DIRECTORY /home/dir
```

group-number,member-number

is the structured view of the user ID of an existing user definition.

group-name.member-name

is the user name of an existing user definition.

/home/dir

is the initial working directory, expressed as an OSS pathname.

For example, the following SAFECOM command assigns the initial working directory `/home/henrysp` to the user definition with the user name SCRIBES.HENRYSP:

```
ALTER USER scribes.henrysp, INITIAL-DIRECTORY /home/henrysp
```

You can assign an initial working directory to an alias instead of to the user definition itself by using the SAFECOM ALTER ALIAS command. You use a SAFECOM command such as the following:

```
ALTER ALIAS alias-name, INITIAL-DIRECTORY /home/dir
```

alias-name

is an existing alias for an existing user definition.

/home/dir

is the initial working directory, expressed as an OSS pathname.

For example, the following SAFECOM command assigns the initial working directory `/user/donl` to the alias `donl` of the user definition with the user name SCRIBES.DONALD:

```
ALTER ALIAS donl, INITIAL-DIRECTORY /user/donl
```

The assignment process can be partially automated by creating a TACL macro file and then editing that file appropriately for each new user. Figure 8-2 on page 8-16 contains a Guardian TACL macro that adds a user and an alias for the user with an OSS initial working directory.

Figure 8-2. TACL Macro to Configure an OSS User (page 1 of 2)

```
?TACL MACRO
==
== This unnamed macro adds FTP user logins to the system.
==
==
#FRAME
    #SET #INLINEPREFIX>
    INLECHO ON
==
==
    SAFECOM /INLINE/
==
==
== Add a user definition for the user LINDA in user group
== SCRIBE.
==
== This definition provides the user with only Guardian access
== and is appropriate for a user who wants initial FTP
== access to the Guardian file system.
==
== The lack of an INITIAL-DIRECTORY value in the following user
== definition causes the Guardian default volume and subvolume
== to become the OSS initial working directory.
    >ADD USER scribe.linda, 168,10, PASSWORD Abcdef1
==
== Assign the user definition appropriate Guardian file
== security:
    >ALTER USER scribe.linda, GUARDIAN DEFAULT SECURITY NUNU
==
== Assign the user definition appropriate Guardian default
== volume and subvolume names:
    >ALTER USER scribe.linda, GUARDIAN DEFAULT VOLUME $data2.q9552
==
== Display the configuration for the primary user definition:
    >INFO USER scribe.linda, DETAIL
==
==
== Add an alias named lindaw for the user definition.
==
== This alias is used for either OSS or Guardian environment
== access.
==
== Assign the same password as for the user definition, to
== avoid user confusion.
    >ADD ALIAS lindaw, 168,10, PASSWORD Abcdef1
==
== Assign the alias appropriate Guardian file security:
    >ALTER ALIAS lindaw, GUARDIAN DEFAULT SECURITY NUNU
==
== Assign the alias consistent Guardian default volume and
== subvolume names:
    >ALTER ALIAS lindaw, GUARDIAN DEFAULT VOLUME $data2.q9552
==
== Assign the alias an initial program to run in the OSS
== environment, in case of access through a direct service:
    >ALTER ALIAS lindaw, INITIAL-PROGRAM /bin/sh
==
== Assign the alias an OSS initial working directory:
    >ALTER ALIAS lindaw, INITIAL-DIRECTORY /home/lindaw
==
```

Figure 8-2. TACL Macro to Configure an OSS User (page 2 of 2)

```
== Display the configuration for the alias:
    >INFO ALIAS lindaw, DETAIL

    >EXIT
==
#UNFRAME
```

Assigning an Initial Working Directory Using a TACLCSTM File

You can also assign an initial working directory to a user by either inserting the following entry in the user's TACLCSTM file or having the user enter the following at a TACL prompt before using the OSH command:

PARAM HOME *pathname*

where *pathname* is the pathname of an existing OSS directory to be used as the initial working directory.

When the user gains access to the OSS environment by logging in at a TACL prompt, the OSH command uses this PARAM value to override any default initial working directory assigned in the user or alias definition. Users can also temporarily change their initial working directories in this manner.

Assigning an Initial Program

An initial program is the program that is executed for a user upon entry to the OSS environment. An initial program is normally a command interpreter, but it can be any application that can read a standard input file.

If you log on and enter the OSS environment without specifying an initial program, Open System Services assumes that you are launching the initial program. This concept is similar to the concept of the default command interpreter in the Guardian environment.

You should provide each OSS user with an initial program in the OSS environment. The initial program can be assigned in either of these ways:

- You can use the Safeguard subsystem.
- For Telserv indirect users only, you can use a TACLCSTM file.

HP recommends that you use the Safeguard subsystem and its INITIAL-PROGRAM attribute.

Unless a user requests otherwise or your site has other specific requirements, you should assign the OSS shell (`/bin/sh`) as the initial program for all users.

Assigning an Initial Program Using Safeguard

You can assign an initial program to an existing user definition by using the SAFECOM ALTER USER command to change the value of the INITIAL-PROGRAM attribute. You use a SAFECOM command such as the following:

```
ALTER USER group-number,member-number,
   INITIAL-PROGRAM /code-dir/program
```

or

```
ALTER USER group-name.member-name,
   INITIAL-PROGRAM /code-dir/program
```

group-number,member-number

is the structured view of the user ID of an existing user definition.

group-name.member-name

is the user name of an existing user definition.

/code-dir/program

is the initial program, expressed as an OSS pathname.

For example, the following SAFECOM command assigns the initial program `/bin/sh` to the user definition with the user name SCRIBES.HENRYSP:

```
ALTER USER scribes.henrysp, INITIAL-PROGRAM /bin/sh
```

You can assign an initial program to an alias instead of to the user definition itself by using the SAFECOM ALTER ALIAS command. You use a SAFECOM command such as the following:

```
ALTER ALIAS alias-name, INITIAL-PROGRAM /code-dir/program
```

alias-name

is an existing alias for an existing user definition.

/code-dir/program

is the initial program, expressed as an OSS pathname.

For example, the following SAFECOM command assigns the initial program `/user/deml.o` to the alias `donl` of the user definition with the user name SCRIBES.DONALD:

```
ALTER ALIAS donl, INITIAL-PROGRAM /user/deml.o
```

The assignment process can be partially automated by creating a TACL macro file and then editing that file appropriately for each new user. Figure 8-2 on page 8-16 contains a Guardian TACL macro that adds a user and an alias for the user with an OSS initial program.

Assigning an Initial Program Using a TACLCSTM File

You can also assign an initial program to a user by either inserting the following entry in the user's TACLCSTM file or having the user enter the following at a TACL prompt:

```
RUN OSH -ls -prog pathname
```

where *pathname* is the pathname of an existing OSS program file to be used as the initial program. The `-ls` specification causes execution behaviors appropriate for a UNIX shell program.

When the user gains access to the OSS environment by logging in at a TACL prompt, the OSH command in the TACLCSTM file ignores any default initial program assigned in the user definition. Users can also temporarily change their initial programs in this manner.

Using this method to assign an initial program might cause unexpected behavior by OSS shell commands such as `newgrp`. HP does not recommend using this method.

See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for additional information on using the OSH command to launch programs other than the OSS shell.

Hints and Suggestions

The following subsections provide some suggestions for configuring users at your site:

Also refer to Utility File Security on page C-8.

Licensing the OSS Monitor to the Super Group

The OSS Monitor contains privileged procedures but is not a licensed program. If anyone other than the super ID (255,255 in the Guardian environment, 65535 in the OSS environment) attempts to start it, the TACL error message

```
Unlicensed privileged program
```

is issued.

HP strongly suggests that you do not license the OSS Monitor. In most cases, the only user of the OSS Monitor should be the system administrator (or other authorized person).

However, in situations where several users are required to have the authority to start or restart OSSMON, it might be advisable to license the OSS Monitor to members of the super group.

This method is preferable to allowing many users to access the super ID logon. The process for licensing the OSS Monitor is:

1. At a TACL prompt, set the volume to $SYSTEM.SYS*nn*, where *nn* indicates the current system subvolume.
2. Enter the following commands:

```
FUP LICENSE OSSMON
FUP SECURE OSSMON, "N-G-", PROGID
```

This procedure allows anyone in the super group to start OSSMON but also allows OSSMON to run under the super ID. In this way, OSSMON can start the other servers that must run under the super ID.

For more details on SCF security issues, see the section that describes configuring and managing generic processes in the *Storage Subsystem Configuration and Management Manual*.

Configuring Expand Users

By default, a new user is configured without access to other NonStop nodes through the Expand product. This default configuration makes all files that would be available through the OSS `/E` directory inaccessible to OSS users other than the user logged in with the super ID.

To configure a user for access to files in `/E`:

- Specify the REMOTEPASSWORD attribute in the Safeguard SAFECOM ADD USER, ALTER USER, ADD ALIAS, or ALTER ALIAS command.
- Provide a remote password for each remote node on which you want to allow that user to have file access.
- Make sure that the user name on a remote node has the same user ID value as that associated with the user name on your local node. For example, if BOOKS.DONL has the user ID 1,2 on your local node, BOOKS.DONL must have the user ID 1,2 on each remote node on which OSS files should be visible to that user.

For a user ID, you can specify a remote password in the local system's authentication records only, such that access is possible from your local node to the remote node but not vice versa. For an alias, you must create a Safeguard user authentication record containing remote passwords on both the local and remote nodes.

For a more detailed description of REMOTEPASSWORD configuration, see the *Safeguard Administrator's Manual*.

Configuring FTP Users

If an FTP server user requests that his or her user definition be configured for initial access to the Guardian environment:

1. Leave that user's definition with the default null value for its OSS initial working directory.
2. Use the TACLCSTM file method to assign an OSS initial working directory for indirect Telserv access (see Assigning an Initial Working Directory Using a TACLCSTM File on page 8-17).

Allowing Anonymous FTP Use

To configure FTP for anonymous access to the OSS file system, use the Safeguard product. For example, enter SAFECOM commands similar to:

```
ADD USER NULL.FTP, 0,254, OWNER 255,255, PASSWORD guest
ALTER USER NULL.FTP, GUARDIAN DEFAULT SECURITY OOOO
ALTER USER NULL.FTP, GUARDIAN DEFAULT VOLUME $guest.ftp
ADD ALIAS anonymous, 0,254, PASSWORD guest
ALTER ALIAS anonymous, GUARDIAN DEFAULT SECURITY OOOO
ALTER ALIAS anonymous, GUARDIAN DEFAULT VOLUME $guest.ftp
ALTER ALIAS anonymous, INITIAL-DIRECTORY /user/guest
ADD ALIAS ftp, 0,254, PASSWORD guest
ALTER ALIAS ftp, GUARDIAN DEFAULT SECURITY OOOO
ALTER ALIAS ftp, GUARDIAN DEFAULT VOLUME $guest.ftp
ALTER ALIAS ftp, INITIAL-DIRECTORY /user/guest
FREEZE USER NULL.FTP
FREEZE ALIAS anonymous
FREEZE ALIAS ftp
```

Note that:

- The Safeguard BLIND-LOGON attribute for the node must be set to OFF.
- The group number for the anonymous FTP user name NULL.FTP must be 0.
- The member number for the anonymous FTP user name NULL.FTP is not 0 or 255.
    - The member number 0 is reserved for a different user name in the group with the group name NULL.
    - The member number 255 is assigned to the group manager of a group; users with the user name NULL.FTP should not have group manager access for the group with the group name NULL.
- Although other aliases are case-sensitive, FTP aliases for anonymous users are case-insensitive. For example, you need to configure only "`ftp`" in the FTPUSERS file to bar access by both the user aliases "`ftp`" and "FTP."

- An alias must have a valid initial working directory (in the previous example, `/user/guest`):
  - If no valid initial working directory is specified for an anonymous alias, then FTP access for that alias is denied.
  - An initial working directory of `/E`, `/G`, or a directory in `/G` is invalid.
- The aliases `anonymous` and `ftp` must be frozen under the Safeguard product, so that those names cannot be used for access through any server process other than FTP.

  Freezing the user NULL.FTP disables anonymous access to the Guardian environment.
- No OSS initial working directory is specified for the user NULL.FTP. As a result, the user NULL.FTP is not allowed access to the OSS file system.
- The aliases `anonymous` and `ftp` do not have access to the Guardian environment.
- The user definition NULL.FTP and its aliases must all use the same Guardian default subvolume.
- The initial working directory for an anonymous FTP user should be set up to have appropriate security in the OSS environment for the user as a type “other” user of the directory.
  - A read-only anonymous user would have the OSS file security for the directory set to “`drwxrwxr-x`”.
  - A write-only anonymous user would have the OSS file security for the directory set to “`drwxrwx-wx`”.
- Guardian or OSS environment access by any user can be disabled by adding the appropriate user name or alias to the FTPUSERS file in $SYSTEM.ZTCPIP. Allowing anonymous user access means omitting the corresponding user name or alias from the FTPUSERS file.

See the *Safeguard Administrator’s Manual*, the *Safeguard Reference Manual*, and the *Security Management Guide* for additional security considerations. See the *TCP/IP Applications and Utilities User Guide* for additional information and recommendations about anonymous user FTP access. See the appropriate manual for information about the corresponding concepts and facilities of a third-party product.

Disallowing OSS Use by Specific Users

Some sites assign a nonexistent OSS pathname as the initial working directory to all user and alias definitions that are not explicitly configured for OSS environment access. For example, they would use a SAFECOM command such as:

```
ALTER USER scribes.donald, INITIAL-DIRECTORY /contact_site_admin
```

This definition prevents use of the Guardian environment default volume and subvolume as the initial working directory. It also causes use of the OSH command to fail for that user and give the user the suggested pathname as part of an OSH error message.

Note that this approach to user and alias definitions can add significantly to logs of error messages or of process failures.

Configuring Special Users

UNIX administrators traditionally reserve certain user names for special uses. The user name `root` is almost universally used for the user who has super ID permissions (appropriate privileges for the use of all restricted system facilities).

Consider configuring the super ID with the alias of `root`. That configuration provides behavior consistent with most UNIX systems and might prevent confusion.

To configure the super ID with the alias `root`, you would enter an appropriate version of the following SAFECOM commands:

```
ADD ALIAS root, 255,255, PASSWORD Doom1
ALTER ALIAS root, GUARDIAN DEFAULT SECURITY ----
ALTER ALIAS root, GUARDIAN DEFAULT VOLUME $SYSTEM.SYSTEM
ALTER ALIAS root, INITIAL-DIRECTORY /
```

The user name SUPER.SUPER is predefined in the security database as the user ID (255,255), which is the super ID with appropriate privileges in the OSS environment.

OSS Security Auditing

As it does in the Guardian environment, the Safeguard audit service records and retrieves information about file access decisions that have occurred within the OSS subsystem. The audit service records the outcome of requests for permission to create, open, or delete files; change file content, permissions, or ownership; add or alter filesets; and create or delete directories. Actions that create or change the state of OSS processes can also be audited, such as the `kill` command or any of the `tdm_exec`, `tdm_spawn`, `exec`, and `tdm_fork()` or `fork()` function calls.

The following subsections provide more information about:

- Audit Records for OSS Objects on page 8-23
- Auditing of OSS Shell Commands on page 8-26

Audit Records for OSS Objects

Audited events are recorded in the Safeguard audit files (collectively referred to as the audit trail). Every audited event describes:

- The user or process initiating the event
- The object affected by the event

- The operation, including whether the operation succeeded or failed, and the details of a defined list of appropriate attributes

Audit records are characterized by the following information:

- If the object of an operation has a pathname, then either the audit record includes the pathname or the operation is associated with another record that includes the pathname.
- OSS filenames stored in the audit record are uniquely identifiable.
- Operations and outcomes are specified by enumerated values defined by Safeguard.
- An operation that modifies an object's attributes provides before and after images of the attributes in the audit record.
- An operation that creates a new object specifies the new object's attributes in the audit record.
- An operation that deletes an object specifies the object's attributes in the audit record.
- Failure to search a directory during name resolution is audited. The audit record indicates the pathname of the directory being searched, up to and including the failure.

This information can be retrieved by using SAFEART, the Safeguard audit-file reduction tool.

Object Names in Audit Records

When the value of an OSS file attribute must appear in an audit record, the OSS name server writes the file's object names in its request to the file system. OSS objects have two kinds of object names, an external name and an internal name.

For objects in the OSS file system, the external name is the fully qualified pathname for the object. For OSS filesets, the external name is the name of the fileset as seen through SCF.

In most audit records, the external and internal names for the object are both included and separated by an equal sign (=). For example:

```
/bin=$ZPNS.Z00000.Z0000004G:56876483
/bin/sh=$OSS1.ZYQ00000.Z000005R:45736652
```

Sometimes only the internal name appears, in which case a preceding RESOLVE record contains both names.

The OSS name server maintains the absolute pathname of the mount point for each fileset that it manages. To ensure that they are generated quickly, all pathnames that are stored in audit records are normalized as follows:

- All dots (.), double dots (..), multiple slashes, and symbolic-link references are resolved.
- The maximum length of the stored pathname is 1023 bytes. If the actual pathname length exceeds 1023 bytes, the audited name consists of three periods (…) followed by the last 1020 bytes of the pathname.

Which audit records are generated depends on the operation (see Auditing of OSS Shell Commands on page 8-26 for a list of what is generated in the audit record of different operations). A record is generated for an object in an audited fileset after the process that manages the object checks the user ID to determine whether the user has the authority to perform the requested operation.

When the operation terminates because of an error and a security ruling has not yet been obtained, no auditing is performed. An operation can also fail after an audit record is logged.

The name logged in an operation depends on the type of object being audited. Formats are:

OSS fileset	$ZPMON.Z*nnnnn*:*yyyymmddhhmmss*, where *nnnnn* is the fileset device number and *yyyymmddhhmmss* is the local civil time (LCT) when the fileset was created. Example: `$ZPMON.Z00000:19980119152451`
OSS regular file (disk file)	$*vol*.ZYQ*nnnnn*.*Ziiiiiii*:*ccccccccccc*, where *nnnnn* is the fileset device number, *iiiiiii* is the file's inode number, and *ccccccccccc* is the file's creation version serial number (CRVSN). Example: `$OSS1.ZYQ00000.Z00004G6:19934568735`
Other OSS files (such as `AF_UNIX` sockets)	$ZPNS.Z*nnnnn*.Z*iiiiiii*:*ccccccccccc*, where *nnnnn* is the fileset device number, *iiiiiii* is the file's inode number, and *ccccccccccc* is the file's CRVSN. Example: `$ZPNS.Z00000.Z00004G5:19387764537`

Object Name Changes

When a directory that is on the path to a fileset mount point is renamed, that renaming is propagated to the fileset mount point on that path. However, this propagation takes place after the call on the rename function has finished. If an audited operation is performed on a file in that path before the rename is propagated to the fileset mount point, the audit record might contain the old pathname rather than the new one.

For example, assume that a fileset is mounted on `/usr/src/projects/mine`. The following sequence of calls occur:

```
rename("/usr/src/projects", "/usr/src/tasks");
open("/usr/src/tasks/mine/main.c");
```

The audit record for the `open` call might contain `/usr/src/projects/mine/main.c` (the old pathname) rather than `/usr/src/tasks/mine/main.c` (the new pathname).

For a description of the OSS subsystem message that occurs under these conditions, see OSS subsystem message 20 in the *Operator Messages Manual*.

Auditing of OSS Shell Commands

Many OSS shell commands cause audit records to be generated by the OSS name server when auditing is enabled. The contents of each audit record depend on which operation is being performed. In cases where the operation is terminated because of an error and a security ruling has not yet been obtained, no auditing is performed.

Some of the shell commands that cause audit records to be created are `mkdir`, `chmod`, `chown`, `kill`, and `rmdir`.

Protecting Your System

This subsection covers the following topics:

OSS Shell Commands Useful for Security Administration

This subsection covers the following topics:

Creating a Logon Session With the su Command

The `su` command provides an alternative to login for accessing an OSS account. The `su` command can change:

- The login name of the current shell (thus changing the user ID of the current shell)
- The password for the user ID of the current shell
- The login name of a new shell (thus changing the user ID from that of the current shell)

- The password for the user ID of a new shell

Security is enforced by requiring the user to complete a normal login procedure for the new login name. The new user ID stays in force until the shell exits. The new password stays in force until it is changed again.

For super ID users, the shell substitutes a # (number sign) for its usual prompt.

You can specify a login shell using `/bin/sh` with the appropriate environment variables. You can also specify a string to be passed to the shell as a command to execute. See the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual* for a discussion about using this option.

For more information about the `su` command, see the `su(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Displaying Your User Login Name

The `logname` utility writes to the standard output file the name you used to log in to the system. You can use this utility after you have issued the `su` command and want to check your current user ID.

For more information about the `logname` command, see the `logname(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Changing Your User Group

If you are logged in as the super ID, the `newgrp` command allows you to become a member of any defined group. This ability can be a convenience when you are administering files in the OSS file system.

Correct behavior of the `newgrp` command depends on whether you have a `SHELL` environment variable defined for your current terminal session or your user ID has an INITIAL-PROGRAM attribute defined for it. For more information about the `newgrp` command, see the `newgrp(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Use of suid Scripts

Some scripts, known as `suid` scripts, enable users to perform some activities that require appropriate privileges; that is, the scripts could be used by an intruder to assume the identity of the super ID (255,255 in the Guardian environment, 65535 in the OSS environment).

As a result, it is a good practice to make sure that there are no such script files on your system. As installed, the OSS environment does not contain `suid` script files.

suid script files have permissions that either start with the digit 4 or have “s” as the owner’s execution permission bit, as shown in the following example:

```
$ ls -l dfile
-rwxr-xr-x   1 MANUALS.HENRYP   MANUALS       29 Jun 27 15:24 dfile
$ chmod 4755 dfile
$ ls -l dfile
-rwsr-xr-x   1 MANUALS.HENRYP   MANUALS       29 Jun 27 15:24 dfile
```

You can detect suid script files by using the find command with the -perm flag, as shown in the following example:

```
$ find .  -perm -4000 -print
./dfile
```

This command searches for files that have permissions of 4000 or greater. The minus sign before the 4000 specifies “greater than or equal to.” Files with permissions of 4000 or greater would all be executable suid scripts. This command found such a script, named ./dfile.

For details about suid script files, consult a book about UNIX security. Some books about UNIX security are listed in UNIX Security on page xix.

Preventing Security Problems

To improve the security of your system:

- All OSS Monitor and server database files should always be owned by the super ID (255, 255) and secured as having read, write, execute, and purge privileges for only the super ID (----) in the Guardian environment.
- The /etc/install_obsolete directory and the Pcleanup command should be secured such that only the super ID or a trusted user has access to them. A single file surreptitiously added to /etc/install_obsolete could be used by an intruder to cause severe damage to mounted filesets.
- The Guardian subvolume from which OSS files are installed by COPYOSS or PINSTALL should be carefully secured and its contents periodically checked for validity. A pax archive surreptitiously added to that subvolume could be used by an intruder to install files such as:
  - Viruses, worms, Trojan horses, or rabbit programs
  - Substitutes for standard utilities, containing logic bombs or back doors
- Use the -R option when starting the inetd process to reduce the chance of denial-of-service attacks.
- Use the same security policies for FTP and remote exec services servers as you use for Telserv terminal access. For example, the rexecd server is equivalent to a context-free, one-line telnet session with enforced user validation; at a minimum, Safeguard should be used to set up its users and, ideally, all requests to rexecd should be filtered by your node’s firewall.

- Do not provide network services servers such as `rshd` where mechanisms such as the `hosts.equiv` file can be used to bypass Safeguard validations.
- For remote procedure call (RPC) applications, encourage your developers to use the application program interface provided by the HP NonStop Distributed Computing Environment (DCE). The authentication security is better than the AUTH_UNIX (AUTH_SYS) level authentication used by the undocumented RPC interfaces underlying such products as the Network File System (NFS) for Open System Services.

Identifying Attempts to Break Security

Checking the file system for changes in the ownership and permissions of important files and directories can reveal the presence of an intruder. You can monitor permissions by entering the following OSS shell command periodically:

`ls -alt` *`pathname`* `>` *`file`*

`pathname`

is the OSS pathname for the mount-point directory of a fileset you want to monitor (such as `/bin` and `/etc`).

`file`

is the OSS filename of a file to receive the output.

Use a different OSS filename each time you use this `ls` command, and use the `diff` command to compare the different listings.

If a file on which only the super ID (255,255 in the Guardian environment, 65535 in the OSS environment) had permissions has changed to have more general permissions (and if this change was not authorized), a break-in might have occurred.

9 Managing With the Shell

The shell is the interactive interface to the Open System Services (OSS) environment. The OSS shell is a UNIX Korn shell. This section describes how to set up the shell to best serve your users.

Information about using the shell is in the *Open System Services User's Guide*. Reference information is in the `sh(1)`reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. Read the reference page to make sure that the shell feature you want to use is available and behaves in the way you expect.

Note. Beginning with the G06.08 release version update (RVU), many OSS shell utilities that had been unsupported became formally supported and were moved from the `/bin/unsupported` directory to the `/bin` directory. Any application program or site-written shell script that uses a utility in `/bin/unsupported` should be modified to use the version in `/bin` when such a version exists.

OSS Management With the Shell

Managing the OSS environment with the shell involves:

- Customizing the OSS shell to make using it convenient for your users. You do this by:
  - Offering a default `.profile` file for your users, as described in Setting Up a Default .profile File on page 9-2
  - Including commands and setting variables and options in the `/etc/profile` file, as described in Setting Up an /etc/profile File on page 9-2
  - Making sure that tracked command aliases are used, as described in Adding Commands for User Convenience on page 9-3
  - Ensuring access to all needed online reference material, as described in Controlling Reference Page Searches and Display on page 9-4
  - Offering localization features, as described in Localizing Software on page 9-5 and Localizing Reference Pages on page 9-7
- Monitoring the OSS environment to maintain optimal performance as described in Monitoring the OSS Environment With the Shell on page 9-8.
- Finding and removing unwanted files, such as old temporary files and large files that have not been accessed in a long time, as described in Controlling the Growth of Directories on page 9-8.
- Defragmenting disks periodically as described in Defragmenting Disks on page 9-9.

The OSS environment does not support the `/etc/passwd` file. You add users, modify their descriptions, delete them from the system, and alter various security attributes

through either a third-party product or the Safeguard subsystem, run from the Guardian environment, as described in Section 8, Managing Security.

Customizing the OSS Shell

You can customize the OSS shell by providing a custom default `.profile` file for your users, setting up the `/etc/profile` file to meet your needs, and using the localization features in both of these files.

Setting Up a Default .profile File

Each user's environment can be set up by means of a `.profile` file. This file is executed automatically every time a user logs in. Although a `.profile` file is not essential, it can make use of the OSS environment easier.

A default `.profile` file provides your users with a basic environment (beyond that of the `/etc/profile` file) that they can then alter to suit their needs. To set up a default `.profile` file for your users, create the file in a convenient administrative directory with an OSS text editor.

The content appropriate in a `.profile` file can depend on the needs of the user groups to which a user ID belongs. Some generally useful commands to add to this file are:

```
export PATH=default.user.path #Replace with a default path
#
#
#You can add command shortcuts, such as:
#
alias m=more
alias h=history
#
#
# You can provide expected commands that are not supported
# by the OSS environment:
#
alias logout=exit
```

As part of adding a new user to the system, copy the default `.profile` file to the user's initial working directory.

Setting Up an /etc/profile File

The `/etc/profile` file is similar to the `.profile` file, but it applies to the shell and, therefore, to all users of the shell rather than to one user. The `/etc/profile` file is run every time a user logs in to the shell, and it is owned by the super ID.

This subsection describes how to set up an `/etc/profile` file to maximize user convenience and improve security. It also describes how to use the file as a substitute for the UNIX `motd` (message of the day) command.

Figure 9-1 shows the `/etc/profile.sample` file that HP provides. You might want to copy this file to `/etc/profile` and use it as your system `/etc/profile` file. (Beginning with RVU G06.14, if your system was ordered preconfigured or your initial OSS configuration was performed using the OSSSETUP utility, this copy was made automatically.)

Figure 9-1. Sample /etc/profile File /etc/profile.sample

```
# Remove /bin/unsupported from PATH if you DO NOT want to
# use the unsupported utilities on your system
# Note: /usr/ucb contains the OSS ftp client
#export PATH=/bin:/usr/ucb
export PATH=/bin:/bin/unsupported:/usr/bin:/usr/ucb
export TERM=xterm
export PS1='$PWD: '
```

Adding Commands for User Convenience

A user can override commands that are in the `/etc/profile` file, therefore commands in this file are for the convenience of your users, not a means of enforcing security provisions.

Some useful commands you can put in the `/etc/profile` file are:

```
umask 022 #Only users have write permission on their files.
set -o noclobber #Redirection can't overwrite files.
set -o trackall #Track all aliases.
export MANPATH=/usr/share/man #Match PATH use.
```

For information about the `set` and `umask` commands, see the `set(1)` and `umask(1)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual*.

Setting the `trackall` flag creates tracked aliases. The tracked alias for a command takes effect the second time the alias is used.

A tracked alias improves performance by automatically starting the aliased command with the full OSS pathname of the system copy of the command file. This behavior avoids time-consuming searches through all the directories in the user's `PATH` environment variable. Several commands have built-in aliases; for details, see the `sh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Tracked aliases take priority over executable files; when a tracked alias refers to a system utility, the file in the `/bin` directory is executed instead of a file with the same utility name that might reside in another directory in the user's search path. This behavior prevents intruders from using a commonly used utility name for a substitute program that they create to damage data on your system.

OSS environment characteristics usually default to those of the Guardian environment on your node. However, many of these characteristics can be changed using standard

UNIX methods so that the OSS environment uses values separate from those of the Guardian environment.

For example, the time values used by an OSS shell default to those of the system that the shell runs on. If you maintain an OSS environment for users in a time zone other than that used for your Guardian environment, you can add the TZ environment variable to /etc/profile to make the time zone for your OSS users appropriate to their location. The following entry would be appropriate for California users of a node located in New York City:

```
export TZ=:PST-8PDT-7,M4.5.0,M10.5.0
```

See the environ(5) reference page either online or in the *Open System Services System Calls Reference Manual* for the format of the TZ environment variable.

Controlling Reference Page Searches and Display

HP recommends setting the MANPATH environment variable in the /etc/profile file to /usr/share/man so that OSS reference pages can always be delivered to users by the man command, and so that the whatis and apropos commands work correctly for OSS information. Individual users can then set their .profile files to use other online reference material, such as that installed for some open source packages into the /usr/local/man directories.

Setting the MANPATH environment variable can be very important if your system has third-party or open source reference page (man page) source files in any of the /usr/local/man/man* directories but you have not installed an nroff utility. The OSS man command can call an nroff formatting tool to provide automatically formatted updates, even though the OSS shell product does not provide such a tool. When the MANPATH variable is not defined, the man command searches directories in the following order:

```
/usr/share/man/man*
/usr/local/man/man*
/usr/share/man/cat*
/usr/local/man/cat*
```

If the man command finds a source file in a /usr/share/man/man* directory that has the name a user is searching for as a formatted file provided by HP in /usr/share/man/cat*, the man command appears to fail with the error message:

```
Nroff/troff is not currently installed, this must be
installed in order to use formatted man pages.
```

HP does not currently ship unformatted source files in /usr/share/man/man*. However, if that set of directories is used, similar man command behavior occurs.

If you install open-source reference page source files in /usr/local/man/man*, you should install an open source formatter such as groff and create a symbolic link to it from the pathname /bin/nroff; the man command then successfully formats the source files and displays them for the user.

Using the /etc/profile File Instead of a motd Command

You can also use the `/etc/profile` file to send a message to all users when they log in. You can inform them of new features, projected downtime, or any other matter that you think they should know. To do so, include a line in the `/etc/profile` file such as:

`echo "`*`message`*`"`

where *`message`* is the message you want to send. You must include the quotation marks.

If this command line is in the `/etc/profile` file, then each time a user logs in to the OSS environment, the `echo` command is executed and the message appears on the user's terminal.

Localizing Software

You can use the localization environment variables in:

- An `/etc/profile` file to customize the behavior of all compatible applications launched from an OSS shell for a specific locale
- A default `.profile` file to customize the behavior of all compatible applications launched from a specific user's OSS shell for a specific locale

You can use a different locale value for each environment variable in either file. The values for these variables are inherited by all the child processes for that shell.

The localization environment variables are listed in Table 9-1 on page 9-5, along with their meanings and default values.

The version of the OSS shell released by HP fully supports only the default C locale. That is, HP does not provide message catalogs for OSS shell commands and utilities when nondefault locales are used. Nondefault values for localization environment variables should be used only after testing.

Table 9-1. Localization Environment Variables (page 1 of 2)

Variable	Meaning and default
`LANG`	The locale of your shell, which consists of a language, territory, and code set. The default locale is the C locale. The default language is English. The default territory is US. The default code set is ASCII.
`LC_ALL`	The behavior for all aspects of the locale, unless overridden by another variable.
`LC_COLLATE`	The collating sequence. The default value is C/POSIX.
`LC_CTYPE`	The character classification information. The default value is C/POSIX.

Table 9-1. Localization Environment Variables (page 2 of 2)

Variable	Meaning and default
LC_MESSAGES	The language for shell messages. The default value is C/POSIX.
LC_MONETARY	The monetary format for your shell. The default value is C/POSIX.
LC_NUMERIC	The numeric format for your shell. The default value is C/POSIX.
LC_TIME	The time format for your shell. The default value is C/POSIX.

Table 9-2 on page 9-6 lists all the locales released as part of the OSS product set. An additional locale, GB18030, is provided for use only within the `vi_gb18030` and `print_gb18030` utilities; see the online reference pages `vi_gb18030(1)` and `print_gb18030(1)` for more information. You can create a unique locale for your server node by using the following tools from the command line of an OSS shell:

```
dspcat
dspmsg
gencat
genxlt
iconv
locale
mkcatdefs
runcat
```

Before using these tools, you must change the following entry in the EDIT file $SYSTEM.SYSTEM.UNISTDH and the OSS text (header) file `/usr/include/unistd.h`:

```
#undef  _POSIX2_LOCALEDEF    /* no support for the creation of locales */
```

to say:

```
#define _POSIX2_LOCALEDEF 1 /* New locales allowed */
```

This change might need to be made again after any software product revision or release version update is installed.

For further information, see the `sh(1)`, `locale(1)`, and `locale(4)` reference pages either online or in the *Open System Services Shell and Utilities Reference Manual* and see the *Software Internationalization Guide*.

Table 9-2. Locale Names and Filenames (page 1 of 2)

Language of locale	Filename for locale definition
Danish	da_DK.ISO8859-1
Dutch, Netherlands	nl_NL.ISO8859-1
Dutch (Belgian), Belgium	nl_BE.ISO8859-1
English, Great Britain	en_GB.ISO8859-1
English, USA	en_US.ISO8859-1
Finnish	fi_FI.ISO8859-1

Table 9-2. Locale Names and Filenames (page 2 of 2)

Language of locale	Filename for locale definition
French, Belgium	fr_BE.ISO8859-1
French, Canada	fr_CA.ISO8859-1
French, France	fr_FR.ISO8859-1
French, Switzerland	fr_CH.ISO8859-1
German	de_DE.ISO8859-1
German, Switzerland	de_CH.ISO8859-1
Greek	el_GR.ISO8859-7
Icelandic	is_IS.ISO8859-1
Italian	it_IT.ISO8859-1
Japanese, EUC	ja_JP.AJEC
Japanese, SJIS	ja_JP.SJIS
Japanese-English, Japan	en_JP.ISO8859-1
Korean, EUC	ko_KR.eucKR
Norwegian	no_NO.ISO8859-1
Portuguese	pt_PT.ISO8859-1
Spanish	es_ES.ISO8859-1
Swedish	sv_SE.ISO8859-1
Taiwanese, EUC	zh_TW.eucTW
Turkish	tr_TR.ISO8859-9

Localizing Reference Pages

HP provides a set of reference pages appropriate for the en_US.ISO8859-1 locale in the default MANPATH environment variable value /usr/share/man.

Access to reference pages is not controlled through locale variables. However, you can provide users of other locales with equivalent sets of reference pages that have been translated into the language corresponding to a locale. You do this by setting the value of a MANPATH environment variable in either the /etc/profile file or the default .profile file.

If you have translations of reference pages, you can put them in any directory and set the MANPATH variable to the corresponding value. The best practice is to put each translated set into its own reference page fileset.

Provide a MANPATH environment variable value for the directory of each language for which you supply reference pages. For example, suppose your site has users of the ja_JP.AJEC locale. You could define a fileset that is mounted at the directory man in directory /usr/EUC and then specify the following in your default .profile file:

```
setenv MANPATH /usr/EUC/man
```

The directory `/usr/EUC/man` would contain Japanese translations of reference pages in its subdirectories. To allow access to the translated reference pages first and then the versions provided by HP, you would specify:

```
setenv MANPATH /usr/EUC/man:/usr/share/man
```

Monitoring the OSS Environment With the Shell

Potential problems that you might want to monitor using the shell include:

- Slow performance
- Overuse of resources

The examples given in this subsection are not exhaustive. You might also want to remove files from directories that expand automatically, as discussed in Controlling the Growth of Directories on page 9-8.

Slow Performance

Slow performance might be the result of many processes left running that are no longer being used. One way to check is to enter an OSS command such as:

```
ps | sort -nr +2
```

This command lists the output of the `ps` command in reverse order by the TIME field. The processes that have run for the longest time are at the top of the list. You can then decide whether to remind the users to stop their processes, let things be, or terminate the offending processes with the OSS shell `kill` command.

Overuse of Resources

Large files that haven't been accessed in a long time might waste resources and prevent users from being able to create files. You might want to find users who have such files in their directories and discuss the situation with them.

You can list the owners of large files using an OSS shell command such as the following, which lists detailed information (including the owners) about files in `/usr` that are larger than 1000 kilobytes:

```
find /usr -size +1000K | xargs ls -l
```

Open System Services also provides the disk usage utilities `du` and `df`.

Controlling the Growth of Directories

The `vi` text editor and other programs produce temporary files that you might later want to remove. You might also want to remove large files that have not been accessed in a long time. This subsection describes how to remove such files.

The `find` command searches for files that match criteria you set; for example, it can find all the files in a directory that have not been accessed in a specified number of days. It can then perform an action you specify, such as deleting the files it finds. For detailed information about the `find` command, see the `find(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

You can create shell scripts that use the `find` command to delete files and then invoke the scripts periodically with either the OSS shell `cron` command or the optional Guardian NetBatch product, as described in Scheduling Periodic Tasks on page 2-34.

Note. OSS file opens do not work on Guardian processes such as the NetBatch process $ZBAT. To work around this, redirect `stdin`, `stdout`, and `stderr` to files that can be opened by the OSS environment, or close these files if they are not being used.

You would invoke the OSS environment with the TACL OSH command. You can run a single OSS shell command such as `find` from within a NetBatch job by using a job file that contains the following OSH command line:

```
OSH <- >outlog 2>errlog -c "command"
```

This command closes `stdin`, directs `stdout` to the file specified as `errlog`, directs `stderr` to the file specified as `outlog`, then runs the specified command.

Here are some examples of using the `find` command to locate and delete files:

- Delete all files in `/var/tmp` whose names begin with TMP that have not been accessed in thirty days:

  ```
  find /var/tmp -name 'TMP*' -atime -30 | xargs rm
  ```

- Delete all files that are named `a.out` or whose names end in `.o` that have not been accessed in sixty days:

  ```
  find / -W NOG -W NOE \( -name a.out -o -name '*.o' \) \
      -atime -60 | xargs rm
  ```

Defragmenting Disks

OSS development environments might be similar to UNIX environments in that users can create a large number of small files. Such files fragment your disks and slow down disk access considerably. To restore optimum disk access speed, run the Guardian Disk Space Analysis Program/Disk Compression Program (DSAP/DCOM) utility periodically to defragment the disks.

You can do this:

- Automatically with either the OSS shell `cron` command or the optional Guardian NetBatch product
- While the OSS environment is running
- Without stopping any filesets before you run DCOM

The syntax for the DCOM command to defragment a disk volume is:

```
DCOM volume
```

volume

is the name of the disk volume you want to defragment.

Compressing Files

You can compress files to create more space on a disk. You can use:

- The pack utility, which is a POSIX.2 utility
- The compress utility, which provides more compression than pack

Both utilities compress a file and store the specified file in a compressed form. The compressed (packed) file replaces the input file and has a name derived from the original filename (*filename*.z in the case of the pack utility, *filename.*Z in the case of the compress utility).

If you have the appropriate privileges, pack preserves the access modes, access and modification dates, and owner of the original file. (For details about these attributes, see the chmod(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual.*) Otherwise, pack compresses the file and assigns your owner and group ID to the new file. The compress utility always preserves access modes, access and modification dates, and owner information.

You can force compression of input files even if the files cannot benefit from compression. You can also display statistics about the input files.mpression is not done under certain conditions; for example, the file is already compressed, has hard links, is a directory, or cannot be opened. For more information about the pack or compress command, see the pack(1) or compress(1) reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Executing Remote Shell Commands

The rsh command executes a specified shell command remotely. It executes commands at the host system where the commands are to be run. The rsh command sends standard input from the local host to the remote command and receives standard output and standard error file data from the remote command.

Note. If the remote host is a NonStop S-series or NonStop NS-series server, you must specify the -l flag and provide your password in clear text form. Using clear text for passwords is not a good practice, so the rsh command should be avoided if possible.

You can do the following tasks by using the rsh command:

- Turn on debugging of the TCP sockets used for communication with the remote host

- Choose to log into the remote host using a specified user name rather than the local user name
- Specify an argument to the command you are performing remotely

The remote host allows access only if at least one of the following conditions is satisfied:

- The local user ID is not the super ID, and the name of the local host is listed as an equivalent host in the remote `/etc/hosts.equiv` file.
- The remote user's home directory contains a `$HOME/.rhosts` file that lists the local host and user name.

For security reasons, any `$HOME/.rhosts` file must be owned by either the remote user or the super ID, and only the owner should have write access.

For more information about the `rsh` command, see the `rsh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

Parsing Command Options With the getopts Command

The `getopts` command is used only in shell scripts, not at the shell prompt. It parses command options and checks a specified command for legal options. Each time it is invoked, `getopts` places the next option letter it finds into a variable name that you specify. You can also specify letters that the `getopts` command is to recognize as valid option values and an option argument to parse.

The `getopts` command differs from regular OSS shell commands in that it does not open a new shell process when it executes.

For more information about the `getopts` command, see the `getopts(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

10 Managing OSS Devices

This section briefly discusses Open System Services (OSS) devices and describes how to manage printers in the OSS environment.

The Scope of OSS Device Management

The only devices you can manage in the OSS environment are printers. All other devices are managed through the Guardian environment.

In the OSS environment, printer and tape-drive definitions are not stored in the `/dev` directory. Do not modify the contents of the `/dev` directory.

Device Access

Tape drives are not supported in the OSS environment as devices and do not appear in the `/dev` directory. Thus, users cannot access tape drives directly.

The only OSS shell utility that can communicate directly with a tape drive is the `pax` utility, which accesses tape devices through low-level software in the Guardian interface. Guardian tape processes are visible to OSS shell users through Guardian file system entries in the `/G` directory (for example, `/G/TAPE4`), but they are not visible to OSS programs.

Printers, like tape drives, do not appear in the `/dev` directory. Therefore, application programs cannot access printers directly in the OSS environment, although you can access them indirectly through the shell.

Only network virtual terminals can gain command-line access to the OSS environment. For information about configuring such terminals, see Section 7, Managing Terminal Access, and the *Telserv Manual*.

Managing Printers in the OSS Environment

The OSS printing utilities use the spooler product. Make sure that the spooler is running on your system. For information about the spooler, see the *Spooler Utilities Reference Manual*.

To configure printers in the OSS environment:

1. Specify a system default printer for use when the OSS environment encounters a print command that does not specify a particular destination printer. Instructions for doing this are in Specifying a Default Printer on page 10-2.
2. Optionally assign aliases for other printers. Instructions for doing this are in Using the /etc/printcap or printcap File on page 10-3.

OSS shell commands that print files require Guardian spooler-location names for access to printers unless shell aliases have been defined as alternative names.

Printer management in the OSS environment consists of defining aliases for Guardian spooler-location names. The aliases in the `/etc/printcap` file provide a system-wide set of definitions. At least one alias should be provided in that file.

A user can also define aliases in his or her own `printcap` file. When you add a new user to the system, consider creating a `printcap` file in the user's initial working directory. Providing a `printcap` file unique to each user allows you to assign a conveniently located printer to that user.

Specifying a Default Printer

You specify a default printer:

- By specifying the Guardian spooler-location name for a system default printer in the `/etc/printcap` file. Specifying a system default printer allows you to control which printer is used for high-volume print jobs.
- By specifying the Guardian spooler-location name for a default printer for a specific user in one of the following ways:
  - Specify the Guardian spooler-location name for the chosen printer in the user's `printcap` file.
  - Define the `LPDEST` or `PRINTER` environment variable for the user. See Using a Printer Environment Variable on page 10-4 for more information about these variables.
  - Use the `env` shell command, described in the `env(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*. (The `env` command can be included in the `/etc/profile` file or in the default `.profile` file.)

When a user invokes an OSS printing utility such as the OSS shell `lp` command, the OSS environment searches the following items, in the order listed, to determine the destination printer:

1. The command line. For example, in the `lp` command, you can name the printer in the `-d` flag or the `-W spl` flag. If you specify both flags, the value for the last flag specified is the destination printer.
2. The `LPDEST` environment variable value.
3. The `PRINTER` environment variable value.
4. The printer listed as the first entry in the `/etc/printcap` file.
5. The printer listed in the first line of the `printcap` file in the directory specified by the `HOME` environment variable. By default, the `HOME` environment variable specifies the user's initial working directory.

If no printer is specified in any of these items, an error is generated and the print job is not completed. You should provide at least the entry for `default` in the

/etc/printcap file in Item 4, because all the other items can be omitted or accidentally removed by user actions.

Using the /etc/printcap or printcap File

Both the /etc/printcap and printcap files contain a list of aliases and the corresponding Guardian spooler-location names for printers. A valid /etc/printcap or printcap file must contain at least one entry to establish the default printer. In the sample file /etc/printcap.sample that is installed as part of the OSS file system, this entry is the Guardian spooler-location name associated with the alias default.

Both files contain lines with the following fields separated by white space:

1. The alias (required). The first alias need not be named default.
2. The Guardian spooler-location name recognized by the spooler (required). Guardian spooler-location names are not case-sensitive, so you can write the spooler-location name with lowercase letters.
3. The spooler-supervisor process name (optional). If the spooler-supervisor process name is not specified, the system assumes the name $SPLS. Guardian process names are not case-sensitive, so you can write the spooler-supervisor process name with lowercase letters.

The /etc/printcap.sample file contains the alias definitions shown in Figure 10-1. You can edit this sample file to change its Guardian spooler-location names and then make it either the /etc/printcap file or a user's printcap file.

Figure 10-1. Sample /etc/printcap File /etc/printcap.sample

```
default             \COMM.$S.#DEFAULT
ps                  \COMM.$S.#POST7
titan               \COMM.$S.#TITAN7
land                \COMM.$S.#LAND7
build               \COMM.$S.#BUILD
```

You assign aliases other than that of the default printer so users can refer to printers by names that are more convenient than the Guardian spooler-location names. For example, you might want to use the alias titan in place of the Guardian spooler-location name \COMM.$S.#TITAN7.

When a user enters a print command that specifies an alias, the printing utility checks the directory specified by the HOME environment variable for a printcap file. If a printcap file is found, the printing utility searches it for the specified alias.

If the alias is found in the printcap file, the printing utility maps the alias to the Guardian spooler-location name given in printcap. This behavior allows a user to have a different value for the alias default from the one in the /etc/printcap file;

however, an alias specified in a printcap file is not actually used as a default value in a shell command line. For example, assume that /etc/printcap contains

```
default    \COMM.$S.#DEFAULT
```

and printcap contains

```
default    \FORTY.$S1.#BOOK
```

If the user enters the following OSS shell command:

```
lp file
```

the file *file* is printed at the Guardian spooler-location named \COMM.$S.#DEFAULT, because that command does not specify a printer definition and therefore uses the default definition in /etc/printcap. However, if the user enters the following OSS shell command:

```
lp -D default file
```

the file *file* is printed at the Guardian spooler-location named \FORTY.$S1.#BOOK, because that is the Guardian spooler-location name specified for default in printcap.

When a user enters a print command that specifies an alias, and if either the alias is not found in the printcap file or the printcap file does not exist, the printing utility searches for the alias in the /etc/printcap file in order to map the alias to the Guardian spooler-location name.

Once you have set up an alias for a printer, you can validate this alias by using it in an lp command to print a sample text file, as described in the *Open System Services User's Guide*.

Using a Printer Environment Variable

Assign a value to the environment variable LPDEST or PRINTER to indicate the name of the default printer for a user. You can specify the same default printer for all users by placing the specification in the /etc/profile file, or you can specify a different default printer for each user by placing the specification in the .profile file.

For example, the /etc/printcap file in Figure 10-1 on page 10-3 defines the printer with the alias default and the Guardian spooler-location name \COMM.$S.#DEFAULT as the default printer. To set the environment variable LPDEST for a specific user to the alias default, so that the default printer for that user is \COMM.$S.#DEFAULT, place the following specification in that user's .profile file:

```
export LPDEST=default
```

11 Managing Problems

Most operational problems are easily resolved by following the recovery recommendations listed in Appendix A, Messages. However, some of the messages indicate that a problem should be reported to HP. This section discusses that possibility.

Problem-Reporting Procedures

Your site should have a formal procedure for reporting problems detected in its own software or in HP software. The specific steps in your site's reporting procedure will vary according to your location and site management practices. However, one step is always necessary: you must know the version of an installed file in order to report a problem with it. The following subsection provides some hints on gathering that information for HP products that are used with the OSS environment.

Gathering Version Information About OSS Files

OSS files are either nonexecutable or executable. Executable program files for HP products always contain product-version information; executable OSS scripts sometimes contain product-version information. Nonexecutable files cannot contain such information.

You can determine whether a Guardian file is executable by using specifying the Guardian filename in a FILEINFO command; the FILEINFO command returns a file code (100, 101, 180, 700, 800, and so forth) indicating the type of file. See the *File Utiliity Program (FUP) Reference Manual* for a list of file codes and what they mean.

You can determine whether a file in the OSS file system is a program file using the OSS shell `file` command. For example, the following command shows that the OSS shell `cd` command file is a text file (a script) and is unlikely to contain product-version information:

```
file /bin/cd /bin/ls /bin/ipcs
  /bin/cd:      commands text
  /bin/ls:      ELF object format,executable,NonStop OSS target
  /bin/ipcs:    TNS object format,executable,axcel region,
                binder region,...
```

For more information on the OSS shell `file` command, see the `file(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

When a problem occurs with a nonexecutable file such as a reference page, use the `ls -al` command to determine the creation timestamp of the file. Report the creation timestamp of the file to HP as the file's product-version information.

When a problem occurs with an executable file, the method you use to obtain product-version information depends on the kind of file involved.The possible methods are:

- For a product with a Subsystem Control Facility (SCF) module, such as the OSS Monitor, you can use the SCF VERSION SUBSYS, VERSION MON, and VERSION PROCESS Commands on page 12-82.
- For other files, you can use the Guardian VPROC utility.

VPROC is the primary tool for gathering product-version information about an executable file. VPROC and its use in the Guardian environment are completely described in the *Guardian User's Guide*. VPROC can also be used from the OSS shell through the `vproc` command or the `gtacl` command.

For example, to obtain information about an OSS-related file in the Guardian file system such as the PINSTALL utility, you can enter the following OSS command:

```
vproc /G/SYSTEM/ZOSS/PINSTALL
```

That command produces a display similar to the following:

```
VPROC-T9617D31-(14 APR 95) SYSTEM \NODE Date 18 DEC 1995, 12:12:45
COPYRIGHT TANDEM COMPUTERS INCORPORATED 1986 - 1995

/G/SYSTEM/ZOSS/PINSTALL
     Binder timestamp:  17MAR95 15:38:44
    Version procedure:  T8626D30_26MAY95_OSSUTL_AAB
           Target CPU:  UNSPECIFIED
      AXCEL timestamp:  17MAR95 15:39:02
```

For help with interpreting the information in VPROC displays, see the `vproc(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.

To use the `vproc` command, you must first know the location of the file for which you want information. To locate a Guardian file, follow the procedure in the *Guardian User's Guide*. To locate an OSS file, use the `find` command. (You can also use the `whence` or `type` command to find a file if you are interested only in those files accessible through your PATH environment variable values.)

For example, to determine the location of the OSS shell file itself (`sh`) and then display the product-version information for that file, enter:

```
find / -name sh -W NOE -W NOG
```

This command returns:

```
find:
   .
   .
   .
/bin/sh
```

Next, enter the following `gtacl` command to use the Guardian VPROC utility directly:

```
gtacl -p vproc '/bin/sh'
```

This command produces a display similar to the following:

```
VPROC-T9617D31-(14 APR 95) SYSTEM \NODE Date 18 DEC 1995, 11:21:14
COPYRIGHT TANDEM COMPUTERS INCORPORATED 1986 - 1995

/bin/sh
     Binder timestamp:  14MAR95 16:32:52
    Version procedure:  T8626D30_26MAY95_OSSUTL_AAB
           Target CPU:  UNSPECIFIED
      AXCEL timestamp:  14MAR95 16:33:22
```

The same information appears if you use the OSS version of the `vproc` command, as shown in the following example:

```
vproc /bin/sh
VPROC-T9617D31-(14 APR 95) SYSTEM \NODE Date 18 DEC 1995, 11:21:14
COPYRIGHT TANDEM COMPUTERS INCORPORATED 1986 - 1995

/bin/sh
     Binder timestamp:  14MAR95 16:32:52
    Version procedure:  T8626D30_26MAY95_OSSUTL_AAB
           Target CPU:  UNSPECIFIED
      AXCEL timestamp:  14MAR95 16:33:22
```

12 Open System Services Monitor

The Open System Services (OSS) Monitor enables you to perform operations on filesets, OSS servers, and itself. Those operations are described in detail in Section 5, Managing Filesets, Section 4, Managing Servers, and Section 2, Operating the OSS Environment. This section provides:

OSS Monitor Overview

The OSS Monitor is a Guardian process that configures and administers the OSS environment. The system manager issues commands to the OSS Monitor through the Subsystem Control Facility (SCF) OSS product module (OSS PM) to:

- Make objects available for use
- Make objects unavailable for use
- Alter the attributes of objects

The OSS Monitor uses the Subsystem Programmatic Interface (SPI) to communicate with SCF and an internal protocol to communicate with the OSS servers.

Note. HP does not support customer use of SPI tokens for programmatic control of the OSS Monitor. OSS Monitor tokens are not documented.

All SCF commands pass through the Subsystem Control Point (SCP) to reach the OSS Monitor.

The OSS PM is part of the OSS product. The installation procedure installs this SCF product module in subvolume $SYSTEM.SYSTEM, the default location for any SCF product module.

See the *SCF Reference Manual for SCP* for more information about SCP and the *SCF Reference Manual for G-Series RVUs* or the *SCF Reference Manual for H-Series RVUs* for information about SCF commands used by more than one SCF product module. The OSS Monitor and its OSS PM provide the features described under OSS Monitor Features on page 12-1.

OSS Monitor Features

The following subsections discuss these features of the OSS Monitor:

Online Help Facility

The OSS PM includes a context-sensitive online help facility similar to that provided in other SCF product modules. This help facility describes:

- The syntax and semantics of the OSS Monitor SCF commands, objects, and object states
- Each of the error messages returned by the OSS Monitor
- The possible causes of each error
- The recommended action to recover from each error

You invoke the help facility from an SCF prompt by entering HELP OSS. This command produces the display shown in Figure 12-1.

Figure 12-1. SCF HELP OSS Display

```
The Open Systems Services (OSS) subsystem is HP's open
computing interface implementation.  It provides an open
interface to the NonStop operating system for supporting
portable applications.  OSS conforms to the POSIX.1 and POSIX.2
standards, and partially conforms to the XPG4 specifications.

The OSS SCF commands and their applicable object types are
summarized in the following table:

+-------------------------------------------------------+
|           |           |          |           |  MON    |
|  Cmd/Obj  |  FILESET  |  NULL    |  SERVER   |  PROCESS|
|           |           |          |           |  SUBSYS |
|===========|===========|==========|===========|=========|
| ADD       |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| ALTER     |     X     |          |     X     |    X    |
|-----------|-----------|----------|-----------|---------|
| CONTROL   |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| DELETE    |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| DIAGNOSE  |     X     |          |           |         |
|-----------|-----------|----------|-----------|---------|
| INFO      |     X     |          |     X     |    X    |
|-----------|-----------|----------|-----------|---------|
| NAMES     |     X     |    X     |     X     |    X    |
|-----------|-----------|----------|-----------|---------|
| RENAME    |     X     |          |           |         |
|-----------|-----------|----------|-----------|---------|
| START     |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| STATUS    |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| STOP      |     X     |          |     X     |         |
|-----------|-----------|----------|-----------|---------|
| VERSION   |           |    X     |           |    X    |
+-------------------------------------------------------+
```

You can get more help for the OSS Monitor by entering HELP at an SCF prompt, then entering OSS at the HELP prompt. This action displays the menu shown in Figure 12-2 on page 12-3.

Figure 12-2. SCF HELP Command OSS Menu

```
+-----------------------------OSS Menu-------------------------------------+
|                                                                          |
| Commands:                                                                |
| ADD       ALTER     CONTROL  DELETE   DIAGNOSE   INFO   NAMES   RENAME   |
| START     STATUS    STOP     VERSION                                     |
|                                                                          |
| Objects:                                                                 |
| FILESET   NULL   MON      PROCESS   SERVER   SUBSYS                      |
| Attributes:                                                              |
|                                                                          |
|                                                                          |
| Error Numbers:                                                           |
| [ E | W ] [ - ] <integer>                                                |
|                                                                          |
+-----------------------Next Menu Selection-------------------------------+
|                                                                          |
| Options:                                                                 |
| QUIT  MAIN  RETURN                                                       |
|                                                                          |
+--------------------------------------------------------------------------+
Enter a command, an object, an error, or an option:
HELP OSS :
```

The menu allows you to request information about OSS object types and commands. You request information about a specific object type or command by entering the object type or command name.

For example, if you enter FILESET, the menu shown in Figure 12-3 is displayed for that OSS object type. The prompt also changes to include the name of the object type; in this example, the prompt becomes:

```
HELP OSS <command> FILESET:
```

Figure 12-3. SCF HELP OSS FILESET Menu

```
+-------------------------OSS FILESET Menu---------------------------------+
|                                                                          |
| Commands                                                                 |
| ADD       ALTER     CONTROL   DELETE    DIAGNOSE   INFO   NAMES    RENAME |
| START     STATUS    STOP                                                 |
|                                                                          |
+-----------------------Next Menu Selection-------------------------------+
|                                                                          |
| Options:                                                                 |
| QUIT  MAIN  RETURN                                                       |
|                                                                          |
+--------------------------------------------------------------------------+
Enter a command, or an option:
HELP OSS <command> FILESET:
```

If you then enter the name of the START command, a display similar to that shown in Figure 12-4 on page 12-4 appears for that command.

Figure 12-4. SCF Help OSS START FILESET Display

```
The START FILESET command makes an existing fileset available to OSS
users.  This action is also known as mounting a fileset.


Syntax of START FILESET:

START [ /OUT <filename>/ ] FILESET <filesetname>


.
.
.
Considerations for START FILESET:

.
.
.
More text? ([Y],N)
```

If you answer “Y” to the prompt for more text, you get more text or, if there is no more relevant text, you are returned to the previous screen (in this case, the SCF HELP command OSS FILESET menu).

The HELP OSS command also returns brief definitions of objects. For example, if you enter the following command at an SCF prompt:

```
HELP OSS FILESET
```

the following PM help text is returned:

```
FILESET is a collection of related files.
```

When you enter the following command at an SCF prompt, a display similar to the one shown for OSS Monitor error 00006 in Figure 12-5 on page 12-4 appears:

```
HELP OSS error_number
```

error_number

specifies the 5-digit number of the error to display an explanation for.

Figure 12-5. SCF HELP OSS Command Sample Error Number Display

```
HELP OSS 00006
OSS E00006 No attribute provided to be altered

Probable Cause

An ALTER command was issued, but without any attributes to be altered.

The OSS Monitor stops processing the command.

Recommended Action

Reissue the command with appropriate attributes and values.
```

Fault Tolerance and Continuous Availability

All servers controlled through the OSS Monitor (except for the $ZTA*nn* transport agent server) can be run as fault-tolerant process pairs. The OSS Monitor itself can be run as a persistent process, as described in Starting the OSS Monitor as a Persistent Process on page 2-9.

In the unlikely event that both the primary and backup OSS name server processes fail, the OSS Monitor restarts an OSS name server and restarts any filesets that that OSS name server was managing. If the OSS name server is not running with a backup process and fails because of a processor halt, the OSS Monitor restarts the OSS name server when the processor becomes available again.

The OSS Monitor remembers the state of all filesets (DIAGNOSING, STARTED, STOPPED, or UNKNOWN) if it stops and is restarted, even after a system load.

Software Requirements

SCF is required for the OSS environment, and the version of SCF must be compatible with that of the OSS PM. You can determine the versions of SCF and the OSS PM on your system by using the SCF VERSION command described in VERSION SUBSYS, VERSION MON, and VERSION PROCESS Commands on page 12-82.

Localization

The OSS PM does not use the National Language Support (NLS) product. All error messages and help text are displayed in English.

Networking and Distributed Processing

The OSS Monitor controls one local NonStop S-series or NonStop NS-series node.

Error Handling

The OSS PM defines a set of SPI error messages specific to the OSS environment that are returned from the OSS Monitor. For normal successful completion of all commands, the standard SPI status value of ZSPI^ERR^OK is returned.

Some OSS Monitor commands require that the OSS Monitor access its configuration files. If the OSS Monitor cannot access these files or if they contain invalid data, an error is returned.

For each command involving a fileset, the OSS Monitor first checks whether the fileset exists by searching the ZOSSFSET file for an entry whose fileset name matches the desired name. If the fileset name is successfully located in the ZOSSFSET file, the OSS Monitor then verifies that the current object state of the fileset is compatible with the initial state required by the command being executed. If the fileset is not successfully located, the appropriate SPI error is returned.

If the OSS Monitor encounters a serious internal inconsistency, it attempts to issue the “internal error” message (message E00001). For additional information about OSS messages, see Appendix A, Messages.

Subsystem and Process Attributes

For the OSS Monitor:

- The subsystem ID is 143.
- The device type is 24.
- The device subtype is 0.
- The name of the program file is $SYSTEM.SYS*nn*.OSSMON.

OSS Monitor SCF Command Reference Information

Use the SCF commands described on the indicated pages to manage the OSS Monitor:

ADD FILESET Command

The ADD FILESET command adds a fileset to the configuration administered through the OSS Monitor. This command affects the contents of the ZOSSFSET file.

The syntax of the ADD FILESET command is:

```
ADD [ /OUT filename1/ ] FILESET [$ZPMON.]filesetname

      , CATALOG volume

      , MNTPOINT { "directory" | 'directory' }

      , POOL filename2

    [ , AUDITENABLED { ON | OFF } ]

    [ , BUFFERED { CREATE | LOG | NONE } ]

    [ , DEVICELABEL devicelabel ]

    [ , DESIREDSTATE { STARTED | STOPPED } ]

    [ , FSCKCPU processor ]

    [ , FTIOMODE
              { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP }

    [ , MAXDIRTYINODETIME seconds2 ]

    [ , MAXINODES maxinodes ]

    [ , NAMESERVER servername  ]

    [ , NFSPOOL kbytes ]

    [ , NFSTIMEOUT seconds ]

    [ , NORMALIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP |
              OSSBUFFEREDCP | DP2BUFFERED | OSSBUFFERED }

    [ , REPORT filename3 ]
```

`OUT filename1`

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]filesetname`

specifies the name of the OSS fileset to be added. *filesetname* can contain up to 32 alphabetic or numeric characters. The first character must be a letter.

Only names not currently defined in the ZOSSFSET file are valid values. Fileset names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

`CATALOG` *`volume`*

specifies the name of the Guardian disk volume to contain the catalog files for the fileset.

Disk volume names are not case-sensitive. The volume specified should be mirrored.

If the BUFFERED CREATE option is specified, any value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

`MNTPOINT { "`*`directory`*`" | '`*`directory`*`' }`

specifies the absolute pathname of the OSS directory that serves as the mount point for the fileset. The specified directory must already exist in the OSS file system.

The pathname specified can contain extra `/`, `.`, or `..` characters; such characters are removed during normalization of the pathname. Do not include symbolic links in mount point pathname specifications. Normalization of pathnames for mount points does not include resolution of symbolic links.

The quotation marks are required. If a pathname contains a quotation mark, either specify that quotation mark twice or use the alternate set of marks to delimit the value.

Directory names are case-sensitive.

`POOL` *`filename2`*

specifies the Guardian filename of the storage-pool file that contains the volume list to use for the fileset. The specified file must exist and must reside on the same Guardian volume and subvolume as the ZOSSFSET and ZOSSSERV files.

Storage-pool filenames are not case-sensitive.

If the BUFFERED CREATE option is specified, the value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

You can specify the following options in any order:

`AUDITENABLED { ON | OFF }`

specifies whether the fileset is audited (ON) or not audited (OFF).

ON

Safeguard global auditing controls govern which objects are audited.

OFF

Security-sensitive operations against this fileset are not logged in the Safeguard security audit trail. This is the default value.

BUFFERED { CREATE | LOG | NONE }

specifies the amount of catalog write buffering used for the fileset.

CREATE	When a request or transaction requires a write to the PXINODE or PXLINK file, buffer the corresponding write to the PXLOG file. Also use the fast file-creation feature for writing new file labels. The CREATE option requires the fileset and its catalog files to reside on the same disk volume.
LOG	When a request or transaction requires a write to the PXINODE or PXLINK file, buffer the corresponding write to the PXLOG file. Do not use fast file creation. Do not restrict the fileset and its catalog files to the same disk volume. This is the default specification.
NONE	Do not buffer anything. Any request or transaction that requires a write to the PXINODE or PXLINK file also requires a write to the PXLOG file. Do not use fast file creation. Do not restrict the fileset and its catalog files to the same disk volume.

If the BUFFERED option is omitted, the default specification is BUFFERED LOG.

If the BUFFERED CREATE option is specified, any value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

DESIREDSTATE { STARTED | STOPPED }

specifies the desired end state of the fileset when the SUBSYS AUTOSTART AUTO feature is used.

STARTED

Attempts are made to start the fileset when a system load occurs. Attempts are made to start the fileset when the OSS environment is restarted, unless the fileset has been manually stopped.

STOPPED

No attempt is made to start the fileset when a system load occurs. No attempt is made to start the fileset during a restart of the OSS environment, unless the fileset has been manually started.

This is the default value.

DEVICELABEL *devicelabel*

specifies the device label assigned to this fileset. The specified label must not already be in use by another fileset in the database. The device label has 6 characters (the first of which is always zero) and consists of numeric characters (0 - 9) and uppercase letters (A - Z), excluding E, I, O, and U. The root fileset always has a device label of 000000. Valid device labels are in the range 000001 through 0ZZZZZ. The default value for DEVICELABEL is the lowest unused device label in the database.

FSCKCPU *processor*

specifies the processor number of the processor that should run the FSCK program when a fileset recovery is automatically initiated, where *processor* is in the range 0 through 15 or is -1. The value -1 indicates that the processor that runs the primary copy of DP2 for the fileset catalog should be used.

If the processor indicated by a nonnegative *processor* value is not available, then the processor specified for the SUBSYS FSCKCPU option is used. If that processor is also unavailable, the processor in which the OSS Monitor is running is used.

If an FSCKCPU option is omitted from the FILESET configuration, then the processor specified for the SUBSYS FSCKCPU option is used when a fileset recovery is automatically started.

FTIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP }

specifies the input/output buffering and fault tolerance for application file opens that use the O_SYNC option:

UNBUFFEREDCP	Use unbuffered input/output with checkpointing. This behavior provides maximum fault tolerance but with reduced performance.
DP2BUFFEREDCP	Use disk-process-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than UNBUFFEREDCP. DP2 buffers file data and checkpoints the file state to its backup process to ensure recovery from single failures.

OSSBUFFEREDCP Use OSS-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than DP2BUFFEREDCP. OSS filesystem processes and DP2 share responsibility for buffering file data; OSS provides the buffering whenever possible. DP2 checkpoints the file state to its backup process to ensure recovery from single failures.

If FTIOMODE is not specified, the default behavior is UNBUFFEREDCP.

MAXDIRTYINODETIME *seconds2*

specifies the maximum number of seconds that an inode of the fileset remain in the OSS name server's inode cache before being flushed to disk. *seconds2* must be a value in the range 1 through 600; the default value is 30.

MAXINODES *maxinodes*

specifies the approximate maximum number of inodes that can be created for the fileset. *maxinodes* must be a value in the range 100000 through 2200000; the default value is 500000.

NAMESERVER *servername*

specifies the server name of the OSS name server that should administer the fileset. The specified server must already be part of the OSS configuration.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

If the NAMESERVER parameter is omitted, the default server is the OSS name server for the root fileset, #ZPNS.

NFSPOOL *kbytes*

specifies the number of kilobytes that the OSS name server uses for buffers for nonretryable Network File System (NFS) operations for the fileset.

Possible values are 4 through 128 kilobytes. The default value is 16 kilobytes.

See the *Open System Services NFS Management and Operations Guide* for more information about NFS.

NFSTIMEOUT *seconds*

specifies the number of seconds that the OSS name server retains the results of nonretryable Network File System (NFS) operations for the fileset.

Valid values are in the range 60 through 300. The default value is 120.

See the *Open System Services NFS Management and Operations Guide* for more information about NFS.

```
NORMALIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP |
DP2BUFFERED | OSSBUFFERED }
```

specifies the input/output buffering and fault tolerance for application file opens that do not use the `O_SYNC` option:

`UNBUFFEREDCP`	Use unbuffered input/output with checkpointing. This behavior provides maximum fault tolerance but with reduced performance.
`DP2BUFFEREDCP`	Use disk-process-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than UNBUFFEREDCP. DP2 buffers file data and checkpoints the file state to its backup process to ensure recovery from single failures.
`OSSBUFFEREDCP`	Use OSS-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than DP2BUFFEREDCP. OSS filesystem processes and DP2 share responsibility for buffering file data; OSS provides the buffering whenever possible. DP2 checkpoints the file state to its backup process to ensure recovery from single failures.
`DP2BUFFERED`	Use disk-process-buffered input/output without checkpointing. This behavior provides no fault tolerance, but better performance than DPBUFFEREDCP.
`OSSBUFFERED`	Use OSS-buffered input/output without checkpointing. This behavior provides no fault tolerance, but better performance than OSSBUFFEREDCP.

If NORMALIOMODE is not specified, the default behavior is OSSBUFFEREDCP.

`REPORT` *`filename3`*

specifies the Guardian filename for the report file to be generated when FSCK performs an automatic recovery for this fileset. The file specified must be a Guardian spooler location.

If this option is omitted, FSCK uses the Guardian filename specified for the SUBSYS REPORT option.

Considerations

- The ADD FILESET command applies only to filesets other than the root fileset.
- The ADD FILESET command can be used only by super-group users (255,*nnn*).
- You cannot use the ADD FILESET command on filesets associated with an OSS name server that is not currently configured.

- You can use the ADD FILESET command on filesets associated with an OSS name server that is not currently running.
- You should not add all filesets to one OSS name server.
- A newly added fileset has no catalog files. The catalog files (PXINODE, PXLINK, and PXLOG) are created the first time the fileset is started.
- The fileset READONLY attribute cannot be set when a fileset is first added. New filesets always have a READONLY attribute value of FALSE. This attribute can be changed with the SCF ALTER FILESET command.
- The pathname specified by the MNTPOINT option must be an existing directory, but the OSS Monitor does not validate this until an attempt is made to start the fileset. Once validated, a normalized version of the pathname is used for the mount point for sorting purposes, so the apparent pathname for the mount point in an INFO FILESET command might not appear to be the same as the specified pathname.
- The MAXDIRTYINODETIME option is meaningful only for filesets that use the BUFFERED LOG option.

  A fileset with an updated (flushed) inode cache is considered “clean” instead of “dirty” and does not need recovery after a failure. The more often the inode cache is flushed, the less likely a fileset is to be corrupted by a failure and to need recovery after the failure.

  The larger the value specified for *seconds2*, the more likely that a fileset recovery is needed after a failure, but the faster fileset access becomes because fewer disk writes occur to update the cache from memory. The smaller the value for *seconds2*, the less likely that a fileset recovery is needed after a failure, but the slower fileset access becomes because more disk writes occur to update the cache from memory.

  Fileset recovery delays subsequent availability of the fileset, so the tradeoff for slightly faster current access is increased probability of delayed access after a failure.
- The MAXINODES value specifies an upper bound on the number of inodes that can be created in the fileset. It does not guarantee that MAXINODES number of inodes will be created in the fileset. Specifying a large MAXINODES value increases the fileset recovery time in the case of an outage. HP recommends that you specify a MAXINODES value less than or equal to 1000000.
- FTIOMODE must have a setting equal to or higher than the setting of NORMALIOMODE. NORMALIOMODE settings are ranked, from highest to lowest:

  ```
  UNBUFFEREDCP
  DP2BUFFEREDCP
  OSSBUFFEREDCP
  DP2BUFFERED
  OSSBUFFERED
  ```

- If an FTIOMODE setting of OSSBUFFEREDCP or a NORMALIOMODE setting of OSSBUFFEREDCP or OSSBUFFERED is used, the OSS filesystem buffers the data unless there are file opens from processes in more than one processor and at least one of the file opens has write permission. In that case, DP2BUFFEREDCP behavior occurs instead of OSSBUFFEREDCP behavior or DP2BUFFERED behavior occurs instead of OSSBUFFERED behavior.

Examples

- To add the fileset USER1 with the storage-pool file ZOSSPOOL using the volume catalog on $DATA1 and the mount point `/user1`, enter the following command:

  ```
  ADD /OUT CMDLOG/ FILESET $ZPMON.USER1, CATALOG $DATA1, &
  POOL ZOSSPOOL, MNTPOINT "/user1", &
  BUFFERED LOG, NAMESERVER #ZPNS
  ```

  This command:

  - Sends informational messages to the file CMDLOG.
  - Specifies the fileset name as USER1 and the mount point as `/user1`.
  - Specifies that files created in the fileset USER1 are stored on the disk volumes listed in the storage-pool file ZOSSPOOL.
  - Specifies that the catalog files for the fileset `/user1` are stored on the disk volume $DATA1.
  - Assigns the fileset USER1 to the OSS name server process $ZPNS, which has the server name #ZPNS.
  - Buffers only output to the PXLOG file.

- To add the fileset USER2 with the storage-pool file ZOSSPOOL using the volume catalog on $DATA2 and the mount point `/user2`, enter the following command:

  ```
  ADD /OUT $S.#USR/ FILESET $ZPMON.USER2, CATALOG $DATA2, &
  POOL ZOSSPOOL, MNTPOINT "/user2", NAMESERVER #ZPNS, &
  NFSTIMEOUT 60, BUFFERED NONE
  ```

  This command:

  - Sends informational messages to the spooler location $S.#USR.
  - Specifies the fileset name as USER2 and the mount point as `/user2`.
  - Specifies that files created in the fileset USER2 are stored on the disk volumes listed in the storage-pool file ZOSSPOOL.
  - Specifies that the catalog files for the fileset USER2 are stored on the disk volume $DATA2.
  - Assigns the fileset USER2 to the OSS name server process $ZPNS, which has the server name #ZPNS.

- Specifies that a saved reply for an NFS request to the fileset USER2 is considered to be obsolete after 60 seconds.
- Specifies that no catalog writes are buffered.

- To add the fileset USER3 with the storage-pool file ZOSSPOOL using the volume catalog on $DATA2 and the mount point `/user3`, enter the following command:

```
ADD /OUT $S.#USR/ FILESET $ZPMON.USER3, CATALOG $DATA2, &
POOL ZOSSPOOL, MNTPOINT "/user3", DESIREDSTATE STARTED, &
MAXDIRTYINODETIME 10, FSCKCPU 5, REPORT $S.#USER3
```

  This command:

  - Sends informational messages from the command to the spooler location $S.#USR.
  - Specifies the fileset name as USER3 and the mount point as `/user3`.
  - Specifies that files created in the fileset USER3 are stored on the disk volumes listed in the storage-pool file ZOSSPOOL.
  - Specifies that the catalog files for the fileset USER3 are stored on the disk volume $DATA2.
  - Assigns the fileset USER3 to the OSS name server process $ZPNS by default.
  - Uses the BUFFERED LOG specification by default.
  - Specifies that the inode cache for the fileset USER3 should be flushed to disk after approximately 10 seconds to make its recovery faster by decreasing the time when the cache is considered dirty.
  - Performs automatic recovery of the fileset USER3 using a copy of the FSCK program in processor 5 after a shutdown or failure. The recovered fileset is left in a started state.
  - Sends FSCK output after an automatic recovery to the spooler $S.#USER3.

- To add the fileset USER1 to the storage-pool configuration file USRPOOL using the volume catalog on $DATA, the mount point `/user1`, and allowing a maximum of 600000 inodes, enter the following command:

```
ADD /OUT CMDLOG/ FILESET $ZPMON.USER1, CATALOG $DATA, POOL
USRPOOL, MNTPOINT "/user1", MAXINODES 600000
```

ADD SERVER Command

The ADD SERVER command adds an OSS server to the configuration administered through the OSS Monitor.

The syntax of the ADD SERVER command is:

```
ADD [ /OUT filename/ ] SERVER server_processname
      [ , TYPE NAME  ]
      , CPU primary_processor
      [ , AUTORESTART ntimes  ]
      [ , BACKUPCPU backup_processor  ]
      [ , BACKUPCPUOK { TRUE | FALSE }  ]
      [ , INODECACHE size1  ]
      [ , LINKCACHE size2  ]
      [ , MAXWAITTIME seconds2  ]
      [ , SQLTIMEOUT seconds  ]
```

`OUT` *`filename`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`SERVER` *`server_processname`*

specifies the server being added. *`server_processname`* has the following form:

`[ $ZPMON.]`*`servername`*

`servername`

specifies the name of the server to add. *`servername`* must begin with a pound sign (#) followed by a letter, which can be followed by 0 through 5 letters or digits. Only names not currently defined in the ZOSSSERV file are valid values. Server names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

`TYPE NAME`

specifies that an OSS name server is being added. Currently, OSS name servers are the only OSS servers that can be added.

CPU *primary_processor*

specifies the processor number of the processor where the primary server process will run. *primary_processor* is an integer in the range 0 through 15.

You can specify the following options in any order:

AUTORESTART *ntimes*

specifies the persistence count of the server, where *ntimes* is an integer value in the range 0 through 10. The persistence count is the number of times the OSS Monitor will automatically restart the server during a 10-minute period.

The persistence count helps to prevent excessive attempts at restarting or an endless loop during restart. The OSS Monitor counts the number of restart attempts for each server during the previous 10 minutes; this number is the current persistence count.

Once the current persistence count for a server reaches the number specified for it for AUTORESTART, the OSS Monitor no longer attempts to restart it and generates an EMS event so that the operator can attempt to identify and correct a possible problem. After the problem is corrected, the operator must take an action that causes the current persistence count to be reset to 0; otherwise, the server and any associated filesets will not be automatically restarted.

A server's current persistence count is reset to 0 at the following times:

- When the OSS Monitor starts or restarts
- When an SCF STOP SERVER, START SERVER, or ALTER SERVER command for the affected server is completed
- When a START FILESET command for a fileset managed by the affected OSS name server is completed
- After the server has run for 10 minutes without being restarted

The current persistence count for each server is incremented when that server terminates abnormally or is stopped by something other than SCF. The current persistence count is not incremented because of processor failure.

When the AUTORESTART option is omitted from the ADD SERVER command, the default specification is AUTORESTART 3.

BACKUPCPU *backup_processor*

specifies the processor number of the processor where the backup server process will run. *backup_processor* is either an integer in the range 0 through 15 or equal to -1.

The value -1 specifies that an OSS name server.

If the BACKUPCPU parameter is omitted, the processor for the backup server process is unchanged.

BACKUPCPUOK { TRUE | FALSE }

specifies whether the server should be started in its configured backup processor when its configured primary processor is unavailable and the server is being restarted automatically.

When the server is started in its configured backup processor and its configured primary processor subsequently becomes available, the server automatically switches to its primary processor.

When this option is omitted from the ADD SERVER command, the default specification is BACKUPCPUOK TRUE.

INODECACHE *size1*

specifies the number of entries permitted in the inode cache used by an OSS name server. *size1* is in the range 128 through 500000. The default value is 4096 entries.

LINKCACHE *size2*

specifies the number of entries permitted in the link cache used by an OSS name server. *size2* can have a value in the range 128 through 500000. The default value is 4096 entries.

MAXWAITTIME *seconds2*

specifies the maximum number of seconds that the OSS Monitor waits for the OSS name server's primary processor to become available during an automatic restart of the server; *seconds2* must be in the range 0 through 32767. A value of 0 means that the OSS Monitor does not wait for the primary processor to become available.

If the BACKUPCPUOK attribute for the server is TRUE and the primary processor does not become available after the specified period has elapsed, the OSS Monitor attempts to start the server in its backup processor.

When the MAXWAITTIME option is omitted from the ADD SERVER command, the default specification is MAXWAITTIME 0.

SQLTIMEOUT *seconds*

specifies the number of seconds an OSS name server waits for a response from SQLCAT to a request. *seconds* can have a value in the range 60 through 300. The default value is 60.

Considerations

- The ADD SERVER command does not start the server; it merely stores information about the server within the OSS Monitor configuration file. An OSS name server is started when the first fileset managed by that OSS name server is mounted. The

OSS sockets local server, OSS message-queue server, and OSS transport agent servers are started by the START SERVER command.

- You must be a member of the super group (255, *nnn*) to use this command.
- The ADD SERVER command is intended to be used for OSS name servers of filesets other than the root fileset.
- When Open System Services is first installed, the OSS Monitor adds the default OSS name server for the root fileset the first time the OSS Monitor is run. The OSS sockets local server, OSS message-queue server, and OSS transport agent servers are added, if needed, at the startup of the OSS Monitor process.

Example

To add the OSS name server #ZPNS1 to the system, with processor 0 as the primary processor and processor 1 as the backup processor or as the primary processor when processor 0 is unavailable after 30 seconds during a automatic restart, a maximum of two attempts at restart, a maximum of 4096 inode cache entries and 4096 link cache entries, and a timeout of 60 seconds for responses from SQLCAT, enter the following command:

```
ADD SERVER #ZPNS1, CPU 0, BACKUPCPU 1, INODECACHE 4096, &
LINKCACHE 4096, SQLTIMEOUT 60, BACKUPCPUOK TRUE, &
MAXWAITTIME 30, AUTORESTART 2
```

ALTER FILESET Command

The ALTER FILESET command changes the configuration of a fileset administered through the OSS Monitor. The information entered in the command is added to or altered in the ZOSSFSET file.

The syntax of the ALTER FILESET command is:

```
ALTER [ /OUT filename1/ ] FILESET [$ZPMON.]filesetname

      { [ , AUDITENABLED { ON | OFF } ]

        [ , BUFFERED { NONE | LOG | CREATE } ]

        [ , CATALOG volume ]

        [ , DESIREDSTATE { STARTED | STOPPED } ]

        [ , FSCKCPU processor ]

       [ , FTIOMODE
             { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP }

        [ , MAXDIRTYINODETIME seconds2 ]

        [ , MNTPOINT { "directory" | 'directory' } ]

        [ , NAMESERVER servername ]

        [ , NFSPOOL kbytes ]

        [ , NFSTIMEOUT seconds ]

       [ , NORMALIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP |
             OSSBUFFEREDCP | DP2BUFFERED | OSSBUFFERED }

        [ , POOL filename2 ]

        [ , READONLY { FALSE | TRUE } ]

        [ , REPORT filename3 ] }
```

`OUT` *`filename1`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

specifies the name of the OSS fileset to be altered. Only names currently defined in the ZOSSFSET file are valid values.

Fileset names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order. You must specify at least one option:

`AUDITENABLED { ON | OFF }`

specifies whether the fileset is audited (ON) or not audited (OFF).

`ON`

Safeguard global auditing controls govern which objects are audited.

`OFF`

Security-sensitive operations against this fileset are not logged in the Safeguard security audit trail.

Only the super ID (255,255) can change the value of AUDITENABLED. If the AUDITENABLED option is omitted, the AUDITENABLED option setting is unchanged.

`BUFFERED { NONE | LOG | CREATE }`

specifies the amount of catalog write buffering used for the fileset.

`CREATE`	When a request or transaction requires a write to the PXINODE or PXLINK file, buffer the corresponding write to the PXLOG file. Also use the fast file-creation feature for writing new file labels. The CREATE option requires the fileset and its catalog files to reside on the same disk volume.
`LOG`	When a request or transaction requires a write to the PXINODE or PXLINK file, buffer the corresponding write to the PXLOG file. Do not use fast file creation. Do not restrict the fileset and its catalog files to the same disk volume.
`NONE`	Do not buffer anything. Any request or transaction that requires a write to the PXINODE or PXLINK file also requires a write to the PXLOG file. Do not use fast file creation. Do not restrict the fileset and its catalog files to the same disk volume.

If the BUFFERED CREATE option is specified and the READONLY TRUE option is not specified, any value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

If the BUFFERED CREATE option is specified and the READONLY TRUE option is also specified, the BUFFERED CREATE option is ignored.

If the BUFFERED option is omitted, the BUFFERED option setting is unchanged.

CATALOG *volume*

specifies the name of the Guardian disk volume to contain the catalog files for the fileset. Any existing catalog files are moved to this disk volume; the subvolume of the catalog files is unchanged.

Disk volume names are not case-sensitive.

If the BUFFERED CREATE option is specified and the READONLY TRUE option is not specified, any value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

If the CATALOG parameter is omitted, the catalog volume for the fileset is not changed unless it is affected by the BUFFERED CREATE option.

You must stop the fileset in order to change the value of the CATALOG parameter.

DESIREDSTATE { STARTED | STOPPED }

specifies the desired end state of the fileset when the SUBSYS AUTOSTART AUTO feature is used.

STARTED

Attempts are made to start the fileset when a system load occurs. Attempts are made to start the fileset when the OSS environment is restarted, unless the fileset has been manually stopped.

STOPPED

No attempt is made to start the fileset when a system load occurs. No attempt is made to start the fileset during a restart of the OSS environment, unless the fileset has been manually started.

If the DESIREDSTATE option is omitted, the previous value for the fileset is unchanged.

FSCKCPU *processor*

specifies the processor number of the processor that should run the FSCK program when a fileset recovery is automatically initiated, where *processor* is in the range 0 through 15 or is -1. The value -1 indicates that the processor used by the copy of DP2 for the fileset catalog should be used.

If the processor indicated by a nonnegative *processor* value is not available, then the processor specified for the SUBSYS FSCKCPU option is used. If that processor is also unavailable, the processor in which the OSS Monitor is running is used.

If the FSCKCPU option is omitted, the previous value for the fileset is unchanged.

`FTIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP }`

specifies the input/output buffering and fault tolerance for application file opens that use the `O_SYNC` option:

`UNBUFFEREDCP`	Use unbuffered input/output with checkpointing. This behavior provides maximum fault tolerance but with reduced performance.
`DP2BUFFEREDCP`	Use disk-process-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than UNBUFFEREDCP. DP2 buffers file data and checkpoints the file state to its backup process to ensure recovery from single failures.
`OSSBUFFEREDCP`	Use OSS-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than DP2BUFFEREDCP. OSS filesystem processes and DP2 share responsibility for buffering file data; OSS provides the buffering whenever possible. DP2 checkpoints the file state to its backup process to ensure recovery from single failures.

If FTIOMODE is not specified, the default behavior is UNBUFFEREDCP.

`MAXDIRTYINODETIME` *`seconds2`*

specifies the approximate number of seconds that cached inodes of the fileset remain in the OSS name server's inode cache without being updated. *`seconds2`* must be a value in the range 1 through 600.

When the MAXDIRTYINODETIME option is omitted, the previous value for the fileset is unchanged.

`MAXINODES` *`maxinodes`*

specifies the approximate maximum number of inodes that can be created for the fileset. *`maxinodes`* must be a value in the range 100000 through 2200000. If the MAXINODES option is omitted, the previous value for the fileset is unchanged.

`MNTPOINT { "`*`directory`*`" | '`*`directory`*`' }`

specifies the pathname of the OSS directory that serves as the mount point for the fileset. The specified directory must already exist in the OSS file system.

The quotation marks are required. If a pathname contains a quotation mark, either specify that quotation mark twice or use the alternate set of marks to delimit the value.

Directory names are case-sensitive.

The pathname described in MNTPOINT must be an existing directory, but the OSS Monitor does not validate this until an attempt is made to start the fileset. Do not include symbolic links in mount-point pathname specifications. Normalization of pathnames for mount points does not include resolution of symbolic links.

If the MNTPOINT option is omitted, the mount-point directory for the fileset is not changed.

NAMESERVER *servername*

specifies the server name of the OSS name server that should administer the fileset. Only names currently defined in the ZOSSSERV file are valid values.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

If the NAMESERVER option is omitted, the OSS name server process for the fileset is not changed.

NFSPOOL *kbytes*

specifies the number of kilobytes that the OSS name server uses for buffers for nonretryable Network File System (NFS) operations for the fileset.

Valid values are in the range 4 through 128 kilobytes.

If the NFSPOOL option is omitted, the number of kilobytes that the OSS name server uses for buffers is not changed.

NFSTIMEOUT *seconds*

specifies the number of seconds that the OSS name server retains the results of nonretryable Network File System (NFS) operations for the fileset.

Valid values are in the range 60 through 300.

If the NFSTIMEOUT option is omitted, the timeout interval for the fileset is not changed.

NORMALIOMODE { UNBUFFEREDCP | DP2BUFFEREDCP | OSSBUFFEREDCP | DP2BUFFERED | OSSBUFFERED }

specifies the input/output buffering and fault tolerance for application file opens that do not use the O_SYNC option:

UNBUFFEREDCP	Use unbuffered input/output with checkpointing. This behavior provides maximum fault tolerance but with reduced performance.

DP2BUFFEREDCP	Use disk-process-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than UNBUFFEREDCP. DP2 buffers file data and checkpoints the file state to its backup process to ensure recovery from single failures.
OSSBUFFEREDCP	Use OSS-buffered input/output with checkpointing. This behavior provides fault tolerance for single failures, with better performance than DP2BUFFEREDCP. OSS filesystem processes and DP2 share responsibility for buffering file data; OSS provides the buffering whenever possible. DP2 checkpoints the file state to its backup process to ensure recovery from single failures.
DP2BUFFERED	Use disk-process-buffered input/output without checkpointing. This behavior provides no fault tolerance, but better performance than DPBUFFEREDCP.
OSSBUFFERED	Use OSS-buffered input/output without checkpointing. This behavior provides no fault tolerance, but better performance than OSSBUFFEREDCP.

If NORMALIOMODE is not specified, the default behavior is OSSBUFFEREDCP.

POOL *filename2*

specifies the Guardian filename of the storage-pool file that contains the volume list to use for file creation for the fileset. The specified file must exist and must reside on the same Guardian volume and subvolume as the ZOSSFSET and ZOSSSERV files.

Storage-pool filenames are not case-sensitive.

If the BUFFERED CREATE option is specified and the READONLY TRUE option is not specified, the value specified for the POOL parameter is ignored and the disk volume specified for the CATALOG parameter is used as the creation pool.

If the POOL option is omitted, the storage-pool file for the fileset is not changed unless it is affected by the BUFFERED CREATE option.

READONLY { FALSE | TRUE }

specifies the write permission granted to users of the fileset.

FALSE	Files within the fileset can be created, read, written, and deleted.
TRUE	Files within the fileset can only be read. Using this specification causes a specification of the BUFFERED CREATE option to be ignored.

If the READONLY option is omitted, the write permission granted to users of the fileset is not changed.

REPORT *filename3*

specifies the Guardian filename for the report file to be generated when FSCK performs an automatic recovery for this fileset; *filename3* must be a Guardian spooler location.

If the specified spooler is unavailable, FSCK uses a file-code-180 file (a C language text file) named $SYSTEM.SYS*nn*.ZX*devicelabel* as its outfile; *nn* indicates the currently running system installation and *devicelabel* is the value specified for the DEVICELABEL option of the fileset. If the file $SYSTEM.SYS*nn*.ZX*devicelabel* already exists, FSCK appends its new output to the existing file. The CTOEDIT command can be used to convert the file code-180-file to a file-code-101 EDIT file.

When the REPORT option is omitted, the previous value for the fileset is unchanged.

Considerations

- The ALTER FILESET command can be used only by super-group users (255,*nnn*).
- You can use the ALTER FILESET command on a fileset that is not in the STOPPED state. However, the changes do not take effect until the fileset is stopped and restarted.
- Assigning a new storage-pool file to a fileset has no effect on existing OSS files in the fileset. New OSS files will be created only on disk volumes listed in the new storage-pool file. Disk volumes with existing OSS files in the fileset remain a part of the fileset's storage pool even when they are not listed in the storage-pool file (are not in the creation pool).
- If a fileset is in the STARTED state, you cannot change its OSS name server process.
- The pathname specified by the MNTPOINT option must be an existing directory, but the OSS Monitor does not validate this until an attempt is made to start the fileset. Once validated, a normalized version of the pathname is used for the mount point for sorting purposes, so the apparent pathname for the mount point in an INFO FILESET command might not appear to be the same as the specified pathname.
- The MAXDIRTYINODETIME option is meaningful only for filesets that use the BUFFERED LOG option.

  A fileset with an updated (flushed) inode cache is considered "clean" instead of "dirty" and does not need recovery after a failure. The more often the inode cache is flushed, the less likely a fileset is to be corrupted by a failure and to need recovery after the failure.

The larger the value specified for *seconds2*, the more likely that a fileset recovery is needed after a failure, but the faster fileset access becomes because fewer disk writes occur to update the cache from memory. The smaller the value for *seconds2*, the less likely that a fileset recovery is needed after a failure, but the slower fileset access becomes because more disk writes occur to update the cache from memory.

Fileset recovery delays subsequent availability of the fileset, so the tradeoff for slightly faster current access is increased probability of delayed access after a failure.

- The MAXINODES value specifies an upper bound on the number of inodes that can be created in the fileset. It does not guarantee that MAXINODES number of inodes will be created in the fileset. Specifying a large MAXINODES value increases the fileset recovery time in the case of an outage. HP recommends that you specify a MAXINODES value less than or equal to 1000000.
- FTIOMODE must have a setting equal to or higher than the setting of NORMALIOMODE. NORMALIOMODE settings are ranked, from highest to lowest:

  ```
  UNBUFFEREDCP
  DP2BUFFEREDCP
  OSSBUFFEREDCP
  DP2BUFFERED
  OSSBUFFERED
  ```

- If an FTIOMODE setting of OSSBUFFEREDCP or a NORMALIOMODE setting of OSSBUFFEREDCP or OSSBUFFERED is used, the OSS filesystem buffers the data unless there are file opens from processes in more than one processor and at least one of the file opens has write permission. In that case, DP2BUFFEREDCP behavior occurs instead of OSSBUFFEREDCP behavior or DP2BUFFERED behavior occurs instead of OSSBUFFERED behavior.

Examples

- To alter the read-write fileset USER1 to be a read-only fileset using the volume catalog on $DATA1 and the mount point `/user1`, enter the following command:

  ```
  ALTER /OUT CMDLOG/ FILESET $ZPMON.USER1, BUFFERED LOG, &
  CATALOG $DATA1, &
  MNTPOINT "/user1", NAMESERVER #ZPNS1, NFSTIMEOUT 60, &
  READONLY TRUE
  ```

  This command:

  - Sends informational messages to the file CMDLOG.
  - Specifies a buffered write to the PXLOG file whenever a request or transaction requires a write to the corresponding PXINODE or PXLINK file.
  - Specifies that $DATA contains the catalog files for the fileset.
  - Specifies that `/user1` is the mount point for the fileset.

- Assigns the fileset to the OSS name server process $ZPNS1, which has the server name #ZPNS1.
- Specifies that the OSS name server for the fileset retains the results of nonretryable Network File System (NFS) operations for 60 seconds.
- Specifies that files within the fileset are read-only.
- Causes any previously specified BUFFERED CREATE attribute to be ignored. A fileset to which the READONLY TRUE attribute is assigned is not buffered.

- To change the maximum number of inodes for the fileset USER1 to the value recommended by HP, enter the following command:

```
ALTER FILESET $ZPMON.USER1, MAXINODES 1000000
```

ALTER SERVER Command

The ALTER SERVER command changes the configuration of a server in the set of servers configured for administration through the OSS Monitor. The information entered in the command is added to or altered in the ZOSSSERV file.

This command is not valid for OSS transport agent servers.

The syntax of the ALTER SERVER command is:

```
ALTER  [ /OUT filename/ ] SERVER server_processname
       { [ , AUTORESTART ntimes  ]
         [ , BACKUPCPU backup_processor ]
         [ , BACKUPCPUOK { TRUE | FALSE }  ]
         [ , CPU primary_processor ]
         [ , DESIREDSTATE { STARTED | STOPPED }  ]
         [ , INODECACHE size1 ]
         [ , LINKCACHE  size2 ]
         [ , MAXMQID maxmqid ]
         [ , MAXMSG maxmsg ]
         [ , MAXWAITTIME seconds2  ]
         [ , MSGMQB msgmqb ]
         [ , MSGMSIZE msgmsize ]
         [ , SQLTIMEOUT seconds ]  }
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server being altered. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of the server to be altered. Only names currently defined in the ZOSSSERV file are valid values. The first character of the name must be a pound sign (#).

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order. You must specify at least one option:

AUTORESTART *ntimes*

specifies the persistence count of the server, where *ntimes* is an integer value in the range 0 through 10. The persistence count is the number of times the OSS Monitor will automatically restart the server during a 10-minute period.

The persistence count helps to prevent excessive attempts at restarting or an endless loop during restart. The OSS Monitor counts the number of restart attempts for each server during the previous 10 minutes; this number is the current persistence count.

Once the current persistence count for a server reaches the number specified for it by AUTORESTART, the OSS Monitor no longer attempts to restart the server and generates an EMS event so that the operator can attempt to identify and correct a possible problem. After the problem is corrected, the operator must take an action that causes the current persistence count to be reset to 0; otherwise, the server and any associated filesets will not be automatically restarted.

A server's current persistence count is reset to 0 at the following times:

- When the OSS Monitor starts or restarts
- When an SCF STOP SERVER, START SERVER, or ALTER SERVER command for the affected server is completed
- When a START FILESET command for a fileset managed by the affected OSS name server is completed
- After the server has run for 10 minutes without being restarted

The current persistence count for each server is incremented when that server terminates abnormally or is stopped by something other than SCF. The current persistence count is not incremented because of processor failure.

When the AUTORESTART option is omitted from the ALTER SERVER command, the previous value for the persistence count is unchanged.

BACKUPCPU *backup_processor*

specifies the processor number of the processor where the backup server process runs, where *backup_processor* is either in the range 0 through 15 or equal to -1.

The value -1 specifies that an OSS name server or the OSS sockets local server does not run as a process pair, or that the OSS message-queue server selects as its backup processor the next available processor in the node.

If the BACKUPCPU option is omitted, the processor for the backup server process is unchanged.

BACKUPCPUOK { TRUE | FALSE }

specifies whether the server should be started in its configured backup processor when its configured primary processor is unavailable and the server is being restarted automatically.

When the server is started in its configured backup processor and its configured primary processor subsequently becomes available, the server automatically switches to its primary processor.

When this option is omitted, the value in the server's current configuration is not changed.

CPU *primary_processor*

specifies the processor number of the processor where the primary server process runs, where *primary_processor* is in the range 0 through 15.

If the CPU option is omitted, the processor for the primary server process is unchanged.

DESIREDSTATE { STARTED | STOPPED }

specifies the desired end state of an OSS sockets local server or an OSS message-queue server when the SUBSYS AUTOSTART AUTO feature is used.

STARTED

Attempts are made to start the server when the OSS environment is restarted.

STOPPED

No attempt is made to start the server during a restart of the OSS environment.

If the DESIREDSTATE option is omitted, the previous value for the server is unchanged.

`INODECACHE` *size1*

specifies the number of entries permitted in the inode cache used by the OSS name server.

Possible values are in the range 128 through 500000. The functional upper limit might be smaller than the theoretical maximum, because the upper limit depends upon the amount of memory available when the OSS name server is started.

If the INODECACHE option is omitted, the number of entries permitted in the inode cache is unchanged.

This option is valid only for OSS name server processes.

`LINKCACHE` *size2*

specifies the number of entries permitted in the link cache used by the OSS name server.

Possible values are in the range 128 through 500000. The functional upper limit might be smaller than the theoretical maximum, because the upper limit depends upon the amount of memory available when the OSS name server is started.

If the LINKCACHE option is omitted, the number of entries permitted in the link cache is unchanged.

This option is valid only for OSS name server processes.

`MAXMQID` *maxmqid*

specifies the maximum number of OSS message queue IDs allowed at any time. Valid values are in the range 1 through 1024.

If the MAXMQID option is omitted, the maximum number of queue IDs allowed is unchanged. The default value is the 32.

This option is valid only for OSS message-queue servers.

`MAXMSG` *maxmsg*

specifies the maximum number of messages allowed on all OSS message queues on a system. Valid values are in the range 1 through 16384.

If the MAXMSG option is omitted, the maximum number of messages allowed is unchanged. The default value is the 32 times the current value of *maxmqid*.

This option is valid only for OSS message-queue servers.

MAXWAITTIME *seconds2*

specifies the maximum number of seconds that the OSS Monitor waits for the server's primary processor to become available during an automatic restart of the server; *seconds2* must be in the range 0 through 32767. A value of 0 means that the OSS Monitor does not wait for the primary processor to become available.

If the BACKUPCPUOK attribute for the server is TRUE and the primary processor does not become available after the specified period has elapsed, the OSS Monitor attempts to start the server in its backup processor.

When the MAXWAITTIME option is omitted, the value in the server's current configuration is not changed.

MSGMQB *msgmqb*

specifies the maximum number of bytes allowed in an OSS message queue. Valid values are in the range 1 through 65535.

If the MSGMQB option is omitted, the maximum number of bytes allowed is unchanged. The default value is the maximum, 65535.

This option is valid only for OSS message-queue servers.

MSGMSIZE *msgmsize*

specifies the maximum number of bytes allowed for a message. Valid values are in the range 1 through 32000.

If the MSGMSIZE option is omitted, the maximum number of bytes allowed is unchanged. The default value is the maximum, 32000.

This option is valid only for OSS message-queue servers.

SQLTIMEOUT *seconds*

specifies the number of seconds that the OSS name server waits for a response from SQLCAT to a request. Valid values are in the range 60 through 300.

If the SQLTIMEOUT option is omitted, the timeout interval is unchanged.

This option is valid only for OSS name server processes.

Considerations

- You can use the ALTER SERVER command on servers that are not currently running.
- Changes to the DESIREDSTATE attribute take effect immediately after using the ALTER SUBSYS command, without stopping and restarting a server. Otherwise, the ALTER SERVER command does not change the operation of a running server; changes do not take effect until the server is stopped and restarted.

- The primary and backup server processes cannot be configured in the same processor.
- The ALTER SERVER command can be used only by super-group users (255,*nnn*).
- For the OSS message-queue server, a BACKUPCPU value of -1 means the server is to use for its backup process the processor specified by the TACL PARAM BACKUPCPU (see Currently Used TACL PARAMs for the OSS Monitor on page 2-11). If the TACL PARAM BACKUPCPU is not defined or is invalid, the OSS message-queue server automatically picks an available processor with the next higher or lower processor number than the primary processor. When the TACL PARAM BACKUPCPU specifies a valid processor number, the OSS Monitor tries to add that processor to the OSS message-queue server configuration before bringing up the OSS message-queue server.
- The INODECACHE, LINKCACHE, MSGMQB, MAXMQID, MAXMSG, MSGMSIZE, and SQLTIMEOUT parameters are not valid when the OSS sockets local server is specified in an ALTER SERVER command.
- The INODECACHE, LINKCACHE, and SQLTIMEOUT parameters are not valid when the OSS message-queue server is specified in an ALTER SERVER command.
- OSS transport agent servers cannot be modified and are not a valid server type for the ALTER SERVER command.

Examples

- To change the configuration of the OSS name server process $ZPNS to use processor 0 as its primary processor and processor 1 as its backup processor, enter the following command:

  ```
  ALTER /OUT CMDLOG/ SERVER #ZPNS, CPU 0, BACKUPCPU 1, &
  INODECACHE 2048, LINKCACHE 4096
  ```

  This command assigns the OSS name server process the default value of 4096 entries for its link cache and a value of 2048 entries for its inode cache. This command also sends informational messages to the file CMDLOG.

- To change the configuration of the OSS message-queue server process $ZMSGQ to use processor 3 as its primary processor and processor 1 as its backup processor, enter the following command:

  ```
  ALTER /OUT CMDLOG/ SERVER #ZMSGQ, CPU 3, BACKUPCPU 1, &
  MAXMSG 8192
  ```

  This command assigns the OSS message-queue server process the value 8192 as the maximum number of messages allowed on the message queue. This command also sends informational messages to the file CMDLOG.

- To change the OSS sockets local server process $ZPLS so that three attempts to restart it in its primary or backup processor are made when its primary processor is

unavailable for more than 30 seconds during an automatic restart, enter the following command:

```
ALTER /OUT CMDLOG/ SERVER #ZPLS, BACKUPCPUOK TRUE, &
MAXWAITTIME 30, AUTORESTART 3, DESIREDSTATE STARTED
```

This command also sends informational messages to the file CMDLOG.

ALTER SUBSYS, ALTER MON, and ALTER PROCESS Commands

The ALTER SUBSYS, ALTER MON, and ALTER PROCESS commands all change the configuration of the OSS Monitor. The information entered in the command is added to or altered in the ZOSSPARM file.

The syntax of the ALTER SUBSYS, ALTER MON, and ALTER PROCESS commands is:

```
ALTER  [ /OUT filename1/ ] [ object-type ] [ process ]
       { [ , AUTOSTART { AUTO | MANUAL } ]
         [ , FSCKCPU processor_number ]
         [ , IOTIMEOUT seconds ]
         [ , REPORT [ filename2 | NULL ]
         [ , ZOSSVOL volume ] }
```

OUT *filename1*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

object-type

is one of the following:

```
MON | PROCESS | SUBSYS
```

MON, PROCESS, and SUBSYS all specify the OSS Monitor.

If *object-type* is omitted, *process* must be specified unless ASSUME PROCESS $ZPMON was previously specified.

process

specifies the manager process of the subsystem whose parameter values you want to alter. *process* has the following form:

process_name

$ZPMON is the only supported value for *process_name*.

process_name must be specified unless ASSUME PROCESS $ZPMON or ASSUME $ZPMON was previously specified.

You can specify the following options in any order. You must specify at least one option:

AUTOSTART { AUTO | MANUAL }

specifies whether the automatic startup service is enabled.

AUTO

All filesets and servers that are configured to use the automatic startup service are automatically restarted after a system load or restart.

MANUAL

Filesets and servers are not automatically restarted after a system load or restart. This is the default value.

When you use AUTOSTART MANUAL, automatic recovery and remounting of filesets that were not in a stopped state when a failure or shutdown occurred are still recovered or restarted when an OSS name server or the OSS Monitor is restarted; this activity is described in Automatic Restart of Filesets During OSS Monitor Startup on page 5-8 and Automatic Restart of Filesets After OSS Name Server Failure on page 5-10.

FSCKCPU *processor_number*

specifies the processor number of the processor in which FSCK runs when the SCF DELETE FILESET or DIAGNOSE FILESET command is executed and no processor was configured for FSCK use with that fileset or the configured processor for the fileset is unavailable. Valid values are either in the range 0 through 15 or equal to -1.

The value -1 specifies that FSCK should run in the processor used by the copy of DP2 for the fileset catalogs.

The initial value (set when the configuration database is created) is -1.

If the FSCKCPU option is omitted, the configured value is unchanged.

IOTIMEOUT *seconds*

specifies the number of seconds that the OSS Monitor waits for a response from an OSS name server to an OSS Monitor request.

Valid values are in the range 1 through 300. The initial value (set when the configuration database is created) is 60.

If the IOTIMEOUT option is omitted, the configured timeout interval is unchanged.

`REPORT [ filename2 | NULL ]`

specifies the target for the FSCK log file and for CVT and FUP output from SCF commands when no report file was configured for use with that fileset or the configured report file for the fileset is unavailable. The specified value must be either NULL, a blank, or the Guardian filename of a spooler location.

If *filename2* is not available when it is needed, reports are created as if NULL had been specified.

The initial value (set when the configuration database is created) is blank.

The value of NULL or blank indicates that the output file should be a file created in the same volume and subvolume as the OSS Monitor OSSMON object file. The created file is a file-code-180 file (a C language text file) with a file identifier that consists of the characters ZX0 followed by the device label of the fileset processed by the command that launched the FSCK utility:

$SYSTEM.SYS*nn*.ZX0*devicelabel*

where *nn* indicates the currently running system installation and *devicelabel* is the value specified for the DEVICELABEL option of the fileset. If the file $SYSTEM.SYS*nn*.ZX*devicelabel* already exists, FSCK appends its new output to the existing file. The CTOEDIT command can be used to convert the file-code-180 file to a file-code-101 EDIT file.

The value specified in this REPORT parameter can be overridden by using the REPORT option of the SCF DIAGNOSE FILESET command.

If the REPORT option is omitted, the configured value is unchanged.

`ZOSSVOL volume`

specifies the name of the Guardian disk volume that contains the program files for the CVT utility, the OSS Monitor, and other OSS components.

Disk volume names are not case-sensitive.

The initial value (set when the configuration database is created) is $SYSTEM.

If the ZOSSVOL option is omitted, the configured value is unchanged.

Considerations

- The ALTER SUBSYS command can be used only by super-group users (255,*nnn*).
- Changes take effect immediately and affect subsequent OSS Monitor SCF commands. The subsystem does not need to be stopped and restarted.
- In the SCF object hierarchy, SUBSYS is the highest of the possible objects for this command.
- When you enter the SCF command ALTER SUBSYS, AUTOSTART AUTO, the OSS Monitor immediately begins providing the service. This causes restart of any

object configured for the service if that object failed since the last system load and is not currently started.

- The AUTOSTART attribute of the subsystem can also be set by starting the OSS Monitor with a PARAM value or RUN command line parameter of AUTOSTART AUTO or AUTOSTART MANUAL. The value entered in the PARAM or command line takes precedence over the previously configured value for the subsystem. The precedence is, in descending order:

  1. Command line parameter
  2. PARAM specification
  3. Configured value

Examples

- To change the configuration of the OSS Monitor $ZPMON to use processor 1 as its default processor for FSCK, enter the following command:

  ```
  ALTER /OUT CMDLOG/ SUBSYS $ZPMON, FSCKCPU 1
  ```

  This command also sends informational messages to the file CMDLOG.

- To change the configuration of the OSS Monitor $ZPMON to enable the automatic startup service for the OSS filesets and servers that it administers, enter the following command:

  ```
  ALTER /OUT CMDLOG/ SUBSYS $ZPMON, AUTOSTART AUTO
  ```

  This command also sends informational messages to the file CMDLOG.

CONTROL FILESET Command

The CONTROL FILESET command can do either or both of the following:

- Reset the highwater mark for inode use by the fileset to the number of inodes currently in use
- Update the fileset attributes used by an OSS name server for a mounted fileset from the current values in the ZOSSFSET configuration file

The syntax of the CONTROL FILESET command is:

```
CONTROL [ /OUT filename1/ ] FILESET filesetname
        { [ , RESET MAXINODEUSED ]
          [ , SYNC ] }
```

OUT *filename1*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

specifies the name of the OSS fileset to be altered. Only the names of currently started filesets defined in the ZOSSFSET file are valid values.

Fileset names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order. You must specify at least one option:

`RESET MAXINODEUSED`

Resets the highwater mark for the maximum number of inodes used by the fileset to the number of inodes currently in use in the fileset. If you omit the RESET option, the highwater mark is not changed from its current value.

`SYNC`

Updates the following attributes of the specified fileset to the values currently in the ZOSSFSET configuration file:

AUDITENABLED
READONLY
BUFFERED
FTIOMODE
MAXDIRTYINODETIME
MAXINODES
NFSPOOL
NFSTIMEOUT
NORMALIOMODE
POOL

The following fileset attributes are updated as soon as their ZOSSFSET values are changed with the SCF ALTER FILESET command and do not require use of the CONTROL FILESET command SYNC option:

DESIREDSTATE
FSCKCPU
REPORT

The following fileset attributes cannot be updated when their ZOSSFSET values are changed unless the fileset is stopped and restarted with the SCF STOP FILESET and START FILESET commands. The CONTROL FILESET command SYNC option has no effect on these attributes:

CATALOG
MNTPOINT
NAMESERVER

If you omit the SYNC option, the values in use for the fileset are not updated from the ZOSSFSET file.

Considerations

- The CONTROL FILESET command can be used only by super-group users (255,*nnn*).
- When you use the SYNC option and the fileset has the READONLY TRUE attribute, the BUFFERED attribute value in the ZOSSFSET file is ignored and the OSS name server uses the BUFFERED NONE attribute.
- When you use the SYNC option and the fileset has the BUFFERED CREATE attribute, the POOL attribute is ignored and the catalog disk volume is the only disk volume used for the storage pool.
- When you use the SYNC option and the MAXINODES attribute value in the ZOSSFSET configuration file is less than the number of inodes currently in use, the command is ignored and an error message is returned.
- When you use the SYNC option and the FTIOMODE or NORMALIOMODE attributes have changed, existing file opens are not affected. The new values are used for new opens.

Examples

- To update the AUDITENABLED attribute of the ROOT fileset after changing it with an ALTER FILESET command, enter:

```
CONTROL FILESET $ZPMON.ROOT, SYNC
```

- To reset the highwater mark for the inodes used by the ROOT fileset to the value currently in use, enter:

```
CONTROL FILESET $ZPMON.ROOT, RESET MAXINODESUSED
```

CONTROL SERVER Command

The CONTROL SERVER command can do either or both of the following:

- Reset the highwater mark for:
  - the maximum number of inodes used by each of its filesets to the number of inodes currently in use in that fileset
  - the maximum number of links used by each of its filesets to the number of links currently in use in that fileset
- Update the attributes used by an OSS name server from the current values in the ZOSSSERV configuration file

The syntax of the CONTROL SERVER command is:

```
CONTROL [ /OUT filename1/ ] SERVER server_processname
        { [ , RESET { ALL |
                      MAXINODECACHEUSED |
                      MAXLINKCACHEUSED  } ]
        [ , SYNC ] }
```

OUT *filename1*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server being altered. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of a running OSS name server to be altered. Only names currently defined in the ZOSSSERV file are valid values. The first character of the name must be a pound sign (#).

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order. You must specify at least one option:

RESET { ALL | MAXINODECACHEUSED | MAXLINKCACHEUSED }

Resets the highwater mark for the indicated resource to the number currently in use by the server. If you omit the RESET option, the highwater marks are not changed from their current values.

SYNC

Updates the following attributes of the specified OSS name server to the values currently in the ZOSSSERV configuration file:

BACKUPCPU
CPU
INODECACHE
LINKCACHE
SQLTIMEOUT

The following OSS name server attributes are updated as soon as their ZOSSSERV values are changed with the SCF ALTER SERVER command and do not require use of the CONTROL SERVER command SYNC option:

AUTORESTART
BACKUPCPUOK
MAXWAITTIME

If you omit the SYNC option, the values in use for the server are not updated from the ZOSSSERV file.

Considerations

- The CONTROL SERVER command can be used only by super-group users (255,*nnn*).
- You cannot use the CONTROL SERVER command on an OSS message-queue server, the OSS local sockets server, or an OSS sockets transport-agent server.
- The CONTROL SERVER command with the SYNC option can be used after the ALTER SERVER command to migrate the primary or backup OSS name server process to another processor. Successful execution of a CONTROL SERVER command with the SYNC option does not mean immediate migration has occurred. Use the TACL STATUS command to confirm migration of the process to the processor specified by the current BACKUPCPU or CPU attribute.

Examples

- To update the INODECACHE attribute of the OSS name server for the ROOT fileset after changing it with an ALTER SERVER command, enter:

```
CONTROL SERVER $ZPMON.#ZPNS, SYNC
```

- To reset the highwater mark for the inode cache used by the OSS name server for the ROOT fileset to the value currently in use, enter:

```
CONTROL SERVER $ZPMON.#ZPNS, RESET MAXINODECACHEUSED
```

DELETE FILESET Command

The DELETE FILESET command removes an OSS fileset from the configuration administered through the OSS Monitor. This command affects the contents of the ZOSSFSET file.

The syntax of the DELETE FILESET command is:

```
DELETE [ /OUT filename/ ] FILESET [$ZPMON.]filesetname
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

FILESET [$ZPMON.]*filesetname*

specifies the name of the fileset you are deleting. Only names currently defined in the ZOSSFSET file are valid values.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- The DELETE FILESET command can be used only by the super ID (255,255).
- You can use the DELETE FILESET command only on a fileset that is in the STOPPED or UNKNOWN state.
- If the fileset contains files, all of those files and the catalog files are deleted.
- The user is prompted for confirmation before the fileset is deleted.
- The DELETE FILESET command does not delete the root fileset.

Example

To delete the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

```
DELETE /OUT CMDLOG/ FILESET $ZPMON.USER1
```

DELETE SERVER Command

The DELETE SERVER command removes an OSS name server, except the root OSS name server, from the configuration administered through the OSS Monitor. This command does not apply to any of the other OSS servers.

The syntax of the DELETE SERVER command is:

```
DELETE [ /OUT filename / ] SERVER server_processname
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server being deleted. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of the server to delete. Only names currently defined in the ZOSSSERV file are valid values.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- You must be in the super group (255, *nnn*) to use this command.
- You can delete only OSS name servers that are not running.
- You cannot delete an OSS name server that is associated with a fileset. You must first alter or delete that fileset so that it is not associated with the OSS name server you want to delete.
- You cannot delete the OSS name server for the root fileset.

Example

To delete the OSS name server #ZPNS1 and send informational messages to the file CMDLOG, enter the following command:

```
DELETE / OUT CMDLOG /SERVER $ZPMON.#ZPNS1
```

DIAGNOSE FILESET Command

The DIAGNOSE FILESET command causes the Guardian FSCK utility to check the integrity of an existing fileset. The fileset must be stopped (unmounted) before it can be checked.

The syntax of the DIAGNOSE FILESET command is:

```
DIAGNOSE [ /OUT filename1 / ] FILESET [$ZPMON.]filesetname

         [ , CPU processor_number ]

           , DETAIL ]

         [ , OPTION  { START | STOP } ]

         [ , REPORT filename2 ]

        [ , { STATUS                                  }
            { REPAIR { ALL | NONE | OPEN | SERIOUS } }
            { UPGRADE                                 }
             { DOWNGRADE                                } ]
```

`OUT` *`filename1`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

is the name of the fileset you are diagnosing. Only names currently defined in the ZOSSFSET file are valid values.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order:

`CPU` *`processor_number`*

specifies the processor number of the processor where the FSCK utility will run. Valid values are in the range 0 through 15.

You can also specify the default processor with the FSCKCPU parameter of the SCF ALTER SUBSYS command.

If you do not specify a processor by either means, the default processor is the processor on which the copy of DP2 used for the fileset catalog is running.

`DETAIL`

produces a detailed report and places it in the FSCK log file.

If the DETAIL option is omitted, only a summary report is produced.

`OPTION { START | STOP }`

starts or stops a diagnosis by the FSCK utility.

`START`	Starts a new diagnosis and sets the state of the fileset to DIAGNOSING.
`STOP`	Stops a diagnosis that is in progress and sets the state of the fileset to STOPPED.

If this option is omitted, the default value is OPTION START.

`REPORT` *`filename2`*

specifies the Guardian filename of the Guardian disk file or spooler location to receive the FSCK log file output and any related CVT or FUP output.

You can also specify the default target for output with the REPORT parameter of the SCF ALTER SUBSYS command.

If you do not specify a file by either means, the file is written to the location described in FSCK Log File on page 12-47.

STATUS

reports any inconsistencies that the OSS name server has detected in the fileset and places the report in the FSCK log file.

When the STATUS option is specified, you cannot specify the REPAIR, UPGRADE, or DOWNGRADE option.

When the STATUS, REPAIR, UPGRADE, and DOWNGRADE options are all omitted, the default value is REPAIR NONE.

REPAIR { ALL | NONE | OPEN | SERIOUS }

specifies which inconsistencies to repair in the fileset.

Serious inconsistencies can make files unavailable, can make users unable to create new files, and have the potential for serious catalog damage. Such inconsistencies are reported in the FSCK log file with messages categorized as SERIOUS.

Minor inconsistencies are unlikely to cause problems by themselves. Such inconsistencies are reported in the FSCK log file with messages categorized as MINOR.

Serious and minor inconsistencies and their repair actions are described in Table 5-2 on page 5-29. Unrepairable inconsistencies are reported with FSCK log file ERROR messages.

This option has the following values:

ALL	Repair all inconsistencies (serious and minor).
NONE	Do not repair any inconsistencies. This is the default specification.
OPEN	Same as SERIOUS, but also assumes that there might be open files in this fileset. This option indicates that inodes cannot be reused until the next unmount or remount.
SERIOUS	Repair only serious inconsistencies. This option can speed up the execution of a fileset check.

If the REPAIR option is omitted, the default value is REPAIR NONE.

When the REPAIR option is specified, you cannot specify the STATUS, UPGRADE, or DOWNGRADE option.

UPGRADE

reformats the catalog for the fileset from an earlier version to a current version. For example, you can upgrade a D30-version catalog to a D40-version catalog to add support for symbolic links.

If the UPGRADE option is omitted, no upgrade occurs.

When the UPGRADE option is specified, you cannot specify the STATUS, REPAIR, or DOWNGRADE option.

`DOWNGRADE`

reformats the catalog for the fileset from a current version to an earlier version. For example, you can downgrade a D40-version catalog to a D30-version catalog to remove support for symbolic links.

△ **Caution.** When you downgrade a fileset's catalog to a D30 version, all symbolic links in that fileset are removed. Attempts to access files in that fileset through symbolic links from other filesets will fail.

If the DOWNGRADE option is omitted, no downgrade occurs.

When the DOWNGRADE option is specified, you cannot specify the STATUS, REPAIR, or UPGRADE option.

Considerations

- You might specify a new processor with the CPU parameter if the processor on which FSCK normally runs has stopped. You might also specify a particular processor for load balancing.
- You cannot use the DIAGNOSE FILESET command on a fileset that is in the STARTED state (mounted).
- When the DIAGNOSE FILESET command is used, the fileset is placed in the DIAGNOSING state until the integrity check is complete. Then the fileset is placed in the STOPPED state and can be remounted with the SCF START FILESET command unless serious inconsistencies remain unrepaired.
- You can use the SCF STATUS FILESET command to determine whether the integrity check is complete.
- If FSCK fails, the fileset is placed in the UNKNOWN state.
- The FSCK utility checks for the inconsistencies listed in Table 5-2 on page 5-29.
- The DIAGNOSE FILESET command can be used only by super-group users (255,*nnn*).

Examples

- To check the integrity of the fileset USER1, not repair any inconsistencies, and send informational messages to the file CMDLOG, first use the SCF STOP FILESET command to stop (unmount) the fileset and then enter the following command:

```
DIAGNOSE /OUT CMDLOG/ FILESET USER1
```

- To check the integrity of the fileset USER1, repair all inconsistencies, produce a detailed report, and send informational messages to the file CMDLOG, first use the

SCF STOP FILESET command to stop (unmount) the fileset and then enter the following command:

```
DIAGNOSE /OUT CMDLOG/ FILESET $ZPMON.USER1,DETAIL,REPAIR ALL
```

- To stop the integrity check in progress on the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

```
DIAGNOSE /OUT CMDLOG/ FILESET $ZPMON.USER1, OPTION STOP
```

FSCK Log File

The FSCK utility writes its output to a log file with a Guardian filename that you can specify with the REPORT parameter of either the DIAGNOSE FILESET command or the ALTER SUBSYS command. If you do not specify a name for the log file, the log file is put on the same volume and subvolume as the OSS Monitor program file, OSSMON, which is usually the SYS*nn* subvolume.

The Guardian file identifier of the default log file consists of the characters ZX0 followed by the rightmost portion of the device label of the fileset. The device label of a fileset is a field of the SCF INFO FILESET display; for information about this command, see INFO FILESET Command on page 12-47.

For example, for the root fileset, which has the device label 0, the file identifier of the default log file is ZX000000.

The log file is a text file. See FSCK Log File on page 5-25 for more information about the format of this file.

Figure 5-4 on page 5-27 shows an example of FSCK output.

INFO FILESET Command

The INFO FILESET command displays information about the attributes of an OSS fileset. The information displayed is from the ZOSSFSET and ZOSSSERV files.

The syntax of the INFO FILESET command is:

```
INFO [ /OUT filename/ ] FILESET [$ZPMON.]filesetname
     [ , DETAIL ]
     [ , SEL [ NOT ] state ]
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

is the name of the fileset whose information is to be displayed. Only names currently defined in the ZOSSFSET file are valid values.

`filesetname` can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order:

`DETAIL`

produces a detailed report.

If the DETAIL option is omitted, only a summary report is produced.

`SEL [ NOT ]` *`state`*

directs the command to apply only to filesets that are in the specified state or, when the NOT field is used, only to filesets that are not in the specified state. Valid values for *`state`* are:

Value	Meaning
DIAGNOSING	The fileset is being diagnosed by the FSCK utility.
STARTED	The fileset is started (mounted).
STOPPED	The fileset is ready to be started (mounted) or diagnosed.
UNKNOWN	The fileset is in an unknown state.

The information returned for a summary report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS Info FILESET fileset-devicename

FilesetName                *MntPoint
filesetname                directory
```

`fileset-devicename`

is the name of the fileset whose information is displayed, shown as a device of the OSS Monitor process.

`filesetname`

is the name of the fileset whose information is being displayed.

directory

is the OSS pathname of the mount point for the fileset. This mount point might not appear to be the one most recently specified for the fileset. The value stored and displayed is the result of transforming the specified pathname to its minimum absolute form.

The information returned for a detailed report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS Detailed Info FILESET fileset-devicename

 DeviceLabel............. nnnnnn
*Catalog................. volume        *NameServer.............. servername1
*Buffered................ choice        *NFSPool................. kbytes
*ReadOnly................ setting       *NFSTimeout.............. seconds
*Pool.................... filename
 CreateBy................ user-name1
 CreateTime.............. time1
 AlterBy................. user-name2
 AlterTime............... time2
*MntPoint-FilesetName.... filesetname
*MntPoint-NameServer..... servername2
*MntPoint................ directory
*AuditEnabled............ audit-value
*DesiredState............ desired-state
*FsckCPU................. processor
*Report.................. filename2
*MaxDirtyInodeTime....... seconds2
*MaxInodes............... maxinodes
*FTIOmode ............... ftiomode
*NormalIOmode ........... normaliomode
```

fileset-devicename

is the name of the fileset whose information is displayed, shown as a device of the OSS Monitor process.

nnnnnn

is the device label associated with the inodes of the fileset.

volume

is the name of the Guardian disk volume that contains the catalog files for the fileset.

servername1

is the server name of the OSS name server that administers the fileset.

choice

indicates the amount of catalog write buffering used for the fileset.

Value	Meaning
CREATE	Buffer catalog writes during file creation within the fileset. This is the fast-create option.
LOG	Buffer only log file activity.
NONE	Do not buffer anything.

kbytes

is the number of kilobytes that the OSS name server uses for buffers for Network File System (NFS) operations for the fileset.

setting

indicates the write permission granted to users of the fileset. The value is one of the following:

Value	Meaning
FALSE	Files within the fileset can be created, read, written, and deleted.
TRUE	Files within the fileset can only be read.

seconds

is the number of seconds that the OSS name server retains the results of nonretryable Network File System (NFS) operations for the fileset.

filename

is the Guardian filename of the storage-pool file that contains the volume list to use when creating new files in the fileset.

user-name1

is the user name of the user or process that created the initial configuration of the fileset.

time1

is the timestamp for the time when the initial configuration of the fileset was created, in the form *dd mmm yyyy hh:mm:ss.mil*.

user-name2

is the user name of the user who last modified the configuration or state of the fileset.

time2

is the timestamp for the last time that the fileset configuration was changed, in the form *dd mmm yyyy hh:mm:ss.mil*.

filesetname

is the name of the fileset on which this fileset is mounted.

servername2

is the server name of the OSS name server for the fileset identified by *filesetname*.

directory

is the OSS pathname of the mount point for the fileset. This mount point might not appear to be the one most recently specified mount point for the fileset. The value stored and displayed is the result of transforming the specified pathname to its minimum absolute form.

audit-value

is the value of the AUDITENABLED attribute of the fileset, which is either ON or OFF.

desired-state

is the value of the DESIREDSTATE attribute of the fileset to use with the AUTOSTART AUTO attribute of the subsystem; *desired-state* is either STARTED or STOPPED.

processor

is the processor number of the processor used by FSCK when that program automatically recovers the fileset.

filename2

is the name of the Guardian file used by FSCK for its report output when that program automatically recovers the fileset.

seconds2

is the number of seconds allowed between updates of the fileset's inode cache if the fileset is using the BUFFERED LOG option.

maxinodes

is the approximate maximum number of inodes allowed for the fileset. If a fileset has not yet been started after installing the first RVU that supports this feature, the word unknown is displayed.

ftiomode

is the fault-tolerance and buffering attribute to be used when file opens use O_SYNC in the fileset. The value displayed is one of the keywords described for the FTIOMODE attribute of the ADD FILESET Command on page 12-7.

normaliomode

is the fault-tolerance and buffering attribute to be used when file opens do not use O_SYNC in the fileset. The value displayed is one of the keywords described for the NORMALIOMODE attribute of the ADD FILESET Command on page 12-7.

Example

To obtain a detailed report of the configuration of the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

```
INFO /OUT CMDLOG/ FILESET $ZPMON.USER1, DETAIL
```

A display such as the following is written to CMDLOG:

```
OSS Detailed Info FILESET \NODE1.$ZPMON.#USER1

 DeviceLabel............. 00001
*Catalog................. $DATA1       *NameServer.............. #ZPNS2
*Buffered................ LOG          *NFSPool................. 16
*ReadOnly................ TRUE         *NFSTimeout.............. 120
*Pool.................... ZOSSPOOL
 CreateBy................ SUPER.SUPER
 CreateTime.............. 12 Dec 2001 13:29:30.433
 AlterBy................. SUPER.SUPER
 AlterTime............... 03 Mar 2002 01:28:14.221
*MntPoint-FilesetName.... USRLIB
*MntPoint-NameServer..... #ZPNS1
*MntPoint................ /user1/data/fixed/sub/binary/lib/user1
*AuditEnabled............ ON
*DesiredState............ STARTED
*FsckCPU................. 5
*Report.................. $S.#SPL
*MaxDirtyInodeTime....... 30
*Maxinodes............... 100000
*FTIOmode................ UNBUFFEREDCP
*NormalIOmode............ OSSBUFFEREDCP
```

INFO SERVER Command

The INFO SERVER command displays information about a server administered by the OSS Monitor. The information displayed is from the ZOSSSERV file.

The syntax of the INFO SERVER command is:

```
INFO [ /OUT filename/ ] SERVER server_processname

     [ , DETAIL ]

     [ , SEL [ NOT ] state ]
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server whose information is to be displayed. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of the server whose information is to be displayed. Only names currently defined in the ZOSSSERV file are valid values. The first character of the name must be a pound sign (#).

servername can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order:

DETAIL

produces a detailed report.

If the DETAIL option is omitted, only a summary report is produced.

SEL [NOT] *state*

directs the command to apply only to servers that are in the specified state or, when the NOT field is used, only to servers that are not in the specified state. Valid values for *state* are:

Value	Meaning
STARTED	The server is started.
STOPPED	The server is ready to be started.

The information returned for a summary report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS INFO SERVER server-devicename

ServerName        Type        *CPU            *BackupCPU
servername        type        processor1      processor2
```

server-devicename

is the server name of the server whose information is displayed, shown as a device of the OSS Monitor process.

servername

is the server name of the server whose information is being displayed.

type

is the type of the server.

Value	Meaning
LOCAL	The server is an OSS sockets local server for AF_UNIX sockets.
NAME	The server is an OSS name server.
MSGQ	The server is an OSS message-queue server.
TAGENT	The server is an OSS transport agent server.

processor1

is the processor number of the processor where the primary server process runs. The returned value is from the database; it might not reflect the actual processor of the running process because the active process might have been switched or recreated its backup process in the next available processor, as implemented in the OSS message-queue server.

processor2

is the processor number of the processor where the backup server process runs. The returned value is from the database; it might not reflect the actual processor of the running process because the active process might have been switched or recreated its backup process in the next available processor, as implemented in the OSS message-queue server.

The information returned for a detailed report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS Detailed Info SERVER server-devicename

 Type.................. type
*CPU................... processor1
*BackupCPU............. processor2
 CreateBy.............. user-name1
 CreateTime............ time1
 AlterBy............... user-name2
 AlterTime............. time2
*Params................ INODECACHE size1, LINKCACHE size2, SQLTIMEOUT
seconds, MSGMQB msgmqb, MAXMQID maxmqid, MAXMSG maxmsg, MSGMSIZE msgmsize
*BackupCPUOK........... choice
*MaxWaitTime........... seconds
*DesiredState.......... state
*AutoRestart........... tries1
 PersistenceCount...... tries2
```

server-devicename

is the server name of the server whose information is displayed, shown as a device of the OSS Monitor process.

type

is the type of the server.

Value	Meaning
LOCAL	The server is an OSS sockets local server for AF_UNIX sockets.
NAME	The server is an OSS name server.
MSGQ	The server is an OSS message-queue server.
TAGENT	The server is an OSS transport agent server.

processor1

is the processor number of the processor where the primary server process runs. The returned value is from the database; it might not reflect the actual processor of the running process because the active process might have been switched or recreated its backup process in the next available processor, as implemented in the OSS message-queue server.

processor2

is the processor number of the processor where the backup server process runs. The returned value is from the database; it might not reflect the actual processor of the running process because the active process might have been switched or recreated its backup process in the next available processor, as implemented in the OSS message-queue server.

user-name1

is the user name of the user or process that created the initial configuration of the server.

time1

is the timestamp for the time when the initial configuration of the server was created, in the form *dd mmm yyyy hh:mm:ss.mil*.

user-name2

is the user name of the user who last modified the configuration or state of the server.

time2

is the timestamp for the last time that the server configuration was changed, in the form *dd mmm yyyy hh:mm:ss.mil*.

size1

is the number of entries permitted in the inode cache used by the OSS name server.

This entry is displayed only for OSS name server processes.

size2

is the number of entries permitted in the link cache used by the OSS name server.

This entry is displayed only for OSS name server processes.

seconds

is the number of seconds that the OSS name server waits for a response from SQLCAT to a request.

This entry is displayed only for OSS name server processes.

msgmqb

specifies the maximum number of bytes allowed in an OSS message queue. Valid values are in the range 1 through 65535.

This parameter is valid only for OSS message-queue servers.

maxmqid

specifies the maximum number of OSS message queue IDs allowed at any time. Valid values are in the range 1 through 1024.

This parameter is valid only for OSS message-queue servers.

maxmsg

specifies the maximum number of messages allowed on all OSS message queues on a system. Valid values are in the range 1 through 16384.

This parameter is valid only for OSS message-queue servers.

msgmsize

specifies the maximum number of bytes allowed for a message. Valid values are in the range 1 through 32000.

This parameter is valid only for OSS message queue-servers.

choice

specifies whether the server can be started in its backup processor during a restart when its primary processor is unavailable. Valid values are TRUE or FALSE.

seconds

specifies the number of seconds that the OSS Monitor waits for the server's primary processor to become available during an automatic restart. Valid values are 0 through 600.

state

specifies the desired state of the server after the next time the automatic startup service is used. Valid values are STARTED or STOPPED.

tries1

specifies the number of times that the OSS Monitor attempts to restart the server when the automatic startup service is used. Valid values are 0 through 10.

tries2

specifies the number of times that the OSS Monitor actually restarted the server during the previous 10 minutes.

Example

To obtain a detailed report of the configuration of the server $ZPNS and send informational messages to the file CMDLOG, enter the following command:

```
INFO /OUT CMDLOG/ SERVER $ZPMON.#ZPNS, DETAIL
```

A display such as the following is written to CMDLOG:

```
OSS Detailed Info SERVER \NODE1.$ZPMON.#ZPNS

 Type.................. NAME
*CPU................... 1
*BackupCPU............. 0
 CreateBy.............. $ZPMON
 CreateTime............ 12 Dec 2001 13:28:28.331
 AlterBy............... SUPER.SUPER
 AlterTime............. 03 Mar 2002 01:29:14.123
*Params................ INODECACHE 4096, LINKCACHE 4096, SQLTIMEOUT 60
*BackupCPUOK........... TRUE
*MaxWaitTime........... 60
*DesiredState.......... STARTED
*AutoRestart........... 3
 PersistenceCount...... 3
```

INFO SUBSYS, INFO MON, and INFO PROCESS Commands

The INFO SUBSYS, INFO MON, and INFO PROCESS commands all display information about the OSS subsystem. The information displayed is from the ZOSSPARM file.

The syntax of the INFO SUBSYS, INFO MON, and INFO PROCESS commands is:

```
INFO [ /OUT filename/ ] object-type [ process ]
     [ , DETAIL ]
```

`OUT filename`

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

object-type

is one of the following:

`MON | PROCESS | SUBSYS`

MON, PROCESS, and SUBSYS all specify the OSS Monitor.

process

specifies the manager process of the subsystem whose parameter values you want to display. *process* has the following form:

process_name

$ZPMON is the only supported value for *process_name*. *process_name* can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

process_name must be specified unless ASSUME PROCESS $ZPMON or ASSUME $ZPMON was previously specified.

`DETAIL`

produces a detailed report.

If the DETAIL option is omitted, only a summary report is produced.

The information returned for a summary report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS Info SUBSYS process-filename

MgrName         *IOTimeout       *FsckCPU        *Report           *ZOSSVol
process_name    seconds          processor       filespec          volume
```

process-filename

is the process name of the subsystem whose information is displayed, shown as the Guardian filename of the OSS Monitor process.

process_name

is the process name of the subsystem whose information is displayed, $ZPMON.

seconds

is the number of seconds that the OSS Monitor waits for a response from an OSS name server to an OSS Monitor request.

processor

is the processor number of the processor in which the FSCK program runs when the SCF DIAGNOSE FILESET or DELETE FILESET command is used and either no processor is configured for the fileset or the processor configured for the fileset is unavailable.

filespec

is the Guardian filename or spooler location of the default FSCK log file.

volume

is the name of the Guardian disk volume that contains the program files for the CVT utility, the OSS Monitor, and other OSS components.

The information returned for a detailed report has the following format (an asterisk indicates an attribute whose value can be changed using an SCF command):

```
OSS Detailed Info SUBSYS process-filename

*IOTimeout.............. seconds
*FsckCPU................ processor
*Report ................ filespec
*ZOSSVol................ volume
 CreateBy............... user-name1
 CreateTime............. time1
 AlterBy................ user-name2
 AlterTime.............. time2
*AutoStart.............. setting
```

process-filename

is the process name of the subsystem whose information is displayed, shown as the Guardian filename of the OSS Monitor process.

seconds

is the number of seconds that the OSS Monitor waits for a response from an OSS name server to an OSS Monitor request.

processor

is the processor number of the processor in which the FSCK program runs when the SCF DIAGNOSE FILESET or DELETE FILESET command is used and either

no processor is configured for the fileset or the processor configured for the fileset is unavailable.

filespec

is the Guardian filename or spooler location of the default FSCK log file.

volume

is the name of the Guardian disk volume that contains the program files for the CVT utility, the OSS Monitor, and other OSS components.

user-name1

is the user name of the user or process that created the initial configuration of the subsystem.

time1

is the timestamp for the time when the initial configuration of the subsystem was created, in the form *dd mmm yyyy hh:mm:ss.mil*.

user-name2

is the user name of the user who last modified the configuration or state of the subsystem.

time2

is the timestamp for the last time that the subsystem configuration was changed, in the form *dd mmm yyyy hh:mm:ss.mil*.

setting

is the automatic startup service setting (AUTO or MANUAL) in effect for the next system load or restart.

Considerations

- The only valid *process_name* is $ZPMON.
- In the SCF object hierarchy, SUBSYS is the highest of the possible objects for this command.

Example

To obtain a detailed report of the configuration of the OSS subsystem and send informational messages to the file CMDLOG, enter the following command:

```
INFO /OUT CMDLOG/ SUBSYS $ZPMON, DETAIL
```

A display such as the following is written to CMDLOG:

```
OSS Detailed Info SUBSYS \NODE1.$ZPMON

*IOTimeout.............. 60
*FsckCPU................  1
*Report ................ $SYSTEM.MONDATA.FSCKOUT
*ZOSSVol................ $SYSTEM.ZOSS
 CreateBy............... $ZPMON
 CreateTime............. 12 Dec 2001 12:48:28.554
 AlterBy................ SUPER.SUPER
 AlterTime.............. 03 Mar 2002 01:19:11.335
*AutoStart.............. MANUAL
```

NAMES Command

The NAMES command lists objects managed through the OSS Monitor. The syntax of the NAMES command is:

```
NAMES [ /OUT filename/ ] [ object-type ] [ object-name ]

      [, SEL [ NOT ] state ]
```

`OUT` *`filename`*

specifies the name of a Guardian output file to receive the list of objects. You can either read this file with a text editor or display it with the FUP COPY command.

`object-type`

specifies the types of objects whose names are to be listed. *`object-type`* is one of the following:

`MON | PROCESS | SUBSYS | FILESET | SERVER`

MON, PROCESS, and SUBSYS object types all specify the OSS Monitor.

If *`object-type`* is omitted, then the names of all objects that match the criteria for the *`object-name`* specification are displayed.

`object-name`

is the name of the object whose name is to be displayed. *`object-name`* can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

If *`object-name`* is omitted, then the names of all filesets, all servers, and the subsystem are displayed.

SEL [NOT] *state*

directs the command to apply only to objects that are in the specified state or, when the NOT field is used, only to objects that are not in the specified state. This option is valid only if an object type is provided.

For the SERVER object type, valid values for *state* are:

Value	Meaning
STARTED	The server is started.
STOPPED	The server is ready to be started.

For the FILESET object type, valid values for *state* are:

Value	Meaning
DIAGNOSING	The fileset is being diagnosed by the FSCK utility.
STARTED	The fileset is started (mounted).
STOPPED	The fileset is ready to be started (mounted) or diagnosed.
UNKNOWN	The fileset is in an unknown state.

The OSS Monitor does not recognize the SEL parameter for object types other than SERVER or FILESET.

The information returned has the following format:

```
$ZPMON Names SUBSYS node.$ZPMON

SUBSYS
$ZPMON

OSS Names FILESET node.$ZPMON.*

FILESET
filesetname       ...

OSS Names SERVER node.$ZPMON.*

SERVER
servername        ...
```

node

is the Expand name of the node on which the subsystem is running.

filesetname

is the name of an OSS fileset.

servername

is the server name of a configured OSS server.

Considerations

- The SEL option applies only to FILESET and SERVER objects.
- The OSS Monitor version of the NAMES command does not support the SUB option that can appear in other SCF NAMES commands. The OSS Monitor does not have subordinate objects.
- In the SCF object hierarchy, SUBSYS is the highest of the possible objects for this command.

Example

To list the names of all OSS filesets, enter the following SCF commands:

```
ASSUME $ZPMON
NAMES FILESET *
```

The following information is displayed:

```
OSS Names FILESET \NODE1.$ZPMON.*

FILESET
DATA1   DATA2   ROOT   TEMP
```

RENAME FILESET Command

The RENAME FILESET command changes an existing fileset name. The syntax of the RENAME FILESET command is:

```
RENAME [ /OUT filename/ ] FILESET [$ZPMON.]filesetname,
       [$ZPMON.]newname
```

filename

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

FILESET [$ZPMON.]*filesetname*

is the name of a fileset to be renamed. Only names currently defined in the ZOSSFSET file are valid values.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

[$ZPMON.]*newname*

is the fileset's new name. Only names not currently defined in the ZOSSFSET file are valid values.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- Only super-group users (255,*nnn*) can use the RENAME FILESET command.
- You can rename a running fileset, and the new name takes effect immediately.

Example

To change the name of fileset USER1 to NEWUSER1, enter the following SCF command:

```
RENAME FILESET $ZPMON.USER1, $ZPMON.NEWUSER1
```

START FILESET Command

The START FILESET command makes an existing OSS fileset available to users (also known as mounting the fileset).

The syntax of the START FILESET command is:

```
START [ /OUT filename/ ] FILESET [$ZPMON.]filesetname
```

`OUT` *`filename`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

specifies the name of the fileset you are starting. The specified fileset:

- Must already be part of the OSS configuration
- Must not be in the UNKNOWN state
- Must not be already started

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- The START FILESET command can be used only by super-group users (255,*nnn*).
- The mount point for the specified fileset must be in a fileset that is already started.
- If the OSS name server for the fileset is not already running, it is started when the fileset is started.
- The MAXINODES attribute is ignored when an OSS name server tries to mount a fileset. If 110% of the number of inodes in use is greater than the current value of

MAXINODES, the OSS name server changes the value of MAXINODES to the minimum of 2200000 or 110% of the number of inodes in use, rounded up to the nearest thousand. The new value for MAXINODES is stored in the ZOSSFSET file.

This algorithm allows a fileset to be mounted and a reasonable number of files created in it before its limit is reached and operator intervention is required.

Examples

- To start (mount) the fileset USER1 and send informational messages to the file CMDLOG, enter the following SCF command:

```
START /OUT CMDLOG/ FILESET $ZPMON.USER1
```

- To start (mount) the ROOT fileset, enter the following SCF commands:

```
ASSUME $ZPMON
START FILESET ROOT
```

To start (mount) the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

```
START /OUT CMDLOG/ FILESET $ZPMON.USER1
```

START SERVER Command

The START SERVER command starts an OSS server. The syntax of the START SERVER command is:

```
START [ /OUT filename/ ] SERVER server_processname
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server being started. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of the server to start. Only names currently defined in the ZOSSSERV file are valid values.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- You can use the START SERVER command to start the OSS sockets local server, the OSS message-queue server, and the OSS transport agent servers. An OSS name server is automatically started when one of the filesets managed by that OSS name server is started.
- The START SERVER command can be used only by super-group users (255,*nnn*).

Example

To start the OSS sockets local server $ZPLS and send informational messages to the file CMDLOG, enter the following command:

```
START /OUT CMDLOG/ SERVER $ZPMON.#ZPLS
```

STATUS FILESET Command

The STATUS FILESET command displays status information about a fileset administered by the OSS Monitor.The syntax of the STATUS FILESET command is:

```
STATUS [ /OUT filename/ ] FILESET [$ZPMON.]filesetname
       [ , DETAIL ]
       [ , SEL [ NOT ] state ]
```

`OUT` *`filename`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

specifies the name of the fileset whose status you want to display. Only names currently defined in the ZOSSFSET file are valid values.

`filesetname` can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order:

`DETAIL`

produces a detailed report.

If the DETAIL option is omitted, only a summary report is produced.

`SEL [ NOT ] state`

directs the command to apply only to filesets that are in the specified state or, when the NOT field is used, only to filesets that are not in the specified state. Valid values for *state* are:

Value	Meaning
DIAGNOSING	The fileset is being diagnosed by the FSCK utility.
STARTED	The fileset is started (mounted).
STOPPED	The fileset is ready to be started (mounted) or diagnosed.
UNKNOWN	The fileset is in an unknown state.

The information returned for a summary report has the following format:

```
OSS Status FILESET fileset-devicename

FilesetName                 State       LastError     ErrorDetail
filesetname                 state       error         error-detail
```

fileset-devicename

is the name of the fileset whose status is displayed, shown as a device of the OSS Monitor process.

filesetname

is the name of the fileset whose status is displayed.

state

describes the state of the fileset.

Value	Meaning
DIAGNOSING	The fileset is being diagnosed by the FSCK utility.
STARTED	The fileset is started (mounted).
STOPPED	The fileset is ready to be started (mounted) or diagnosed.
UNKNOWN	The fileset is in an unknown state.

error

is the error number of the last OSS Monitor error that occurred during an operation on the fileset. See Numbered Messages on page A-35 for an explanation of a specific error number.

error-detail

is the error number of the Guardian file-system error (if any) reported with the *error* value. For information about Guardian file-system errors, see the *Guardian Procedure Errors and Messages Manual*.

The information returned for a detailed report has this format:

```
OSS Detailed Status FILESET fileset-devicename

 State................................ state
 MountTime............................ time1
 LastError............................ error
 ErrorDetail.......................... error-detail
 ErrorTime............................ time2
 FsckName............................. volume
 LastControlSyncTime.................. time3
 AlterAfterSyncOrMount................ status1
*AuditEnabled......................... status2
*Buffered............................. type
*ReadOnly............................. status3
*NFSTimeout........................... time4
*NFSPool.............................. number1
*MaxInodes............................ number2
 InuseInodes.......................... number3
 MaximumCountInodesUsed............... number4
 TimestampMaximumCountInodesUsed...... time5
 TimestampMaxiximumCountInodesReset..... time6
*FTIOmode ............................ ftiomode
*NormalIOmode ........................ normaliomode
 NumVols.............................. number5
 Volumes:
 volume_list1
*VolumesElligibleForFileCreation:
 volume_list2
 VolumesNotElligibleForFileCreation:
 volume_list3
 VolumesWithZeroFiles:
 volume_list4
 VolumesWithFiles:
 volume_list5
 VolumeInformation:
              -- Capacity (Mb) --      %     -- Free Extents --
Volume  (M)   Total      Free         Free    Count    Biggest
$vol          size1      size2        nnn      num     size3
```

fileset-devicename

is the name of the fileset whose status is displayed, shown as a device of the OSS Monitor process.

state

describes the state of the fileset.

Value	Meaning
DIAGNOSING	The fileset is being diagnosed by the FSCK utility.
STARTED	The fileset is started (mounted).
STOPPED	The fileset is ready to be started (mounted) or diagnosed.
UNKNOWN	The fileset is in an unknown state.

time1

is the timestamp for the time when the fileset was last mounted, in the form *dd mmm yyyy hh:mm:ss.mil*.

time2

is the timestamp for the time when the last error for the fileset was reported, in the form *dd mmm yyyy hh:mm:ss.mil*. If no error has been reported, this value is the same as the *time1* value.

status1

indicates whether the configuration of the fileset has changed after its most recent start or synchronization with ZOSSFSET:

Value	Meaning
FALSE	The fileset configuration has not changed.
TRUE	The fileset configuration has changed.

This field is displayed only if the fileset is in the STARTED state.

error

is the error number of the last OSS Monitor error that occurred during an operation on the fileset. See Numbered Messages on page A-35 for an explanation of a specific error number.

error-detail

is the error number of the Guardian file-system error (if any) reported with the *error* value. For information about Guardian file-system errors, see the *Guardian Procedure Errors and Messages Manual*.

time3

is the timestamp for the time when the configuration of the fileset was last synchronized with ZOSSFSET, in the form *dd mmm yyyy hh:mm:ss.mil*. If a CONTROL FILESET command with the SYNC option has not been issued since the fileset was last started, this field is blank.

This field is displayed only if the fileset is in the STARTED state.

volume

is the name of the Guardian disk volume used by the FSCK utility to store its log file.

status2

indicates whether changes to the content of the fileset are being audited:

Value	Meaning
OFF	The fileset is not audited.
ON	The fileset is audited.

This field is displayed only if the fileset is in the STARTED state.

type

indicates the type of buffering performed for the fileset:

Value	Meaning
CREATE	When a request or transaction requires a write to the PXINODE or PXLINK file, the corresponding write is buffered to the PXLOG file. Also uses the fast file-creation feature for writing new file labels.
LOG	When a request or transaction requires a write to the PXINODE or PXLINK file, the corresponding write is buffered to the PXLOG file. Does not use fast file creation.
NONE	Does not buffer anything. Any request or transaction that requires a write to the PXINODE or PXLINK file also requires a write to the PXLOG file. Does not use fast file creation.

This field is displayed only if the fileset is in the STARTED state.

status3

indicates whether the fileset is mounted for read-only access:

Value	Meaning
FALSE	The fileset has read and write access.
TRUE	The fileset is mounted for read-only access.

This field is displayed only if the fileset is in the STARTED state.

time4

is the number of seconds that the OSS name server retains the results of nonretryable Network File System (NFS) operations for the fileset.

This field is displayed only if the fileset is in the STARTED state.

number1

is the number of kilobytes that the OSS name server uses for buffers for nonretryable NFS operations for the fileset.

This field is displayed only if the fileset is in the STARTED state.

number2

is the approximate maximum number of inodes that the OSS name server allows for the fileset.

This field is displayed only if the fileset is in the STARTED state.

number3

is the current number of inodes that the OSS name server is using for the fileset, displayed as a decimal number and as a percentage of the maximum number allowed.

This field is displayed only if the fileset is in the STARTED state.

number4

is the last highwater mark of inode use in the cache for the fileset, displayed as a decimal number and as a percentage of the maximum number allowed.

This field is displayed only if the fileset is in the STARTED state.

time5

is the timestamp for the time when the fileset last reached its highwater mark for inode use, in the form *dd mmm yyyy hh:mm:ss.mil*.

This field is displayed only if the fileset is in the STARTED state.

time6

is the timestamp for the time when the fileset highwater mark for inode use was last reset, in the form *dd mmm yyyy hh:mm:ss.mil*.

This field is displayed only if the fileset is in the STARTED state.

ftiomode

is the fault-tolerance and buffering attribute to be used when file opens use O_SYNC in the fileset. The value displayed is one of the keywords described for the FTIOMODE attribute of the ADD FILESET Command on page 12-7.

normaliomode

is the fault-tolerance and buffering attribute to be used when file opens do not use O_SYNC in the fileset. The value displayed is one of the keywords described for the NORMALIOMODE attribute of the ADD FILESET Command on page 12-7.

number5

is the number of disk volumes in the storage-pool file for the fileset.

volume_list1

lists the names of the disk volumes in the fileset's storage pool. If the fileset is in the STARTED state, the complete volume list for the storage pool is displayed; otherwise only the creation pool volume list is displayed.

The following fields are displayed only when the fileset is in the STARTED state:

volume_list2

lists the names of the disk volumes in the fileset's storage pool that have space available for new file creation. If the fileset is mounted with the READONLY attribute set to TRUE, this field is blank.

volume_list3

lists the names of the disk volumes in the fileset's storage pool that do not have space available for new file creation.

volume_list4

lists the names of the disk volumes in the fileset's storage pool that currently contain no files and can be safely removed from the storage pool.

volume_list5

lists the names of the disk volumes in the storage-pool for the fileset that already contain OSS files.

The `VolumeInformation` fields also are displayed only when the fileset is in the STARTED state:

`M`

indicates whether the volume is mirrored.

$vol

is the name of a disk volume in the fileset's storage pool.

size1

is the number of megabytes allocatable for OSS or Guardian files.

size2

is the number of megabytes available for new OSS or Guardian files.

nnn

is the percentage of space available for new OSS or Guardian files.

num

is the number of extents allocatable for OSS or Guardian files.

size3

is the number of megabytes available for OSS or Guardian files in the largest of those extents.

Considerations

- The DETAIL display for an unmounted fileset differs from that of a mounted (started) fileset. If a fileset is not in the STARTED state, its catalog usage information, disk configuration, disk status information, and fileset attributes are not displayed.

Examples

- To determine the status of the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

  ```
  STATUS /OUT CMDLOG/ FILESET $ZPMON.USER1
  ```

  A display such as the following is written to CMDLOG:

```
OSS Status FILESET USER1

FilesetName                State         LastError      ErrorDetail
USER1                      STARTED           0               0
```

- To see a listing of all information available for the fileset USER1, enter the following command:

  ```
  STATUS /OUT CMDLOG/ FILESET $ZPMON.USER1, DETAIL
  ```

A display such as the following appears:

```
OSS Detailed Status FILESET \NODE1.$ZPMON.USER1

 State................................. STARTED
 MountTime............................. 14 Nov 2004, 14:25:52.396
 LastError............................. 0
 ErrorDetail........................... 0
 ErrorTime............................. 22 Nov 2004, 16:25:38.835
 FsckName.............................. $DATA
 LastControlSyncTime................... 15 Nov 2004, 12:23:43.234
 AlterAfterSyncOrMount................. TRUE
*AuditEnabled.......................... OFF
*Buffered.............................. NONE
*ReadOnly.............................. FALSE
*NFSTimeout............................ 120
*NFSPool............................... 16
*MaxDirtyInodeTime..................... 30
*MaxInodes............................. 300000
 InuseInodes........................... 200000 (67% of MaxInodes)
 MaximumCountInodesUsed................ 250000 (83% of MaxInodes)
 TimestampMaximumCountInodesUsed....... 18 Nov 2004, 22:34:43.343
 TimestampMaximumCountInodesReset...... 17 Nov 2004, 12:23:43:293
*FTIOmode.............................. UNBUFFEREDCP
*NormalIOmode.......................... OSSBUFFEREDCP
 NumVols............................... 6
 Volumes:
 $OSS1 $OSS2 $OSS3 $OSS4 $OSS5 $OSS6
*VolumesEligibleForFileCreation:
 $OSS1 $OSS2 $OSS3
 VolumesNotEligibleForFileCreation:
 $OSS4 $OSS5 $OSS6
 VolumesWithZeroFiles:
 $OSS4 $OSS1
 VolumesWithFiles:
 $OSS2 $OSS3 $OSS5 $OSS6
 VolumeInformation:
           -- Capacity (Mb) -- % -- Free Extents --
 Volume (M) Total Free        Free Count Biggest
 $OSS2      4238  2622.40      61      1 2622.40
 $OSS3      4238  2860.44      67      1 2860.44
 $OSS4      4238  2642.02      62      3 2642.01
 $OSS5      4238  2735.44      64      4 2735.40
 $OSS6 (unavailable -- device error 201 )
```

To determine the status of the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

```
STATUS /OUT CMDLOG/ FILESET $ZPMON.USER1
```

A display such as the following is written to CMDLOG:

```
OSS Status FILESET USER1

FilesetName                 State         LastError      ErrorDetail
USER1                       STARTED           0               0
```

STATUS SERVER Command

The STATUS SERVER command displays status information about a server administered by the OSS Monitor. The syntax of the STATUS SERVER command is:

```
STATUS [ /OUT filename/ ] SERVER server_processname

       [ , DETAIL ]

       [ , SEL [ NOT ] state ]
```

`OUT filename`

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`SERVER server_processname`

specifies the server whose status you want to display. *server_processname* has the following form:

`[$ZPMON.]servername`

servername

specifies the name of the server whose status you want to display. Only names currently defined in the ZOSSSERV file are valid values.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

servername can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

You can specify the following options in any order:

`DETAIL`

produces a detailed report.

If the DETAIL option is omitted, only a summary report is produced.

`SEL [ NOT ] state`

directs the command to apply only to servers that are in the specified state or, when the NOT field is used, only to servers that are not in the specified state. Valid values for *state* are:

Value	Meaning
STARTED	The server is started.
STOPPED	The server is ready to be started.

The information returned for a summary report has the following format:

```
OSS Status SERVER server-devicename

ServerName                  State       LastError     ErrorDetail
servername                  state       error         error-detail
```

server-devicename

is the server name of the server whose status is displayed, shown as a device of the OSS Monitor process.

servername

is the server name of the server whose status is displayed.

state

describes the state of the server.

Value	Meaning
STARTED	The server process is running.
STOPPED	The server process is not running.

error

is the error number of the last OSS Monitor error that occurred during an operation on the server. See Numbered Messages on page A-35 for an explanation of a specific error number.

error-detail

is the error number of the Guardian file-system error (if any) reported with the *error* value. For information about Guardian file-system errors, see the *Guardian Procedure Errors and Messages Manual*.

The information returned for a detailed report has this format:

```
OSS Detailed Status SERVER server-devicename

 State.................. state
 StartTime.............. time1
 LastError.............. error
 ErrorDetail............ error-detail
 ErrorTime.............. time2
 LastControlSyncTime.... time3
 AlterAfterSyncOrStart.. status
*CPU.................... processor1
*BackupCPU.............. processor2
*SQLTimeout............. time4
*InodeCache............. size1
 InodeCacheInUse........ size2
 MaxInodeCacheUsed...... size3
 TimeMaxInodeCacheUsed.. time5
 TimeMaxInodeCacheReset. time6
*LinkCache.............. size4
 LinkCacheInUse......... size5
 MaxLinkCacheUsed....... size6
TimeMaxLinkCacheUsed.... time7
TimeMaxLinkCacheReset... time8
```

server-devicename

is the server name of the server whose status is being displayed, shown as a device of the OSS Monitor process.

state

is the state of the server. The value is one of the following:

Value	Meaning
STARTED	The server process is running.
STOPPED	The server process is not running.

time1

is the timestamp for the time when the server was last started, in the form *dd mmm yyyy hh:mm:ss.mil*.

error

is the error number of the last OSS Monitor error that occurred during an operation on the server. See Numbered Messages on page A-35 for an explanation of a specific error number.

error-detail

is the error number of the Guardian file-system error (if any) reported with the *error* value. For information about Guardian file-system errors, see the *Guardian Procedure Errors and Messages Manual*.

time2

is the timestamp for the time when the last error for the server was reported, in the form *dd mmm yyyy hh:mm:ss.mil*. If no error has been reported, this value is the same as the *time1* value.

The following fields are displayed only when an OSS name server is in the STARTED state:

time3

is the timestamp for the time when the OSS name server was last synchronized with ZPOSFSERV, in the form *dd mmm yyyy hh:mm:ss.mil*. If a CONTROL SERVER command with the SYNC option has not been issued since the OSS name server was last started, this field is blank.

status

indicates whether the configuration of the OSS name server has changed after its most recent start or synchronization with ZOSSSERV:

Value	Meaning
FALSE	The OSS name server configuration has not changed.
TRUE	The OSS name server configuration has changed.

processor1

indicates the processor in which the primary copy of the OSS name server process runs.

processor2

indicates the processor in which the backup copy of the OSS name server process runs.

time4

indicates the number of seconds that the OSS name server waits for a response from SQLCAT to a request.

size1

indicates the number of inodes that the OSS name server is allowed to cache.

size2

indicates current inode cache usage by the OSS name server, displayed as the number of entries in use and as a percentage of *size1*.

size3

indicates the most recent highwater mark for inode cache use, displayed as the number of entries used and as a percentage of *size1*.

time5

is the timestamp for the time when the OSS name server last reached its highwater mark for inode cache use, in the form *dd mmm yyyy hh:mm:ss.mil*.

time6

is the timestamp for the time when the OSS name server last had its highwater mark for inode cache use reset, in the form *dd mmm yyyy hh:mm:ss.mil*.

size4

indicates the number of links that the OSS name server is allowed to cache.

size5

indicates current link cache usage by the OSS name server, displayed as the number of entries in use and as a percentage of *size4*.

size6

indicates the most recent highwater mark for link entries in cache, displayed as the number of entries used and as a percentage of *size4*.

time7

is the timestamp for the time when the OSS name server last reached its highwater mark for link cache use, in the form *dd mmm yyyy hh:mm:ss.mil*.

time8

is the timestamp for the time when the OSS name server last had its highwater mark for link cache use reset, in the form *dd mmm yyyy hh:mm:ss.mil*.

Considerations

- The DETAIL display for an unstarted OSS name server differs from that of a started OSS name server. If an OSS name server is not in the STARTED state, its inode cache information, link cache configuration, and OSS name server attributes are not displayed.

Examples

- To obtain detailed information about the status of the OSS name server for the root fileset (#ZPNS), enter the following command:

```
STATUS SERVER #ZPNS, DETAIL
```

A display similar to the following is returned:

```
OSS Detailed Status SERVER \NODE1.$ZPMON.#ZPNS

 State..................... STARTED
 StartTime................. 25 Nov 2002, 10:23:20.123
 LastError.................       0
 ErrorDetail...............       0
 ErrorTime................. 25 Nov 2002, 10:23:20.123
 LastControlSyncTime....... 25 Nov 2002, 12:23:43.234
 AlterAfterSyncOrStart..... TRUE
*CPU....................... 1
*BackupCPU................. 0
*SQLTimeout................ 60
*InodeCache................ 4096
 InodeCacheInuse........... 100 (2% of InodeCache)
 MaxInodeCacheUsed......... 2993 (73% of InodeCache)
 TimeMaxInodeCacheUsed..... 19 Nov 2002,23:23:34.343
 TimeMaxInodeCacheReset.... 19 Nov 2002,12:22:34.343
*LinkCache................. 4096
 LinkCacheInuse............ 200 (5% of LinkCache)
 MaxLinkCacheUsed.......... 1933 (47% of LinkCache)
 TimeMaxLinkCacheUsed...... 19 Nov 2002,23:23:34.344
 TimeMaxLinkCacheReset..... 19 Nov 2002,12:24:12.123
```

- To obtain information about the OSS sockets local server $ZPLS and send informational messages to the file CMDLOG, enter the following command:

  ```
  STATUS /OUT CMDLOG/ SERVER #ZPLS
  ```

- To obtain summary information about all configured servers and send informational messages to the file CMDLOG, enter the following command:

  ```
  STATUS /OUT CMDLOG/ SERVER *
  ```

STOP FILESET Command

The STOP FILESET command makes an existing, started OSS fileset unavailable to users (also known as unmounting the fileset).

The syntax of the STOP FILESET command is:

```
STOP [ /OUT filename/ ] FILESET [$ZPMON.]filesetname
```

`OUT` *`filename`*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

`FILESET [$ZPMON.]`*`filesetname`*

specifies the name of the fileset you are stopping. *`filesetname`* is the name in the NAME field of the ZOSSFSET file entry for the fileset.

`filesetname` can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- The STOP FILESET command does not close any open OSS files or directories on the affected fileset.
- The STOP FILESET command can be used only by super-group users (255,*nnn*).
- A fileset cannot be stopped while other filesets are mounted on it.

Examples

- To stop (unmount) the fileset USER1 and send informational messages to the file CMDLOG, enter the following command:

  ```
  STOP /OUT CMDLOG/ FILESET $ZPMON.USER1
  ```

- To stop (unmount) all started filesets and send informational messages to the file CMDLOG, enter the following command:

  ```
  STOP /OUT CMDLOG/ FILESET $ZPMON.*
  ```

STOP SERVER Command

The STOP SERVER command stops an OSS server.

The syntax of the STOP SERVER command is:

```
STOP [ /OUT filename/ ] SERVER server_processname
```

OUT *filename*

specifies the name of a Guardian output file for informational messages. You can either read this file with a text editor or display it with the FUP COPY command.

SERVER *server_processname*

specifies the server being stopped. *server_processname* has the following form:

[$ZPMON.]*servername*

servername

specifies the name of the server to stop. Only names currently defined in the ZOSSSERV file are valid values.

The first character of the name must be a pound sign (#). Server names are not case-sensitive.

servername can contain wildcard characters. (See Using Wildcard Characters in OSS Monitor Commands: on page 2-13 for the definition of UNIX wildcard characters.)

The $ZPMON prefix can be omitted if you have previously specified $ZPMON in an SCF ASSUME command.

Considerations

- The STOP SERVER command can be used to stop the OSS sockets local server, the OSS message-queue server, and the OSS transport agent servers. An OSS name server is automatically stopped when all filesets managed by that OSS name server are stopped.
- The STOP SERVER command can be used only by super-group users (255,*nnn*).

Examples

- To stop the OSS sockets local server $ZPLS and send informational messages to the file CMDLOG, enter the following command:

```
STOP /OUT CMDLOG/ SERVER #ZPLS
```

- To stop all OSS sockets local servers, OSS message-queue servers, and OSS transport agent servers and send informational messages to the file CMDLOG, enter the following command:

```
STOP /OUT CMDLOG/ SERVER *
```

VERSION SUBSYS, VERSION MON, and VERSION PROCESS Commands

The VERSION SUBSYS, VERSION MON, and VERSION PROCESS commands all display product-version information for a specified OSS object known to SCP.

The syntax of the VERSION SUBSYS, VERSION MON, and VERSION PROCESS command is:

```
VERSION [ /OUT filename/ ] [ object-type ] [ processname ]
        [ , DETAIL ]
```

OUT *filename*

directs any output generated for this command to the specified file. You can either read this file with a text editor or display it with the FUP COPY command.

object-type

specifies the types of objects whose version information is to be returned. *object-type* has the following form:

```
MON | PROCESS | SUBSYS
```

MON, PROCESS, and SUBSYS all specify the OSS Monitor.

If *object-type* is omitted, *processname* must be specified unless ASSUME PROCESS $ZPMON was previously specified.

processname

specifies the name of the process whose version is to be displayed. Only the process name $ZPMON is supported.

processname must be specified unless ASSUME PROCESS $ZPMON or ASSUME $ZPMON was previously specified.

DETAIL

indicates that all available version information is to be returned for the specified object. If DETAIL is omitted, only one line of information is returned.

The format of a one-line VERSION display is:

```
VERSION process-filename: subsystem-name version-information-banner
```

process-filename

is the name of the object whose version information is displayed, shown as a Guardian process filename.

subsystem-name

is the name of the SCF subsystem associated with the object.

version-information-banner

provides the product-version information for the object. See Section 11, Managing Problems, for more information about product-version information and the related Guardian VPROC utility.

The format of a detailed VERSION display is:

```
Detailed VERSION process-filename
[  SYSTEM system-name   ]
      subsystem-name1 version-information-banner
[     GUARDIAN NonStop-OS-version-information-banner   ]
   SYSTEM system-name
      subsystem-name1 version-information-banner
      GUARDIAN NonStop-OS-version-information-banner
      SCF KERNEL version-information-banner
      subsystem-name2 PM version-information-banner
```

process-filename

is the name of the object whose version information is displayed, shown as a Guardian process filename.

system-name

is the node name of the system executing the object.

subsystem-name1

is the name of the SCF subsystem associated with the object.

version-information-banner

provides the product-version information for the software named on the same display line. See Section 11, Managing Problems, for more information about product-version information and the related Guardian VPROC utility.

NonStop-OS-version-information-banner

provides the product-version information for the copy of the NonStop operating system that is running on the node indicated by the preceding *system-name*. See Section 11, Managing Problems, for more information about product-version information and the related Guardian VPROC utility.

subsystem-name2

is the name of the SCF subsystem product module associated with the object.

Considerations

- The product-version information for the SCF Kernel subsystem is displayed as part of the SCF startup banner and when the SCF ENV command is used.
- In the SCF object hierarchy, SUBSYS is the highest of the possible objects for this command.

Examples

- The command

  ```
  VERSION $ZPMON
  ```

  produces output such as the following:

  ```
  VERSION SUBSYS \NODE.$ZPMON: OSS Monitor-T8622D46-01SEP98-AAF
  ```

  where the information after the colon (:) is the subsystem name and product-version information associated with $ZPMON.

- The command

  ```
  VERSION SUBSYS $ZPMON, DETAIL
  ```

  produces output similar to the following:

  ```
  Detailed VERSION \SIERRA.$ZPMON
      SYSTEM \SIERRA
        OSS Monitor - T8622G09-01FEB01
        GUARDIAN - T9050 - (Q06)
        SCF KERNEL - T9082G02 - (19JUN01) (11APR01)
        OSS PM - T8623G09 - (01FEB01)
  ```

A Messages

Status, warning, and error messages are sent to the console by:

- The Open System Services (OSS) EasySetup utilities. These messages are described under OSS EasySetup Utility Messages on page A-2.
- The Guardian Catalog Volume Tool (CVT) utility. These messages are described under CVT Messages on page A-3.
- The FSCK fileset integrity checker when the Subsystem Control Facility (SCF) DELETE FILESET or DIAGNOSE FILESET command is used. These messages are described under FSCK Messages on page A-6.
- The OSS Monitor process at startup. These messages are described under Unnumbered Messages on page A-27.
- The OSS product module (OSS PM) when SCF commands are used. These messages are described under Numbered Messages on page A-35.
- The OSSTTY utility at startup. These messages are described under Startup Messages on page A-58.

Event Management Service (EMS) event messages are described in the *Operator Messages Manual* only. EMS event messages are issued by:

- The OSS message-queue server, under the NSK subsystem ID.
- The OSS Monitor, under the OSS subsystem ID.
- An OSS name server, under the OSS subsystem ID.
- The FSCK fileset integrity checker, under the OSS subsystem ID, when the SCF DIAGNOSE FILESET command is used.
- The OSS sockets local server, under the OSS subsystem ID.
- An OSS transport agent server, under the TAG subsystem ID.
- The OSS terminal helper server, under the OSS subsystem ID.
- The OSS EasySetup utilities or OSS server processes such as `inetd`, under the OSS subsystem ID.
- The OSSTTY server, under the OSSTTY subsystem ID.

The OSS, OSSTTY, and NonStop Kernel subsystems generate EMS event messages that are common to many SCF product modules. Such messages are described in the *SCF Reference Manual for G-Series RVUs* or the *SCF Reference Manual for H-Series RVUs* and are not described in this guide. Such messages have negative message numbers (for example, E-00019); only messages specific to the OSS subsystem with positive message numbers (for example, E00019) are described in this guide.

For further information about EMS messages, see the *Operator Messages Manual*.

OSS EasySetup Utility Messages

The following console messages can appear on the terminal that issued an EasySetup command. The OSS EasySetup utilities can also generate EMS events, which are issued under the OSS subsystem ID. For information about these messages, see the *Operator Messages Manual*.

```
WARNING - Event definitions file (filename) not loaded
because file could not be found
```

filename

indicates the Guardian file that cannot be found, which is usually $SYSTEM.ZSPIDEF.ZEMSTACL or $SYSTEM.ZSPIDEF.ZOSSTACL.

Cause. The utility cannot find the indicated event definition file, a Subsystem Programmatic Interface (SPI) file containing the Event Management Service (EMS) event definitions that the utility needs to issue event messages. This problem commonly occurs when the SPI files either have not been installed, have been installed on a volume other than $SYSTEM, or have been installed in a subvolume other than ZSPIDEF.

Effect. All utilities continue processing. No EMS events are generated.

Recovery. If EMS events are needed, install or reinstall the missing SPI file, carefully following the instructions for DSM/SCM use.

```
WARNING - Variable (variable_name) needed but does not exist
```

variable_name

indicates the Event Management Service (EMS) message token for which a definition cannot be found.

Cause. The utility cannot find the event definition file (usually $SYSTEM.ZSPIDEF.ZEMSTACL or $SYSTEM.ZSPIDEF.ZOSSTACL) that contains the token definition. The event definition file is a Subsystem Programmatic Interface (SPI) file containing the EMS message definitions that the utility needs to issue event messages. This problem commonly occurs when the SPI files either have not been installed, have been installed on a volume other than $SYSTEM, or have been installed in a subvolume other than ZSPIDEF.

Effect. All utilities continue processing. No EMS events are generated.

Recovery. If EMS events are needed, install or reinstall the missing SPI file, carefully following the instructions for DSM/SCM use.

CVT Messages

The following warning and error messages appear on the terminal that issued the CVT command. The Guardian CVT utility does not generate Event Management Service (EMS) events.

CVT Warning Message

CVT issues the following warning message.

```
Warning:filename - No such File
```

filename

is a Guardian filename used in the command.

Cause. The named file does not exist.

Effect. CVT continues.

Recovery. Check the command and correct it as necessary.

CVT Error Messages

CVT issues the following error messages, which are listed in alphabetic order. After issuing an error message, CVT terminates.

```
***Command Error*** - token
```

token

identifies the invalid parameter.

Cause. The value *token* appears where CVT expects something else.

Effect. CVT terminates.

Recovery. Check the command and correct it as necessary.

```
FILE_OPEN_ Error error on file filename
```

error

is a Guardian file-system error value returned by the underlying Guardian FILE_OPEN_ procedure call. See the *Guardian Procedure Errors and Messages Manual* for information about the indicated error.

filename

identifies the affected file.

Cause. The Guardian FILE_OPEN_ procedure call could not be completed.

Effect. CVT did not open the indicated file.

Recovery. Check the command and correct it as necessary.

```
FILE_PURGE_ Error error on file filename
```

error

is a Guardian file-system error value returned by the underlying Guardian FILE_PURGE_ procedure call. See the *Guardian Procedure Errors and Messages Manual* for information about the indicated error.

filename

identifies the affected file.

Cause. The Guardian FILE_PURGE_ procedure call could not be completed.

Effect. CVT did not purge the indicated file.

Recovery. Check the command and correct it as necessary.

```
FILE_RENAME_ Error error on file filename
```

error

is a Guardian file-system error value returned by the underlying Guardian FILE_RENAME_ procedure call. See the *Guardian Procedure Errors and Messages Manual* for information about the indicated error.

filename

identifies the affected file.

Cause. The Guardian FILE_RENAME_ procedure call could not be completed.

Effect. CVT did not rename the indicated file.

Recovery. Check the command and correct it as necessary.

```
***fsck needed -- subvolume.PXCKSTAT exists***
```

subvolume

identifies the affected subvolume.

Cause. There is a PXCKSTAT file on one of the subvolumes involved in processing the command. CVT does not move catalogs to, from, or within a subvolume that contains a PXCKSTAT file—nor does it purge a saved catalog from such a subvolume.

Effect. CVT terminates.

Recovery. Run the FSCK utility against the catalog before attempting the CVT command again.

```
***Incomplete Command***
```

Cause. The command entered was incomplete.

Effect. CVT terminates.

Recovery. Check the command and correct it as necessary.

```
***Internal Error***
```

Cause. CVT has detected an internal inconsistency.

Effect. CVT terminates.

Recovery. Enter the command again with the INSPECT SAVEABEND RUN option to produce a saveabend file. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
***Invalid Serial Number*** - token
```

token

identifies the invalid value found in place of an FSCK serial number (FSN).

Cause. The value *token* followed the keyword SERIAL in the command line but is not a valid FSN value.

Effect. CVT terminates.

Recovery. List the files in the subvolume to see the FSN values used in their Guardian file IDs. Try the command again with an FSN from those shown in the file IDs. See Generated Catalog Files on page 5-33 for more information about FSN values.

```
***Invalid Subvolume Name*** - token
```

token

identifies the invalid subvolume name.

Cause. The value *token* followed the keyword IN in the command line, but it is not a valid Guardian subvolume name.

Effect. CVT terminates.

Recovery. Check the command and correct it as necessary.

```
***Unexpected Argument*** - token
```

token

identifies the extra parameter.

Cause. The value *token* appears in the command line, but what precedes it already constitutes a complete CVT command.

Effect. CVT terminates.

Recovery. Check the command and correct it as necessary.

FSCK Messages

The FSCK utility runs when you use the SCF DIAGNOSE FILESET or DELETE FILESET command. The FSCK messages appear in the FSCK log file. The FSCK utility can issue the following kinds of messages:

- Consistent-fileset messages, which indicate that FSCK found no relevant inconsistencies or errors in a fileset
- Inconsistency and error messages, which indicate that FSCK found an inconsistency or an error in a fileset or in fileset-related information

For more information about the inconsistencies that FSCK checks for, see Inconsistencies Checked by FSCK on page 5-29.

FSCK Consistent-Fileset Messages

If the FSCK utility finds no relevant inconsistencies or errors in a fileset, it issues one of the following messages (listed in alphabetic order). After issuing one of these messages, FSCK clears the mounted flag and the bitmap indicating the types of inconsistencies previously detected by the OSS name server.

```
*** CATALOG AND ZYQ SUBVOLUMES HAVE BEEN PURGED ***
```

Cause. FSCK successfully deleted a fileset.

Effect. None.

Recovery. Informational message only; no corrective action is needed.

```
*** NO INCONSISTENCIES DETECTED -- EXISTING CATALOG RETAINED
***
```

Cause. FSCK detected no inconsistencies and retained the current catalog files.

Effect. None. A new catalog is not created.

Recovery. Informational message only; no corrective action is needed.

```
*** NO SERIOUS INCONSISTENCIES DETECTED -- EXISTING CATALOG
RETAINED ***
```

Cause. You specified REPAIR SERIOUS as a repair option in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, and FSCK detected no serious inconsistencies.

Effect. Minor inconsistencies are corrected if necessary. A new catalog is not created.

Recovery. Informational message only; no corrective action is needed.

FSCK Inconsistency and Error Messages

The FSCK utility produces a message in one of the following classes if it detects at least one inconsistency or error:

ERROR	These messages are issued when FSCK encounters an irrecoverable error. Such messages report unexpected conditions that FSCK cannot manage. After issuing an ERROR message, FSCK always terminates abnormally.
MINOR	These messages are issued when FSCK detects a minor inconsistency. A catalog integrity problem exists.

SERIOUS	These messages are issued when FSCK detects a serious inconsistency. A catalog integrity problem exists.
WARNING	These messages are issued when a recoverable error occurs or when FSCK takes an action (such as purging an empty orphan ZYQ file) that is not reported as an inconsistency. WARNING messages report conditions that FSCK encounters or actions that FSCK takes that do not by themselves indicate problems with the catalog.

To recover files for which no parent directory can be located, FSCK must create a new parent directory. This directory is given the name `lost+found`, and its parent directory is the root directory of the fileset.

When an inode is placed in `/lost+found`, it is given the name `#`*`inode-number`*; for example, `/lost+found/#3926`.

The class of a message is given by the first word of the message (ERROR, MINOR, SERIOUS, or WARNING). The inconsistency and error messages that FSCK produces are listed on the following pages in order by message number.

3

```
{ WARNING | ERROR } 3 - operation Error error-number
(description) on filename
```

operation

identifies the uncompleted operation.

error-number

provides the Guardian file-system error number of the error that occurred.

description

describes the error that occurred.

filename

identifies the affected file.

Cause. The FSCK utility encountered a file-management error on an operation (given by *operation*) on the indicated file because of the error described by *error-number* and *description*. Some common examples of this message are:

Message	Meaning
OPEN Error 11 (Record Not Found) on PXINODE	There is no OSS fileset catalog in the specified (or implied) catalog subvolume. The ZOSSFSET file specifies an incorrect volume.
OPEN Error 12 (File In Use) on PXCKSTAT	Another copy of FSCK is checking the catalog.
OPEN Error 14 (No Such Device) on PXCKSTAT	The specified (or implied) catalog subvolume refers to a nonexistent volume. The ZOSSFSET file specifies a nonexistent volume.

Note that the severity of this message can be either WARNING or ERROR.

Effect. If the severity of the message is ERROR, FSCK terminates abnormally. If the severity of the error is WARNING, FSCK attempts to recover from the error by retrying the operation.

Recovery. If the severity of the message is WARNING, the message is an informational message only; no corrective action is needed.

If the severity of the message is ERROR, recovery actions depend on the specific cause of the error. See the related Guardian file-system error explanation in the *Guardian Procedure Errors and Messages Manual* for possible actions.

6

```
ERROR 6 - Catalog Version not supported by this program
```

Cause. The catalog is at a version higher than this version of the FSCK utility can handle. The catalog was created by an incompatible version of the OSS file system, perhaps from a software release version update (RVU) more recent than the RVU containing this copy of FSCK.

This message can indicate that you are attempting an operation on a D40 catalog using a version of FSCK preceding the D30.02 RVU.

Effect. The FSCK utility terminates abnormally.

Recovery. Check that the correct catalog is being used. Use a more recent version of the OSS Monitor or invoke a more recent version of FSCK.

8

```
WARNING 8 - FSCK Run Number nnnn was Interrupted
```

nnnn

identifies the FSCK serial number (FSN) of the affected invocation of FSCK.

Cause. The indicated invocation of FSCK was interrupted before it finished.

Effect. A new FSN is not assigned. If FSCK did not finish creating a new catalog, the existing saved catalog files (PXIN*nnnn*, PXLI*nnnn*, and PXLO*nnnn*) are used. If a new catalog was successfully created, the current invocation of FSCK purges the PXCKSTAT file, issues the following message:

```
*** FSCK RUN NUMBER nnnn COMPLETED SUCCESSFULLY -- NEW CATALOG
RETAINED ***
```

and terminates normally.

Recovery. Informational message only; no corrective action is needed.

13

```
WARNING 13 - Can't UPGRADE/DOWNGRADE catalog with
CORRUPT/MISSING Super Block
```

Cause. The FSCK utility was asked to upgrade or downgrade a catalog, but it cannot determine the catalog version due to a corrupt or missing superblock.

Effect. FSCK does not upgrade or downgrade the catalog.

Recovery. Repair the catalog with the version of FSCK that matches the actual version of the catalog (perhaps by running a different version of the OSS Monitor), then retry the upgrade or downgrade operation.

14

```
WARNING 14 - Catalog Already Upgraded
```

Cause. The FSCK utility was asked to upgrade a catalog to the current format, but the catalog is already in that format.

Effect. FSCK does not upgrade the catalog.

Recovery. Informational message only; no corrective action is needed.

15

```
WARNING 15 - Catalog Already Downgraded
```

Cause. The FSCK utility was asked to downgrade a catalog to the previous format, but the catalog is already in that format.

Effect. FSCK does not downgrade the catalog.

Recovery. Informational message only; no corrective action is needed.

16

```
WARNING 16 - Dirty Catalog using Fast Create; REPAIR ALL will
be performed
```

Cause. In a Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the REPAIR option was specified for a catalog that used the fast-create fileset option and had not been unmounted cleanly.

Effect. The FSCK utility proceeds as if the REPAIR ALL option were selected, because catalogs that use the fast-create option can be repaired only by using the REPAIR ALL option.

Recovery. Informational message only; no corrective action is needed.

98

```
ERROR 98 - INTERNAL ERROR
```

Cause. The FSCK utility detected a condition that indicates an internal logic error.

Effect. FSCK terminates abnormally.

Recovery. Set the SAVEABEND attribute of the FSCK program file using the `nld` command. Rerun FSCK to produce a saveabend file. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

99

```
ERROR 99 - HEAP OVERFLOW
```

Cause. The FSCK utility has exhausted its internal memory.

Effect. FSCK terminates abnormally.

Recovery. Set the SAVEABEND attribute of the FSCK program file using the `nld` command. Rerun FSCK to produce a saveabend file. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

100

```
SERIOUS 100 - Corrupt PXLINK Record - Parent:parent,
Child:child, Name:linkname
```

parent

indicates the inode number of the parent end of the link.

child

indicates the inode number of the child end of the link.

linkname

identifies the affected link.

Cause. The FSCK utility detected a corrupt record in the PXLINK file.

Effect. This record is not included in the new PXLINK file. Any inode orphaned as a result is placed in the OSS `/lost+found` directory with the name *#inode-number*.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

101

```
SERIOUS 101 - Duplicate Link ID - Parent:parent, Child:child,
Name:link linkname
```

parent

indicates the inode number of the parent end of the link.

child

indicates the inode number of the child end of the link.

linkname

identifies the affected link.

Cause. The indicated link has the same link ID as another link between the same two inodes.

Effect. When FSCK creates a new PXLINK file, it assigns a new unique link ID.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

102

```
SERIOUS 102 - Missing Link - Parent:parent, Child:child,
Name:link linkname
```

parent

indicates the inode number of the parent end of the link.

child

indicates the inode number of the child end of the link.

linkname

identifies the affected link.

Cause. The indicated PXLINK record was missing in the old PXLINK file.

Effect. The indicated record is added to the new PXLINK file. The missing link will be for one of the reserved inodes (/, /G, /E, /lost+found, /dev, /dev/tty, /dev/null, or the superblock).

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

200

```
SERIOUS 200 - Invalid Parent List, Inode= inode-number
```

inode-number

indicates the affected inode.

Cause. Each record in PXINODE contains a list of parent inode numbers. (The parent of an inode is a directory containing a link to the inode.) This error indicates that the list in the indicated inode did not accurately reflect its parents.

Effect. If, after correction, the list of parent inode numbers is empty, the inode is placed in the OSS /lost+found directory.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

201

```
SERIOUS 201 - Broken Free List, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. The PXINODE file contains two free lists of inode numbers. This error indicates that one of these lists is corrupted at the indicated inode number.

Effect. The corrupted free list is rebuilt.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the

catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

202

```
SERIOUS 202 - Corrupt Inode, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. The FSCK utility has detected a corrupt record in the PXINODE file.

Effect. The record is omitted from the new PXINODE file.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

204

```
SERIOUS 204 - Too Many Parents, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. After correction, the parent list for an inode contains more than the permissible number of parent links (1 for directories and symbolic links, and 128 for other files).

Effect. The FSCK utility drops enough links (PXLINK records) to bring the number down below the limit.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

205

```
MINOR 205 - Missing ZYQ File, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. The indicated regular file inode has no corresponding ZYQ file.

Effect. The inode and any links to it are omitted from the new catalog.

Recovery. If the REPAIR option was not specified or the REPAIR NONE, REPAIR SERIOUS, or REPAIR OPEN option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

206

```
SERIOUS 206 - Parent Inode Not Directory, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. A parent of the indicated inode is not a directory.

Effect. Any links from the nondirectory parent are omitted from the new PXLINK file. If this results in all links to the file being omitted, the file is placed in the OSS `/lost+found` directory.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

207

```
MINOR 207 - OSS Name Server Failed while Fileset was Mounted
```

Cause. The mounted flag in the superblock was nonzero, indicating that the catalog may contain inconsistencies.

Effect. The mounted flag is reset to zero in the new PXINODE file.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

208

```
MINOR 208 - There are nnn Inode Numbers Unaccounted For
```

nnn

indicates the number of affected inode numbers.

Cause. There are *nnn* inode numbers for which there were neither records in the PXINODE file nor entries in the free inode tables.

Effect. These inode numbers are recorded as free in the new PXINODE file.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

209

```
{ MINOR | SERIOUS } 209 - Invalid Inode Number,
Inode=inode-number
```

inode-number

indicates the affected inode.

For the MINOR message:

Cause. A record with the indicated invalid inode number as a key was found in the current PXINODE file. The inode number falls within the range reserved for future special inodes.

Effect. The record is assigned a currently unused inode number in the new PXINODE file, and all links to and from the file are mapped to the new inode number. If the inode represents a regular file, the corresponding ZYQ file is renamed to reflect the new inode number.

Recovery. Informational message only; no corrective action is needed.

For the SERIOUS message:

Cause. A record with the indicated invalid inode number as a key was found in the current PXINODE file. The inode number is less than or equal to zero or is greater than 2^{31}.

Effect. The record is omitted from the new PXINODE file.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

210

```
SERIOUS 210 - Missing Inode, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. There are references to the indicated inode, either in parent lists or in PXLINK records, but the inode does not exist in the PXINODE file.

Effect. If the inode is one of the reserved inodes (`/`, `/G`, `/E`, `/lost+found`, `/dev`, `/dev/tty`, `/dev/null`, or the superblock), it is added to the PXINODE file.

If the inode has appeared as a parent in one or more PXLINK records, the inode is added to the PXINODE file as a directory.

If there is a nonempty ZYQ file and the disk process indicates a nonzero number of links corresponding to this inode, the inode is added as a regular file that refers to the ZYQ file; otherwise, all references to the inode are dropped in the new catalog.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

211

```
SERIOUS 211 - Loop in Directory Graph, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. The directory graph forms a loop that includes the indicated inode. The inode is linked through other inodes back to itself; this means that the directory referred to by the inode number is its own parent.

Effect. The inode is unlinked and placed in the OSS `/lost+found` directory.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

212

```
SERIOUS 212 - Orphan Inode, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. This inode appears in the PXINODE file but has no links in the PXLINK file.

Effect. The inode is added to the OSS `/lost+found` directory.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

213

```
MINOR 213 - Orphan ZYQ File - filename
```

filename

indicates the affected file.

Cause. The indicated ZYQ file has no corresponding inode in the PXINODE file.

Effect. A currently unused inode is allocated, the ZYQ file is renamed to correspond to the new inode, and the inode is placed in the OSS `/lost+found` directory.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

214

```
MINOR 214 - Catalog/File Label Mismatch, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. For the indicated inode, either of the following is true:

1. The disk process's copy of the number of links to the file is incorrect.
2. The file creation time or creation version serial number (CRVSN) in the fileset catalog does not match that in the file label of the ZYQ file.

Effect. For Cause 1, FSCK sends a request to the disk process to correct the number. For Cause 2, FSCK updates the inode with the values from the file label.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

215

```
MINOR 215 - Invalid Timestamp, Inode=inode-number
```

inode-number

indicates the affected inode.

Cause. One of the time fields (`atime`, `ctime`, `mtime`, or `creationtime`) in the indicated inode is invalid.

Effect. The invalid time field is set to the current Coordinated Universal Time (UTC) in the new catalog.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

216

```
MINOR 216 - ZYQ File Conflict - filename
```

filename

indicates the affected file.

Cause. The inode corresponding to the indicated file does not refer to that file.

Effect. A currently unused inode is allocated, the ZYQ file is renamed to correspond to the new inode, and the inode is placed in the OSS `/lost+found` directory.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

217

```
WARNING 217 - Fileset is Full and there are still ZYQ File
Conflicts
```

Cause. While attempting to resolve a ZYQ conflict previously reported in FSCK message 216, FSCK was unable to find an available inode.

Effect. No further attempt is made to resolve these conflicts.

Recovery. Informational message only; no corrective action is needed.

218

```
SERIOUS 218 - Not a Root Fileset
```

Cause. The FSCK utility was run with the FILESET ROOT option, but the fileset catalog does not correspond to a root fileset.

Effect. FSCK converts the fileset to a root fileset.

Recovery. If the REPAIR option was not specified or the REPAIR NONE option was specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR SERIOUS or REPAIR OPEN option.

If the REPAIR SERIOUS, REPAIR OPEN, or REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

219

```
WARNING 219 - Root Fileset
```

Cause. The FILESET ROOT option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, but the fileset catalog is a root catalog.

Effect. The FSCK utility does not check for the existence of those files that must be in a root fileset.

Recovery. Informational message only; no corrective action is needed.

220

```
WARNING 220 - The Inode Table has Overflowed to Disk
```

Cause. FSCK was forced to write information about some files to its swap file. This can occur because there was not enough extended swap disk space for FSCK to allocate heap space or because there are more than 131,072 records in the PXINODE file.

Effect. None. FSCK proceeds with its checking.

Recovery. Informational message only; no corrective action is needed.

221

```
WARNING 221 - File Omitted from New Catalog,
Inode = inode-number
```

inode-number

indicates the affected inode.

Cause. During a downgrade operation, the indicated inode cannot be included in the new catalog; the new catalog version does not support files of this inode's type.

The FSCK utility issues this message for each symbolic link found in the catalog if you are downgrading the catalog to a D30 version.

Effect. The file does not appear in the new catalog.

Recovery. Informational message only; no corrective action is needed.

222

```
WARNING 222 - Catalog will be converted from up-level format
```

Cause. A catalog at a higher version than the current version is being repaired (for example, a catalog in D40 format is being repaired by a D30.02 release version update, or RVU, copy of the FSCK utility).

Effect. A new catalog is produced that is suitable for mounting by an OSS name server from the same software RVU as the FSCK utility that issued the message.

Recovery. Informational message only; no corrective action is needed.

223

```
WARNING 223 - More than 255 disk volumes associated with this
fileset
```

Cause. An OSS fileset has files on 256 disk volumes. The maximum number of disk volumes that can be cataloged in the volume list is 256.

Effect. Cannot add any more disk volumes.

Recovery. Informational message only; no corrective action is needed.

300

```
ERROR 300 - Catalog Subvolume Full
```

Cause. The FSCK utility is unable to allocate an FSCK serial number (FSN) because there exists a PXIN*nnnn*, PXLI*nnnn*, or PXLO*nnnn* file for all *nnnn*, where 0 <= *nnnn* <= 9999.

Effect. FSCK terminates abnormally.

Recovery. Use the Guardian CVT utility to purge unneeded old catalog files.

301

```
MINOR 301 - Invalid file in ZYQ Subvolume - filename.
```

filename

indicates the affected file.

Cause. The indicated file exists in a subvolume associated with this catalog, but it is not a valid OSS file.

Effect. A currently unused inode is allocated, the invalid file is renamed to correspond to this inode, and the inode is placed in the OSS `/lost+found` directory.

Recovery. If the REPAIR ALL option was not specified in the Subsystem Control Facility (SCF) DIAGNOSE FILESET command, the catalog is still in an inconsistent state. Reissue the command with the REPAIR ALL option.

If the REPAIR ALL option was specified in the command, this is an informational message only; no corrective action is needed.

302

```
ERROR 302 - Invalid or Corrupt PXCKSTAT File
```

Cause. The PXCKSTAT file in the catalog subvolume is not a valid FSCK status file.

Effect. FSCK terminates abnormally.

Recovery. Check that PXINODE and PXLINK files still exist in the subvolume; rename appropriate PXIN*nnnn* and PXLI*nnnn* files, if needed. Purge the PXCKSTAT file and reissue the command.

If the problem persists, set the SAVEABEND attribute of the FSCK program file using either the Guardian Binder program (on a D40.00 system) or the `nld` command (on a D42.00 or subsequent D4x-series system). Rerun FSCK to produce a saveabend file. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

303

```
ERROR 303 - Catalog Inconsistent
```

Cause. The catalog subvolume is missing a PXINODE or PXLINK file and the PXIN*nnnn*, PXLI*nnnn*, or PXLO*nnnn* creation timestamps do not match those stored in the PXCKSTAT file.

Effect. The FSCK utility terminates abnormally.

Recovery. Informational message only; no corrective action is needed.

304

```
WARNING 304 - filename Purged
```

filename

indicates the affected file.

Cause. The indicated Guardian disk file is either an empty orphan ZYQ file or a file belonging to an incomplete catalog left behind when an earlier execution of FSCK was interrupted.

Effect. The indicated file is purged and FSCK proceeds with its checking.

Recovery. Informational message only; no corrective action is needed.

305

```
ERROR 305 - Invalid { PXINODE | PXLOG | PXLINK } File
```

Cause. The indicated file was not a valid OSS fileset catalog file. It had an incorrect file type, file code, or key structure.

Effect. The FSCK utility terminates abnormally.

Recovery. Check for a typographical error in the volume, subvolume, or file identifier of the filename for the correct file. Remove the invalid file, rename the appropriate PXIN*nnnn*, PXLI*nnnn*, or PXLO*nnnn* file if needed, and reissue the command.

306

```
ERROR 306 - DP2 Cache Flush Write Error
```

Cause. To improve performance, FSCK uses buffered access to the new catalog files, then uses the Guardian SETMODE 95 operation to flush any dirty cache blocks to disk. The SETMODE 95 operation returned an indication that one of the new files has its broken-file flag on because of a disk write error.

Effect. The FSCK utility terminates abnormally.

Recovery. Determine the cause of the disk write failure, correct it, then reissue the command.

307

```
ERROR 307 - Can't Create SQLCAT Process - PROCESS_CREATE_
Error error:detail
```

error:*detail*

is the error and error detail returned by the Guardian PROCESS_CREATE_ procedure. For further information, see the *Guardian Procedure Errors and Messages Manual*.

Cause. The FSCK utility detected a file that contains SQL/MP data but failed to start the SQLCAT process needed to delete the file. The failure might be caused by a temporary shortage of system resources.

Effect. The file cannot be deleted from the OSS fileset, and fileset repair or deletion cannot be completed.

Recovery. Use the Subsystem Control Facility (SCF) DIAGNOSE FILESET command to run FSCK in a different processor.

308

```
ERROR 308 - Unexpected SQLCAT Error error Purging File
filename
```

error

is the Guardian file-system error returned by the underlying Guardian procedure call. For further information, see the *Guardian Procedure Errors and Messages Manual*.

filename

indicates the affected file.

Cause. The SQLCAT process could not delete an OSS file containing an SQL/MP program because it cannot find or access the SQL catalog for the file. This message can occur when fileset deletion is attempted.

Effect. The file cannot be deleted from the OSS fileset, and fileset repair or deletion cannot be completed.

Recovery. If the value of *error* is 48, Safeguard is preventing update of the SQL catalog associated with the file. Either:

- Use SAFECOM to grant the super ID temporary update privileges for the SQL catalog, and then repeat the activity that produced this message.

- Perform the following steps:

  1. Use the Subsystem Control Facility (SCF) START FILESET command to remount the fileset containing the affected file.
  2. Use the File Utility Program (FUP) INFO, DETAIL command or the OSS `pname` utility to determine the pathname of the affected file.
  3. Have a user with update authority for the SQL catalog delete the file using the OSS *`rm`* utility.
  4. Use the SCF STOP FILESET command on the affected fileset.
  5. Repeat the activity that produced this message.

If the value of *`error`* is not 48, refer to the *Guardian Procedure Errors and Messages Manual* for possible causes and recovery actions. For example, relabeling the disk volume containing the SQL catalog can make the catalog inaccessible to the OSS name server for the fileset.

OSS Monitor Messages

The messages in this subsection are sent by the Open System Services (OSS) Monitor subsystem. These messages do not correspond to Event Management Service (EMS) events.

The OSS Monitor sends messages directly to the terminal from which it is started.

When the SCF product module for the OSS Monitor is running, the OSS Monitor sends numbered messages to the terminal using the SCF interface.

The OSS Monitor can also generate EMS events, which are issued under the OSS subsystem ID. For information about these messages, see the *Operator Messages Manual*.

Unnumbered Messages

Unnumbered messages can appear during startup of the OSS Monitor. Each message is prefixed by Subsystem Programmatic Interface (SPI) termination information in the following form:

```
ABENDED: processname
CPU time: interval
3: Premature process termination with fatal errors or
diagnostics
Termination info: code
Subsystem: TANDEM.143.release
text-message
```

processname

is the name of the failed process. Unless otherwise indicated in the specific message description, this name is always $ZPMON.

interval

is the elapsed time since the start of the failed process, in the form *dd mmm yyyy hh:mm:ss.mil*.

code

is the SPI error code of the failure. Values for this are indicated in the specific message descriptions.

release

is the major release version identifier for the failed process. For example, the version of the OSS Monitor in the D44.00 release version update (RVU) has the *release* value D40.

text-message

is the unnumbered message that describes the cause of the failure.

The possible text messages are described on the following pages in alphabetic order.

```
Invalid value specified for the AUTOSTART PARAM -- must be
AUTO or MANUAL
```

Cause. The OSS Monitor detected an invalid value for the AUTOSTART PARAM during startup.

Effect. The OSS Monitor terminates. The existing AUTOSTART attribute value for the subsystem is unchanged.

Recovery. Restart the OSS Monitor with a correct value, as indicated in the message.

```
Invalid value specified on the command line for the AUTOSTART
attribute -- must be AUTO or MANUAL
```

Cause. The OSS Monitor detected an invalid value for the AUTOSTART attribute on its command line during startup.

Effect. The OSS Monitor terminates. The existing AUTOSTART attribute value for the subsystem is unchanged.

Recovery. Restart the OSS Monitor with a correct value, as indicated in the message.

```
OSS Monitor failed in adding converted record to filename
file -- Error:  err
```

filename

is the Guardian filename of the affected OSS configuration file.

err

is the Guardian file-system error number that describes the failure.

Cause. During automatic conversion of the OSS configuration files from a previous release version update (RVU), the OSS Monitor found an entry in the ZPCONFIG file that could not be written to the indicated configuration file.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following steps:

1. See the *Guardian Procedure Errors and Messages Manual* for a description of the file-system error *err* and possible corrective actions. Correct the problem.
2. Remove the invalid database files by entering the following at a TACL prompt:

   ```
   VOLUME $SYSTEM.ZXOSSMON
   PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
   ```
3. Restart the OSS Monitor to restart the database creation process:

   ```
   OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
   ```

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor failed in adding default record to filename file
-- Error: err
```

filename

is the Guardian filename of the affected OSS configuration file.

err

is the Guardian file-system error number that describes the failure.

Cause. The OSS Monitor was unable to add a default record to the indicated OSS configuration file, during the initial startup of a database conversion from the D40 product. (For example, the entry for the root OSS name server could not be added to the ZOSSSERV file.)

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following steps:

1. See the *Guardian Procedure Errors and Messages Manual* for a description of the file-system error *err* and possible corrective actions. Correct the problem.
2. Remove the invalid database files by entering the following at a TACL prompt:

```
VOLUME $SYSTEM.ZXOSSMON
PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
RENAME ZXCONFIG ZPCONFIG
RENAME ZXMNTTAB ZPMNTTAB
```

3. Restart the OSS Monitor to restart the database creation process:

```
OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
```

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor failed in creating filename file -- Error: err
```

filename

is the Guardian filename of the affected OSS configuration file.

err

is the Guardian file-system error number that describes the failure.

Cause. The OSS Monitor could not create the indicated OSS configuration file. This message might indicate that the automatic conversion of the configuration files from a previous release version update (RVU) has failed.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following steps:

1. See the *Guardian Procedure Errors and Messages Manual* for a description of the file-system error *err* and possible corrective actions. Correct the problem.
2. Remove the invalid database files by entering the following at a TACL prompt:

   ```
   VOLUME $SYSTEM.ZXOSSMON
   PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
   ```

3. Restart the OSS Monitor to restart the database creation process:

   ```
   OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
   ```

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor failed in opening filename file -- Error: err
```

filename

is the Guardian filename of the affected OSS configuration file.

err

is the Guardian file-system error number that describes the failure.

Cause. The OSS Monitor could not open the indicated OSS configuration file. This message might indicate that the configuration file is corrupted.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following steps:

1. See the *Guardian Procedure Errors and Messages Manual* for a description of the file-system error and possible corrective actions. Correct the problem.
2. Remove the invalid database files by entering the following at a TACL prompt:

   ```
   VOLUME $SYSTEM.ZXOSSMON
   PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
   ```

3. Restart the OSS Monitor to restart the database creation process:

   ```
   OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
   ```

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor failed in reading filename file -- Error: err
```

filename

is the Guardian filename of the affected OSS configuration file.

err

is the Guardian file-system error number that describes the failure.

Cause. The OSS Monitor could not read the indicated OSS configuration file. This message might indicate that the configuration file is corrupted.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following steps:

1. See the *Guardian Procedure Errors and Messages Manual* for a description of the file-system error *err* and possible corrective actions. Correct the problem.
2. Remove the invalid database files by entering the following at a TACL prompt:

   ```
   VOLUME $SYSTEM.ZXOSSMON
   PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
   ```

3. Restart the OSS Monitor to restart the database creation process:

   ```
   OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
   ```

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor failed to get its process name
```

Cause. The OSS Monitor could not determine its own process name. An internal error might have occurred.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value -7. The OSS environment is not available.

Recovery. Restart the OSS Monitor with the process name $ZPMON. If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

```
OSS Monitor found invalid format in ZPCONFIG [ reason ]
```

reason

is a text message that further describes the problem.

Cause. During automatic conversion of the OSS configuration files from a previous release version update (RVU), the OSS Monitor found an entry in the ZPCONFIG file that could not be interpreted.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value 17. The OSS environment is not available.

Recovery. Perform the following actions:

1. Remove the invalid database files by entering the following at a TACL prompt:

   ```
   VOLUME $SYSTEM.ZXOSSMON
   PURGE ZOSSFSET ZOSSFS00 ZOSSFS01 ZOSSSERV ZOSSPARM
   ```

2. Check that the OSS Monitor is using the correct copy of the ZPCONFIG file for the conversion. If necessary, move the correct copy of the ZPCONFIG file to $SYSTEM.ZXOSSMON.
3. Check the format of the ZPCONFIG file entries against the sample file shown in Appendix D, Falling Back to a Previous Release Version Update. Correct any format errors.

4. Restart the OSS Monitor to restart the database creation process:

   ```
   OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
   ```

```
Process name of OSS Monitor must be $ZPMON
```

Cause. The OSS Monitor was started with the name shown as *processname* in the SPI termination information for this message.

Effect. The OSS Monitor process terminated abnormally with the SPI error *code* value -33. The OSS environment is not available.

Recovery. Restart the OSS Monitor with the process name $ZPMON:

```
OSSMON /NAME $ZPMON, TERM $ZHOME, NOWAIT/
```

```
This version of OSSMON only runs on Himalaya S Series servers
```

Cause. The version of the OSS Monitor is G09 or later and the system is an HP NonStop K-series system (release versions D4x).

Effect. The OSS Monitor process issues the error message on its home terminal and terminates.

Recovery. Restore the previous version of the OSS Monitor.

```
Unknown keyword specified on the command line
```

Cause. The OSS Monitor does not recognize a keyword specified as a parameter on the command line during startup. This most commonly indicates a typographical error.

Effect. The OSS Monitor terminates.

Recovery. Restart the OSS Monitor using a corrected keyword in its command line.

Numbered Messages

Numbered OSS Monitor SCF messages are described on the following pages in order by message number.

Note. Negative-numbered messages are common to most SCF subsystem modules. If you receive a negative-numbered message, see the description of “Common Error Messages” in the appendix “SCF Error Messages” in the *Subsystem Control Facility (SCF) Reference Manual*.

1

```
OSS E00001 Internal error
```

Cause. The OSS Monitor has detected a serious internal error during the execution of a Subsystem Control Facility (SCF) command.

Effect. The OSS Monitor stops processing the command.

Recovery. Check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

2

```
OSS E00002 Server did not respond server-name
```

server-name

is the server name of the OSS name server affected by the error.

Cause. During the execution of a Subsystem Control Facility (SCF) command, the OSS Monitor did not receive a response from an OSS name server process within the time specified by the IOTIMEOUT attribute of the OSS subsystem. This might indicate any of the following conditions:

- The IOTIMEOUT attribute value is too small.
- The OSS name server is overloaded with requests because of insufficient processor or memory resources or because of a software problem.

Effect. The OSS Monitor stops processing the command.

Recovery. Perform one or more of the following actions, depending on how often the message occurs:

- Reissue the SCF command.
- Use the SCF ALTER SUBSYS command to increase the value of the IOTIMEOUT attribute.

- Examine the load on the processor that is running the primary OSS name server process, then do one or more of the following:
  - Increase the execution priority of the OSS name server process.
  - Start the OSS name server on a less-busy processor.
  - Increase the cache size for the OSS name server and restart the server.

3

```
OSS E00003 Failed to start process -- error err, error-detail
```

process

is the process name of the OSS subsystem process that failed to start.

err

is the error number returned by the underlying call to the Guardian PROCESS_CREATE_ procedure.

error-detail

is the error detail information returned by the underlying call to the Guardian PROCESS_CREATE_ procedure.

Cause. During the execution of a Subsystem Control Facility (SCF) command, an OSS name server process failed. This error most commonly occurs when the HP NonStop operating system cannot start a server process.

Effect. If an OSS name server process fails, the OSS Monitor restarts that process automatically.

Recovery. Perform one or more of the following actions:

- Use the SCF INFO SERVER command to check for invalid information in the configuration of the failing server.
- Verify that the processors used by the server are running.
- Check the server program file security permissions.
- Look up the PROCESS_CREATE_ error information in the *Guardian Procedure Errors and Messages Manual*. Correct the problem and start the server.

5

```
OSS E00005 Invalid value for parameter paramname
```

paramname

identifies the command parameter with the invalid value.

Cause. A Subsystem Control Facility (SCF) ADD or ALTER command contained a parameter value that was invalid.

Effect. The command is not processed.

Recovery. Reissue the command with a valid value.

6

```
OSS E00006 No attributes have been specified for this
command.
```

Cause. A Subsystem Control Facility (SCF) command contained a parameter without a required value or was missing a required parameter.

Effect. The command is not processed.

Recovery. Reissue the command with an appropriate parameter or value.

8

```
OSS E00008 Exceeded maximum I/O error retry limit on name
```

name

identifies the affected entity.

Cause. During the execution of a Subsystem Control Facility (SCF) command, an OSS server had a recurring I/O error and the retry limit was reached. This error usually occurs when either a processor fails, an OSS name server process is not running, or a disk device is malfunctioning.

Effect. The OSS Monitor stops processing the command.

Recovery. Perform one or more of the following actions:

- Try to determine and correct the cause of the I/O retries. If the failing entity is an OSS name server, do one of the following:
  - If the OSS name server is still running, stop all filesets managed by that OSS name server. Remount those filesets.
  - If the OSS name server is not running, stop and restart the OSS Monitor so that it executes the automatic fileset restart sequence.
- If a server processor has failed, bring the processor up, which should cause the OSS Monitor to execute the automatic fileset restart sequence.
- Execute the SCF command again.

9

```
OSS E00009 Failed to start fileset filesetname
```

filesetname

identifies the affected fileset.

Cause. A Subsystem Control Facility (SCF) START FILESET command failed. Either another fileset is mounted on the mount point you want to use for the indicated fileset, or the fileset containing the mount point you want to use is not started.

Effect. The OSS name server rejects the mount-point mount operation, the OSS Monitor backs out of the START FILESET operation, and the fileset is not mounted.

Recovery. Check the fileset configurations with the SCF INFO FILESET * command to look for duplicated mount points. Reissue the START FILESET command. For information about the INFO FILESET command, see INFO FILESET Command on page 12-47.

10

```
OSS E00010 Failed to stop fileset filesetname
```

filesetname

identifies the affected fileset.

Cause. The most common cause of this message is that another fileset is mounted on the indicated fileset.

Effect. The indicated fileset is still mounted. There are two possible effects on the state of that fileset:

- If there is another fileset mounted on the fileset that you want to unmount, the state of the indicated fileset remains STARTED.
- If the OSS name server decides that the fileset is not mounted, an inconsistency exists between the OSS Monitor and the OSS name server. In this case, the state of the indicated fileset is set to UNKNOWN.

Recovery. Enter a Subsystem Control Facility (SCF) STATUS FILESET command to determine whether the indicated fileset is in the STARTED or UNKNOWN state.

- If the fileset is in the STARTED state, make sure that there are no filesets mounted on the fileset that you want to unmount.
- If the fileset is in the UNKNOWN state, follow these steps:

  1. Enter an SCF START FILESET command for the indicated fileset.
  2. Enter another SCF STOP FILESET command for the indicated fileset.

3. Enter an SCF DIAGNOSE FILESET command for the indicated fileset.
4. If the problem persists, contact your service provider and provide all relevant information as follows:
   - Description of the problem and accompanying symptoms
   - Details from the message or messages generated
   - Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

   If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

14

```
OSS E00014 Invalid disk volume volname
```

volname

is the name of the affected disk volume.

Cause. A Subsystem Control Facility (SCF) command was issued that affected the named disk volume. Either the disk volume does not exist or one or more disk volumes needed to execute the command do not exist or are unavailable.

Effect. The command is not processed.

Recovery. Perform one or more of the following actions:

- Check the spelling of any disk volume name specified in the command. Make sure that the named disk volume is available for use.
- Check the spelling of the catalog disk volume for any fileset specified in the command. Make sure that the named disk volume is available for use.
- Examine the storage-pool file for any fileset specified in the command. Check the spelling of all disk volume names and make sure that all the named disk volumes are available for use.
- Reissue the command.

15

```
OSS E00015 Invalid mount point in fileset filesetname
```

filesetname

identifies the affected fileset.

Cause. A Subsystem Control Facility (SCF) START FILESET command was issued for a fileset whose mount point does not represent a valid OSS pathname. (The pathname does not exist or is not a directory.)

Effect. The fileset is not started.

Recovery. Perform the following steps:

1. Use the SCF INFO FILESET command to check the spelling of the mount point pathname.
2. Use the OSS shell `ls` command to verify that an OSS directory exists to serve the mount point.
3. Use the SCF ALTER FILESET command to change the configuration of the fileset so that it uses a valid OSS directory as the mount point, then reissue the START FILESET command.

For information about the INFO FILESET, ALTER FILESET, and START FILESET commands, see Section 5, Managing Filesets.

16

```
OSS E00016 Unable to access catalog volume volname
```

volname

is the name of the affected disk volume.

Cause. A Subsystem Control Facility (SCF) START FILESET command failed because the catalog volume is inaccessible.

Effect. The fileset is not started.

Recovery. Perform one or more of the following actions:

- Make sure that the catalog disk volume is not down. If the volume is down, bring it up.
- Examine the configuration file with the SCF INFO FILESET command. Use the SCF ALTER FILESET command to correct the disk volume name if necessary.
- Reissue the START FILESET command.

For information about the INFO FILESET, ALTER FILESET, and START FILESET commands, see Section 5, Managing Filesets.

17

```
OSS E00017 Unable to access configuration file filename --
error err
```

filename

is the Guardian filename of the OSS configuration file that could not be opened, read, or written.

err

is the Guardian file-system error number returned by the input or output operation.

Cause. The OSS Monitor could not open, read from, or write to a configuration file.

Effect. No processing can be performed on the objects in the affected configuration file. Depending on the configuration file affected, all filesets or OSS servers could be inaccessible.

Recovery. This message can occur when you use the Subsystem Control Facility (SCF) STATUS FILESET command on the root fileset and another user has the ZOSSFSET file open at the console. This problem goes away after the ZOSSFSET file is closed.

If that is not the cause of the message:

1. Check that the security on the indicated configuration file is correct.
2. Check the Event Management Service (EMS) log for any events related to the problem.
3. Refer to the *Guardian Procedure Errors and Messages Manual* to determine how to correct the reported Guardian file-system error.
4. Correct the situation and reissue the command.

If the message recurs:

1. Check that the $SYSTEM.ZXOSSMON.ZPOSSMON file is present.
2. Check that the security on that file is correct for its execution and for access by the OSS Monitor to the configuration files.
3. Restart the OSS Monitor.

18

```
OSS E00018 Configuration contains invalid data
```

Cause. A fileset cannot be started because the configuration record for it has become corrupted.

Effect. The command is not processed.

Recovery. A serious problem exists. Perform the following actions:

1. If possible, check and correct the configuration of the fileset using the Subsystem Control Facility (SCF) INFO FILESET and ALTER FILESET commands. Reassigning an existing attribute to the fileset might fix the configuration record problem.
2. Restart the fileset.

If the problem persists, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

The fileset probably cannot be used again. If you have a recent backup, use the SCF DELETE FILESET command to delete the fileset entry from the configuration file, then recreate the configuration record with the SCF ADD FILESET command and restore the fileset files from the backup copy.

19

```
OSS E00019 There is no disk volume in pool filename
```

filename

is the Guardian filename of the affected storage-pool file.

Cause. A Subsystem Control Facility (SCF) START FILESET command was issued but the fileset has no disk volumes. The indicated storage-pool file contains no entries.

Effect. The command is not processed.

Recovery. Make sure that the storage-pool file contains at least one disk volume name. Reissue the command.

22

```
OSS E00022 Invalid value for attribute attribute-name
```

attribute-name

is the name of the affected attribute.

Cause. A Subsystem Control Facility (SCF) ADD or ALTER command failed because a specified attribute has a value that is invalid.

Effect. The command is not processed.

Recovery. Reissue the command using a valid value for the attribute.

23

```
OSS E00023 Name Server server rejected the request
```

server

is the server name of the affected OSS name server.

Cause. A Subsystem Control Facility (SCF) START FILESET command failed.

Effect. The command cannot be processed.

Recovery. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- A copy of the OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

24

```
OSS E00024 Name Server server gave unexpected response to OSS
Monitor: status.
```

server

is the server name of the affected OSS name server.

status

is the status value returned to the OSS Monitor.

Cause. An OSS name server returned status information that is not recognized by the OSS Monitor. An OSS name server process might be malfunctioning.

Effect. The command cannot be processed.

Recovery. Contact your service provider and provide all relevant information as follows:

- Saveabend file
- A copy of the OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

25

```
OSS E00025 Name Server server detected invalid data.
```

server

is the server name of the affected OSS name server.

Cause. An OSS name server received invalid data from the OSS Monitor. The OSS Monitor might be malfunctioning.

Effect. The command cannot be processed.

Recovery. Reissue the command. If the problem persists, contact your service provider and provide all relevant information as follows:

- Saveabend file

- A copy of the OSS fileset catalog files
- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as Event Management Service (EMS) logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

26

```
OSS E00026 Repair is needed for corrupted fileset filesetname
```

filesetname

identifies the affected fileset.

Cause. The indicated fileset is corrupted. This can happen when the fileset was started with the BUFFERED CREATE or BUFFERED LOG attribute set and its OSS name server failed.

Effect. The indicated fileset cannot be started.

Recovery. Recover the fileset with the Subsystem Control Facility (SCF) DIAGNOSE FILESET command using the REPAIR option. Restart the fileset with the SCF START FILESET command.

27

```
OSS E00027 Root fileset is not started
```

Cause. A Subsystem Control Facility (SCF) START FILESET command was issued, but the root fileset is not started yet. No other filesets can be started until the root fileset is started.

Effect. The command is not processed.

Recovery. Use the SCF START FILESET command to start the root fileset, then reissue the original START FILESET command.

28

```
OSS E00028 Failed in moving catalog files
```

Cause. A Subsystem Control Facility (SCF) ALTER FILESET command was issued to change the catalog volume of a fileset, but the catalog files could not be moved to the specified volume.

During successful processing of this command, the CVT utility renames the current catalog files to a temporary subvolume with a name that begins with ZZ on the original catalog disk volume; the renamed files have a Guardian file identifier that ends with 9999. The File Utility Program (FUP) then duplicates the temporary subvolume and files on the new catalog disk volume. Finally, the CVT utility restores the temporary subvolume and files to their original names on the new disk volume.

This message indicates that a step in that process has failed.

This error commonly occurs when the ZOSSVOL attribute of the subsystem is invalid or Guardian file permissions prevent execution of the CVT or FUP utilities.

This error can also occur when:

- The specified disk volume is managed by the NonStop Storage Management Foundation (SMF). Open System Services and SMF cannot manage the same disk volume.
- The specified disk volume is an optical disk. OSS fileset catalogs cannot reside on optical disks.

Effect. The command is not processed. The catalog files remain on the original catalog disk volume, and the fileset configuration record is unchanged.

Recovery. Perform the following actions:

1. Ensure that the temporary subvolumes are empty on the original catalog disk volume and on the target catalog disk volume. If the temporary subvolumes are not empty, processing of the command could not successfully be reversed and you must:
    a. Use the CVT utility to restore the temporary files to their original names on either the source or target catalog disk volume. If the temporary files exist on both disk volumes, restore the files on the original catalog volume; when the ALTER FILESET command cannot find catalog files on a current volume but can find them on a target disk volume, it assumes the latter are valid files.
    b. Delete any remaining temporary copies of the catalog files.
2. Check the ZOSSVOL value being used:
    a. Locate the disk volume containing the ZOSS subvolume with the CVT utility. The default disk volume for this subvolume is $SYSTEM.

b. Use the SCF INFO SUBSYS command to check the disk volume being used for the ZOSSVOL attribute.

c. If the ZOSSVOL attribute does not specify the correct disk volume, use the SCF ALTER SUBSYS command to correct the problem.

3. Check the security permissions of the CVT and FUP utilities to ensure that the OSS Monitor can execute them. Correct them if necessary.

Reissue the ALTER FILESET command to change the CATALOG value. If the problem persists:

1. Back up the fileset as described in Backing Up and Restoring OSS Files on page 6-11.
2. Use the SCF DELETE FILESET command to delete the fileset.
3. Recreate the fileset configuration using the SCF ADD FILESET command.
4. Restore the fileset from its backup as described in Backing Up and Restoring OSS Files on page 6-11.
5. Reissue the ALTER FILESET command.

29

```
OSS E00029 Need to start mount point fileset first
```

Cause. A Subsystem Control Facility (SCF) START FILESET command was issued, but the fileset containing the mount point for the specified fileset is not yet started.

Effect. The command is not processed.

Recovery. Start the fileset that contains the mount point. Reissue the original START FILESET command.

30

```
OSS E00030 Failed to stop server servername
```

servername

is the server name of the affected OSS server.

Cause. The indicated OSS server could not be stopped. Either OSS files that are read or written through the server remain open or a software problem exists.

Effect. The server process continues to run and might become an unstoppable process.

Recovery.

If the affected server is...	Take this action...
OSS message-queue server	Use the Subsystem Control Facility (SCF) STOP SERVER command to stop the server.
OSS name server	Use the SCF STOP FILESET command for all filesets managed by the server.
OSS sockets local server	1. Stop all applications with open `AF_UNIX` sockets. 2. Use the SCF STOP SERVER command to stop the server.
OSS transport agent server	1. Stop all applications with open sockets. 2. Use the SCF STOP SERVER command to stop the server.

If the problem persists and appears to be a software error, check the Event Management Service (EMS) log for related messages. Contact your service provider and provide all relevant information as follows:

- Description of the problem and accompanying symptoms
- Details from the message or messages generated
- Supporting documentation such as EMS logs, trace files, and a processor dump, if applicable

If your local operating procedures require contacting the Global Customer Support Center (GCSC), supply your system number and the numbers and versions of all related products as well.

31

```
OSS E00031 Invalid value for command option optionname
```

optionname

is the name of the command option for which the invalid value was specified.

Cause. A Subsystem Control Facility (SCF) DIAGNOSE FILESET command was issued, but the indicated command option has an invalid value.

Effect. The command is not processed.

Recovery. Reissue the command with a valid option value. See DIAGNOSE FILESET Command on page 12-43.

32

```
OSS E00032 Invalid combination of command options
```

Cause. A Subsystem Control Facility (SCF) DIAGNOSE FILESET command was issued, but the command contained an invalid combination of options. This condition usually occurs when more than one of the options STATUS, REPAIR, UPGRADE, and DOWNGRADE are specified.

Effect. The command is not processed.

Recovery. Reissue the command with a single option or a valid combination of options. See DIAGNOSE FILESET Command on page 12-43.

33

```
OSS E00033 Too many disk volumes in pool filename
```

filename

is the Guardian filename of the affected OSS storage-pool file.

Cause. A Subsystem Control Facility (SCF) START FILESET command was issued, but the storage-pool file for the fileset contains too many disk volume names. The current maximum number of disk volumes allowed in a fileset for the creation of new OSS files is 20.

Effect. The command is not processed.

Recovery. Remove as many disk volume names as necessary from the storage-pool file and reissue the START FILESET command. Remember that disk volumes remain in the storage pool even when no longer in the storage-pool file; disk volumes that are close to capacity are the best choices to remove.

34

```
OSS E00034 Invalid parameter parameter-name
```

parameter-name

is the parameter that is not supported by the command.

Cause. A Subsystem Control Facility (SCF) ADD SERVER or ALTER SERVER command was issued, but the indicated parameter is not supported by the type of server being added or reconfigured.

Effect. The command is not processed.

Recovery. Reissue the ADD SERVER or ALTER SERVER command without the parameter.

35

```
OSS E00035 Command reserved for SUPER.SUPER only
```

Cause. A Subsystem Control Facility (SCF) command that requires the super ID was issued but you are not logged on with the super ID.

Effect. The command is not processed.

Recovery. Exit SCF, change your user ID to the super ID, and re-enter SCF. Reissue the command.

If you are not permitted to use the super ID, contact your system administrator to have the command issued for you.

36

```
OSS E00036 Alter mount point of Root fileset is not allowed
```

Cause. A Subsystem Control Facility (SCF) ALTER FILESET command was issued for the root fileset, and the command contained a MNTPOINT value other than `/`. The mount point for the root fileset must always be `/`.

Effect. The command is not processed.

Recovery. Reissue the ALTER FILESET command without the MNTPOINT option.

37

```
OSS E00037 Alter Name Server of Root fileset is not allowed
```

Cause. A Subsystem Control Facility (SCF) ALTER FILESET command was issued for the root fileset, and the command contained a NAMESERVER value other than #ZPNS. The OSS name server for the root fileset must always be #ZPNS.

Effect. The command is not processed.

Recovery. Reissue the ALTER FILESET command without the NAMESERVER option.

38

```
OSS E00038 Duplicate attribute attribute
```

attribute

identifies the duplicate attribute.

Cause. An attribute is specified more than once in a Subsystem Control Facility (SCF) command.

Effect. The command is not processed.

Recovery. Reissue the command with a single specification for the attribute.

39

```
OSS E00039 Missing required attribute attribute
```

attribute

identifies the missing attribute.

Cause. A Subsystem Control Facility (SCF) command was entered without a required attribute.

Effect. The command is not processed.

Recovery. Reissue the command with the required attribute specification.

44

```
OSS E00044 Reserved name for OSS transport agent server
```

Cause. An ADD SERVER command was attempted to create a server configuration record with the OSS transport agent server reserved name.

Effect. The OSS Monitor stops processing the command.

Recovery. Reissue the command with a different server name.

45

```
OSS E00045 Device label number is being used by fileset
filesetname
```

filesetname

identifies the fileset using the affected device label number.

Cause. A Subsystem Control Facility (SCF) ADD FILESET command attempted to assign a device label number to a new fileset, but the specified device label number is already in use.

Effect. The command is not processed.

Recovery. Reissue the command with an unused value for the device label number.

46

```
OSS E00046 Alter Name Server of active fileset is not allowed
```

Cause. A Subsystem Control Facility (SCF) ALTER FILESET command attempted to change the OSS name server process for a fileset while the fileset is mounted.

Effect. The command is not processed.

Recovery. Stop the fileset and any filesets mounted on it, then reissue the command and restart all affected filesets.

47

```
OSS E00047 Mount point pathname must be an absolute pathname
```

Cause. The mount point pathname specified in a Subsystem Control Facility (SCF) ADD FILESET or ALTER FILESET command was invalid or is a relative pathname.

Effect. The command is not processed.

Recovery. Determine the correct absolute pathname to use for the mount point, then reissue the command.

48

```
OSS E00048 Mount Point Pathname may not start with /E or /G
```

Cause. The mount point pathname specified in a Subsystem Control Facility (SCF) ADD FILESET or ALTER FILESET command begins with `/E` or `/G`.

Effect. The command is not processed.

Recovery. Select a mount point in the local node’s OSS file system and reissue the command.

49

```
OSS E00049 OSSMON/Database Record version incompatibility
```

Cause. A record in the OSS Monitor database has been updated by a newer version of the OSS monitor than is currently running. The record was updated in such a way that the OSS monitor currently running cannot reliably interpret the record's contents.

Effect. The record is not accessible to the current OSS monitor.

Recovery. Upgrade to a later version of the OSS monitor.

52

```
OSS E00052 Fileset fileset is not mounted.
```

fileset

identifies the affected fileset.

Cause. A Subsystem Control Facility (SCF) CONTROL FILESET command was entered, but the fileset targeted by it is not mounted.

Effect. The command is not processed.

Recovery. Use a START FILESET command to start the fileset and reissue the CONTROL FILESET command.

53

```
OSS E00053 Name Server server is not running.
```

server

identifies the affected server process.

Cause. A Subsystem Control Facility (SCF) CONTROL SERVER command was entered, but the OSS name server targeted by it is not running.

Effect. The command is not processed.

Recovery. Use a START FILESET command to start the server and reissue the CONTROL SERVER command.

54

```
OSS E00054 Fileset fileset is corrupt and needs to be
repaired.
```

fileset

identifies the affected fileset.

Cause. A Subsystem Control Facility (SCF) CONTROL FILESET command was entered, but the fileset targeted by it is corrupted.

Effect. The command is not processed.

Recovery. Use the SCF STOP FILESET command to stop the fileset. Recover the fileset using the SCF DIAGNOSE FILESET command with the REPAIR option. Restart the fileset with the START FILESET command, then reissue the CONTROL FILESET command if necessary.

55

```
OSS E00055 The MAXINODES value is lower than the number of
currently inuse inodes inuseinodes for the fileset fileset.
```

inuseinodes

is the current number of inodes in use.

fileset

identifies the affected fileset.

Cause. A Subsystem Control Facility (SCF) CONTROL FILESET command was entered, but the specified MAXINODES value is lower than the current number of inodes in use.

Effect. The command is not processed.

Recovery. Use the ALTER FILESET command to change the MAXINODES value to a value greater than the current number of inodes in use and reissue the CONTROL FILESET command with the SYNC option.

56

```
OSS E00056 The primary Name Server server failed to migrate
to the processor specified by the CPU attribute.
```

server

identifies the affected server process.

Cause. A Subsystem Control Facility (SCF) CONTROL SERVER command was entered, but the processor specified by its CPU attribute is down.

Effect. The command is not processed.

Recovery. Use the ALTER SERVER command to change the CPU attribute, and reissue the CONTROL SERVER command with the SYNC option.

57

```
OSS E00057 The backup Name Server server failed to migrate to
the processor specified by the BACKUPCPU attribute.
```

server

identifies the affected server process.

Cause. A Subsystem Control Facility (SCF) CONTROL SERVER command was entered, but the processor specified by its BACKUPCPU attribute is down.

Effect. The command is not processed.

Recovery. Use the ALTER SERVER command to change the BACKUPCPU attribute, and reissue the CONTROL SERVER command with the SYNC option.

58

```
OSS E00058 The migration of a primary or a backup Name Server
server to a different processor failed.
```

server

identifies the affected server process.

Cause. A Subsystem Control Facility (SCF) CONTROL SERVER command was entered, but the backup OSS name server process is not in a valid state.

Effect. The command is not processed.

Recovery. Retry the command at a later time. If that does not work, stop (unmount) all the filesets mounted by that OSS name server and restart them.

59

```
OSS E00059 Unable to make all the volumes in the POOL edit
file eligible for file creation
```

Cause. A Subsystem Control Facility (SCF) CONTROL FILESET command was entered, but one or more of the disk volumes configured to be in the storage creation pool for the fileset cannot be used to store newly created OSS files. The storage pool for the fileset previously approached the limit of 256 disk volumes and the storage creation pool configuration now contains one or more new disk volume names.

Effect. The command is not processed.

Recovery. Use the SCF STATUS FILESET command with the DETAIL option to check the number of volumes in the storage pool. If the storage pool has reached its disk volume limit but the fileset requires space for new files, possible actions include:

- Delete unused files from the fileset to free space on disk volumes already in the storage creation pool, remove the new disk volumes from the storage pool file, then reenter the CONTROL FILESET command.
- Move files from this fileset to another fileset with more growth capacity to free space on disk volumes already in the storage creation pool, remove the new disk volumes from the storage pool file, then reenter the CONTROL FILESET command.

60

```
OSS W00060 The fileset is started, but MAXINODES value is
changed to maxinodesvalue
```

maxinodesvalue

indicates the MAXINODES value being used. This value is also stored in the configuration database for the fileset.

Cause. The specified MAXINODES value is less than 110 percent of the number of in-use inodes. This message can occur when an OSS name server tries to mount an existing fileset and the previous maximum value did not allow reasonable growth.

Effect. This is an informational message only.

Recovery. No immediate action is required. The disk capacity of the fileset should be checked to determine if its MAXINODES value should be increased further.

61

```
OSS W00061 The fileset is started, but not all the volumes in
the pool edit file are eligible for file creation
```

Cause. A Subsystem Control Facility (SCF) START FILESET command was entered, but one or more of the disk volumes configured for the storage creation pool for the fileset cannot be used to store newly created OSS files. The storage pool for the fileset has reached the limit of 256 disk volumes and the storage creation pool file now contains one or more new disk volume names.

Effect. This is an informational message only.

Recovery. No immediate action is required. However, the disk capacity of the fileset should be checked in case new OSS files will need to be created.

Use the SCF STATUS FILESET command with the DETAIL option to check the disk volumes in the storage creation pool. If the storage creation pool contains no disk volumes eligible for new file creation, possible actions include:

- Delete unused files from the fileset to free space on disk volumes already in the storage pool (when all OSS files have been removed from a disk volume, it is no longer part of the storage pool unless it is also in the storage creation pool). Then use an SCF CONTROL FILESET command with the SYNC option to update the storage creation pool from the pool edit file.
- Move files from this fileset to another fileset with more growth capacity to free space on disk volumes already in the storage pool (when all OSS files have been removed from a disk volume, it is no longer part of the storage pool unless it is also in the storage creation pool). Then use an SCF CONTROL FILESET command with the SYNC option to update the storage creation pool from the pool edit file.

OSSTTY Subsystem Messages

The messages in this section are generated by the OSSTTY subsystem. OSSTTY generates the unnumbered messages described in Startup Messages on page A-58 when it is first started. It can log the EMS messages, described in the *Operator Messages Manual,* while running; the subsystem ID displayed by these messages is OSSTTY.

Startup Messages

```
Usage : run osstty / NAME <process-name>,
  IN <stdin-redirection-target>,
  OUT <stdout-redirection-target>,
  TERM <stderr-redirection-target>, NOWAIT / [options]

<stdin-redirection-target>
  {$process-name}

<stdout-redirection-target>
  {$process-name | EDIT-filename}

<stderr-redirection-target>
  {$process-name}

[options]
  [-backupcpu <cpu-num>]
  [-access < ALL | OWNER >]
  [-secure <security-string>]
  [-[no]server]
  [-[no]wrap]
  [-[no]prefixpid]
  [-coll <collector>]
  [-idletimeout <seconds>]
  [-[no]quiet]
  [-help]

<security-string>
  <sec-char><sec-char><sec-char><sec-char>

<sec-char>
  { A | G | O } -- local security
  { N | C | U } -- remote security
  { - } -- "super" security
```

Cause. The user entered the OSSTTY command with the `-help` flag specified.

Effect. OSSTTY is not running.

Recovery. This is an informative message only; no action is needed.

```
OSSTTY: Both the attributes attr1 and attr2 are mutually
exclusive
```

attr1

indicates the first of two conflicting command flags or values.

attr2

indicates the second of two conflicting command flags or values.

Cause. The identified flags or values cannot be used in the same command.

Effect. OSSTTY is not running.

Recovery. Reenter the command without specifying one of the conflicting flags.

```
OSSTTY: Invalid attribute value for stdxx_redirection_target
```

value

indicates the value that caused the diagnosed condition.

stdxx_redirection_target

indicates the redirection target specification (IN, OUT, or TERM) that contains an invalid object type.

Cause. The Guardian file-system object specified in the command is not supported for use as the indicated redirection target file. For example, this message occurs when an EDIT file is specified for the OSS standard input file (IN).

Effect. OSSTTY is not running.

Recovery. Reenter the command with a different Guardian object specified. (Process names are valid for all three redirection targets, EDIT files are valid only for the OUT target.)

```
OSSTTY: Unable to OPEN file/process name. error error_number
```

name

identifies the process or file.

error_number

indicates the Guardian file-system error returned by the underlying call to the Guardian FILE_OPEN_ procedure. For more information about this error, refer to the *Guardian Procedure Errors and Messages Manual*.

Cause. Access to one of the redirection targets failed.

Effect. OSSTTY is not running.

Recovery. Check that:

- A specified process is running
- An existing EDIT output file is not corrupted, has the correct access permissions for use, and is not already opened by another process

Reenter the command after correcting the problem.

```
OSSTTY: Value out of range for attribute attr
```

attr

identifies the command flag that could not be processed.

Cause. The value specified is not one of the allowed values or is out of range.

Effect. OSSTTY is not running.

Recovery. See Starting OSSTTY on page C-1 for a description of the valid values and flags for the OSSTTY command. Reenter the command with a corrected flag or value.

```
OSSTTY: Error in creating/securing file file_name,
error error_number
```

file_name

indicates the Guardian filename assigned in the command to the OUT target file that could not be created.

error_number

indicates the Guardian file-system error returned by the underlying call to the Guardian FILE_CREATE_ procedure. For more information about this error, refer to the *Guardian Procedure Errors and Messages Manual*.

Cause. OSSTTY could not create the specified EDIT file to receive redirected OSS standard output file data.

Effect. OSSTTY stops.

Recovery. Check that:

- File creation is allowed in the subvolume to be used for the file
- The file name was specified using a valid Guardian filename format

Reenter the command after correcting the problem.

```
OSSTTY: ****Warning**** Unable to create the backup process,
error error_number. Continuing...
```

error_number

indicates the Guardian file-system error returned by the underlying call to the Guardian PROCESS_LAUNCH_ procedure. For more information about this error, refer to the *Guardian Procedure Errors and Messages Manual*.

Cause. The primary copy of the OSSTTY process could not start its backup copy.

Effect. OSSTTY is running without a backup copy and cannot provide the fault tolerance of a process pair.

Recovery. This is an informative message only; no action is needed unless the backup process is required by your site procedures. The OSSTTY process can be stopped and restarted after the problem is corrected.

```
OSSTTY: ****Warning**** All the three re-directional targets
are TELSERV terminals, which are accessible directly.
Continuing...
```

Cause. The OSSTTY command was entered without any of the redirection target specifications (IN, OUT, or TERM).

Effect. OSSTTY is running but has no apparent function to perform other than providing duplication of the terminal interface of the Telserv process. OSS application response time will be slower than for direct access of terminals through the Telserv process.

Recovery. This is an informative message only; no action is needed.

```
OSSTTY: Invalid parameter: parameter
```

parameter

indicates the command flag that was not recognized.

Cause. The OSSTTY command was entered with a flag that is not recognized. The most common cause of this message is a typographical error in the command line.

Effect. OSSTTY is not running.

Recovery. See Starting OSSTTY on page C-1 for a description of the valid values and flags for the OSSTTY command. Reenter the command with a corrected flag or value.

```
OSSTTY: ****Warning**** EMS Collector coll_name is not
accessible, error error_number, Using $0 as the EMS
collector.
```

coll_name

indicates the process name specified in the command.

error_number

indicates the Guardian file-system error returned by the underlying call to the Guardian FILE_OPEN_ procedure. For more information about this error, refer to the *Guardian Procedure Errors and Messages Manual*.

Cause. The specified collector is not accessible. The process might not be running.

Effect. OSSTTY is running. Event Management Service (EMS) messages are logged to the default collector process, $0.

Recovery. This is an informative message only; no action is needed unless a specific nondefault collector process is required by your site procedures. The OSSTTY process can be stopped and restarted after the problem is corrected.

```
OSSTTY: ****Warning**** The EDIT file specified in the OUT
parameter already exists or is a process, ignoring the
–secure option. Continuing...
```

Cause. The OSSTTY command was entered with the `-secure` flag specified. The `-secure` flag is only used when an EDIT file must be created as the specified output redirection target; that condition does not exist.

Effect. OSSTTY is running. If the OUT parameter specified an existing EDIT file, its security permissions are unchanged.

Recovery. This is an informative message only; no action is needed.

```
OSSTTY: ****Warning**** The –wrap option has no effect when
OUT is a process. Continuing...
```

Cause. The OSSTTY command was entered with the `-wrap` flag specified. The `-wrap` flag is only used when an EDIT file is the specified output redirection target; that condition does not exist.

Effect. OSSTTY is running.

Recovery. This is an informative message only; no action is needed.

B

Manually Setting Up an OSS Environment

Table B-1 summarizes the procedures for configuring and starting a new OSS environment by using manually entered operator commands instead of the OSSSETUP utility. Table B-2 on page B-9 summarizes the procedures for completing a new OSS environment preconfigured by HP or created by the OSSSETUP DEFAULTS command. Be careful when using DSM/SCM in a network of systems running both G-series RVUs and H-series RVUs; the default for **Manage OSS Files** is unchecked for G-series RVUs and checked for H-series RVUs.

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 1 of 8)

Task	Subtask	Example	See
Ensure that your target system is ready for software updates	Perform all site preparation procedures, such as backups		*Guardian User's Guide*
Update DSM/SCM (product T6031) to the D46 product version update (PVU) if necessary	Perform all DSM/SCM steps		*DSM/SCM User's Guide*
Install current Guardian product files	Perform all DSM/SCM steps without the **Manage OSS Files** check box selected in the planner interface		*DSM/SCM User's Guide*
	Perform a cold start if necessary		
Prepare for configuration	Log on as the super ID	LOGON SUPER.SUPER	*Guardian User's Guide*

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 2 of 8)

Task	Subtask	Example	See
Prepare for configuration (*continued*)	Verify system resources: ● 64 MB memory/ processor ● 500 MB disk space for root fileset ● Running software release version update (RVU) documented in this guide ● Have Safeguard product or third-party software supporting Security Event Exit Process ● Have TCP/IP configured and Telserv running ● Have a VT100 or Xterm emulator or terminal		*Guardian User's Guide*
		FILEINFO $SYSTEM.SYS*.RLSEID to determine complete *filename* of correct version of RLSEID file, then TYPE *filename*	
			Safeguard Administrator's Manual
			TCP/IPv6 Configuration and Management Manual or *TCP/IP Configuration and Management Manual*, and *Telserv Manual*
Start related processes	Start the security manager process	OSMP / NAME $ZSMP, PRI 190, NOWAIT, CPU 0 / 1	Starting the OSS Monitor on page 2-7

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 3 of 8)

Task	Subtask	Example	See
Provide required configuration files	Create or modify a storage-pool file for each fileset	FUP DUP ZXOSSMON.OSSPOOL, ZXOSSMON.ROOTPOOL EDIT ROOTPOOL	Creating a Storage Pool on page 5-6
	Ensure that preconfigured databases are accessible after a node number change	FUP INFO $SYS*.*.ZOSSFSET, DETAIL FUP ALTER ZOSSFSET, ALTFILE (0,ZOSSFS00) FUP ALTER ZOSSFSET, ALTFILE (1,ZOSSFS01)	HP strongly discourages changes in node (system) numbers. However, in a few situations, such as during the first startup of a newly delivered system, you must change node numbers. on page 4-12
Configure the OSS file system	Optionally configure the OSS Monitor for super-group access	VOLUME $SYSTEM.SYS00 FUP LICENSE OSSMON FUP SECURE OSSMON, "N-G-", PROGID	Licensing the OSS Monitor to the Super Group on page 8-19
	Start the OSS Monitor	OSSMON / NAME $ZPMON, NOWAIT, CPU 0, TERM $ZHOME /	Starting the OSS Monitor as a Normal Process on page 2-8
	Configure the root fileset	SCF ALTER FILESET $ZPMON.ROOT, CATALOG $OSS, POOL ROOTPOOL ALTER SERVER $ZPMON.#ZPNS, BACKUPCPU 1 START FILESET $ZPMON.ROOT EXIT	Creating a Unique Fileset on page 5-1 and Configuring an OSS Name Server on page 4-29

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 4 of 8)

Task	Subtask	Example	See
Install OSS product files	Perform all DSM/SCM steps with the **Manage OSS Files** check box selected in the planner interface		*DSM/SCM User's Guide*

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 5 of 8)

Task	Subtask	Example	See
Finish configuring the OSS environment	Start an OSS shell	OSH	`osh(1)` reference page in the *Open System Services Shell and Utilities Reference Manual*

	Create the general profile file	`/bin/cd /etc`
		`/bin/cp profile.sample profile`
		`/bin/vi profile`
		`umask 022 #Only users have write permission on their files.`
		`set -o noclobber #Redirection can't overwrite files.`
		`set -o trackall #Track all aliases.`
		`export MANPATH=/usr/share/man #Match PATH use.`
			Setting Up an /etc/profile File on page 9-2

	Configure and start network services	`/bin/ln -s /G/system/ztcpip/resconf resolv.conf`
		`/bin/ln -s /G/system/ztcpip/networks networks`
		`/bin/ln -s /G/system/ztcpip/protocol protocols`
		`/bin/ln -s /G/system/ztcpip/services services`
		`/bin/ln -s /G/system/ztcpip/hosts hosts`
		`/bin/ln -s /G/system/ztcpip/ipnodes ipnodes`
			Configuring Network Services Servers, Tools, and Applications on page 4-31

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 6 of 8)

Task	Subtask	Example	See

Finish configuring the OSS environment (*continued*)	Configure and start network services (*continued*)	`/bin/cp smplinetd.conf inetd.conf`
		`/bin/vi inetd.conf`
		`shell stream tcp nowait root /bin/rshd`
		`/bin/vi hosts.equiv`
		`/bin/chmod 1775 *`
		`/usr/ucb/inetd -R 10&`
			Starting a Network Services Server on page 4-38

	Configure utilities for periodic tasks	`/bin/cd /usr/bin/cron`
		`/bin/cp at.deny.sample at.deny`
		`/bin/chmod 1775 *`
		`/bin/cd /var/adm/cron`
		`/bin/cp queuedefs.sample queuedefs`
		`/bin/cp .proto.sample .proto`
		`/bin/cp cron.deny.sample cron.deny`
			Configuring the cron Process on page 2-35

	Modify the files as necessary for your site	`/bin/vi *` `/bin/chmod 1775 *`	
	Start the cron process	`/bin/cron &`	`cron(8)` reference page, either online or in the *Open System Services Shell and Utilities Reference Manual*

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 7 of 8)

Task	Subtask	Example	See

Finish configuring the OSS file system	Secure fileset mount points	`/bin/cd /`
		`/bin/chmod 0777 /`
		`/bin/chmod 0775 /bin`
		`/bin/chmod 1775 /etc`
		`/bin/chmod 1775 /var`
		`/bin/mkdir tmp`
		`/bin/chmod 1777 /tmp`
		`/bin/mkdir home`
		`/bin chmod 0777 /home`
			The ZOSSFSET File on page 4-8 and Creating a Unique Fileset on page 5-1

	Create the whatis database for the OSS shell `apropos`, `man`, and `whatis` commands	`/bin/merge_whatis`
		`/bin/merge_whatis /nonnative/usr/share/man`
		`/bin/exit`
			Updating the whatis Database Files on page 6-10

Start related servers	Start the OSS message-queue server	SCF START SERVER $ZPMON.#ZMSGQ	Starting the OSS Message-Queue Server on page 4-37
	Start the OSS sockets local server	START SERVER $ZPMON.#ZPLS, CPU 1, BACKUPCPU 0 EXIT	Starting the OSS Sockets Local Server on page 4-37

Table B-1. Creating a Basic OSS Environment Without Using the OSSSETUP Utility (page 8 of 8)

Task	Subtask	Example	See
Configure OSS users	Add users and specify an alias for each	SAFECOM ADD USER USER.ONE, 001,001, PASSWORD Secure1 ALTER USER USER.ONE, GUARDIAN DEFAULT SECURITY NUNU ALTER USER USER.ONE, GUARDIAN DEFAULT VOLUME $DATA.USER1 ADD ALIAS user1, 001,001, PASSWORD Secure1 ALTER ALIAS user1, GUARDIAN DEFAULT SECURITY NUNU ALTER ALIAS user1, GUARDIAN DEFAULT VOLUME $DATA.USER1	Managing Users and Groups on page 8-9
	Specify an initial program and an initial (home) directory for each user	ALTER ALIAS user1, INITIAL-PROGRAM /bin/sh ALTER ALIAS user1, INITIAL-DIRECTORY /home/user1 EXIT	Assigning an Initial Working Directory on page 8-13 and Assigning an Initial Program on page 8-17
	Add the user home directories to the OSS file system	OSH /bin/mkdir /home/user1 /bin/chmod 0744 /home/user1	Creating a Unique Fileset on page 5-1
Configure the default printer alias	Specify a Guardian spooler process	/bin/cd /etc /bin/cp printcap.sample printcap /bin/vi printcap default /NODE1.$S.#LPR1	Specifying a Default Printer on page 10-2

Table B-2. Completing a Preconfigured Basic OSS Environment (page 1 of 4)

Task	Subtask	Example	See
Prepare for configuration	Log on as the super ID	LOGON SUPER.SUPER	*Guardian User's Guide*
	Ensure that preconfigured databases are accessible after a node number change	FUP INFO $SYS*.*.ZOSSFSET, DETAIL FUP ALTER ZOSSFSET, ALTFILE (0,ZOSSFS00) FUP ALTER ZOSSFSET, ALTFILE (1,ZOSSFS01)	HP strongly discourages changes in node (system) numbers. However, in a few situations, such as during the first startup of a newly delivered system, you must change node numbers. on page 4-12
Configure OSS users	Add users and specify an alias for each	SAFECOM ADD USER USER.ONE, 001,001, PASSWORD `Secure1` ALTER USER USER.ONE, GUARDIAN DEFAULT SECURITY NUNU ALTER USER USER.ONE, GUARDIAN DEFAULT VOLUME $DATA.USER1 ADD ALIAS `user1, 001,001,` PASSWORD `Secure1` ALTER ALIAS `user1`, GUARDIAN DEFAULT SECURITY NUNU ALTER ALIAS `user1`, GUARDIAN DEFAULT VOLUME $DATA.USER1	Managing Users and Groups on page 8-9
	Specify an initial program and an initial (home) directory for each user	ALTER ALIAS `user1`, INITIAL-PROGRAM `/bin/sh` ALTER ALIAS `user1`, INITIAL-DIRECTORY `/home/user1` EXIT	Assigning an Initial Working Directory on page 8-13 and Assigning an Initial Program on page 8-17
Configure the OSS file system	Start all processes and filesets	VOLUME $SYSTEM.ZOSSINS STARTOSS	

Table B-2. Completing a Preconfigured Basic OSS Environment (page 2 of 4)

Task	Subtask	Example	See
Finish configuring the OSS environment	Start an OSS shell	OSH	`osh(1)` reference page in the *Open System Services Shell and Utilities Reference Manual*

Table B-2. Completing a Preconfigured Basic OSS Environment (page 3 of 4)

Task	Subtask	Example	See

Finish configuring the OSS environment (*continued*)	Create the general profile file	`/bin/cd /etc`
		`/bin/vi profile`
		`umask 022 #Only users have write permission on their files.`
		`set -o noclobber #Redirection can't overwrite files.`
		`set -o trackall #Track all aliases.`
		`export MANPATH=/usr/share/man #Match PATH use.`
			Setting Up an /etc/profile File on page 9-2

	Configure and start network services	`/bin/ln -s /G/system/ztcpip/resconf resolv.conf`
		`/bin/ln -s /G/system/ztcpip/networks networks`
		`/bin/ln -s /G/system/ztcpip/protocol protocols`
		`/bin/ln -s /G/system/ztcpip/services services`
		`/bin/ln -s /G/system/ztcpip/hosts hosts`
		`/bin/ln -s /G/system/ztcpip/ipnodes ipnodes`
			Configuring Network Services Servers, Tools, and Applications on page 4-31

		`/bin/cp smplinetd.conf inetd.conf`
		`/bin/vi inetd.conf`
		`shell stream tcp nowait root /bin/rshd`
		`/bin/vi hosts.equiv`
		`/bin/chmod 1775 *`
		`/usr/ucb/inetd -R 10&`
			Starting a Network Services Server on page 4-38

Table B-2. Completing a Preconfigured Basic OSS Environment (page 4 of 4)

Task	Subtask	Example	See

Finish configuring the OSS environment (*continued*)	Configure utilities for periodic tasks	`/bin/cd /usr/bin/cron`
		`/bin/cp at.deny.sample at.deny`
		`/bin/chmod 1775 *`
		`/bin/cd /var/adm/cron`
		`/bin/cp queuedefs.sample queuedefs`
		`/bin/cp .proto.sample .proto`
		`/bin/cp cron.deny.sample cron.deny`
			Configuring the cron Process on page 2-35

	Modify the files as necessary for your site	`/bin/vi *` `/bin/chmod 1775 *`	
	Start the cron process	`/bin/cron&`	`cron(8)` reference page, either online or in the *Open System Services Shell and Utilities Reference Manual*

Finish configuring the OSS file system	Secure fileset mount points	`/bin/cd /`
		`/bin/chmod 0777 /`
		`/bin/chmod 0775 /bin`
		`/bin/chmod 1775 /etc`
		`/bin/chmod 1775 /var`
		`/bin/chmod 1777 /tmp`
		`/bin chmod 0777 /home`
			The ZOSSFSET File on page 4-8 and Creating a Unique Fileset on page 5-1

	Create the whatis database for the OSS shell `apropos`, `man`, and `whatis` commands	`/bin/merge_whatis`
		`/bin/merge_whatis /nonnative/usr/share/man`
		`/bin/exit`
			Updating the whatis Database Files on page 6-10

C OSS Management Utilities

This appendix describes:

- The optional OSSTTY utility
- The OSS Easy Setup utilities provided in the T0585 product

OSSTTY

OSSTTY can run as a single-use process or as a server. OSSTTY provides an alternative to Telserv for OSS program terminal input or output using OSS standard files. The OSS terminal interface allows input and output only through network virtual terminal devices; using Telserv for access to such devices, OSS standard files cannot be redirected to or from Guardian files such as EDIT files or home terminal processes such as $VHS and $ZHOME. OSSTTY allows input/output redirection using such files and processes.

The OSSTTY server runs as a fault-tolerant process pair with the subsystem ID number 245 and the subsystem name of OSSTTY. The OSSTTY server runs as a subtype 30 Guardian process and uses device type 6 for the three standard process name qualifiers `#stdin`, `#stdout,` and `#stderr`.

Starting OSSTTY

OSSTTY can be started:

- Using the OSSTTY command from a TACL prompt, as described in this appendix.
- Using the OSH command with the `-osstty` flag, as described in the `osh(1)` reference page either online or in the *Open System Services Shell and Utilities Reference Manual*.
- Using the Subsystem Control Facility (SCF) Kernel subsystem to place it under the control of the persistence monitor, so that it starts automatically after a system load. See Examples on page C-6 for a suggested set of commands.

Command

To run OSSTTY, enter this command:

```
[RUN] OSSTTY
      [ [/ NAME process_name,             ]
      [ IN stdin_redirection_target,      ]
      [ OUT stdout_redirection_target,    ]
      [ TERM stderr_redirection_target,   ]
      [ NOWAIT                            ] / ]
      [ -access { ALL | OWNER }           ]
      [ -backupcpu processor              ]
      [ -coll collector                   ]
      [ -help                             ]
      [ -idletimeout seconds              ]
      [ -[no]prefixpid                    ]
      [ -[no]quiet                        ]
      [ -[no]server                       ]
      [ -[no]wrap                         ]
      [ -secure security_string           ]
```

`NAME $process_name`

specifies the Guardian process name for the OSSTTY process. HP recommends $ZTTY.

If you omit this option, the name used is a randomly generated 4-character process name created by the NonStop operating system.

`IN stdin_redirection_target`

specifies the Guardian process name for the process from which data for the OSS standard input file should be read. The process specified must use device type 0, 1, or 6. This process appears as $*process_name.*#stdin to a process using the Guardian file system or /G/*process_name/*#stdin to a process using the OSS file system.

If you specify this option, the process cannot be a collector process or a spooler. If you omit this option, the name used is the device name for the Telserv terminal from which the command is entered.

`OUT stdout_redirection_target`

specifies the Guardian process name for the process or the Guardian file name for the Guardian EDIT file to which data for the OSS standard output file should be written.

A process specified must use device type 0, 1, or 6. The process appears as $*process_name.*#stdout to a process using the Guardian file system or /G/*process_name/*#stdout to a process using the OSS file system.

If an EDIT file (file code 101) is specified but does not exist, the file is created with the security specified by the `-secure` option or the default security setting for the user.

If you omit this option, the name used is the device name for the Telserv terminal from which the command is entered.

`TERM` *`stderr_redirection_target`*

specifies the Guardian process name for the process to which data for the OSS standard error file should be written. The process specified must use device type 0, 1, or 6. This process appears as $*`process_name.`*#stderr to a process using the Guardian file system or /G/*`process_name/`*#stderr to a process using the OSS file system.

If you omit this option, the name used is the device name for the HOMETERM terminal of the session from which the command is entered.

`NOWAIT`

specifies that TACL does not wait while the OSSTTY program runs but returns a command prompt after sending the startup message to the new process.

Use this option when you use the `-server` option if you need to start an OSS program from the same terminal to use the server.

If you omit this option, TACL pauses while OSSTTY runs.

`-access { ALL | OWNER }`

Allows you to give access to the started OSSTTY process to any OSS process with a valid user ID (the value ALL) or restrict access to OSS processes that you have started under your user ID (the value OWNER).

If you omit this option, the value OWNER is used.

`-backupcpu` *`processor`*

specifies the number of the processor on which the backup copy of OSSTTY should be created. You can specify any number valid for your system other than that used by the primary copy of OSSTTY. The processor used by the primary copy of OSSTTY is the processor used by the terminal session you enter the command from.

Specifying this option causes OSSTTY to run with an active backup copy as a fault-tolerant process pair. Running as a process pair improves fault tolerance but can reduce response time slightly.

If you omit this option, OSSTTY runs without a backup copy and does not provide fault-tolerant features.

`-coll` *`collector`*

specifies the Event Management Service (EMS) collector process to receive the OSSTTY event messages. If you specify a collector process that cannot be used, OSSTTY issues a warning message.

If you omit this option or if the specified collector process is not valid, OSSTTY uses $0 as the collector process.

`-help`

requests a summary of OSSTTY command options. This is the first message described in Startup Messages on page A-58.

`-idletimeout` *`seconds`*

specifies the time in seconds that OSSTTY waits before terminating after its last opener closes it. The time waiting allows another OSS process to open it if necessary for additional processing.

Valid values are in the range 0 through 32767. A value of 0 indicates that OSSTTY should terminate immediately after its last opener closes it.

If you specify this option, you cannot specify the `-server` option.

If you omit this option, OSSTTY uses the value of 30 seconds.

`-[no]prefixpid`

controls whether OSSTTY prefixes standard file data for OUT or TERM with information to identify the OSS application being serviced. If a copy of OSSTTY services more than one OSS application, this prefix data can help distinguish which data comes from each application.

When the `-prefixpid` option is specified, a text string of the following form is prefixed to each data block:

`pid``:`*`objame`*

`pid`

indicates the OSS process ID for the application process.

`objame`

indicates the process program filename for the application process.

When the `-noprefixpid` option is specified, no information is added to the data blocks.

If you do not specify either option, OSSTTY uses the `-noprefixpid` option.

`-[no]quiet`

controls whether the warning and error messages described in Startup Messages on page A-58 are displayed. Suppressing message display can be useful when OSSTTY is used within a TACL macro or other script.

When the `-quiet` option is specified, no messages are displayed.

When the `-noquiet` option is specified, messages are displayed as necessary.

If you do not specify either option, OSSTTY uses the `-noquiet` option.

`-[no]server`

controls whether OSSTTY runs as a server that waits indefinitely for new processes to open it after its last opener closes it.

When the `-server` option is specified, OSSTTY waits indefinitely and can only be stopped by external action, such as using the TACL STOP command. This option cannot be specified with the `-idletimeout` option.

When the `-noserver` option is specified, OSSTTY waits after its last opener closes it for the number of seconds specified for the `-idletimeout` option.

If you do not specify either option, OSSTTY uses the `-noserver` option.

`-[no]wrap`

controls whether data wraps around to the beginning of a file and overwrites the data there when a disk file's end-of-file mark is reached. This option is intended for use when the OUT option specifies and EDIT file and should be used with caution to avoid inadvertent loss of data.

When the `-wrap` option is specified, overwrite is enabled.

When the `-nowrap` option is specified, reaching the end-of-file mark with more data to write causes an error.

If you do not specify either option, OSSTTY uses the `-nowrap` option.

`-secure` *`security_string`*

specifies the access permissions for the OUT file in the Guardian file system when the OUT file is an EDIT file and must be created by the command. Valid values are any four-character combination of the read, write, execute, and purge permissions allowed for Guardian file-system objects (N, U, O, A, G, C, or - for the super ID).

If you omit this option when an EDIT file must be created, OSSTTY uses the default file-system permissions for the user ID of your terminal session.

Considerations

- The OSS standard error file cannot be redirected to an EDIT file.

- The OSS shell commands and utilities do not restrict the use of the standard error file to the display of error messages. As is true for many implementations of UNIX, the standard error file can be used when unbuffered output to the terminal is the desired behavior for a utility. TACL macros or scripts that include OSSTTY should not depend upon the OSS standard error file containing only error messages, or the OSS standard output file being the only target for a prompt message.
- You can start as many copies of OSSTTY as needed. Give each copy a unique process name.
- When OSSTTY's redirectional target is a Telserv terminal and OSSTTY loses control of the break key while writing to that terminal, all applications redirecting data through that copy of OSSTTY experience target outage.

Examples

1. The following set of commands, beginning at a TACL prompt, allows redirection of a single `ls` command's standard output from your current terminal to a new EDIT file with the Guardian file identifier EFILE in the current subvolume:

   ```
   RUN OSSTTY / NAME $TTY, OUT EFILE , NOWAIT /
   OSH
   ls -l / > /G/tty/#stdout
   exit
   STATUS *, TERM
   .
   .
   .
   TEDIT EFILE
   ```

   Thirty seconds after the `ls` command completes its output, the $TTY process stops because that is its default idle time. The STATUS command allows you to ensure that OSSTTY has terminated normally so that the EDIT file was properly closed after the data is written.

2. The following command at a TACL prompt starts a fault-tolerant OSSTTY server process pair that writes all OSS application error file output to the virtual home-terminal subsystem (VHS):

   ```
   RUN OSSTTY / NAME $ZTTY, TERM $VHS, NOWAIT / -backupcpu 3
              -server
   ```

3. The following command at a TACL prompt starts a fault-tolerant OSSTTY server process pair that writes all OSS application standard file output to an EDIT file for use only by the super ID:

   ```
   RUN OSSTTY / NAME $ZTTY, OUT LOGS.MYFILE, NOWAIT /
              -backupcpu 3 -server -secure "-N--"
   ```

Processor 3 is used for the backup copy of OSSTTY in both examples 2 and 3.

See the *Open System Services Programmer's Guide* for additional examples of OSSTTY use.

Stopping OSSTTY

To stop a copy of the OSSTTY process started through either the OSH command or from a TACL prompt, use the TACL STOP command. Use the STOP command carefully; when OSSTTY does not terminate normally, any EDIT files it has open might not be properly closed and data could be lost.

To stop a copy of the OSSTTY process monitored by the persistence manager, use the SCF Kernel subsystem ABORT PROCESS $ZZKRN.#ZTTY command.

EasySetup Utilities

The OSS EasySetup utilities can be used to partially automate operation of the OSS environment. These utilities consist of the scripts, data files, and library files described in Table C-1 on page C-7.

Table C-1. The EasySetup Utilities (page 1 of 2)

Component	Description
OSSSETUP Utility on page C-11	A TACL macro (script) that executes either interactively or without interactive dialog.
STARTOSS Utility on page C-14	A TACL macro (script) that executes without interactive dialog.
STOPOSS Utility on page C-16	A TACL macro (script) that executes without interactive dialog.
OSSREMOV Utility on page C-17	A TACL macro (script) that executes interactively.
OSSINF File on page C-18	An EDIT file used by OSSSETUP that contains entries it configured for entities managed through the OSS Monitor. This file is supplied by HP.
OSSINFIL File on page C-19	An EDIT file that initially duplicates the contents of OSSINF. OSSINFIL is used by STARTOSS and is created by it when the file cannot be found. This file is created in the volume and subvolume from which the STARTOSS utility is run. You should create this file yourself before you use STARTOSS for the first time.
OSSJOURN file	An EDIT file containing all messages logged by the OSSSETUP utility. This file is created in the volume and subvolume from which the OSSSETUP utility is run.
OSSLIB file	A library of routines shared among the OSS EasySetup utilities.

Table C-1. The EasySetup Utilities (page 2 of 2)

Component	Description
OSSTREE file and /tmp/oss.tree.*ddmmmyyyy*.*system_name* file	An EDIT file and an ASCII text file, respectively, containing a list of all directories created in the OSS file system.
	These files are created by the OSSSETUP utility and contain the result of the following OSS shell command: `find / -WNOE -WNOG -type d -print`
ddmmmyyyy.*system_name* file (*continued*)	Creation of these files can take a long time. The OSSTREE file is created in the volume and subvolume from which the OSSSETUP utility is run. The pathname of the OSS copy of the file contains the name of the local system node and the timestamp from when the file was created.

Utility File Security

Once installed, the utility files should be properly secured for execution only by site-selected user IDs. Because they are TACL macros, the READ permission (not the EXECUTE permission) must be secured to control execution. For example, NN-N does not prevent execution by any user; however, ---- does restrict execution to the super ID. Commonly, STARTOSS and STOPOSS are secured G--- for use by the super group.

Interactive Dialogs

During interactive execution, the OSSSETUP or OSSREMOV utility prompts the user for needed configuration data or permission to proceed. Prompts have the form of:

prompt_message [*default_response*]

prompt_message

is a detailed description of the processing step about to be taken or the information needed to take the step.

default_response

is the action taken or the value used if the user presses Return.

After you start a utility, you cannot back up within its dialog and choose alternate responses to prompts. Only the OSSSETUP utility provides help as a TACL command line option.

You can stop any of the utilities by pressing the Break key. This action is acknowledged by the following message:

```
Break or error terminated operation.
task of the OSS subsystem on system-name:  FAILED
Utility-name terminated unexpectedly or encountered errors.
```

task

the action (specific to each utility) in process when the break key was pressed:

task	Utility
Installation	OSSSETUP
Removal	OSSREMOV
Shutdown	STOPOSS
Startup	STARTOSS

system-name

is the Expand node name for the system on which the utility stopped.

Utility-name

is the name of the utility issuing the message.

The same message appears, preceded by error diagnostic information, if a utility must stop because of an internally detected error.

Diagnostic Messages

All the OSS EasySetup utilities issue a banner display message and progress and error messages that are intended to be self-explanatory. OSSSETUP writes diagnostic messages to its journal file as well as to these files. Progress messages are written to the user's home terminal.

If the Event Management Service (EMS) event definition files ZOSSTACL and ZEMSTACL have been installed in the ZSPIDEF subvolume of $SYSTEM, the T0585AAA version of the utilities also writes progress messages to the system service log ($ZLOG), and to the EMS collector process $0. Beginning with the T0585AAB version, the utilities suppress progress messages as EMS events unless the EASYSETUP^EMSVOL PARAM has been defined.

Progress messages have the form:

```
text: { STARTING | COMPLETED | FAILED }
```

where COMPLETED indicates successful completion of a task, while FAILED indicates that the task could not be completed.

OSS EasySetup EMS events are logged with the OSS subsystem ID and have the form:

```
TANDEM.OSS.D30 00010 USER NOTICE (userID) : text
```

where *userID* is your user ID, and *text* is the event message text.

Diagnostic messages for nonfatal situations include a suggested response to avoid the problem. See the *Open System Services Installation Guide* for a complete list of possible diagnostic messages

Utility PARAMs

Beginning with the T0585AAB version, the EasySetup utilities support the following TACL PARAM declarations:

PARAM Name	PARAM value	Used by
EASYSETUP^EMSVOL	A valid disk volume name that identifies the disk on which the Event Management Service (EMS) subvolume (ZSPIDEF) resides. This subvolume must contain the event definition files ZOSSTACL and ZEMSTACL. Specifying this subvolume enables EMS message logging by the utilities. By default, no EMS messages are issued.	OSSSETUP STARTOSS STOPOSS OSSREMOV
EASYSETUP^STARTUP	The keyword $NULL, if the utility is to include the $NULL process among its actions. The keyword $ZSMP, if the utility is to include the $ZSMP process among its actions. Both of the above keywords, separated by a blank and enclosed in quotation marks. The default action excludes either process for which a keyword is omitted.	OSSSETUP STARTOSS STOPOSS OSSREMOV
EASYSETUP^UTILVOL	A valid disk volume name that identifies the disk on which the OSS installation subvolume (ZOSSUTL) resides. The specified volume must be on the local NonStop server node (the COPYOSS utility called by OSSSETUP does not allow installation on remote nodes). The default value is $SYSTEM.	OSSSETUP

Quotation marks are optional if only one value is specified for a PARAM. If you specify a value that is not recognized as a keyword, a warning message appears and the value is ignored. To have the intended effect, PARAMs must be declared before a utility is started.

Examples

1. To issue progress messages as EMS events using a ZSPIDEF subvolume on $SYSTEM when initially configuring and starting OSS, enter:

   ```
   PARAM EASYSETUP^EMSVOL $SYSTEM
   OSSSETUP
   ```

2. To issue progress messages as EMS events using a ZSPIDEF subvolume on $SYSTEM on the remote node \NODE when initially configuring and starting OSS, enter:

   ```
   PARAM EASYSETUP^EMSVOL \NODE.$SYSTEM
   OSSSETUP
   ```

   This allows systems in an Expand network to share a ZSPIDEF subvolume instead of maintaining a separate ZSPIDEF subvolume on each node.

3. To configure and start both $ZSMP and $NULL when initially configuring and starting OSS, enter:

   ```
   PARAM EASYSETUP^STARTUP "$ZSMP $NULL"
   OSSSETUP
   ```

4. To start only $NULL when starting OSS, enter:

   ```
   PARAM EASYSETUP^STARTUP "$NULL"
   STARTOSS
   ```

5. To configure and start OSS using an installation subvolume on the disk $DSM, enter:

   ```
   PARAM EASYSETUP^UTILVOL $DSM
   OSSSETUP
   ```

OSSSETUP Utility

The OSSSETUP utility configures a new, basic OSS environment by using the server and fileset configuration file values discussed in Section 4, Managing Servers, and Section 5, Managing Filesets. The T0585AAA version of OSSSETUP also configures the OSSMON object file ($ZPMON) as a persistent object in the Kernel Subsystem; T0585AAB and subsequent versions configure $ZPMON as a generic object.

The T0585AAA version of OSSSETUP configures the OSMP object file ($ZSMP) as a generic object and the NULL object file ($NULL) as a persistent object if they are not already running. The T0585AAB and subsequent versions of OSSSETUP can be configured to do that also, but do not do so by default.

The OSSSETUP utility can be used when:

- Your system was not preconfigured by HP.
- You do not require more than three filesets (ROOT for `/`, TEMP for `/tmp`, and HOME for `/home`).

OSSSETUP is intended to provide a quick way to initially configure an OSS environment. You can later build on the configuration OSSSETUP creates by manually configuring additional resources or altering your initial configuration.

You must not use the OSSSETUP utility when an OSS configuration already exists on your node. For example, you cannot use OSSSETUP when:

- Your system was preconfigured by HP, but OSSREMOV has not been run.
- OSSSETUP has previously been run, but OSSREMOV has not been run.
- You have manually configured your OSS environment.

Command

To run OSSSETUP, enter these commands:

```
VOLUME $SYSTEM.ZOSSINS

[ RUN ] OSSSETUP [ DEFAULTS | ? | HELP ]
```

`DEFAULTS`

causes the utility to execute without interactive dialog. The resulting configuration is described in Configuration Files on page 4-7.

`? | HELP`

suppresses creation of an OSS environment configuration and provides general help text for the utility.

When no parameter is specified on the command line, the OSSSETUP utility runs interactively. However, you cannot enter HELP or ? at subsequent OSSSETUP prompts to see descriptive information about the corresponding dialog screens.

Summary Displays

As OSSSETUP executes, it displays summaries of your responses and asks for confirmation that the corresponding configuration is acceptable before proceeding. Figure C-1, Figure C-2, and Figure C-3 on page C-13 contain samples of these progress displays.

Figure C-1. Example of Servers, Subsystem Processes, and Other Information Display

```
              Servers, Subsystem Processes and other Information

Process    Subsystem/Server       Processors (-1 = N/A)

$ZPMON     *OSS Monitor           System load processor
$ZSMP      *Security Manager      0      1
$NULL      *NULL Monitor          0      1
$ZPLS      Local Server           1      2
$ZMSGQ     Message Queue Server   2      3
$ZPNS      ROOT's Name Server     0      1
$ZPNH      HOME's Name Server     -1     -1
$ZTAxx     *Transport Agents      One configured for each processor

Processes with asterisks (*) cannot be modified using this utility.

Active      0<--CPU Status-->15    TSV Archive location: $SYSTEM.ZOSSUTL
Processors: 1111,0000,0000,0000


Do you want to configure the OSS subsystem processes with the
above recommendations? <yes/[NO]>:
```

Figure C-2. Example of a Storage Volumes for Filesets Display

```
                          Disk Volumes for Filesets

Potential subsystem volume(s):   $WORK     $OSS        $KAOSR20  $KAOSR17
$G0608     $G0607      $G0606    $G0501    $BOAT       $AUDIT2   $AUDIT1

Potential subsystem volume(s) NOT in a started state:  $G0604

Recommended volumes not to use:  $SYSTEM   $DSMSCM

Volumes that CANNOT be used:     $TAPE1    $TAPE0      $SEETHU   $Z4Y4
```

Figure C-3. Example of Filesets, Mount Points, and Associated Name Servers Display

```
             Filesets, Mount Points and Associated Name Servers

Fileset  Name   Catalog   Mount     Pool Space Volume(s)

ROOT     $ZPNS  $WORK     /         $WORK     $OSS        $KAOSR20  $KAOSR17
                                    $G0608    $G0607      $G0606    $G0501
                                    $BOAT     $AUDIT2     $AUDIT1

HOME     $ZPNH  $OSS      /home     $WORK     $OSS        $KAOSR20  $KAOSR17
                                    $G0608    $G0607      $G0606    $G0501
                                    $BOAT     $AUDIT2     $AUDIT1

TEMP     $ZPNS  $KAOSR20  /tmp      $KAOSR20

$ZZSTO.INTERNAL-DISK profile setting for OSSCaching: ON


Do you want to configure the OSS filesets as shown above? <yes/[NO]>:
```

Considerations

- Beginning with the G06.17 release version update (RVU) and the G10 version of the OSS Monitor, the OSS Monitor no longer uses the $NULL process or requires the $ZSMP process for its own startup. However, not configuring or starting $NULL or $ZSMP could affect the functioning of other products or subsystems.
- If $NULL or $ZSMP is already running and not configured as a generic object, you must stop it before you can use OSSSETUP to configure it as a generic object. To stop $NULL, use the TACL STOP command. To stop $ZSMP, use the SAFECOM STOP command.
- The OSSSETUP utility and the files it creates must be appropriately secured for access by users other than the super ID, according to your site's security guidelines.
- The OSSSETUP command installs all code and text files from current `pax` archive files in the ZOSSUTL subvolume into the filesets it creates.
- The OSSSETUP command does not define users or user attributes such as INITIAL-DIRECTORY in the security database.
- The OSSSETUP command uses the storage subsystem OSSCACHING value of the storage-pool file's disk volume for the OSSCACHING attribute of all disks in that storage-pool file. If the value used is different from the value originally configured for the disk, OSSSETUP records the value used in the OSSJOURN file.
- You cannot use this command from a remote Expand node.
- You cannot use this command from the system startup TACL session ($YMIOP.#CLCI). You should use a Telserv session instead.
- Fileset mount points should be checked after installation to ensure that they have security permissions consistent with your site's security policies. The OSS shell `chmod` command can be used to modify permissions as necessary.

STARTOSS Utility

The STARTOSS utility starts the processes and filesets previously configured through the OSSSETUP utility or subsequently added to the OSSINFIL file. STARTOSS is intended for use after a system load or after the STOPOSS utility has been used.

The STARTOSS utility can be used when any of the following is true:

- Your OSS environment was preconfigured by HP or configured by using OSSSETUP and has not been modified.
- Your OSS environment was preconfigured by HP or configured by using OSSSETUP, and all subsequent modifications have been entered into the OSSINFIL file in the STARTOSS volume and subvolume.

- Your OSS environment was not preconfigured by HP or configured by using OSSSETUP, but you have created an OSSINFIL file in the STARTOSS volume and subvolume and maintain your current OSS configuration in that file.

Under other conditions, using the STARTOSS utility might not achieve your intention.

Command

To run STARTOSS, enter these commands:

```
VOLUME $SYSTEM.ZOSSINS

[ RUN ] STARTOSS
```

Considerations

- The T0585AAA version of STARTOSS attempts to start $ZSMP, $NULL, $ZPMON, and any processes and filesets specified in the OSSINFIL file. STARTOSS ignores any process or fileset that is already running.
- Beginning with the T0585AAB version, STARTOSS attempts to start $ZPMON, either process identified by an EASYSETUP^STARTUP PARAM keyword, and any processes and filesets specified in the OSSINFIL file. STARTOSS ignores any process or fileset that is already running.
- Beginning with the G06.17 release version update (RVU) and the G10 version of the OSS Monitor, the OSS Monitor no longer uses the $NULL process or requires the $ZSMP process for its own startup. However, not starting $NULL or $ZSMP could affect the functioning of other products or subsystems.
- The STARTOSS utility, the files it accesses, and the processes it starts must be appropriately secured for access by users other than the super ID, according to your site's security guidelines. See Licensing the OSS Monitor to the Super Group on page 8-19 for a possible approach.
- You must be logged in as a member of the super group (255,*nnn*) to use this utility.
- You cannot use this command from a remote Expand node.
- You can embed this command in the CIIN file or a subsystem startup file such as $SYSTEM.STARTUP.STARTUP as follows:

  ```
  == Start the OSS subsystem, filesets, and servers:
  #Push #Defaults
  Volume $SYSTEM.ZOSSINS
  Run STARTOSS
  #Pop #Defaults
  ```

  The keyword Run is optional, but it is required if #PMSEARCHLIST does not include #DEFAULTS.

- Fileset mount points should be checked after all filesets are started to ensure that they have security permissions consistent with your site's security policies. The OSS shell `chmod` command can be used to modify permissions as necessary.

STOPOSS Utility

The STOPOSS utility stops all filesets, regardless of whether they were configured in OSSINF or OSSINFIL. STOPOSS is intended for use before a system shutdown. It does not stop OSS servers (such as $ZPLS or $ZMSGQ) or subsystem processes, because that action is not needed before a system shutdown.

Command

To run STOPOSS, enter these commands:

```
VOLUME $SYSTEM.ZOSSINS

[ RUN ] STOPOSS
```

Considerations

- Beginning with the G06.17 release version update (RVU) and the G10 version of the OSS Monitor, the OSS Monitor no longer uses the $NULL process or requires the $ZSMP process for its own startup. However, stopping $NULL or $ZSMP could affect the functioning of other products or subsystems.
- If $ZSMP, $NULL, or $ZPMON are not running when the T0585AAA version of STOPOSS is run, STOPOSS starts the missing process so that it can complete its function, then stops the process again before completing.
- If $ZSMP and/or $NULL are specified in the EASYSETUP^STARTUP PARAM and are not running, or if $ZPMON is not running when the T0585AAB version of STOPOSS is run, STOPOSS starts the missing process so that it can complete its function, then stops the process again before completing.

  If you used OSSSETUP to configure $ZSMP or you manually configured $ZSMP as a generic process using the recommended values defined in Starting the OSS Monitor as a Persistent Process on page 2-9, $ZSMP will not restart itself because its AUTORESTART value is 0. If $ZSMP does not have an AUTORESTART value of 0, you must use the SCF ABORT command and then the SAFECOM STOP command to permanently stop $ZSMP.
- The STOPOSS utility, the files it accesses, and the processes it stops must be appropriately secured for access by users other than the super ID, according to your site's security guidelines. See Licensing the OSS Monitor to the Super Group on page 8-19 for a possible approach.
- You must be logged in as a member of the super group (255,*nnn*) to use this utility.
- You cannot use this command from a remote Expand node.

- You can embed this command in the subsystem shutdown file as follows:

```
== Stop the OSS subsystem, filesets, and servers:
#Push #Defaults
Volume $SYSTEM.ZOSSINS
Run STOPOSS
#Pop #Defaults
```

  The keyword Run is optional, but it is required if #PMSEARCHLIST does not include #DEFAULTS.

OSSREMOV Utility

The OSSREMOV utility deletes the configurations of all processes and filesets regardless of whether they are configured in the OSSINF file or OSSINFIL file. If any OSS processes or OSH sessions have been started, you must stop them before you run OSSREMOV. SQL program files cataloged within the OSS file system during use of the OSS environment must be removable for OSSREMOV to successfully complete; such program files should be removed from the SQL catalog before running OSSREMOV.

Command

To run OSSREMOV, enter these commands:

```
VOLUME $SYSTEM.ZOSSINS

[ RUN ] OSSREMOV
```

Considerations

- Beginning with the G06.17 release version update (RVU) and the G10 version of the OSS Monitor, the OSS Monitor no longer uses the $NULL process or requires the $ZSMP process for its own startup. However, removing $NULL or $ZSMP could affect the functioning of other products or subsystems.
- The OSSREMOV utility, the files it accesses, and the processes it stops must be appropriately secured for access by users other than the super ID, according to your site's security guidelines. See Licensing the OSS Monitor to the Super Group on page 8-19 for a possible approach.
- The OSSREMOV command should be used when the OSSSETUP utility does not complete a successful configuration.
- The OSSREMOV utility does not back up any files from the OSS file system before removing filesets.
- The OSSREMOV command removes all OSS files and all fileset catalogs. As a result, it also removes any directories or files kept in the OSS file system by products such as NonStop SQL/MX or iTP WebServer.

- The OSSREMOV command prompts you to determine whether it should remove the security manager server process $ZSMP and the $NULL process. These processes are often required by other products and usually should be allowed to continue running.

  OSSREMOV uses the Safeguard SAFECOM program to stop $ZSMP to ensure that $ZSMP does not restart itself. If your site has not licensed Safeguard, either do not use OSSREMOV or respond “no” to its prompt about stopping $ZSMP.

  If you used OSSSETUP to configure $ZSMP or you manually configured $ZSMP as a generic process using the recommended values defined in Starting the OSS Monitor as a Persistent Process on page 2-9, $ZSMP will not restart itself because its AUTORESTART value is 0. If $ZSMP does not have an AUTORESTART value of 0, use the SCF ABORT command and then the SAFECOM STOP command to permanently stop $ZSMP.
- The OSSREMOV command skips the tasks for any portions of the OSS configuration that are no longer present to be removed and continues processing until all detectable portions have been removed.
- The OSSREMOV command does not remove users or user attributes such as INITIAL-DIRECTORY from the security database.
- The OSSREMOV utility does not restore storage subsystem OSSCACHING settings for disk volumes used in storage pools to their values before OSSSETUP was run. Modified values are retained unless you subsequently restore them manually. You can determine which settings were changed by OSSSETUP from the OSSJOURN file.
- You cannot use this command from a remote Expand node.

OSSINF File

This EDIT file is supplied by HP and contains the initial configuration OSSSETUP creates for management by the OSS Monitor. You should not modify this file because the file can be overwritten by a subsequent software product revision (SPR), which would remove your changes.

Figure C-4 shows the content of a typical OSSINF file.

Figure C-4. Example of an OSSINF File

```
* File: OSS Monitor in-file ( Version 1.0 )  { <--MUST BE FIRST LINE OF FILE}
* This is an in-file for the OSS T0585 utilities STARTOSS and OSSSETUP. The
* OSS Monitor process manages all entries in this file.

* There is no need to refer to any name server or transport agent server.
* Name servers are automatically started when their corresponding filesets
* are started. Transport agent servers are started when a processor is
* reloaded.

* NOTE: Add an asterisk followed with a space in front of a line to prevent
*       that server or fileset from being started.

* Format for server entry:  SERVER #<name>
* Format for fileset entry: FILESET <name>


* Start the OSS Message Queue Server:
SERVER #ZMSGQ


* Start the OSS Local Server:
SERVER #ZPLS


* Add additional filesets to the end of this file. Starting order
* of filesets must be maintained. An incorrect order can cause
* filesets not to start. Do not use a hash mark (#) in fileset names.

* Start the OSS ROOT, HOME and TEMP Filesets:
FILESET ROOT
FILESET HOME
FILESET TEMP
```

OSSINFIL File

This file is used by the STARTOSS utility. It should be created and maintained for each site's specific OSS configuration of servers and filesets managed by the OSS Monitor process.

You should create this EDIT file by using FUP DUP on the OSSINF file. If the STARTOSS utility cannot locate this file, this file is created from the OSSINF file.

Always edit OSSINFIL to update its content any time you add or remove an OSS server or fileset. The rules for adding entries to the OSSINFIL file are summarized in the sample file shown in Figure C-4:

- Comments must begin with the asterisk (*) character followed by a space. The sequence asterisk asterisk (**) does not begin a valid comment, but the sequence asterisk space asterisk (* *) does begin a valid comment.
- FILESET entries must appear in the order of their mount points within the OSS file system hierarchy.
- FILESET entries cannot contain the number sign (#) character.

- SERVER entries must contain the number sign (#) character.
- SERVER entries for #ZTA*nn* are not needed.
- SERVER entries for OSS name servers are not valid.

D

Falling Back to a Previous Release Version Update

You cannot fallback to a G-series RVU from an H-series RVU because of hardware considerations. The considerations in this appendix apply to OSS products only. There might be other considerations when falling back to a previous RVU. See the *Release Version Update Compendium* for the current RVU for fallback notes for products installed on your system and see the *Interactive Upgrade Guide* for detailed information about falling back to a previous RVU.

The procedures described in this appendix assume that you either do not use DSM/SCM to manage OSS files (the **Manage OSS Files** check box is not selected in the planner interface) or you do not have the T6031 D46 product version update (PVU) or later installed.

Use of the DSM/SCM backout feature is described in the *DSM/SCM User's Guide*. Be careful when using DSM/SCM in a network of systems running both G-series RVUs and H-series RVUs; the default for **Manage OSS Files** is unchecked for G-series RVUs and checked for H-series RVUs.

Falling Back to G-series Release Version Update As Far Back as G06.12

The procedure you use for falling back depends on the RVU to which you are falling back and the method you used to install the RVU of OSS from which you are falling back:

- For RVUs after G06.18, you can use the DSM/SCM backout feature for files in the OSS file system if you used DSM/SCM to install and manage OSS files for both RVUs involved in the fallback.
- On RVUs prior to G06.19, you cannot use the DSM/SCM backout feature for files in the OSS file system. DSM/SCM will have no old configuration to fall back to.

The first time a G09 or newer OSS Monitor runs on a system that has a D46 OSS Monitor installed, a copy of the existing D46 Monitor database is created. In the copy, the names of the files have the prefix "ZOSS" instead of the original prefix "ZPOS"; for example, ZPOSFSET is copied to ZOSSFSET. The copied files (whose names begin with "ZOSS") are extended to include new fields that support OSS security auditing. After this, the G09 or newer OSS Monitor uses the converted database and ignores the original D46 Monitor database.

To fall back to a G06.12 or more recent G-series RVU:

- If no new features are used and there are no changes to configuration records, you can safely fall back without any special preparation.

- If there are changes to the configuration records and new features are used, you need to consider the following:
  - If either the INODECACHE or LINKCACHE OSS name server attribute was changed to a value greater than 250000 after installing the T8622G11 or a later release and you fall back to a release with a product version prior to T8622G11, the SCF START FILESET command will fail with error OSS E00008 on any fileset managed by the affected OSS name server. To avoid this error, use the SCF ALTER SERVER command to change the INODECACHE or LINKCACHE attribute value to a value lower than or equal to 250000 before any filesets using that OSS name server are started.

E Environment Limits

This appendix summarizes the architectural and functional limits that apply to the Open System Services environment on NonStop servers. Limits for a specific release version update (RVU) are subject to increases as software product revisions (SPRs) occur.

OSS application programs that conform to POSIX.1 conform to a set of limits defined by the POSIX standards; POSIX limits can be less than those imposed by the environment. Refer to the header files used by a specific application to determine the current settings of that program's programmatic limits; for example, most programs use `/usr/include/limits.h`.

Table E-1. OSS Environment Limits (page 1 of 4)

Description	Maximum Value	Comment	To Check Amount Currently Used

OSS files in one fileset	2.2 million	The limiting factor is the number of inodes used within the fileset; inodes are used for any entry in the fileset: directories, named pipes (FIFOs), files, AF_UNIX sockets, and so forth.
		Prior to RVU G06.24 (prior to PVU T8621G11), the limit was 500,000 inodes per fileset.
		Prior to RVU G06.18 (prior to SPR T8621AAU), the limit was 250,000 inodes per fileset.
			At a TACL prompt in the catalog subvolume:
			`FUP INFO PXINODE, STAT`
			gives a rough approximation; the PXINODE file groups free inodes into a single record.
			At an OSS shell prompt, using the fileset mount point (*`mtpt`*):
			`ls -W NOG -W NOE -R` *`mtpt`* `| wc -l`
			gives a rough approximation (also skips free inodes). If the fileset directory tree contains the mount point for another fileset, the inodes of that fileset will also be listed.

OSS filename	248 characters		
OSS pathname	1024 characters		

Table E-1. OSS Environment Limits (page 2 of 4)

Description	Maximum Value	Comment	To Check Amount Currently Used
OSS file size	For H06.06 and later RVUs: Approximately 1 terabyte (constrained by the space available on the disk volume) For G-series RVUs and for H-series RVUs prior to H06.06: Approximately 2 gigabytes	For RVUs H06.06 and later, the file size limit depends on the function used to open the file. For details, see OSS and Guardian Enscribe File Formats and File Size Limits on page E-5.	
OSS processes per node	Approximately 29,000	Prior to RVU G06.19 (prior to SPR T9050AOU), the limit was approximately 16,000.	At an OSS shell prompt: `ps -lA` lists active and zombie processes.
OSS processes per processor	Approximately 4,000 for S-series servers Approximately 8,000 for NS-series servers	The limiting factor is the number of possible process control blocks (PCBs) per processor: For RVUs G06.00 through G06.07, the limit was approximately 1,800. For RVUs G06.08 through G06.18 (prior to SPR T9050AOU), the limit was approximately 2,200.	At a TACL prompt, use PEEK.
OSS file opens per process, without using the `select()` function to manage I/O	4,500		

Table E-1. OSS Environment Limits (page 3 of 4)

Description	Maximum Value	Comment	To Check Amount Currently Used
OSS file opens per application process, using the `select()` function to manage I/O	1,024 4,096	Prior to RVU G06.26 (prior to NPV T9055 G12), the limit was 1,024.	
OSS file opens per terminal process, when using the `select()` function and nonblocking terminal I/O	1	The OSS terminal helper process only accepts one open per terminal process	

OSS file opens per processor	Approximately 12,000	Limiting resource is OSS POSIX extended segment (PXS) memory.
		Prior to SPR T8620ABT, the limit was approximately 7,750.
			Usage can be checked with Measure by looking at the OSSCPU entity; at a MEASCOM prompt:
			`ADD OSSCPU *`
			`START MEASUREMENT` *`mymeas`*
			`STOP MEASUREMENT` *`mymeas`*
			`LIST OSSCPU *`

OSS pthreads per process	At least 10,000	The limiting factor is available memory.	
OSS sockets per processor	4,096		
OSS pipes per processor	135 to 1,024, depending on message size	Prior to SPR T8620ABT, the limit was 39 to 256, depending on message size.	
Device labels per node	$ZZZZZ_{32}$ - 480_{10}	Certain label designators are skipped.	At a TACL prompt, use the SCF INFO FILESET *, DETAIL command.

Table E-1. OSS Environment Limits (page 4 of 4)

Description	Maximum Value	Comment	To Check Amount Currently Used
OSS filesets per node	Functional limit: 1,000,000 Theoretical maximum: 33,553,952	Functional limit is imposed by functional limit on directories (a fileset must have a directory as a mount point). Theoretical maximum is determined by maximum number of device labels.	At a TACL prompt, use the SCF INFO FILESET *, DETAIL command.
Entries in an inode cache	500,000	A smaller limit is configurable. The default is 4,096. Prior to RVU G06.24, the limit was 250,000.	
Entries in a link cache	500,000	A smaller limit is configurable. The default is 4,096. Prior to RVU G06.24, the limit was 250,000.	
Buffer size for Network File System (NFS) nonretryable operations	128,000 bytes	A smaller limit is configurable. The default is 16,000.	
OSS message-queue IDs that can be cached	1024	A smaller limit is configurable. The default is 32.	
OSS message-queue messages per node	16,384	A smaller limit is configurable. The default depends on the maximum number of message-queue IDs that can be cached.	At an OSS shell prompt: `ipcs -oq`
OSS message queue length	65,535 bytes	A smaller limit is configurable. The default is 65,535.	At an OSS shell prompt: `ipcs -oq`
OSS message-queue message length	32,000 bytes	A smaller limit is configurable. The default is 32,000.	

OSS and Guardian Enscribe File Formats and File Size Limits

The size limit for a file depends on:

- The function used to open the file
- The RVU on which the file was created
- The type of API (OSS or Guardian) used to create the file
- The current size of the file

File Size Limits For Files Created on H06.06 and Later RVUs

Note. For H06.06 and later RVUs, *xxx*( ) functions are mapped to *xxx*64( ) functions if you compile the application using the `#define _FILE_OFFSET_BITS 64` feature test macro or an eqivalent compiler command option.

Table E-2. Size Limits for Files Created on H06.06 and Later RVUs

	Using This Function			
To Create This File...	**32-Bit OSS APIs such as `creat( )`**	**64-Bit OSS APIs such as `creat64( )`**	**Guardian APIs without 64-Bit elections**	**Guardian APIs with 64-Bit elections**
OSS file	2GB - 8KB (2,147,475,456 bytes), Enscribe Format 1	Approximately 1TB-41MB, constrained by disk volume space, Enscribe Format 2	N/A	N/A
Guardian Enscribe file	Approximately 2GB - 2MB, Enscribe Format 1	Approximately 25GB, Enscribe Format 2	4GB - 4KB (4,294,963,200 bytes), Enscribe Format 1	1TB - 1MB (1,099,510,579,200), Enscribe Format 2

File Size Limit Behavior for File Open Operations

Attempts to use 32-bit OSS APIs such as `open( )` to open any Guardian Enscribe file that is larger than approximately 2 gigabytes fail with an error.

Limits for Guardian Enscribe files opened with Guardian APIs are described in the *Guardian Procedure Calls Reference Manual*.

Table E-3. File Format and Limits Table for File Open Behavior

	Using This Function			
To Open This File...	**32-Bit OSS APIs such as open( )**	**64-Bit OSS APIs such as open64( )**	**Guardian APIs without 64-Bit elections**	**Guardian APIs with 64-Bit elections**
OSS small file (< approx. 2 GB) created on H06.06 or later RVUs	2GB-8KB (2,147,475,456 bytes)	Approximately 1TB-41MB, constrained by disk volume space	2GB-8KB (2,147,475,456 bytes)	Approximately 1TB-41MB, constrained by disk volume space
OSS large file (> approx. 2 GB) created on H06.06 or later RVUs	Fails with an EOVERFLOW error	Approximately 1TB-41MB, constrained by disk volume space	Fails with a FEOVERSIZEFILE error	Approximately 1TB-41MB, constrained by disk volume space
OSS small file created on older RVUs and on a disk with a 514-byte sector size	Approximately 2GB-32MB if 64-bit OSS API has never been used on this file 2GB-8KB (2,147,475,456 bytes) if prior open calls have been closed and if the file has previously been opened by a 64-bit OSS API	Approximately 1TB-41MB, constrained by disk volume space	Approximately 2GB-32MB if 64-bit API has never been used on this file 2GB-8KB (2,147,475,456 bytes) if prior open calls have been closed and if the file has previously been opened by a 64-bit API	Approximately 1TB-41MB, constrained by disk volume space
OSS small file created on older RVUs and on a disk with a 512-byte sector size	Approximately 2GB-24MB if 64-bit OSS API has never been used on this file 2GB-8KB (2,147,475,456 bytes) if prior open calls have been closed and if the file has previously been opened by a 64-bit OSS API	Approximately 1TB-41MB, constrained by disk volume space	Approximately 2GB-24MB if 64-bit API has never been used on this file 2GB-8KB (2,147,475,456 bytes) if prior open calls have been closed and if the file has previously been opened by a 64-bit API	Approximately 1TB-41MB, constrained by disk volume space

Table E-3. File Format and Limits Table for File Open Behavior

	Using This Function			
To Open This File...	**32-Bit OSS APIs such as open( )**	**64-Bit OSS APIs such as open64( )**	**Guardian APIs without 64-Bit elections**	**Guardian APIs with 64-Bit elections**
Guardian Enscribe Format 1 file created using OSS APIs on H06.06 or later RVUs	Approximately 2GB-2MB, Enscribe Format 1	Approximately 2GB-2MB, Enscribe Format 1	Approximately 2GB-2MB, Enscribe Format 1	Approximately 2GB-2MB, Enscribe Format 1
Guardian Enscribe Format 1 file created using Guardian APIs on H06.06 or later RVUs	Approximately 2GB-1MB, Enscribe Format 1, constrained by maximum number of extents and extent sizes	Approximately 2GB-1MB, Enscribe Format 1, constrained by maximum number of extents and extent sizes	See the *Guardian Procedure Calls Reference Manual*.	See the *Guardian Procedure Calls Reference Manual*.
Guardian Enscribe Format 2 file created using OSS APIs on H06.06 or later RVUs (and file is less than approx. 2 GB)	2GB-8KB (2,141,475,456 bytes), Enscribe Format 2	Approximately 25 GB, Enscribe Format 2	2GB-8KB (2,141,475,456 bytes), Enscribe Format 2	Approximately 25 GB, Enscribe Format 2
Guardian Enscribe Format 2 file created using Guardian APIs on H06.06 or later RVUs (and file is less than approx. 2 GB)	2GB-8KB (2,141,475,456 bytes), Enscribe Format 2	Approximately 1TB constrained by disk volume space, Enscribe Format 2	See the *Guardian Procedure Calls Reference Manual*.	See the *Guardian Procedure Calls Reference Manual*.
Guardian Enscribe Format 1 file created using OSS APIs on G06.28, any type of magnetic disk	197 MB (206,213,120 bytes)	197 MB (206,213,120 bytes)	197 MB (206,213,120 bytes)	197 MB (206,213,120 bytes)

Table E-3. File Format and Limits Table for File Open Behavior

	Using This Function			
To Open This File...	**32-Bit OSS APIs such as open( )**	**64-Bit OSS APIs such as open64( )**	**Guardian APIs without 64-Bit elections**	**Guardian APIs with 64-Bit elections**
Guardian Enscribe Format 1 file created using Guardian APIs on G06.28 and any type of magnetic disk	2GB-1MB (2,146,435,072 bytes), constrained by maximum number of extents and extent sizes	2GB-1MB (2,146,435,072 bytes), constrained by maximum number of extents and extent sizes	See the *Guardian Procedure Calls Reference Manual*.	See the *Guardian Procedure Calls Reference Manual*.
Guardian Enscribe Format 1 file created using OSS APIs on RVUs prior to G06.28 and on a disk with a 514-byte sector size	Approximately 176MB	Approximately 176MB	Approximately 176MB	Approximately 176MB
Guardian Enscribe Format 1 file created using OSS APIs on RVUs prior to G06.28 and on a disk with a 512-byte sector size	Approximately 197 MB	Approximately 197 MB	Approximately 197 MB	Approximately 197 MB

Glossary

A7CINFO file. A Distributed Systems Management/Software Configuration Manager (DSM/SCM) configuration file distributed with OSS products that contains information about the contents of all `pax` archive files of an OSS distribution subvolume.

absolute pathname. A pathname that begins with a slash (/) character and is resolved beginning with the root directory. Contrast with relative pathname.

address space. The memory locations to which a process has access.

API. See application program interface (API).

application program interface (API). A set of services (such as programming language functions or procedures) that are called by an application program to communicate with other software components. For example, an application program in the form of a client might use an API to communicate with a server program.

backout. The Distributed Systems Management/Software Configuration Manager (DSM/SCM) action of making the last configuration applied to the target system inaccessible and replacing it with the previous configuration.

Coordinated Universal Time (UTC). The standard measure of time from the beginning of the current Epoch. UTC is sometimes called Universal Coordinated Time, CUT, or UCT; the standard appellation is abbreviated as UTC, an arbitrary ordering of the letters. UTC was formerly called Greenwich mean time (GMT). See also Epoch.

creation pool. The set of disk volumes currently used for new file creation for a fileset. The creation pool is defined by the storage-pool file for the fileset and is a subset of the storage pool for the fileset.

creation version serial number (CRVSN). A number assigned by the disk process when a file is created. The CRVSN is used by the disk process and the OSS name server process to verify that access to the correct file occurs. The CRVSN is stored in the catalog entry for a regular file and is passed to the disk process when a Data Definition Language (DDL) request is made that involves the file.

CRVSN. See creation version serial number (CRVSN).

device. A computer peripheral or an object that appears to the application as such. See also terminal.

directory. A type of OSS special file that contains directory entries, which name links to other files. No two directory entries in the same directory have the same name.

distribution subvolume (DSV). A subvolume containing program files for a particular software product along with the software release document (softdoc) file for that product. The format for an HP DSV name is `Ynnnnrrr` or `Rnnnnrrr` where `nnnn` is

the software product number and `rrr` is the base version identifier (such as G01) or software product revision (SPR) identifier (such as AAB.)

DSV. *See* distribution subvolume (DSV).

Epoch. The period beginning January 1, 1970, at 0 hours, 0 minutes, and 0 seconds Coordinated Universal Time (UTC). See also Coordinated Universal Time (UTC).

FIFO. A type of special file that is always read and written in a first-in, first-out manner.

file. An object to which data can be written or from which data can be read. A file has attributes such as access permissions and a file type. In the OSS environment, file types include regular file, character special file, block special file, FIFO, and directory.

filename. In the OSS environment, a component of a pathname containing any valid characters other than a slash (/) or a null. In the Guardian environment, a filename is the set of node name, volume name, subvolume name, and file identifier characters that uniquely identifies a file.

fileset. In the OSS environment, a set of files with a common mount point within the file hierarchy. A fileset can be part or all of a single virtual file system.

On an HP NonStop S-series or NonStop NS-series system, the Guardian file system for a node has a mount point and is a subset of the OSS virtual file system. The entire Guardian file system therefore could be viewed as a single fileset. However, each volume, and each process of subtype 30, within the Guardian file system is actually a separate fileset.

The term “file system” is often used interchangeably with “fileset” in UNIX documentation.

file system. In the OSS environment, a collection of files and file attributes. A file system provides the namespace for the file serial numbers that uniquely identify its files. Open System Services provides a file system (see also ISO/IEC IS 9945-1: 1990 [ANSI/IEEE Std. 1003.1-1990], Clause 2.2.2.38); the Guardian application program interface (API) provides a file system; and OSS Network File System (NFS) provides a file system. (OSS NFS filenames and pathnames are governed by slightly different rules than OSS filenames and pathnames.) Within the OSS and OSS NFS file systems, filesets exist as manageable objects.

On an HP NonStop S-series or NonStop NS-series system, the Guardian file system for a node is a subset of the OSS virtual file system. Traditionally, the API for file access in the Guardian environment is referred to as the “Guardian file system.”

In some UNIX and NFS implementations, the term “file system” is used to mean the same thing as “fileset.” That is, a file system is a logical grouping of files that, except for the root of the file system, can be contained only by directories within the file system. See also fileset.

free list. The list of available inodes that can be allocated to files.

Guardian. An environment available for interactive or programmatic use with the HP NonStop operating system. Processes that run in the Guardian environment usually use the Guardian system procedure calls as their application program interface; interactive users of the Guardian environment usually use the HP Tandem Advanced Command Language (TACL) or another HP product's command interpreter. Contrast with Open System Services (OSS).

inode. A data structure that stores the location of a file.

large OSS file. A regular file that has a size greater than or equal to approximately 2 gigabytes. Contrast with small OSS file.

large file safe. An application or function that causes no loss of data or corruption of data when it encounters a large OSS file. A large file safe application or function is not required to process large OSS files with the same ability as when it processes small files, but it must handle errors or warnings detected during file manipulation operations and fail gracefully. Contrast with large file aware.

large file aware. An application or function that can process large OSS files in addition to small OSS files. For example, it must be able to access large files as input and generate large files as output. Contrast with large file safe.

link. A directory entry for a file.

mount. To make a fileset accessible to the users of a node.

mount point. A directory that contains a mounted fileset. The mounted fileset can be in a different file system.

node. A uniquely identified computer system connected to one or more other computer systems in a network.

Open System Services (OSS). An open system environment available for interactive or programmatic use with the HP NonStop operating system. Processes that run in the OSS environment usually use the OSS application program interface; interactive users of the OSS environment usually use the OSS shell for their command interpreter. Synonymous with Open System Services (OSS) environment. Contrast with Guardian.

Open System Services (OSS) environment. The HP NonStop operating system Open System Services (OSS) application program interface (API), tools, and utilities.

Open System Services (OSS) Monitor. A Guardian utility that accepts commands affecting OSS objects through an interactive Guardian interface named the Subsystem Control Facility (SCF).

orphan file. A file with no corresponding inode in the PXINODE file.

orphan inode. An inode that appears in the PXINODE file but has no links in the PXLINK file.

OSS. See Open System Services (OSS).

OSS environment. See Open System Services (OSS) environment.

OSS Monitor. See Open System Services (OSS) Monitor.

pathname. In the OSS environment, the string of characters that uniquely identifies a file within its file system. A pathname can be either relative or absolute. See also ISO/IEC IS 9945-1:1990 (ANSI/IEEE Std. 1003.1-1990 or POSIX.1), Clause 2.2.2.57.

pathname component. See filename.

pathname resolution. In the OSS environment, the process of associating a single file with a specified pathname.

pipe. An unnamed FIFO, created programmatically by invoking the `pipe()` function or interactively with the shell pipe syntax character (`|`). A shell pipe redirects the standard output of one process to become the standard input of another process. A programmatic pipe is an interprocess communication mechanism.

relative pathname. A pathname that does not begin with a slash (`/`) character. A relative pathname is resolved beginning with the current working directory. Contrast with absolute pathname.

root. See root fileset and root directory. See also super ID.

root directory. A directory associated with a process that the system uses for pathname resolution when a pathname begins with a slash (`/`) character.

root fileset. The fileset with the device identifier of 0, normally containing the root directory for the OSS file system in an HP NonStop S-series or NonStop NS-series system. HP recommends that this fileset be named “root”.

root user. The user with the user name of `root`.

saveabend file. A file containing dump information needed by the system debugging tool on a NonStop system. In UNIX systems, such files are usually called core files or core dump files. A saveabend file is a special case of a save file. See also save file.

save file. A file created through the Inspect or Debug product. A save file contains enough information about a running process at a given time to restart the process at the same point in its execution. A save file contains an image of the process, data for the process, and the status of the process at the time the save file was created.

A save file can be created through an Inspect SAVE command at any time. A save file called a saveabend file can be created by the DMON debug monitor when a process’s SAVEABEND attribute is set and the process terminates abnormally.

small OSS file. A regular file that is smaller than approximately 2 gigabytes. Contrast with large OSS file.

storage pool. The set of disk volumes that contain files for a specific fileset or that can be used to contain files for a specific fileset. The storage pool includes the creation pool.

storage-pool file. A file containing a list of disk volumes to be used when creating new files in a fileset. As these volumes are filled, more volumes can be added to the storage-pool file.

superblock. The part of the OSS environment that contains all the information about the current state of the OSS file system. The superblock contains such items as the free list and the size of inodes.

super ID. On HP NonStop S-series or NonStop NS-series systems, a privileged user who can read, write, execute, and purge all files on the system. The super ID is usually a member of a system-supervisor group.

The super ID has the set of special permissions called appropriate privileges. In the Guardian environment, the structured view of the super ID, which is (255, 255), is most commonly used; in the OSS environment, the scalar view of the super ID, which is 65535, is most commonly used.

system. A single copy of the HP NonStop operating system and the collection of hardware groups in which it runs.

target directory location (TDL). The absolute pathname within the target OSS environment where the OSS files for configuration are placed. One TDL can contain files for a single product or multiple products. TDL names are contained in the A7CINFO file for each product and are transmitted to the target system in the activation package instructions.

target subvolume (TSV). The subvolume name of a disk location on the target system where the software files for configuration are placed. One TSV can contain files for a single product or multiple products. TSV names are contained in the A0CINFO file for each product and are transmitted to the target system in the activation package instructions.

TDL. *See* target directory location (TDL).

terminal. A type of character special file that conforms to the interface description in Clause 7 of ISO/IEC IS 9945-1: 1990.

TSV. *See* target subvolume (TSV).

UID. A nonnegative integer that uniquely identifies a user within a node.

The UID is a scalar number consisting of the group number multiplied by 256 and added to the member number; for example, the UID of the super ID is (255*256) + 255, or 65535. The UID is used in the OSS environment for functions normally associated with a UNIX user ID. See also user ID.

unmount. To make a fileset inaccessible to the users of a node.

user ID. The unique identification of a user within a node.

In the Guardian environment, the term “user ID” usually means the group number and member number pair; for example, the user ID of the super ID is usually described as (255, 255). In the OSS environment, the term “user ID” usually refers to the single number that is the scalar view of the user ID—a number called the UID. See also UID.

UTC. *See* Coordinated Universal Time (UTC)

working directory. A directory, associated with a process, that is used in pathname resolution for pathnames that do not begin with a slash (/) character.

Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Special Characters