Sun™ QFS, Sun™ SAM-FS, and Sun™ SAM-QFS File System Administrator’s Guide

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054 U.S.A.
650-960-1300

Part No. 816-2542-10
August 2002, Revision A

Send comments about this document to: `docfeedback@sun.com`

Please
Recycle

Adobe PostScript™

Contents

Preface

This manual, the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS File System Administrator's Guide*, describes the file system software included in the Sun^{TM} QFS, Sun SAM-FS, and Sun SAM-QFS 4.0 releases. The software products and the file systems they include are as follows:

- Sun SAM-FS file system. The Sun SAM-FS environment includes a general purpose file system along with the storage and archive manager, SAM. The Sun SAM-FS environment's file system allows data to be archived to automated libraries at device-rated speeds. In addition, data can also be archived to files in another file system through a process known as *disk archiving*. The file system in the Sun SAM-FS environment is a complete file system. The user is presented with a standard file system interface and can read and write files as though they were all on primary disk storage.
- Sun QFS and Sun SAM-QFS file systems. The Sun QFS file system can be used as a standalone file system or it can be used in conjunction with the storage and archive manager, SAM. When used in conjunction with SAM, it is known as Sun SAM-QFS. Sun QFS shares most of the Sun SAM-FS file system's features. The Sun QFS file system, however, is designed for high performance and contains more features than are supported within the Sun SAM-FS environment.

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems are technologically similar, but within this manual, differences are noted when necessary.

This manual is written for system administrators responsible for installing, configuring, and maintaining Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. You, the system administrator, are assumed to be knowledgeable about Sun Solaris operating environment (OE) procedures, including installation, configuration, creating accounts, performing system backups, and other basic Sun Solaris system administrator tasks.

How This Book Is Organized

This manual contains the following chapters:

- Chapter 1 provides overview information.
- Chapter 2 provides file system design information.
- Chapter 3 provides volume management information.
- Chapter 4 explains how to perform various tasks for the Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. Tasks covered include initializing a file system, adding a server, adding disk cache, and other system administration activities.
- Chapter 5 explains how to use the Sun QFS shared file system.
- Chapter 6 explains how to use the `samu`(1M) operator utility.
- Chapter 7 explains how to use file system quotas.
- Chapter 8 explains miscellaneous advanced topics such as striping the `.inodes` file, using the SAN-QFS file system, and performance features.

The glossary defines terms used in this and other Sun QFS, Sun SAM-FS, and Sun SAM-QFS documentation.

Related Documentation

This manual is part of a set of documents that describes the operations of the Sun QFS, Sun SAM-FS, and Sun SAM-QFS software products. TABLE P-1 shows the complete release 4.0 documentation set for these products.

TABLE P-1 Related Documentation

Title	Part Number
Sun SAM-Remote Administrator's Guide	816-2094
Sun QFS, Sun SAM-FS, and Sun SAM-QFS Disaster Recovery Guide	816-2540
Sun QFS, Sun SAM-FS, and Sun SAM-QFS File System Administrator's Guide	816-2542
Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide	816-2543
Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide	816-2544
Sun QFS, Sun SAM-FS, and Sun SAM-QFS README File	816-7675

Note that the *Sun SAM-Remote Administrator's Guide* has not been updated for the 4.0 release. An updated version of this manual will be provided at a later date.

Accessing Sun Documentation Online

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS software distribution includes PDF files for the documents for these products. These PDF files can be viewed from the following locations:

1. **At Sun's Network Storage documentation website.**

   This website contains documentation for many storage software products.

   a. **To access this website, go to the following URL:**

`www.sun.com/products-n-solutions/hardware/docs/Software/Storage_Software`

   The Storage Software page displays.

   b. **Click on the appropriate link from the following list:**

   Sun QFS Software

   Sun SAM-FS and Sun SAM-QFS Software

2. **At `docs.sun.com`.**

   This website contains documentation for Solaris and many other Sun software products.

   a. **To access this website, go to the following URL:**

   `docs.sun.com`

   The `docs.sun.com` page displays.

   b. **Find the documentation for your product by searching for one of the following in the search box:**

   - Sun QFS
   - Sun SAM-FS
   - Sun SAM-QFS

Viewing PDF files requires the Acrobat Reader software, which is available for free from the following website:

`www.adobe.com/products/acrobat/readstep.html`

Licensing

For information on obtaining licenses for Sun QFS, Sun SAM-FS, or Sun SAM-QFS software, contact your Sun sales representative or your authorized service provider (ASP).

Diagnostics

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS software includes the `info.sh`(1M) script. This diagnostic script can be very useful to you and to the Sun customer support staff. This script produces a diagnostic report of the server configuration and collects log information. After the software is installed, you can access the `info.sh`(1M) man page for more information about this script.

Installation Assistance

For installation and configuration services please contact Sun's Enterprise Services at 1-800-USA4SUN or contact your local Enterprise Services sales representative.

Using UNIX Commands

This document does not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices.

See one or more of the following for this information:

- *Solaris Handbook for Sun Peripherals*
- AnswerBook2™ online documentation for the Sun Solaris OE
- Other software documentation that you received with your system

Typographic Conventions

TABLE P-2 lists the typographic conventions used in this manual.

TABLE P-2 Typographic Conventions

Typeface or Symbol	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output.	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail`.

`AaBbCc123`	What you type, when contrasted with on-screen computer output.	`%` **`su`**
		`Password:`

AaBbCc123	Book titles; new words or terms; words to be emphasized; and command line variables to be replaced with a real name or value.	Read Chapter 6 in the *User's Guide.* These are called *class* options. You *must* be `root` to do this. To delete a file, type `rm` *filename.*
[ ]	In syntax, brackets indicate that an argument is optional.	`scmadm` [`-d` *sec*] [`-r` *n*[:*n*][,*n*]...] [`-z`]
{ *arg* | *arg*}	In syntax, braces and pipes indicate that one of the arguments must be specified.	`sndradm -b` {*phost* | *shost*}

\	At the end of a command line, the backslash (\) indicates that the command continues on the next line.	`atm90 /dev/md/rdsk/d5 \`
		`/dev/md/rdsk/d1 atm89`

Shell Prompts

TABLE P-3 shows the shell prompts that this manual uses.

TABLE P-3 Shell Prompts

Shell	**Prompt**
C shell	*machine-name*%
C shell superuser	*machine-name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can email your comments to Sun using the following email address:

`docfeedback@sun.com`

Please include the part number (816-2542-10) of your document in the subject line of your email.

CHAPTER **1**

Overview

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems are configurable file systems that present a standard UNIX file system interface to users. TABLE 1-1 shows how these file systems can be used or combined with the storage and archive management (SAM) software.

TABLE 1-1 Product Overview

Product	Components
Sun QFS	Sun QFS standalone file system
Sun SAM-QFS	Sun QFS file system plus the storage and archive management utility, SAM
Sun SAM-FS	Standard file system plus the storage and archive management utility, SAM

While technologically similar, there are differences between each file system. This chapter presents an overview of the features common to these file systems, highlights the features that differentiate the file systems, and explains the commands available with each file system. Specifically, this chapter is divided into the following sections:

- "Common Features" on page 1
- "File System Differences" on page 4
- "Commands" on page 6

Common Features

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems do not require changes to user programs, nor are changes required to the UNIX kernel. These file systems share the features described in the following sections.

`vnode` Interface

The Sun QFS, Sun SAM-FS and Sun SAM-QFS file systems are implemented using the standard Sun Solaris operating environment (OE) virtual file system (`vfs/vnode`) interface.

By using the `vfs/vnode` interface, these file systems work with the standard Sun Solaris kernel and require no modifications to the kernel for file management support. Thus, the file system is protected from operating system changes and typically does not require extensive regression testing when the operating system is updated.

The kernel intercepts all requests for files, including those that reside in Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. If the file is identified as a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file, the request is passed to the file system. The file system handles all requests for files. Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems are identified as type `samfs` in the `/etc/vfstab` file and on the `mount`(1M) command.

Enhanced Volume Management

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support both striped and round-robined disk access. The master configuration file (`mcf`) and the mount parameters specify the volume management features and let the file system know the relationships between the devices it controls. This is in contrast to most UNIX file systems that can address only one device or one portion of a device. Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems do not require any additional volume management applications. An additional package, such as a logical volume manager, is needed if you want to use mirroring.

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS integrated volume management features use the standard Sun Solaris device driver interface to pass I/O requests to and from the underlying devices. The Sun QFS, Sun SAM-FS, or Sun SAM-QFS software groups storage devices into family sets upon which each file system resides.

Support for Paged and Direct I/O

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support two different types of I/O: paged (also called cached or buffered I/O) and direct.

When paged I/O is used, user data is cached in virtual memory pages and then written to disk by the Sun Solaris Virtual Memory Manager (`vm`). The standard Sun Solaris interfaces manage paged I/O. This is the default type of I/O.

When direct I/O is used, user data is written directly to disk. Direct I/O can be specified by using the Sun Solaris `directio`(3C) function call or the `setfa`(1) command with its `-D` option. Large block, sequential, aligned I/O can realize substantial performance improvements by using direct I/O.

Preallocation of file space

You can use the `setfa`(1) command to preallocate contiguous disk space for fast sequential reads and writes.

Application Programmer Interface Routines

The application programmer interface (API) routines enable a program to perform various specialized functions such as the ability to preallocate contiguous disk space or to access a specific striped group. For more information on these routines, see the `intro_libsam`(3) man page.

Unlimited Capacity

You are virtually unlimited with regard to file size, the number of files that can reside in a file system, and the number of file systems that you can specify.

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support files of up to 2^{63} bytes in length. Such very large files can be striped across many disks or RAID devices, even within a single file system. This is true because Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems use true 64-bit addressing. This is in contrast to a UFS file system, which is not a true 64-bit file system.

The number of file systems you can configure is virtually unlimited. The volume manager allows each file system to include up to 252 device partitions (typically disk). Each partition can include up to 1 terabyte of data. This configuration offers virtually unlimited storage capacity.

There is no predefined limit for the number of files on a Sun SAM-FS file system. Because the inode space (which holds information about the files) is dynamically allocated, the maximum number of files is limited only by the amount of disk storage comprising the file system. The inodes are cataloged in the `.inodes` file under the mount point. The `.inodes` file requires 512 bytes of storage per file.

For a Sun QFS or Sun SAM-QFS file system, the inodes are located on the metadata device(s) and are separated from the file data devices. The number of files in these file systems is limited by the size of the metadata (`mm`) devices, but you can increase the number of files by adding more metadata devices.

Fast File System Recovery

A key function of a file system is the ability to recover quickly after an unscheduled outage. Standard UNIX file systems require a lengthy file system check (`fsck`(1M)) to repair inconsistencies after a system failure.

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems often do not require file system checks after a disruption that prevents the file system from being written to to disk (using `sync`(1M)). In addition, they recover from system failures without using journaling. They accomplish this dynamically by using identification records, serial writes, and error checking for all critical I/O operations. After a system failure, Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems can be remounted immediately, even for multiterabyte-sized file systems.

Adjustable Disk Allocation Unit (DAU)

The DAU is the basic unit of online storage. The Sun QFS and Sun SAM-QFS file systems include an adjustable DAU, which is useful for tuning the file system with the physical disk storage device and for eliminating the system overhead caused by read-modify-write operations.

File System Differences

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems share the features described in “Common Features” on page 1. This section, however, describes the areas in which they differ. One area of difference is performance. The Sun QFS and Sun SAM-QFS file systems provide the ability to attain raw, device-rated disk speeds with the administrative convenience of a file system. The following sections note other ways in which the file systems differ.

Metadata Storage

File systems use metadata to reference file and directory information. Typically, metadata resides on the same device as the file data. This is true for the Sun SAM-FS file system.

The Sun QFS and Sun SAM-QFS file systems separate the file system metadata from the file data by storing them on separate devices. The Sun QFS and Sun SAM-QFS file systems enable you to define one or more separate metadata devices in order to reduce device head movement and rotational latency, improve RAID cache utilization, or mirror metadata without mirroring file data.

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems store inode metadata information in a separate file. This enables the number of files, and the file system as a whole, to be enlarged dynamically.

Support for Multiple Striped Groups

To support multiple RAID devices in a single file system, striped groups can be defined in Sun QFS and Sun SAM-QFS file systems. Disk block allocation can be optimized for a striped group, reducing the overhead for updating the on-disk allocation map. Users can assign a file to a striped group either through an API routine or by using the `setfa`(1) command.

SAM Interoperability

The Sun SAM-FS and Sun SAM-QFS file systems combine file system features with the storage and archive management utility, SAM. Users can read and write files directly from magnetic disk, or they can access archive copies of files as though they were all on primary disk storage. The Sun QFS file system is a standalone file system, however, and it does not interoperate with SAM.

When possible, Sun SAM-FS and Sun SAM-QFS products use the standard Sun Solaris disk and tape device drivers. For devices not directly supported under the Sun Solaris OE, such as certain automated library and optical disk devices, Sun Microsystems provides special device drivers in the Sun SAM-FS and Sun SAM-QFS software packages.

Sun QFS Shared File System Support

The Sun QFS shared file system can be implemented in either a Sun QFS environment or in a Sun SAM-QFS environment. This file system enables you to implement a distributed file system that can be mounted on multiple Sun Solaris host systems.

Unlike Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems configured without the Sun QFS shared file system, file systems created as Sun QFS shared file systems do not support the following file types:

- `b`, blocked special files
- `c`, character special files
- `p`, FIFO (named pipe) special files

For more information on this file system, see the "Sun QFS Shared File System" on page 89.

Commands

Specialized Sun QFS, Sun SAM-FS, and Sun SAM-QFS file system commands are included in the Sun QFS, Sun SAM-FS, and Sun SAM-QFS environments. These commands operate in conjunction with the standard UNIX file system commands. Some commands are specific to only one or two of these environments. All the commands are documented in UNIX `man`(1) pages.

The following subsections show the commands supported within each environment.

User Commands

By default, file system operations are transparent to the end user. Depending on your site practices, however, you might want to make some commands available to users at your site to fine-tune certain operations.

TABLE 1-2 summarizes these commands.

TABLE 1-2 User Commands

Command	Description	Used By
`archive`(1)	Archives files and sets archive attributes on files.	Sun SAM-FS, Sun SAM-QFS
`release`(1)	Releases disk space and sets release attributes on files.	Sun SAM-FS, Sun SAM-QFS
`request`(1)	Creates a removable media file.	Sun SAM-FS, Sun SAM-QFS
`sdu`(1)	Summarizes disk usage. The `sdu`(1) command is based on the GNU version of the `du`(1) command.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`segment`(1)	Sets segmented file attributes.	Sun SAM-FS, Sun SAM-QFS
`setfa`(1)	Sets file attributes.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`sfind`(1)	Searches for files in a directory hierarchy. The `sfind`(1) command is based on the GNU version of the `find`(1) command and contains options for displaying file system options.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`sls`(1)	Lists contents of directories. The `sls`(1) command is based on the GNU version of the `ls`(1) command and contains options for displaying file system attributes and information.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`squota`(1)	Reports quota information.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`ssum`(1)	Sets the checksum attributes on files.	Sun SAM-FS, Sun SAM-QFS
`stage`(1)	Sets stage attributes on files and copies offline files to disk.	Sun SAM-FS, Sun SAM-QFS

General System Administrator Commands

TABLE 1-3 summarizes the commands that provide system management and maintenance capabilities.

TABLE 1-3 General System Administrator Commands

Command	Description	Used By
`samcmd`(1M)	Executes one `samu`(1M) operator interface utility command.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samd`(1M)	Starts or stops robotic and removable media daemons.	Sun SAM-FS, Sun SAM-QFS
`samset`(1M)	Changes Sun SAM-FS or Sun SAM-QFS settings.	Sun SAM-FS, Sun SAM-QFS
`samu`(1M)	Invokes the full-screen, text-based operator interface. This interface is based on the `curses`(3X) software library. The `samu` utility displays the status of devices and allows the operator to control automated libraries.	Sun QFS, Sun SAM-FS, Sun SAM-QFS

File System Commands

TABLE 1-4 summarizes the file system commands. These are used to perform file system maintenance operations.

TABLE 1-4 File System Commands

Commands	Description	Used By
`mount`(1M)	Mounts a file system. The man page name for this command is `mount_samfs`(1M).	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`qfsdump`(1M) `qfsrestore`(1M)	Creates or restores a dump file containing the file data and metadata associated with a Sun QFS file system.	Sun QFS
`sambcheck`(1M)	Lists block usage for a file system.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samchaid`(1M)	Changes file admin set ID attribute. For use with quotas.	Sun QFS, Sun SAM-FS, Sun SAM-QFS

TABLE 1-4 File System Commands *(Continued)*

Commands	Description	Used By
`samfsck`(1M)	Checks and repairs metadata inconsistencies in a file system and reclaims allocated, but unused, disk space.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samfsconfig`(1M)	Displays configuration information.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samfsdump`(1M) `samfsrestore`(1M)	Creates or restores a dump file of the metadata associated with a Sun SAM-FS or Sun SAM-QFS file system.	Sun SAM-FS, Sun SAM-QFS
`samfsinfo`(1M)	Displays information about the layout of a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samgrowfs`(1M)	Expands a file system by adding disk devices.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`sammkfs`(1M)	Initializes a new file system from disk devices.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samncheck`(1M)	Returns a full directory path name given the mount point and inode number.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samquota`(1M)	Reports, sets, or resets quota information.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samquotastat`(1M)	Reports on active and inactive file system quotas.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samsharefs`(1M)	Manipulates the Sun QFS shared file system configuration information.	Sun QFS, Sun SAM-QFS
`samtrace`(1M)	Dumps the trace buffer.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`samunhold`(1M)	Releases SANergy file holds.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`trace_rotate.sh`(1M)	Rotates trace files.	Sun QFS, Sun SAM-FS, Sun SAM-QFS

Additional Commands and APIs

Sun Microsystems also provides the following additional types of commands for use in the Sun SAM-FS and Sun SAM-QFS environments:

- Automated library commands
- Archiver, stager, releaser, and recycler commands
- Specialized maintenance commands
- Operational utility commands

The preceding commands are described on individual man pages and in the *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

In addition to the preceding commands, Sun Microsystems provides an application programmer interface (API). The API enables file system requests to be made from within a user application. The requests can be made locally or remotely to the machine upon which the file system is running. The API consists of the `libsam` and `libsamrpc` libraries. These libraries contain library routines for obtaining file status; for setting archive, release, and stage attributes for a file; and for manipulating the library catalog of an automated library. The `sam-rpcd` server process handles remote requests.

For more information on the API, see the `intro_libsam`(3) or `intro_libsam`(3X) man pages. These man pages provide overview information for using the library routines in `libsam` and `libsamrpc`.

CHAPTER 2

File System Design

File system design is critical to ensuring quick and uninterrupted access to information. Good design is also essential to file system recovery, when necessary.

This chapter presents the following topics to consider when configuring a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system:

Design Basics

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems are multithreaded, advanced storage management systems. To take maximum advantage of these capabilities, create multiple file systems whenever possible.

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems use a linear search method when performing directory lookups. They search from the beginning of the directory to the end. As the number of files in a directory increases, the search time through the directory also increases. Users who have directories with thousands of files can experience excessive search times. These search times are also evident when you restore a file system. To increase performance and speed up file system dumps and restores, you should keep the number of files in a directory under 10,000.

Inode Files and File Characteristics

The types of files to be stored in a file system affect file system design. An *inode* is a 512-byte block of information that describes the characteristics of a file or directory. This information is allocated dynamically within the file system.

The inodes are stored in the `.inodes` file located under the file system mount point. A Sun SAM-FS `.inodes` file resides on the same physical device as the file data and is interleaved with the file data. In contrast, a Sun QFS or Sun SAM-QFS `.inodes` file resides on a metadata device that is separate from the file data device.

Like a standard Sun Solaris operating environment (OE) inode, a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system inode contains the file's POSIX standard inode times: file access, file modification, and inode changed times. The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems add a creation time, attribute change time, and a residence time. TABLE 2-1 summarizes the times that are recorded in the inode.

TABLE 2-1 Content of `.inode` Files

Time	Incident
`access`	Time the file was last accessed. POSIX standard.
`modification`	Time the file was last modified. POSIX standard.
`changed`	Time the inode information was last changed. POSIX standard.
`attributes`	Time the attributes specific to the Sun QFS, Sun SAM-FS, or Sun SAM-QFS files systems were last changed. Sun Microsystems extension.
`creation`	Time the file was created. Sun Microsystems extension.
`residence`	Time the file changed from offline to online or vice-versa. Sun Microsystems extension.

The attributes specific to the Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems include both user settings and general file states. The following two sections describe these characteristics.

File Attributes and File States

A file's user-specified attributes and its system-specified states are stored in the file's inode. These inode attributes can be displayed by using the `sls`(1) command with its `–D` option. For more information on `sls`(1) options, see the `sls`(1) man page.

A user can set attributes with the following user commands:

- `archive`(1)
- `ssum`(1)
- `release`(1)
- `segment`(1)
- `setfa`(1)
- `stage`(1)

Users can also set attributes from within an application by using the following application programmer interface (API) routines:

- `sam_archive`(3)
- `sam_release`(3)
- `sam_segment`(3)
- `sam_setfa`(3)
- `sam_ssum`(3)
- `sam_stage`(3)

User-Specified File Attributes

TABLE 2-2 shows the user-specified attributes that are listed in the inode.

TABLE 2-2 User-Specified File Attributes

Command	**Definition**	**Used By**
`archive –c`	The file is marked for concurrent archiving. This means that the file can be archived even if it is open for a write operation. This attribute can be set by using the `archive`(1) command.	Sun SAM-FS, Sun SAM-QFS
`archive –n`	The file is marked to never be archived. This attribute can be set by the superuser with the `archive`(1) command.	Sun SAM-FS, Sun SAM-QFS
`release –a`	This file is marked to be released as soon as one archive copy is made. This attribute can be set from within the `archiver.cmd` file or by using the `release`(1) command.	Sun SAM-FS, Sun SAM-QFS
`release –n`	This file is marked to never be released. This attribute can be set from within the `archiver.cmd` file or it can be set by the superuser using the `release`(1) command.	Sun SAM-FS, Sun SAM-QFS

TABLE 2-2 User-Specified File Attributes *(Continued)*

Command	**Definition**	**Used By**
`release –p`	The file is marked for partial release. This attribute can be set from within the `archiver.cmd` file or by using the `release`(1) command.	Sun SAM-FS, Sun SAM-QFS
`stage –a`	The file is marked for associative staging. This attribute can be set from within the `archiver.cmd` file or by using the `stage`(1) command.	Sun SAM-FS, Sun SAM-QFS
`stage –n`	The file is marked to never be staged. This signifies direct access to removable media cartridges. This attribute can be set from within the `archiver.cmd` file or it can be set by the superuser using the `stage`(1) command.	Sun SAM-FS, Sun SAM-QFS
`setfa -D`	The file is marked for direct I/O.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`setfa -g`*n*	The file is marked for allocation on striped group *n*.	Sun QFS, Sun SAM-QFS
`setfa -s`*m*	The file is marked for allocation with a stripe width of *m*.	Sun QFS, Sun SAM-FS, Sun SAM-QFS
`segment` *n*`m stage_ahead` *x*	The file is marked for segmentation. The *n*`m` notation indicates that the segment is *n* megabytes in size. The `stage_ahead` *x* attribute indicates the number of attributes (*x*) to be staged ahead. You can set these attributes by using the `segment`(1) command.	Sun SAM-FS, Sun SAM-QFS

All the preceding attributes can be set on directories, too. After directory attributes are set, files that are created in the directory inherit all the directory attributes at the time of creation. Files created before an attribute is applied to the parent directory do not inherit directory attributes.

Users can gather information on file attributes by using the `sls`(1) command, which is described in “Displaying File Information” on page 15.

System-Specified File States

TABLE 2-3 shows the various states that the file systems set for a file. These states are stored in the inode.

TABLE 2-3 System-Specified File States

Attribute	Definition	Used By
`archdone`	Indicates that the file's archive requirements have been met. There is no more work the archiver must do on the file. Note that `archdone` does not necessarily indicate that the file has been archived. This attribute is set by the archiver and cannot be set by a user.	Sun SAM-FS, Sun SAM-QFS
`damaged`	The file is damaged. This attribute is set by the stager or by the `samfsrestore`(1M) command. You can use the `undamage`(1M) command to reset this attribute to undamaged. If this attribute has been set by the `samfsrestore`(1M) utility, it means that no archive copies existed for the file at the time a `samfsdump`(1M) was taken. You can reset this attribute to undamaged, but the file might still be unrecoverable.	Sun SAM-FS, Sun SAM-QFS
`offline`	The file data has been released. This attribute is set by the releaser. This attribute can also be set by using the `release`(1) command.	Sun SAM-FS, Sun SAM-QFS

Users can gather information on file states by using the `sls`(1) command, which is described in "Displaying File Information" on page 15.

Displaying File Information

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS `sls`(1) command extends the standard UNIX `ls`(1) command and provides more information about a file. CODE EXAMPLE 2-1 shows detailed `sls`(1) command output that displays the inode information for file `hgc2`.

CODE EXAMPLE 2-1 `sls`(1) Output in a Sun SAM-QFS Environment

```
hgc2:
  mode: -rw-r--r--  links:   1  owner: root      group: other
  length:     14971  admin id:      0  inode:       30.5
  archdone;
  segments 3, offline 0, archdone 3, damaged 0;
  copy 1: ---- Jun 13 17:14     2239a.48   lt MFJ192
```

CODE EXAMPLE 2-1 `sls`(1) Output in a Sun SAM-QFS Environment *(Continued)*

```
   copy 2: ---- Jun 13 17:15      9e37.48   lt AA0006
   access:      Jun 13 17:08  modification: Jun 13 17:08
   changed:     Jun 13 17:08  attributes:   Jun 13 17:10
   creation:    Jun 13 17:08  residence:    Jun 13 17:08
```

TABLE 2-4 describes the meaning of each row of `sls`(1) output shown in CODE EXAMPLE 2-1. In TABLE 2-4, note that lines that pertain to archiving do not appear in `sls`(1) output in a Sun QFS environment.

TABLE 2-4 `sls`(1) Output Explanation

Line Number	First Few Characters	Content
1	`mode:`	The file's mode and permissions, the number of hard links to the file, the owner of the file, and the group to which the owner belongs.
2	`length:`	The file's length in bytes, the file's admin ID number, and the file's inode number. By default, the admin ID number is 0. If this number is greater than 0, it indicates the file's accounting category for counting files and blocks. This number can be set to a value greater than 0 even when file system quotas are not enabled on this file system. For information on file system quotas, see "File System Quotas" on page 197. The inode number is a two-part number that contains the inode number itself, followed by a period (.), followed by the inode generation number.
3	`archdone;`	The file attributes specific to the file. For more information on this line, see the `sls`(1) man page.
4	`segments`	The segment index information. This line does not appear unless the file is a segment index. The general format for this line is as follows: `segments` *n*`, offline` *o*`, archdone` *a*`, damaged` *d*`;` This line indicates that there are 3 data segments. There are zero (0) data segments offline. There are 3 data segments that have met their archiving requirements. The number of damaged data segments is zero (0).
5	`copy 1:`	The first archive copy line. One archive copy line is displayed for each active or expired archive copy. For more information on this, see "Archive Copy Line Explanation" on page 17.
6	`copy 2:`	The second archive copy line. For more information on this, see "Archive Copy Line Explanation" on page 17.

TABLE 2-4 `sls`(1) Output Explanation *(Continued)*

Line Number	First Few Characters	Content
7	`access:`	The time since the file was last accessed and modified.
8	`changed:`	The time since the file content was last changed and since the file's attributes were last changed.
9	`creation:`	The time since the file was created and since the file became resident in the file system.

Archive Copy Line Explanation

The fields in the archive copy lines are as follows:

- The first field indicates the archive copy number.
- The second field contains 4 indicators, each of which is either a dash (-) or a letter. Reading them from left to right, TABLE 2-5 shows the information that the indicators convey.

TABLE 2-5 Archive Copy Line Indicators

Position	Meaning
1	Indicates either an expired or active entry. An `S` indicates that the archive copy is expired. That is, the file was modified and this archive copy is a previous version of the file. A `U` indicates that the copy has been unarchived. *Unarchiving* is the process by which archive entries for files or directories are deleted. A dash (-) indicates that the archive copy is active and valid.
2	Indicates whether the archive copy is to be rearchived. An `r` indicates that the archive copy is scheduled to be rearchived by the archiver. A dash (`-`) indicates that the archive copy is not to be rearchived by the archiver.
3	Unused.
4	Indicates whether the copy is damaged or undamaged. A `D` indicates that the archive copy is damaged. The archive copy is not a candidate for staging. A dash (-) indicates that the archive copy is not damaged. It is a candidate for staging.

- The third field shows the date and time the archive copy was written to the archive media.

- The fourth field contains two hexadecimal numbers separated by a decimal point (.). The first hexadecimal number (2239a) indicates the position of the beginning of the archive file on the cartridge. The second hexadecimal number (48) is the file byte offset (divided by 512) of this copy in the archive file.
- The fifth and sixth fields in the archive copy line indicate the media type and the Volume Serial Name (VSN) where the archive copy resides.

Checksum Line Explanation

If a file has checksum-related attributes, the sls(1) command returns a checksum line. These attributes (generate, use, or valid) are set by using the ssum(1) command. This line appears in sls(1) output in Sun SAM-FS and Sun SAM-QFS environments. The format of the checksum line is as follows:

```
checksum: gen  use  val  algo:  1
```

The preceding line is displayed if checksum attributes are set for a file. If the generate attribute is not set, no_gen appears in place of gen. Similarly, if the use attribute is not set, no_use appears. val is displayed when the file has been archived and a checksum has been computed. If the file has not been archived or if no checksum has been computed, not_val appears. The keyword algo precedes the numeric algorithm indicator that specifies the algorithm that is used to generate the checksum value.

Specifying Disk Allocation Units and Stripe Widths

Disk space is allocated in blocks. These are also called *disk allocation units* (DAUs), which are the basic unit of online disk storage. While sectors, tracks, and cylinders describe the physical disk geometry, the DAU describes the file system geometry. The appropriate DAU setting and stripe can improve performance and improve magnetic disk usage. The DAU setting is the minimum amount of contiguous space that is used when a file is written.

Example. Assume that you have a Sun SAM-FS file system. Your DAU is set to 16 kilobytes and you have disabled striping by setting stripe=0. You are using round robin allocation (because of the stripe=0 setting), and you have 2 files, as follows:

- The first file is a 15-kilobyte file. It occupies 1 DAU. The file data occupies 15 kilobytes of the DAU, and the other 1 kilobyte is not used.
- The second file is a 20-kilobyte file. It occupies 2 DAUs. The file data occupies all 16 kilobytes of the first DAU, and 4 kilobytes of the second DAU. The second DAU contains 12 kilobytes that are not used.

The DAU setting is specified by the `–a` *allocation_unit* option on the `sammkfs`(1M) command.

If striped allocation is used, the stripe width mount option determines the maximum number of DAUs written in one I/O event. This setting is specified by the `–o stripe=`*n* option on the `mount`(1M) command. You must run the `sammkfs`(1M) command before you run the `mount`(1M) command.

The following sections describe how DAU settings and stripe widths can be configured.

DAU Settings and File System Geometry

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems use an adjustable DAU. This adjustable DAU is useful for tuning the file system with the physical disk storage device. This eliminates the system overhead caused by the read-modify-write operation. Applications that manipulate very large files can benefit substantially from this feature. For an example that shows how to control the read-modify-write operation, see the `mount_samfs`(1M) man page's description of the `–o writebehind=`*n* option and the `EXAMPLES` section.

Each file system can have its own unique DAU setting. Thus, several mounted file systems can be active on a server, each with a different DAU setting. The DAU setting is determined when the file system is created using the `sammkfs`(1M) command. It cannot be changed dynamically.

The possible DAU settings differ depending on the file system you are using. The following sections describe the DAU settings for each file system. These sections also introduce the concept of the master configuration file (`mcf` file). You create this ASCII file at system configuration time. It defines the devices and file systems used in your Sun QFS, Sun SAM-FS, or Sun SAM-QFS environment. The `mcf` file is introduced in the following sections, but it is more thoroughly discussed in “Volume Management” on page 41.

Two file allocation schemes are available to you. The following sections describe these schemes.

Dual Allocation Scheme

File systems that use `md` devices use a dual allocation scheme.

A Sun SAM-FS file system is defined as Equipment Type `ms` in your `mcf` file. The only device type allowed in a Sun SAM-FS file system is type `md`. Both metadata and file data are written to the `md` devices in a Sun SAM-FS file system. The `md` device type is a dual allocation device type. By default, the DAU on an `md` device is 16 kilobytes.

A Sun QFS or Sun SAM-QFS file system is defined as Equipment Type `ma` in your `mcf` file. In Sun QFS and Sun SAM-QFS file systems, data devices can be defined as `md`, `mr`, or `g`*XXX*. You can mix `mr` and `g`*XXX* devices in a file system, but you cannot mix `md` devices with either `mr` or `g`*XXX* devices in a file system. The `mr` and `g`*XXX* single allocation data device types are described in "Single Allocation Scheme" on page 20.

- In file systems that use `md` data devices, the small allocation is 4 kilobytes and the large allocation is a DAU (disk allocation unit). The default DAU is 64 kilobytes. You can override this default when the file system is initialized by using the `-a` *allocation_unit* option to the `sammkfs`(1M) command. The DAU size can be either 16, 32, or 64 kilobytes.

  When a file is created, file systems that use `md` devices allocate the first eight addresses of a file in the small allocation. If more space is needed, the file system uses one or more large allocations (a DAU) in expanding the file. As a result, I/O performance improves for large files while minimizing the disk fragmentation that can result from having many small files.

- The `mm` metadata devices use a dual allocation scheme. The small allocation is 4 kilobytes, and the large allocation is 16 kilobytes. The dual allocation scheme allows metadata to be written to disk more efficiently and helps minimize disk fragmentation.

Depending on the type of file data stored in the file system, selecting a larger DAU size can improve file system performance significantly. For information on tuning file system performance, see "Advanced Topics" on page 223.

Single Allocation Scheme

Only Sun QFS and Sun SAM-QFS file systems can include devices that use a single allocation scheme.

The Sun QFS and Sun SAM-QFS file systems are Equipment Type `ma` in your `mcf` file. These file systems consist of separate metadata devices and data devices.

- The metadata devices can be defined only as Equipment Type `mm`.

- The data devices can be defined as Equipment Type md, mr, or g*XXX*. The md devices follow the dual allocation scheme of a Sun SAM-FS file system and are limited to DAU sizes of 16 kilobytes, 32 kilobytes, or 64 kilobytes.

  The mr and g*XXX* devices follow a single allocation scheme. You can mix mr and g*XXX* devices in a file system, but you cannot mix md devices with either mr or g*XXX* devices in a file system.

The DAU size for Sun QFS file systems that use mr and g*XXX* data devices is configurable. The possible DAU sizes that can be used on data devices depend on the Equipment Type assigned to each data device in the mcf file. TABLE 2-6 shows these DAU sizes.

TABLE 2-6 Sun QFS or Sun SAM-QFS Equipment Types

Equipment Type	DAU Sizes
mr or g*XXX*	You can specify different DAU sizes by adjusting the default size in 8 kilobyte increments. The DAU size can be anywhere from 16 kilobytes to 65,528 kilobytes (64 megabytes). The default DAU for an mr or g*XXX* device in a Sun QFS or Sun SAM-QFS environment is 64 kilobytes.
md	This type of device uses a dual allocation in the style of a Sun SAM-FS file system. The DAU can be configured to be 16, 32, or 64 kilobytes in length. The default DAU for an md device in a Sun QFS or Sun SAM-QFS environment is 64. An md device in a Sun QFS or Sun SAM-QFS file system is used to store data only, not metadata. This is the difference between an md device in a Sun QFS or Sun SAM-QFS file system versus an md device in a Sun SAM-FS file system.

Note – If you did not perform a sammkfs(1M) on your file system when the Sun QFS or Sun SAM-QFS 4.0 software was installed, you are using a version 1 superblock. In the version 1 superblock, the mm devices do not use the dual allocation scheme. In the version 1 superblock, the allocation for mm devices is 16 kilobytes. Only a version 2 superblock allows you to define md devices in a Sun QFS or Sun SAM-QFS file system.

The DAU setting is specified using the -a *allocation_unit* option to the sammkfs(1M) command. The following command specifies a DAU of 128 kilobytes:

```
# sammkfs -a 128 samqfs1
```

For more information on the sammkfs(1M) command, see the sammkfs(1M) man page.

Allocation Scheme Summary

TABLE 2-7 shows the Equipment Types that can be used in Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems.

TABLE 2-7 Equipment Types for File System Devices

Equipment Types in `mcf` File	Type of Data Stored	File Systems That Can Include the Equipment Type
`md`	File data and metadata	Sun SAM-FS
`md`	File data	Sun QFS and Sun SAM-QFS
`mm`	Metadata	Sun QFS and Sun SAM-QFS
`mr`	File data	Sun QFS and Sun SAM-QFS
`g`*XXX*	File data	Sun QFS and Sun SAM-QFS

TABLE 2-8 summarizes the allocation schemes used by the various file systems.

TABLE 2-8 File Allocation

File System and Device Type	Allocation Increments
Sun SAM-FS with `md` devices	Up to 8 4-kilobyte blocks, then DAUs
Sun QFS and Sun SAM-QFS with `mr` devices	DAUs
Sun QFS and Sun SAM-QFS with `g`*X* devices	DAUs
Sun QFS and Sun SAM-QFS with `md` devices	Up to 8 4-kilobyte blocks, then DAUs

TABLE 2-9 summarizes the DAU defaults.

TABLE 2-9 Default DAU Sizes

File System and Device Types	Default DAU Size
Sun SAM-FS `md` devices	16 kilobytes
Sun QFS and Sun SAM-QFS `mr` and `md` devices	64 kilobytes
Sun QFS and Sun SAM-QFS `g`*X* devices	256 kilobytes

Stripe Widths on Data Disks

Stripe width defaults differ between Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. The stripe width is specified by the `-o stripe=`*n* option on the `mount`(1M) command. If the stripe width is set to 0, round-robin allocation is used.

The following sections explain the differences that affect stripe widths on the various file systems.

Sun SAM-FS Stripe Widths

On Sun SAM-FS file systems, the stripe width is set at mount time. TABLE 2-10 shows default stripe widths.

TABLE 2-10 Sun SAM-FS Default Stripe Widths

DAU	Default Stripe Width	Amount of Data Written to 1 Disk
16 kilobytes (default)	8	128 kilobytes
32 kilobytes	4	128 kilobytes
64 kilobytes	2	128 kilobytes

For example, if `sammkfs`(1M) is run with default settings, the default large DAU is 16 kilobytes. If no stripe width is specified when the `mount`(1M) command is issued, the default is used, and the stripe width set at mount time is 8.

Note that if you multiply the number in the first column of TABLE 2-10 by the number in the second column, the resulting number is 128 kilobytes. The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems operate more efficiently if the amount of data being written to disk is at least 128 kilobytes.

Sun QFS and Sun SAM-QFS Stripe Widths – Not Using Striped Groups

On Sun QFS and Sun SAM-QFS file systems, the stripe width that is set at mount time depends on whether or not striped groups are configured. A *striped group* is a collection of devices that are striped as a group. For more information on striped groups, see “File Allocation Methods” on page 26. This section describes stripe widths for Sun QFS and Sun SAM-QFS without stripe groups.

If striped groups are not configured, the DAU and stripe width relationships are similar to those for Sun SAM-FS file systems. The differences being that DAUs larger than 64 kilobytes or greater are possible and that the DAU is configurable in 8-kilobyte blocks. The maximum DAU size is 65528 kilobytes.

By default, if no stripe width is specified, the amount of data written to disk is at or near 128 kilobytes. The Sun QFS and Sun SAM-QFS file systems are more efficient if write operations write at least one whole stripe per I/O request. TABLE 2-11 shows the default stripe widths. These are the widths used if you do not specify a stripe width.

TABLE 2-11 Default Stripe Widths

DAU	Default Stripe Width	Amount of Data Written to 1 Disk
16 kilobytes	8	128 kilobytes
24 kilobytes	5	120 kilobytes
32 kilobytes	4	128 kilobytes
40 kilobytes	3	120 kilobytes
48 kilobytes	2	96 kilobytes
56 kilobytes	2	112 kilobytes
64 kilobytes (default)	2	128 kilobytes
72 kilobytes	1	72 kilobytes
128 kilobytes	1	128 kilobytes
> 128 kilobytes	1	DAU size

Sun QFS and Sun SAM-QFS Stripe Widths – Using Striped Groups

If striped groups are configured for your Sun QFS or Sun SAM-QFS file system, the minimum amount of space allocated is the DAU multiplied by the number of devices in the striped group. The amount of the allocation can be very large when using striped groups.

When striped groups are used, data is written to several disk devices at once. This allocation treats a group of disks as if they were one device. Allocations on striped groups are logically equal to the DAU size multiplied by the number of elements in the striped group.

When striped groups are used, the `-o stripe=`*n* mount option determines the number of allocations that occur on each stripe group before the allocation moves to a different striped group. If a file system is mounted with `-o stripe=0`, the allocation is always to one striped group.

By default, the setting is `–o stripe=0`, which is round robin. The setting can be as low as `–o stripe=0` (which disables striping) or as high as `–o stripe=255`. The system sets `-o stripe=0` if mismatched striped groups are present.

Sun QFS and Sun SAM-QFS Data Alignment

Data alignment refers to matching the allocation unit of the RAID controller with the allocation unit of the file system. The optimal Sun QFS file system alignment formula is as follows:

allocation_unit = *RAID_stripe_width* X *number_of_data_disks_in_the_RAID*

For example, if a RAID-5 unit has a total of 9 disks, with 1 of the 9 being the parity disk, the number of data disks is 8. If the RAID stripe width is 64 kilobytes, then the optimal allocation unit is 64 X 8 = 512 kilobytes.

Data files are striped or round-robined through each striped group (`g`*XXX*) or data disk (`mr` or `md`) defined within the same file system.

A mismatched alignment hurts performance because it can cause a read-modify-write operation. The rest of this chapter provides more information for you to consider when setting DAUs and determining stripe widths.

Stripe Widths on Metadata Disks

You can use the `–o mm_stripe=`*n* option to the `mount_samfs`(1M) command to stripe metadata information on the metadata disk. The default stripe width is `–o mm_stripe=1`, which specifies that one 16-kilobyte DAU be written to a metadata disk before switching to the next metadata disk. The small, 4-kilobyte DAU is used for metadata disks.

By default, if you have multiple metadata devices, metadata is allocated using striped or round-robin allocation depending what is specified on the `-o mm_stripe=`*n* option to the mount(1M) command. The setting can be as low as `-o mm_stripe=0`, which disables striping. It can also be as high as `-o mm_stripe=256`. You can override this setting for the `.inodes` file. For more information on striping the `.inodes` file, see “Advanced Topics” on page 223.

File Allocation Methods

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems allow you to specify both round-robined and striped allocation methods. TABLE 2-12 shows the default file allocation methods used.

TABLE 2-12 Default Allocation Methods

File System	Metadata	File Data
Sun SAM-FS	Striped	Striped
Sun QFS and Sun SAM-QFS	Striped	Striped
Sun QFS and Sun SAM-QFS (striped groups)	Striped	Round-robined
Sun QFS shared file system	Striped	Round-robined

The following sections describe round-robined allocation, striped allocation, and striped groups in more detail.

Round-Robined Allocation

The round-robined allocation method writes one data file at a time to each successive device in the family set. Round-robined allocation is useful for multiple data streams because aggregate performance can exceed striping performance in this type of environment.

Round-robined disk allocation allows a single file to be written to a logical disk. The next file is written to the next logical disk. When the number of files written equals the number of devices defined in the family set, the file system starts over again with the first devices selected. If a file exceeds the size of the physical device, the first portion of the file is written to the first device, and the remainder of the file is written to the next device with available storage.

I/O size is determined by the size of the file being written. Round-robined allocation can be explicitly specified in the `/etc/vfstab` file by entering `stripe=0`.

The following figures depict round-robined allocations. In these figures, file 1 is written to disk 1, file 2 is written to disk 2, file 3 is written to disk 3, and so on. When file 6 is created, it is written to disk 1, starting the round-robined allocation scheme over again.

FIGURE 2-1 depicts a Sun SAM-FS file system using round-robined allocation on five devices. FIGURE 2-2 depicts a Sun QFS or Sun SAM-QFS file system using round-robined allocation on five devices.

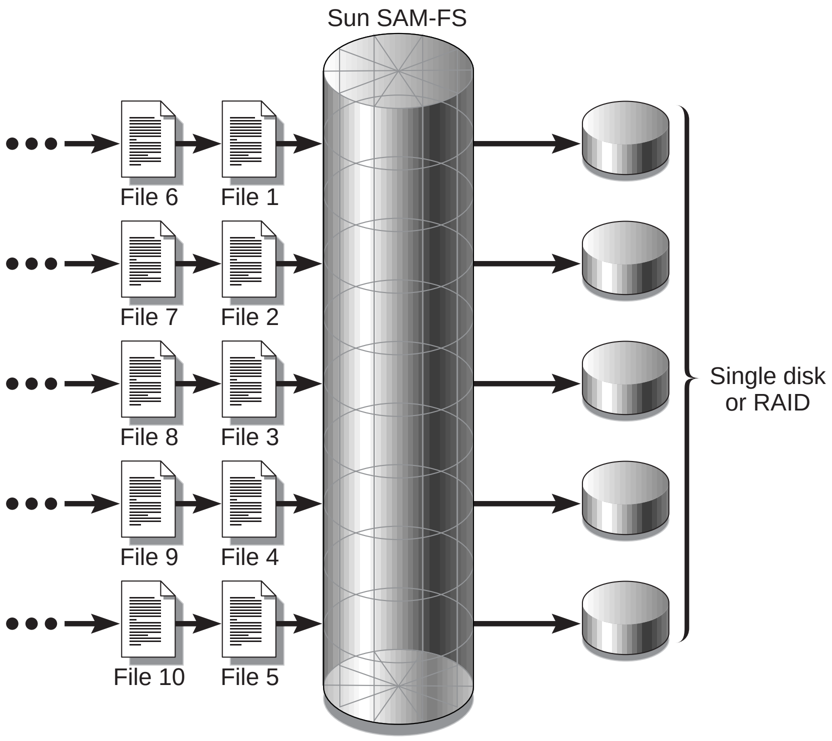

FIGURE 2-1 Round-robined Sun SAM-FS File System Using Five Devices

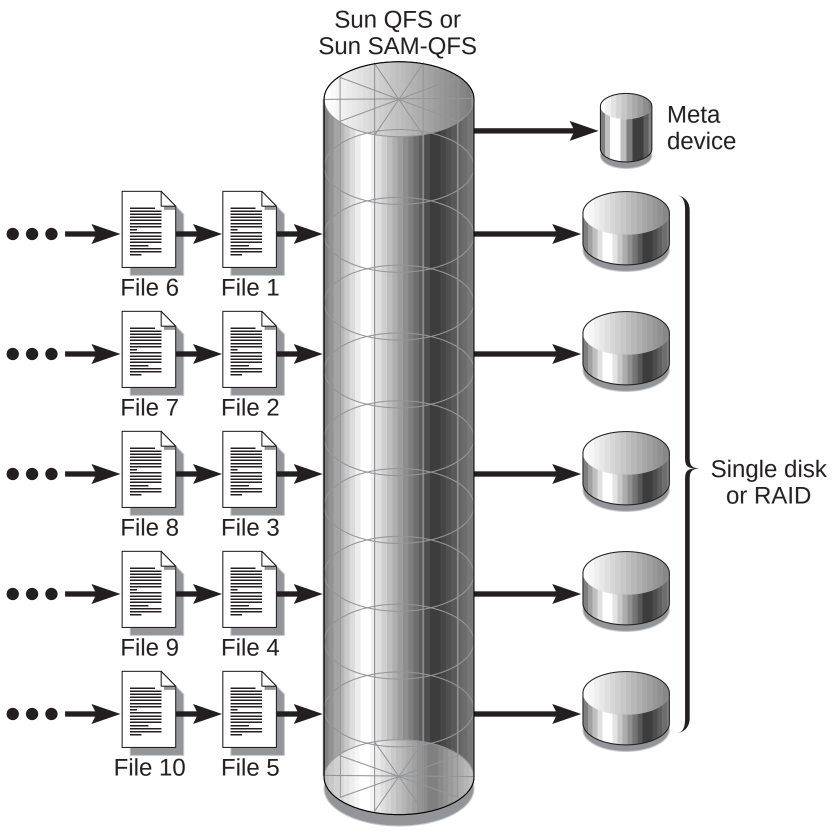

FIGURE 2-2 Round-robined Sun QFS or Sun SAM-QFS File System Using Five Devices

Striped Allocation

By default, Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems use a striped allocation method to spread data over all the devices in the file system family set. Striping is a method of writing files in an interlaced fashion across multiple devices concurrently.

Striping is used when performance for one file requires the additive performance of all the devices. A file system that is using striped devices addresses blocks in an interlaced fashion rather than sequentially. Striping generally increases performance because disk reads and writes are spread concurrently across disk heads. Striped disk access allows multiple I/O streams to simultaneously write a file across multiple disks. The size of the I/O transmission is determined by the DAU and the stripe width.

In a file system using striping, file 1 is written to disk 1, disk 2, disk 3, disk 4, and disk 5. File 2 is written to disks 1 through 5 as well. The DAU multiplied by the stripe width determines the amount of data written to each disk in a block.

When a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system starts to write a file to an `md` device, it first assumes that the file will fit into a small DAU, which is 4 kilobytes. If the file does not fit into the first 8 small DAUs (32 kilobytes) allocated, the file system writes the remainder of the file into one or more large DAUs.

When a Sun QFS or Sun SAM-QFS file system starts to write a file to an `mr` device, it writes first to one DAU, then another, and so on. The `mr` devices have only one DAU size. A Sun QFS or Sun SAM-QFS file system can also write metadata to striped `mm` devices.

Multiple active files cause significantly more disk head movement if striped allocation is used. If I/O is to occur to multiple files simultaneously, round-robin allocation is preferred.

The following figures depict file systems using striped allocations. In these figures, DAU X *stripe_width* bytes of the file are written to disk 1, DAU X *stripe_width* bytes of the file are written to disk 2, DAU X *stripe_width* bytes of the file are written to disk 3, and so on. The order of the stripe is first-in-first-out for the files. Striping spreads the I/O load over all the disks.

FIGURE 2-3 depicts a Sun SAM-FS file system using five striped devices. FIGURE 2-4 depicts a Sun QFS or Sun SAM-QFS file system using five striped devices.

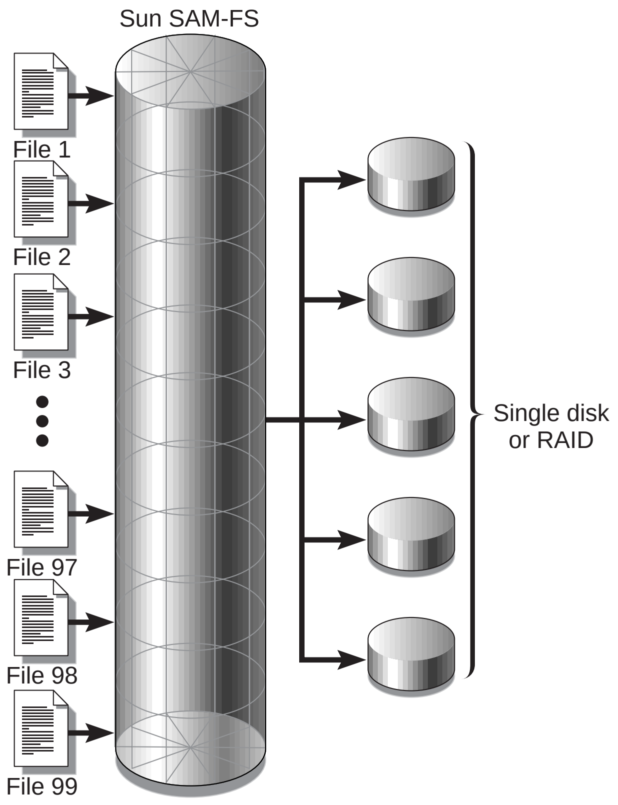

FIGURE 2-3 Sun SAM-FS File System Using Five Striped Devices

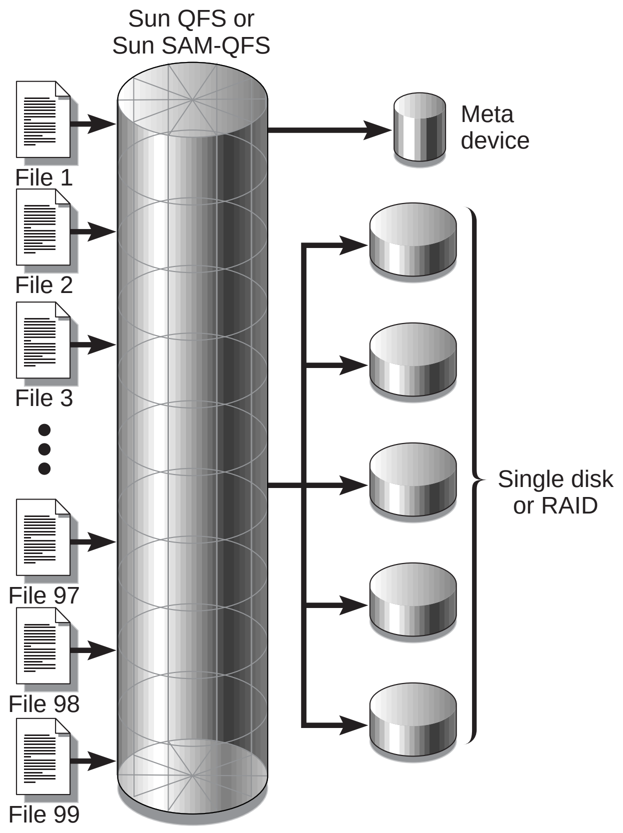

FIGURE 2-4 Sun QFS or Sun SAM-QFS File System Using Five Striped Devices

Striped Groups (Sun QFS and Sun SAM-QFS File Systems Only)

A *striped group* is a special Sun QFS and Sun SAM-QFS allocation method that is designed for file systems that have extremely large I/O requirements and terabytes of disk cache. A striped group allows you to designate an Equipment Type that contains multiple physical disks. Multiple striped group Equipment Types can make up a single Sun QFS or Sun SAM-QFS file system. Striped groups save bit map space and system update time for very large RAID configurations.

A striped group is a collection of devices within a Sun QFS or Sun SAM-QFS file system. Striped groups must be defined in the `mcf` file as `g`*XXX* devices. Striped groups allow one file to be written to and read from two or more devices. You can specify up to 128 striped groups within a file system.

FIGURE 2-5 depicts a Sun QFS or Sun SAM-QFS file system using striped groups and a round-robined allocation. In FIGURE 2-5, files written to the `qfs1` file system are round-robined between groups `g0`, `g1`, and `g2`. Three striped groups are defined (`g0`, `g1`, and `g2`). Each group consists of two physical RAID devices.

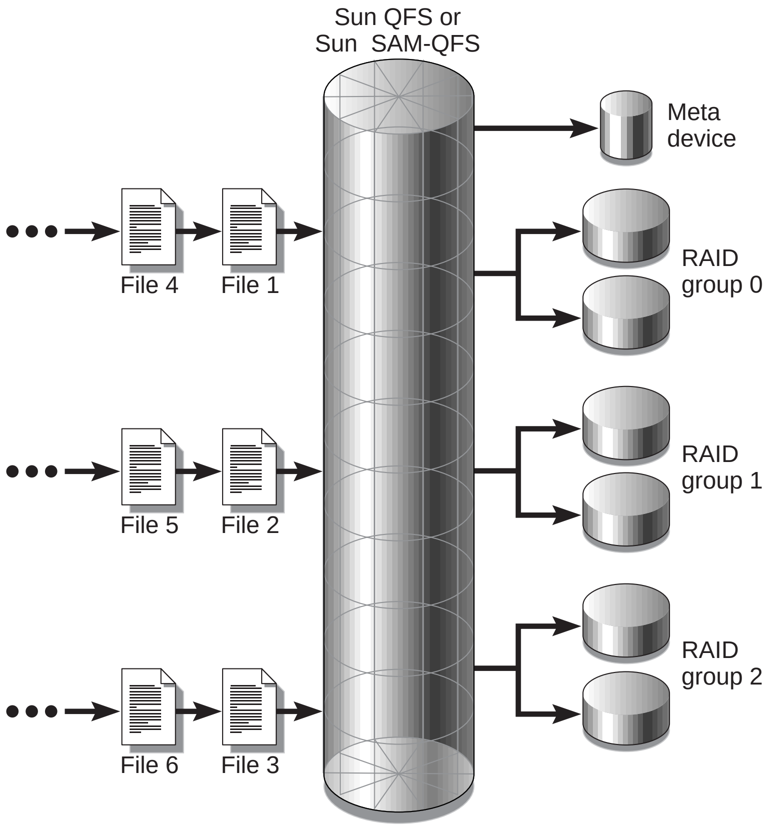

FIGURE 2-5 Sun QFS and Sun SAM-QFS Round-Robined Striped Groups

For the configuration in FIGURE 2-5, the mount point option in `/etc/vfstab` is set to `stripe=0`. These striped groups are declared as follows in the `mcf` file:

```
# Equipment         Eq   Eq    Fam   Dev    Additional
# Identifier        Ord  Type  Set   State  Parameters
#
qfs1                10   ma    qfs1
/dev/dsk/c0t1d0s6   11   mm    qfs1  -
/dev/dsk/c1t1d0s2   12   g0    qfs1  -
/dev/dsk/c2t1d0s2   13   g0    qfs1  -
/dev/dsk/c3t1d0s2   14   g1    qfs1  -
/dev/dsk/c4t1d0s2   15   g1    qfs1  -
/dev/dsk/c5t1d0s2   16   g2    qfs1  -
/dev/dsk/c6t1d0s2   17   g2    qfs1  -
```

FIGURE 2-6 depicts a Sun QFS or Sun SAM-QFS file system using striped groups in which the data is striped across groups. In FIGURE 2-6, files written to the `qfs1` file system are striped through groups `g0`, `g1`, and `g2`. Each group is comprised of four physical RAID devices. The mount point option in `/etc/vfstab` is set to `stripe=1` or greater.

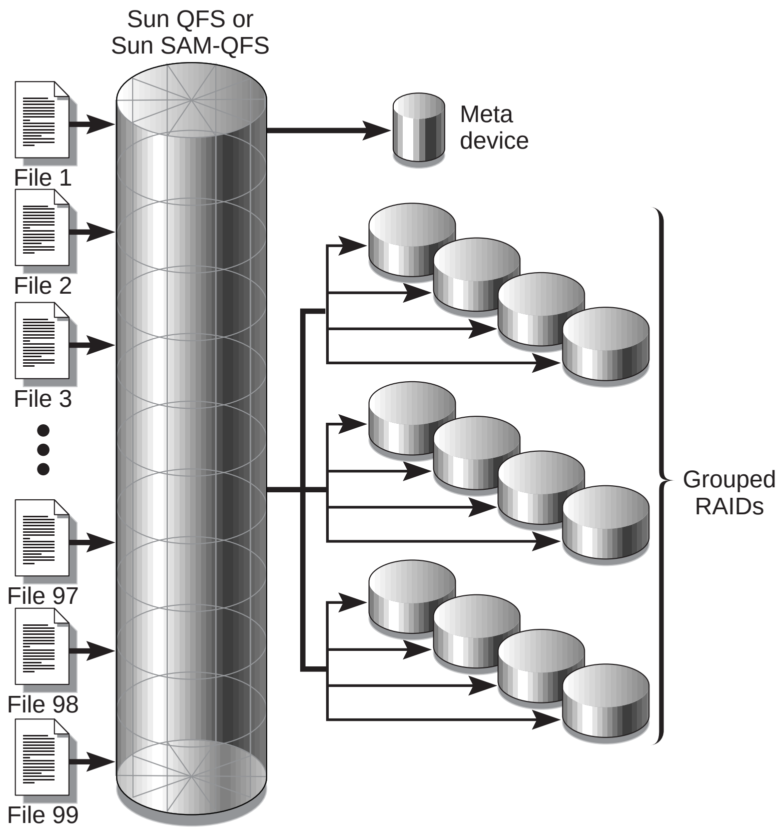

FIGURE 2-6 Sun QFS and Sun SAM-QFS Striped Group Allocation

Mismatched Striped Groups (Sun QFS and Sun SAM-QFS File Systems Only)

It is possible to build a file system with mismatched striped groups. Mismatched striped groups are those that do not contain the same number of devices in each group. Sun QFS and Sun SAM-QFS file systems support mismatched striped groups, but they do not support striping on mismatched groups. File systems with mismatched striped groups are round-robin file systems.

The following example shows how a file system can be set up to store different types of files.

Assumptions

Assume that you have a Sun QFS license, and you need to create a file system at your site that contains video and audio data.

Storing the Video and Audio Files

Video files are quite large and require greater performance than audio files. You want to store them in a file system with a large striped group because striped groups maximize performance for very large files.

Audio files are smaller and require lower performance than video files. You want to store them in a small striped group. One file system can support both video and audio files.

FIGURE 2-7 depicts the file system needed. It is a Sun QFS file system using mismatched striped groups in a striped allocation.

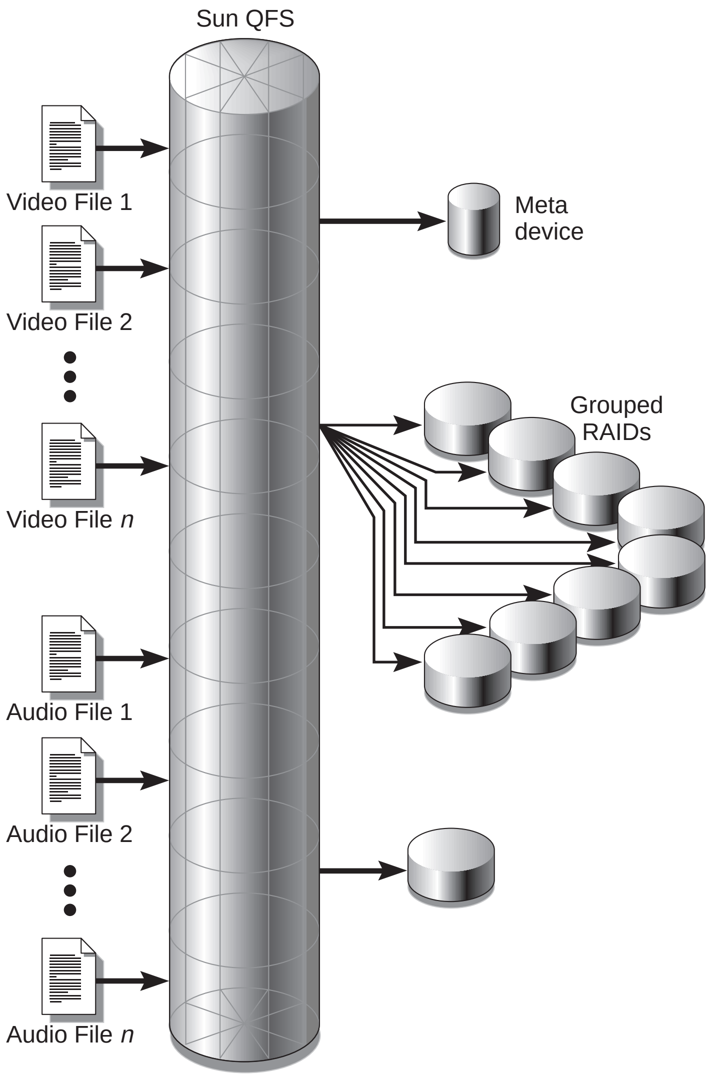

FIGURE 2-7 Sun QFS File System Using Mismatched Striped Groups in a Striped Allocation

TABLE 2-13 shows the characteristics of this file system.

TABLE 2-13 File System `avfs` Characteristics

Characteristics	**Notes**
File system name	`avfs.`
Number of stripe groups	Two. The video file group is `g0`. The audio file group is `g1`.
Stripe width	0.
DAU	128 kilobytes.
Number of disks for `g0`	8.
Minimum block size for `g0`	8 disks X 128-kilobyte DAU = 1024 kilobytes. (This is the amount of data written in one block write. Each disk receives 128 kilobytes of data, so the total amount written to all disks at one time is 1024 kilobytes.)
Number of disks for `g1`	1.
Minimum block size for `g1`	1 disk X 128-kilobyte DAU = 128 kilobytes.

Add the following line to the `/etc/vfstab` file so the environment recognizes the `avfs` file system:

```
avfs    -    /avfs    samfs    -    no    stripe=0
```

Note that in the `/etc/vfstab` file, `stripe=0` is used to specify a round-robin file system. This is used because a value greater than 0 (stripe > 0) is not supported for mismatched striped groups.

The mcf file for this file system is as follows:

```
# Equipment          Eq   Eq     Fam   Dev    Additional
# Identifier         Ord  Type   Set   State  Parameters
#
avfs                 100  ma     avfs
/dev/dsk/c00t1d0s6  101  mm     avfs  -
#
/dev/dsk/c01t0d0s6  102  g0     avfs  -
/dev/dsk/c02t0d0s6  103  g0     avfs  -
/dev/dsk/c03t0d0s6  104  g0     avfs  -
/dev/dsk/c04t0d0s6  105  g0     avfs  -
/dev/dsk/c05t0d0s6  106  g0     avfs  -
/dev/dsk/c06t0d0s6  107  g0     avfs  -
/dev/dsk/c07t0d0s6  108  g0     avfs  -
/dev/dsk/c08t0d0s6  109  g0     avfs  -
#
/dev/dsk/c09t1d0s6  110  g1     avfs  -
```

After the mcf file for this file system is ready, you can enter the following sammkfs(1M) and mount(1M) commands to create and mount the avfs file system:

```
# sammkfs –a 128 avfs
# mount avfs
```

After the file system is mounted, you can create two directories for the two types of files by issuing the following commands:

```
# mkdir video
# mkdir audio
```

After the directories are created, you can use the setfa(1) command to assign the large striped group to video and to assign the small striped group to audio. Files created in these directories are allocated on their respective striped groups because attributes are inherited. The commands are as follows:

```
# setfa –g0 video
# setfa –g1 audio
```

For more information on the `sammkfs`(1M) command, see the `sammkfs`(1M) man page. Fore more information on the `mount`(1M) commands, see the `mount_samfs`(1M) man page. For more information on the `setfa`(1) command, see the `setfa`(1) man page.

CHAPTER **3**

Volume Management

The master configuration file (`mcf`) describes all devices that are under the control of, or used by, the Sun QFS, Sun SAM-FS, or Sun SAM-QFS software. When creating this file, you declared attributes for each device, and you group the devices comprising each file system into family sets.

The configuration process is part of the installation process. The steps in the configuration process are as follows:

1. Create the `/etc/opt/SUNWsamfs/mcf` file.
2. Edit the `/etc/vfstab` file.
3. Use the `sammkfs`(1M) command to construct the new file system.
4. Use the `mount`(1M) command to mount the file system.

The installation and configuration process is described completely in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*. This chapter provides more information on configuring the file systems used in the Sun QFS, Sun SAM-FS, and Sun SAM-QFS environments. It describes the following topics:

- "Creating the mcf File" on page 42
- "Examples of mcf Files" on page 45
- "Interactions Between File Settings, Options, and Directives" on page 49
- "Initializing a File System" on page 50
- "Configuration Examples" on page 52

Creating the `mcf` File

The first step toward configuring a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system is to create a master configuration file in `/etc/opt/SUNWsamfs/mcf`. The `mcf` file contains the information that these file systems need in order to identify and organize RAID and disk devices into file systems. It also contains entries for each automated library or device included in a file system. A sample `mcf` file is located in `/opt/SUNWsamfs/examples/mcf`.

An `mcf` file is an ASCII file that consists of lines of specification code divided into six columns, or fields. The following format shows the six fields that comprise each line in an `mcf` file:

```
Equipment    Equipment    Equipment    Family    Device    Additional
Identifier   Ordinal      Type         Set       State     Parameters
```

The following rules pertain to how data can be entered in the `mcf` file:

- Enter either space or tab characters between the fields in the file.
- You can include comment lines in an `mcf` file. Comment lines start with a pound character (#).

As the following sections indicate, some fields are optional. Use a dash character (-) to indicate that an optional field contains no meaningful information. The following sections describe each field.

For more information on writing the `mcf` file, see the `mcf`(4) man page.

The `Equipment Identifier` Field

The `Equipment Identifier` field must contain either the name of a file system, the keyword `nodev`, a `/dev/dsk` entry, a `/dev/samst` entry, or a `/dev/rmt` entry. This is a required field.

If this field contains the name of a file system, the subsequent lines in the `mcf` file all define the disks or devices included in the file system. More than one file system can be declared in an `mcf` file. Typically, the first data line in an `mcf` file declares the first file system, and subsequent lines specify the devices included in the file system. The other file systems declared in the `mcf` file can be preceded by a blank comment line for readability. File system names must start with an alphabetic character and can contain only alphabetic characters, numeric characters, or underscore (_) characters.

If this field contains the keyword `nodev`, the `mcf` file is being used as a client host in a Sun QFS shared file system. This keyword can only appear in this field as the `Equipment Identifier` for the metadata (`mm`) device that resides on the metadata server. For more information on creating an `mcf` file for the members of a Sun QFS shared file system, see the "Sun QFS Shared File System" on page 89.

If this field is a `/dev/dsk` entry, it identifies a disk partition or slice.

If this field is a `/dev/samst` entry, it identifies an automated library or optical drive. If you are configuring a network-attached automated library, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* and the *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide,* for more information.

If this field is a `/dev/rmt` entry, it identifies a tape drive.

The `Equipment Ordinal` Field

For each row in the `mcf` file, the `Equipment Ordinal` field must contain a numeric identifier for the file system component or device being defined. Specify a unique integer from 1 to 65535. This is a required field.

The `Equipment Type` Field

Enter a 2-, 3-, or 4-character code for the `Equipment Type` field. This is a required field

As TABLE 3-1 shows, a Sun SAM-FS file system can contain either `ms` or `md` in the `Equipment Type` field.

TABLE 3-1 Sun SAM-FS `Equipment Type` Field

Equipment Type Field Content	Meaning
`ms`	Defines a Sun SAM-FS file system.
`md`	Defines a striped or round-robined device for storing file data and metadata information.

As TABLE 3-2 shows, a Sun QFS or Sun SAM-QFS file system can contain either `ma`, `md`, `mm`, `mr`, or `g`*XXX* in the `Equipment Type` field.

TABLE 3-2 Sun QFS and Sun SAM-QFS `Equipment Type` Field

Equipment Type Field Content	Meaning
`ma`	Defines a Sun QFS or Sun SAM-QFS file system.
`md`	Defines a striped or round-robined device for storing file data.
`mm`	Defines a metadata device for storing inode and other non-data information.
`mr`	Defines a round-robined or striped data device.
`g`*XXX*	Striped group data device. Striped groups start with the letter `g` followed by a number. The number must be an integer such that `0` ≤ *XXX* ≤ `127`. For example, `g12`. All members in a striped group must be the same type and size. Different striped groups within one file system are not required to have the same number of members. `md`, `mr`, and `g`*XXX* devices cannot be mixed in one file system.

Besides the file system equipment types, other codes are used to identify automated libraries and other devices. For more information on specific equipment types, see the `mcf`(4) man page.

The `Family Set` Field

The `Family Set` field must contain the name of the group of devices included in a file system. This is a required fild for file system devices. This is optional for other devices. If this is used as an optional field, enter a dash (-) character to indicate that this field is omitted.

Naming conventions for family set names are identical to those for file system names. The names must start with an alphabetic character and can contain only alphabetic characters, numeric characters, or underscore (_) characters.

For a file system, this field is required because a family set associates all devices with the same family set name together as a file system. The family set name is physically recorded on all the devices in the file system when the `sammkfs`(1M) command is issued. It is possible to change this name by using the `–F` and `–R` options together on the `samfsck`(1M) command. For more information on the `sammkfs`(1M) command, see the `sammkfs`(1M) man page. For more information on the `samfsck`(1M) command, see the `samfsck`(1M) man page.

In Sun SAM-FS and Sun SAM-QFS environments, this field can be either a family set name or a dash (-). If the device is associated with a family set (that is, a file system or an automated library), enter the family set name for this device.

If the device is a manually loaded drive, this is an optional field, so enter a dash character (-) to indicate that this field is omitted.

The `Device State` Field

The `Device State` field specifies the state of the device when the file system is initialized. Valid device states are `on` and `off`. This is an optional field. If `on` or `off` are not entered, enter a dash (-) character to indicate that this field is omitted.

The `Additional Parameters` Field

The `Additional Parameters` field is optional and can be left completely blank. By default, library catalog files are written to `/var/opt/SUNWsamfs/catalog/`*family_set_name*. This field can be used to specifiy an alternate path to the library catalog file.

Examples of `mcf` Files

Each file system configuration is unique. System requirements and actual hardware differ from site to site. The following sections show sample `mcf` files for Sun QFS, Sun SAM-FS, and Sun SAM-QFS environments.

Sun SAM-FS Volume Management Examples

For the Sun SAM-FS file system, you can define family sets in the `/etc/opt/SUNWsamfs/mcf` file in the `Equipment Type` field using the following equipment types:

- `ms` for the Sun SAM-FS file system type.
- `md` for the devices. Data is striped or round-robined across these devices. The stripe width is set with the `-o stripe=`*n* option on the `mount`(1M) command. The default stripe width is set based on the DAU size. For more information on stripe widths and DAU sizes, see "File System Design" on page 11.

Both metadata (including inodes, directories, allocation maps, and so on) and file data on Sun SAM-FS file systems are located on the same disk. Data files are striped or round-robined through each disk partition defined within the same file system.

The following example shows an `mcf` file for a Sun SAM-FS file system.

```
# Sun SAM-FS file system configuration example
#
# Equipment       Eq    Eq   Fam.  Dev.    Additional
# Identifier      Ord  Type  Set   State   Parameters
#-----------      ---   --  ------ ------  ------------------
samfs1             10  ms  samfs1
/dev/dsk/c1t1d0s6  11  md  samfs1  -
/dev/dsk/c2t1d0s6  12  md  samfs1  -
/dev/dsk/c3t1d0s6  13  md  samfs1  -
/dev/dsk/c4t1d0s6  14  md  samfs1  -
/dev/dsk/c5t1d0s6  15  md  samfs1  -
```

Sun QFS and Sun SAM-QFS Volume Management Examples

For the Sun QFS and Sun SAM-QFS file systems, family sets are defined in the `/etc/opt/SUNWsamfs/mcf` file in the Equipment Type field using the following device types:

- `ma` for the Sun QFS or Sun SAM-QFS file system type.
- `mm` for a metadata device. File data is not written to this device. You can specify multiple metadata devices. Metadata (including inodes, directories, allocation maps, and so on) on Sun QFS and Sun SAM-QFS file systems is located on the metadata device(s) and is separated from the file data devices. By default, metadata is allocated using round-robin allocation if you have multiple metadata devices.
- `mr` or `md` for devices upon which file data is to be striped or round-robined. The stripe width is defined as a mount option. The default stripe width is set based on the DAU size. For more information on stripe widths and DAU sizes, see “File System Design” on page 11.
- `g`*XXX* for devices upon which file data is to be striped as a group. A striped group is a logical group of devices that are striped as a unit. Data is striped across the members of each group.

  Groups are specified with `g0` through `g127` equipment type numbers with the stripe width on each device being the DAU. All devices in a striped group must be the same size. Different striped groups within one file system are not required to have the same number of members. `mr` and `g`*XXX* devices can be mixed in a file system, but `md` devices cannot be mixed with either `mr` or `g`*XXX* devices in a file system.

Data can be striped (if all groups contain the same number of devices) or round-robined between groups. The default is round robin.

Data files are striped or round-robined through each data disk partition (`mr` or `g`*XXX*) defined within the same file system.

Example 1

The following example shows an `mcf` file for a Sun QFS or Sun SAM-QFS file system with two striped groups.

```
# Sun QFS file system configuration
#
# Equipment        Eq   Eq    Fam.  Dev.    Additional
# Identifier       Ord  Type  Set   State   Parameters
#-----------      ---    --  ------ ------  ------------------
qfs1               10   ma   qfs1    -
/dev/dsk/c2t1d0s7  11   mm   qfs1    -
/dev/dsk/c3t0d0s6  12   g0   qfs1    -
/dev/dsk/c3t0d1s6  13   g0   qfs1    -
/dev/dsk/c4t0d0s6  14   g1   qfs1    -
/dev/dsk/c4t0d1s6  15   g1   qfs1    -
```

Example 2

The following example shows an `mcf` file with three Sun QFS or Sun SAM-QFS file systems.

```
# Sun SAM-QFS file system configuration example
#

# Equipment       Eq    Eq   Fam.  Dev.    Additional
# Identifier      Ord  Type  Set   State   Parameters
#-----------      ---   --  ------ ------  ------------------
qfs1               10  ma   qfs1   -
/dev/dsk/c1t13d0s6 11  mm   qfs1   -
/dev/dsk/c1t12d0s6 12  mr   qfs1   -
#
qfs2               20  ma   qfs2   -
/dev/dsk/c1t5d0s6  21  mm   qfs2   -
/dev/dsk/c5t1d0s6  22  mr   qfs2   -
#
qfs3               30  ma   qfs3   -
/dev/dsk/c7t1d0s3  31  mm   qfs3   -
/dev/dsk/c6t1d0s6  32  mr   qfs3   -
/dev/dsk/c6t1d0s3  33  mr   qfs3   -
/dev/dsk/c5t1d0s3  34  mr   qfs3   -
```

Example 3

The following example shows an `mcf` file with one Sun SAM-QFS file system that uses `md` devices. This `mcf` file also defines a tape library.

```
# Sun SAM-QFS file system configuration example
#

# Equipment        Eq    Eq   Fam.  Dev.    Additional
# Identifier       Ord  Type  Set   State   Parameters
#-----------       ---   --  ------ ------  ----------
samfs1             10    ma   samfs1  -
/dev/dsk/c1t2d0s6  11    mm   samfs1  -
/dev/dsk/c1t3d0s6  12    md   samfs1  -
/dev/dsk/c1t4d0s6  13    md   samfs1  -
/dev/dsk/c1t5d0s6  14    md   samfs1  -
# scalar 1000 and 12 AIT tape drives
/dev/samst/c5t0u0  30    as   adic1   -
/dev/rmt/4cbn     101    at   adic1   on
/dev/rmt/5cbn     102    at   adic1   on
/dev/rmt/6cbn     103    at   adic1   on
/dev/rmt/7cbn     104    at   adic1   off
/dev/rmt/10cbn    105    at   adic1   on
/dev/rmt/11cbn    106    at   adic1   on
/dev/rmt/3cbn     107    at   adic1   on
/dev/rmt/2cbn     108    at   adic1   on
/dev/rmt/1cbn     109    at   adic1   on
/dev/rmt/0cbn     110    at   adic1   on
/dev/rmt/9cbn     111    at   adic1   on
/dev/rmt/8cbn     112    at   adic1   on
```

For more examples showing file system configuration in the `mcf` file, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

Interactions Between File Settings, Options, and Directives

The `mcf` file defines each file system, but file system behavior depends on interactions between default systems settings, settings in the `/etc/vfstab` file, settings in the `samfs.cmd` file, and options on the `mount`(1M) command line.

You can specify some mount options, for example the stripe width, in more than one place. When this happens, settings in one place can override the settings in another.

For information on the various ways to specify mount options, see “To Mount a File System” on page 66.

Initializing a File System

The `sammkfs`(1M) command constructs new file systems, and its `-a` *allocation_unit* option allows you to specify the DAU setting. The number specified for *allocation_unit* determines the DAU setting.

The `sammkfs`(1M) command is also used when restoring file systems. Another command, `samfsinfo`(1M) can be used to gather the configuration information for an existing file system.

The `sammkfs`(1M) command must be issued prior to issuing the `mount`(1M) command when installing and configuring a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system for the first time.

The 4.0 releases of these file systems support two different superblock designs. Both superblock designs are available to you in the 4.0 release. You can use the `samfsinfo`(1M) command, as shown in the following example, to determine which superblock a file system is using.

```
# samfsinfo samfs1
name:     samfs1       version:       2
time:     Wed Feb 21 13:32:18 1996
count:    1
capacity:      001240a0          DAU:         16
space:         000d8ea0
ord  eq   capacity      space   device
  0  10   001240a0   000d8ea0   /dev/dsk/c1t1d0s0
```

The first line of the preceding output indicates that this is a version 2 superblock. Be aware of the following operational and feature differences that pertain to these superblocks:

- The version 1 design is the only superblock design supported in releases prior to 4.0.
- The version 2 superblock is supported in 4.0 and later releases. If you installed the 4.0 software as an upgrade, you must use the 4.0 `sammkfs`(1M) command to reinitialize your existing file systems before you attempt to use any of the features that depend on the version 2 superblock. Certain 4.0 features, such as access

control lists (ACLs) and the Sun QFS shared file system are supported only in the version 2 superblock. Reinitializing a file system is described as a step in the 4.0 software installation upgrade process, but this can be done any time after the software is installed.

- If you want to use the version 1 superblock with the release 4.0 software, you need to use the `-P` option to the `sammkfs`(1M) command every time you reinitialize your file system. The `-P` option directs the `sammkfs`(1M) command to reinitialize the file system using a version 1 superblock.

Caution – File systems that use a version 2 superblock cannot revert to a release prior to 4.0.

For more information on features that require a version 2 superblock, or on using the `sammkfs`(1M) command to obtain use of the version 2 superblock, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

Example 1

The following command initializes a Sun SAM-FS file system using a version 1 superblock.

```
# sammkfs -a 64 -P samfs1
Creating an old format file system disallows some file system
features
Please see 'Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and
Configuration Guide' for a list of the affected feature
Building 'samfs1' will destroy the contents of devices:
                /dev/dsk/c3t4d0s6
                /dev/dsk/c3t5d0s6
Do you wish to continue? [y/N] y
total data kilobytes       = 4168576
total data kilobytes free  = 4168512
total meta kilobytes       = 4168576
total meta kilobytes free  = 4168160
```

Example 2

The following command initializes a Sun SAM-FS file system using a version 2 superblock.

```
# sammkfs -a 64 samfs1
Creating a new file system prevents use with SAM-FS 3.5.0 or
earlier
Use the -P option on sammkfs to create a 3.5.0 compatible file
system
Do you wish to continue? [y/N] y
Building 'samfs1' will destroy the contents of devices:
                /dev/dsk/c3t4d0s6
                /dev/dsk/c3t5d0s6
Do you wish to continue? [y/N] y
total data kilobytes       = 4168576
total data kilobytes free  = 4168512
total meta kilobytes       = 4168576
total meta kilobytes free  = 4168160
```

For more information on the `sammkfs`(1M) command, see the `sammkfs`(1M) man page.

Configuration Examples

The rest of this chapter presents sample configurations and shows various steps and considerations in setting up the `mcf` file on a server. The following procedures are described:

- "To Create a Sun QFS Round-Robined Disk Configuration" on page 53
- "To Create a Sun SAM-FS Round-Robined Disk Configuration" on page 54
- "To Create a Sun QFS Striped Disk Configuration" on page 55
- "To Create a Sun SAM-FS Striped Disk Configuration" on page 57
- "To Create a Sun QFS Striped Groups Configuration" on page 58

Note that all sample Sun QFS configurations could have automated libraries and other removable media devices defined as well, essentially extending the size of the disk cache. Removable media device configurations are shown in only one example. For information on configuring removable media devices see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* and the *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

The sample configurations assume that the file system is loaded on the system and all file systems are unmounted.

▼ To Create a Sun QFS Round-Robined Disk Configuration

This sample configuration illustrates a Sun QFS file system that separates the metadata onto a low-latency disk. Round-robin allocation is used on four partitions. The file system is created using the `sammkfs`(1M) command. Each disk is on a separate controller.

The following assumptions are used:

- The metadata device is a single partition (s6) used on controller 5, LUN 0 of the device designated as equipment ordinal 11.
- The data devices consist of four disks attached to four controllers.

1. **Use an editor to create the `mcf` file.**

```
Sun QFS disk cache configuration – Round-robin mcf example

# Equipment        Eq   Eq    Fam.  Dev     Additional
# Identifier       Ord  Type  Set   State   Parameters
#-----------       ---   --  ------ ------  ------------------
qfs1                 1   ma   qfs1
/dev/dsk/c5t0d0s6  11   mm   qfs1    on
/dev/dsk/c1t1d0s6  12   mr   qfs1    on
/dev/dsk/c2t1d0s6  13   mr   qfs1    on
/dev/dsk/c3t1d0s6  14   mr   qfs1    on
/dev/dsk/c4t1d0s6  15   mr   qfs1    on
```

2. **Use the `mkdir`(1) command to create the `/qfs` mount point for the `/qfs1` file system.**

```
# mkdir /qfs
```

3. **Use the the `sammkfs`(1M) command to initialize the file system.**

   The default 64-kilobyte DAU is used.

```
# sammkfs qfs1
```

4. **Use an editor to modify the `/etc/vfstab` file.**

   The Sun QFS file system with `mr` data devices uses striped allocation as a default, so you must set `stripe=0` for round-robin allocation. To explicitly set round-robin on the file system, set the `stripe=0` as follows.

```
qfs1    -    /qfs    samfs    -    yes    stripe=0
```

5. **Use the `mount`(1M) command to mount the file system.**

```
# mount /qfs
```

▼ To Create a Sun SAM-FS Round-Robined Disk Configuration

This sample configuration illustrates a Sun SAM-FS file system. Striped allocation is used by default on four partitions. You need to set `stripe=0` to specify round robin allocation. The file system is created using the `sammkfs`(1M) command. The data devices consist of four disks attached to four controllers. Each disk is on a separate controller.

1. **Use an editor to create the `mcf` file.**

```
Sun SAM-FS disk cache configuration – Round-robin mcf example

# Equipment       Eq   Eq    Fam.   Dev     Additional
# Identifier      Ord  Type  Set    State   Parameters
#-----------      ---   --  ------  ------  -----------------
samfs1              1   ms   samfs1
/dev/dsk/c1t1d0s6  11   md   samfs1  on
/dev/dsk/c2t1d0s6  12   md   samfs1  on
/dev/dsk/c3t1d0s6  13   md   samfs1  on
/dev/dsk/c4t1d0s6  14   md   samfs1  on
```

2. **Use the `mkdir`(1) command to create the `/samfs` mount point for the `/samfs1` file system.**

```
# mkdir /samfs
```

3. **Use the the `sammkfs`(1M) command to initialize the file system.**

The default DAU is 16 kilobytes, but the following examples sets the DAU size to 64 kilobytes.

```
# sammkfs -a 64 samfs1
```

4. **Use an editor to modify the `/etc/vfstab` file.**

   The Sun SAM-FS file system uses striped allocation as by default, so you must set `stripe=0` for round-robin allocation. To explicitly set round-robin on the file system, set the `stripe=0` as follows.

```
samfs1    -    /samfs    samfs    -    yes    stripe=0
```

5. **Use the `mount`(1M) command to mount the file system.**

```
# mount /samfs
```

▼ To Create a Sun QFS Striped Disk Configuration

This sample configuration illustrates a Sun QFS file system. By default, file data is striped to four partitions. The file system is created using the `sammkfs`(1M) command, and the DAU size is specified.

The following assumptions are used:

- The metadata device is a single partition (s6) used on controller 0, LUN 1. Metadata is written to equipment 11 only.
- The data devices consist of four disks attached to four controllers. Each disk is on a separate controller.

1. **Use an editor to create the `mcf` file.**

```
Sun QFS disk cache configuration – Striped Disk mcf example

# Equipment        Eq   Eq    Fam.  Dev.    Additional
# Identifier       Ord  Type  Set   State   Parameters
#-----------       ---   --  ------ ------  ------------------
qfs1                10   ma  qfs1
/dev/dsk/c0t1d0s6  11   mm  qfs1     on
/dev/dsk/c1t1d0s6  12   mr  qfs1     on
/dev/dsk/c2t1d0s6  13   mr  qfs1     on
/dev/dsk/c3t1d0s6  14   mr  qfs1     on
/dev/dsk/c4t1d0s6  15   mr  qfs1     on
```

2. **Use the `mkdir`(1) command to create the `/qfs` mount point for the `/qfs1` file system.**

```
# mkdir /qfs
```

3. **Use the the `sammkfs`(1M) command to initialize the file system.**

   The default DAU is 64 kilobytes, but the following example sets the DAU size to 128 kilobytes.

```
# sammkfs -a 128 qfs1
```

   With this striped disk configuration, any file written to this file system is striped across all of the devices in increments of 128 kilobytes.

4. **Use an editor to modify the `/etc/vfstab` file.**

   The Sun SAM-FS file system uses striped allocation as by default. This example sets the stripe width as `stripe=1` DAU, which is the default. The following setting stripes data across all four of the `mr` devices with a stripe width of one DAU.

```
qfs1    -    /qfs    samfs    -    yes    stripe=1
```

5. **Use the `mount`(1M) command to mount the file system.**

```
# mount /qfs
```

▼ To Create a Sun SAM-FS Striped Disk Configuration

This sample configuration illustrates a Sun SAM-FS file system. File data is striped to four disk drives. The file system is created using the sammkfs(1M) command. The data devices consist of four disks attached to four controllers. Each disk is on a separate LUN.

1. **Use an editor to create the mcf file.**

```
Sun SAM-FS disk cache config – Striped Disk mcf example

# Equipment        Eq   Eq    Fam.  Dev.    Additional
# Identifier       Ord  Type  Set   State   Parameters
#-----------       ---   --  ------ ------  ------------------
samfs1              10   ms  samfs1
/dev/dsk/c1t1d0s6  11   md  samfs1   on
/dev/dsk/c2t1d0s6  12   md  samfs1   on
/dev/dsk/c3t1d0s6  13   md  samfs1   on
/dev/dsk/c4t1d0s6  14   md  samfs1   on
```

2. **Use the mkdir(1) command to create the /samfs mount point for the /samfs1 file system.**

```
# mkdir /samfs
```

3. **Use the the sammkfs(1M) command to initialize the file system.**

The following example uses the default 64-kilobyte DAU.

```
# sammkfs samfs1
```

With this striped disk configuration, any file written to this file system is striped across all of the devices in increments of 64 kilobytes.

4. **Use an editor to modify the /etc/vfstab file.**

It is not necessary to modify the /etc/vfstab file because this file system uses the default values.

5. **Use the mount(1M) command to mount the file system.**

```
# mount /samfs
```

▼ To Create a Sun QFS Striped Groups Configuration

Striped groups allow you to group RAID devices together for very large files. A DAU is represented by one bit in the bit maps. If the striped group has *n* devices, *n* multiplied by the DAU is the minimum allocation. Only one bit in the bit maps is used to represent *n* X DAU. This method of writing huge DAUs across RAID devices saves bit map space and system update time. Striped groups are useful for writing very large files to a group of RAID devices.

Note – The minimum disk space allocated in a striped group is as follows:

```
minimum_disk_space_allocated = DAU X number_of_disks_in_the_group
```

Writing a single byte of data fills the entire `minimum_disk_space_allocated` of a striped group. Striped groups are used for very specific applications. Make sure that you understand the effects of using striped groups with your file system.

Files less than the aggregate stripe width times the number of devices (in this example, files less than 128 kilobytes X 4 disks = 512 kilobytes in length) still use 512 kilobytes of disk space. Files larger than 512 kilobytes have space allocated for them as needed in total space increments of 512 kilobytes.

The devices within a striped group must be the same size. It is not possible to add devices to increase the size of a striped group. You can use the `samgrowfs`(1M) command to add additional striped groups, however. For more information on this command, see the `samgrowfs`(1M) man page.

This sample configuration illustrates a Sun QFS file system that separates the metadata onto a low-latency disk. Two striped groups are set up on four drives.

The following assumptions are used:

- The metadata device is a single partition (s6) used on controller 0, LUN 1.
- The data devices consist of four disks (two groups of two identical disks) attached to four controllers. Each disk is on a separate LUN. The entire disk is used for data storage assuming that partition 6 is the entire disk.

1. **Use an editor to create the `mcf` file.**

```
Sun QFS disk cache configuration – Striped Groups mcf example

# Equipment       Eq   Eq    Fam.  Dev.    Additional
# Identifier      Ord  Type  Set   State   Parameters
#-----------      ---   --  ------ ------  ------------------
qfs1               10   ma  qfs1
/dev/dsk/c0t1d0s6  11   mm  qfs1     on
/dev/dsk/c1t1d0s6  12   g0  qfs1     on
/dev/dsk/c2t1d0s6  13   g0  qfs1     on
/dev/dsk/c3t1d0s6  14   g1  qfs1     on
/dev/dsk/c4t1d0s6  15   g1  qfs1     on
```

2. **Use the `mkdir`(1) command to create the `/qfs` mount point for the `/qfs1` file system.**

```
# mkdir /qfs
```

3. **Use the the `sammkfs`(1M) command to initialize the file system.**

   The following example sets the DAU size to 128 kilobytes.

```
# sammkfs -a 128 qfs1
```

4. **Use an editor to modify the `/etc/vfstab` file.**

   This example uses the default setting of `stripe=0`, which essentially specifies a round-robined allocation from striped group `g0` to striped group `g1`.

```
 qfs1    -    /qfs    samfs    -    yes   stripe=0
```

   This `/etc/vfstab` file sets the stripe width using the `stripe=` option. In this example, there are two striped groups, `g0` and `g1`. With the `stripe=0` specification, devices 12 and 13 are striped, and files are round-robined around the two striped groups. You are really treating a striped group as a bound entity. That is, the configuration of the striped group, after it is created, cannot change these groups without issuing another `sammkfs`(1M) command.

5. **Use the `mount`(1M) command to mount the file system.**

```
# mount /qfs
```

CHAPTER 4

Operations

This chapter presents topics related to file system operations. It presents the following topics:

- "To Initialize a File System" on page 62
- "To Initialize or Reinitialize an mcf or defaults.conf File" on page 62
- "To Mount a File System" on page 66
- "To Unmount a File System" on page 69
- "To Check File System Integrity" on page 70
- "To Repair a File System" on page 72
- "To Preserve Information for an Upgrade" on page 72
- "To Prepare for a Hardware Upgrade" on page 77
- "To Add Disk Cache to a File System" on page 78
- "To Replace Disks in a File System" on page 80
- "To Upgrade a Host System" on page 82
- "To Upgrade Your Sun Solaris OE in a Sun SAM-FS or Sun SAM-QFS Environment" on page 83
- "To Upgrade Your Sun Solaris OE in a Sun QFS Environment" on page 85

Certain other types of operations and upgrades also need to be performed within a Sun QFS, Sun SAM-FS, or Sun SAM-QFS environment. The following publications describe these other types of operations:

- The *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* describes upgrading Sun QFS, Sun SAM-FS, and Sun SAM-QFS software. It also describes how to create dump files of Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems.
- The *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide* describes how to add slots in an automated library, how to upgrade or replace an automated library, and how to upgrade DLT tape drives.

- The *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Disaster Recovery Guide* describes how to restore Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems.

To Initialize a File System

A Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system can be initialized or reinitialized by using the `sammkfs`(1M) command.

Example 1. This example shows this command in its simplest form, with the file system name as its only argument. This builds a version 2 superblock.

```
# sammkfs samqfs1
```

Example 2. The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support two different superblocks. This example shows the command being used to create a file system with a version 1 superblock.

```
# sammkfs –P samqfs1
```

For more information about the `sammkfs`(1M) command, its options, and the implications of the version 1 and version 2 superblocks, see “Initializing a File System” on page 50, or see the `sammkfs`(1M) man page.

To Initialize or Reinitialize an `mcf` or `defaults.conf` File

The following procedures describe how to reinitialize the `mcf` and how to initialize or reinitialize the `defaults.conf` file. You must perform these procedures under the following circumstances:

- If you update your `mcf` or `defaults.conf` file in order to add, delete, or correct information.
- If you create a `defaults.conf` file after your Sun SAM-QFS, Sun SAM-FS, or Sun SAM-QFS system is already operational.

The procedures differ depending on whether you have the Sun QFS, Sun SAM-FS, or Sun SAM-QFS software. In a Sun SAM-FS or Sun SAM-QFS environment, the procedures differ depending on whether you are changing file system or removable media drive information. The following sections describe the procedures. For more information on these files, see the `defaults.conf`(4) or `mcf`(4) man pages.

▼ To Change `mcf` or `defaults.conf` Information in a Sun QFS Environment

1. **Edit the file and change the file system information.**
2. **Issue the `sam-fsd` command to check the `mcf` file for errors. (Optional)**

   Perform this step if you are changing an `mcf` file. The format of this command is as follows:

```
# sam-fsd
```

   If the output from this command shows errors, correct them prior to proceeding to the next step.

3. **Use the following command to initialize or reinitialize the `mcf` or `defaults.conf` file on a Sun QFS file system:**

```
# pkill -HUP sam-fsd
```

▼ To Change `mcf` or `defaults.conf` File System Information in a Sun SAM-FS or Sun SAM-QFS Environment

1. **Edit the `mcf` or `defaults.conf` file and change the file system information.**
2. **Issue the `sam-fsd` command to check the `mcf` file for errors. (Optional)**

   Perform this step if you are changing an `mcf` file. The format of this command is as follows:

```
# sam-fsd
```

If the output from this command shows errors, correct them prior to proceeding to the next step.

3. **Issue a `samcmd aridle` command for each file system defined in the `mcf` file. (Optional)**

   You must perform this step if you are removing or changing information related to one or more file systems. The format for this command is as follows:

   ```
   # samcmd aridle fs.fsname
   ```

   For *fsname*, specify the name of a file system defined in the `mcf` file. Issue this command for every file system in the `mcf` file that is affected by the change.

4. **Issue a `samcmd idle` command for each equipment ordinal assigned to a drive in the `mcf` file. (Optional)**

   You must perform this step if you are removing or changing information related to one or more drives. The format for this command is as follows:

   ```
   # samcmd idle eq
   ```

   For *eq*, specify the equipment ordinal of a drive defined in the `mcf` file. Repeat this command as necessary for all drives in your `mcf` file affected by the change.

5. **Issue the `umount`(1M) command to unmount the file system.**

   For more information on unmounting the file system, see "To Unmount a File System" on page 69.

6. **Issue the following command to reinitialize the file.**

   ```
   # samd config
   ```

7. **Use the `mount`(1M) command to remount the file system.**

▼ To Change `mcf` or `defaults.conf` Removable Media Drive Information

1. **Edit the file and change the removable media drive information.**

2. **Issue the `sam-fsd` command to check the `mcf` file for errors. (Optional)**

   Perform this step if you are changing an `mcf` file. The format of this command is as follows:

   ```
   # sam-fsd
   ```

   If the output from this command shows errors, correct them prior to proceeding to the next step.

3. **Issue a `samcmd aridle` command for each file system defined in the `mcf` file. (Optional)**

   Perform this step if you are removing or changing information related to one or more file systems. The format for this command is as follows:

   ```
   # samcmd aridle fs.fsname
   ```

   For *fsname*, specify the name of a file system defined in the `mcf` file. Issue this command for every file system in the `mcf` file that is affected by the change.

4. **Issue a `samcmd idle` command for each equipment ordinal assigned to a drive in the `mcf` file. (Optional)**

   Perform this step if you are removing or changing information related to one or more drives. The format for this command is as follows:

   ```
   # samcmd idle eq
   ```

   For *eq*, specify the equipment ordinal of a drive defined in the `mcf` file. Repeat this command as necessary for all drives in your `mcf` file affected by the change.

5. **Issue the following command to stop all archiving:**

   ```
   # samd stop
   ```

6. **Issue the following command to reinitialize the file:**

   ```
   # samd config
   ```

7. Issue the following command to restart the archiver:

```
# samd start
```

To Mount a File System

You can mount a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system by using the Sun Solaris `mount`(1M) command. This command is described on the `mount_samfs`(1M) man page. This section describes the various ways that mount options can be specified.

Mount parameters are used to manipulate file system characteristics. There are several ways to specify mount parameters, and some specification methods override others. You can specify mount options in the following ways:

1. On the `mount`(1M) command using command line options. Highest priority. Options specified on the Sun Solaris `mount`(1M) command override other options specified in the `/etc/vfstab` file, directives specified in the `samfs.cmd` file, and system default settings.
2. As `/etc/vfstab` file settings.
3. In the `samfs.cmd` file using directives.
4. System defaults. Lowest priority. The default system settings are the configurable settings already defined for your Sun Solaris operating environment (OE). These system settings can be overridden by directives in the `samfs.cmd` file, the `/etc/vfstab` file, and on the `mount`(1M) command.

You can also specify mount options by using the the `samu`(1M) operator utility or on the `samcmd`(1M) command. Mount options enabled or disabled by using these utilities persist until the file system is unmounted.

The following sections describe these system components in more detail, explain when to use these files and commands, and show the order in which they take precedence. In addition to the following sections, the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* includes information on mounting a file system.

The mount(1M) Command

The Sun Solaris mount(1M) command mounts the file system and allows you to specify settings that override the settings specified in the /etc/vfstab file and in the /etc/opt/SUNWsamfs/samfs.cmd file. For example, you can specify the stripe width, readahead, writebehind, high and low water marks for disk cache utilization, and so on.

One way to use the mount(1M) command in conjunction with the samfs.cmd file is to use the samfs.cmd file as your main location for mount options and to use options on the mount(1M) command when experimenting with or tuning your system. The mount(1M) command options override both the /etc/vfstab entries and the directives in the samfs.cmd file.

Example. The following command mounts file system qfs1 at /work with setuid execution disallowed. The qfs1 file system name is the equipment identifier. This also appears in the mcf file's Equipment Identifier field for this file system. The mount options are separated by a comma without an intervening space.

```
# mount –o nosuid,qwrite qfs1 /work
```

For more information on the mount(1M) command, see the mount_samfs(1M) man page.

The /etc/vfstab File

The /etc/vfstab Sun Solaris system file must contain a line for each Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system that is defined in the mcf file. For each file system, you must provide information for the seven fields shown in TABLE 4-1.

TABLE 4-1 Fields in the /etc/vfstab File

Field Number	Content
1	The file system family set name.
2	The file system to samfsck(1M).
3	The mount point.
4	The file system type. This is always samfs, even for Sun QFS and Sun SAM-QFS file systems.

TABLE 4-1 Fields in the `/etc/vfstab` File *(Continued)*

Field Number	Content
5	The `samfsck`(1M) pass.
6	Mount at boot options.
7	Mount parameters.

The fields in the `/etc/vfstab` file must be separated by either space or tab characters. The mount parameters in the seventh field, however, must each be separated by a comma character (,) without any intervening spaces.

Example. The following is an example of an `/etc/vfstab` file.

```
qfs1    -    /qfs    samfs    -    yes    stripe=0
```

The mount parameters field can contain any of the mount parameters listed as arguments to the `-o` option on the `mount_samfs`(1M) man page. These parameters are identical to those that you can specify as directive lines in the `samfs.cmd` file or as arguments to the `-o` option on the `mount`(1M) command. As with the `samfs.cmd` file, you can include specifications for various I/O settings, readahead, writebehind, the stripe width, various storage and archive management (SAM) settings, Qwrite, and other features.

For more information on possible mount parameters, see the `mount_samfs`(1M) man page. For more information on modifying the `/etc/vfstab` file, see the `vfstab`(4) man page.

The `samfs.cmd` File

The `/etc/opt/SUNWsamfs/samfs.cmd` file allows you to specify mount parameters for all your Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. This file can be useful when you have multiple file systems configured and you want to specify the same mount parameters for them.

The list of possible mount parameters is very comprehensive. The possible mount parameters you can specify pertain to I/O settings, readahead, writebehind, the stripe width, various storage and archive management (SAM) settings, Qwrite, and other features.

Using this file allows you to define all your mount parameters in one place in an easily readable format. Directives specified toward the beginning of this file are global directives, and they apply to all Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. The second part of this file allows you to indicate the specific parameters that you want to apply to each individual file system. The ability to specify the common parameters once, and only in one place, differentiates this file from the `/etc/vfstab` file, in which you must specify all mount parameters for each file system in the seventh field.

The mount parameters that can be specified in the `samfs.cmd` file are nearly identical to those that you can specify in the `/etc/vfstab` file or as arguments to the `-o` option on the `mount`(1M) command. For more information on the mount parameters that can be specified in this file, see the `samfs.cmd`(4) man page.

In the `samfs.cmd` file, the directives are written one per line. The file can contain comments, which must begin with a pound character (#). Characters that appear to the right of the pound character are treated as comments.

Directives that appear before any `fs =` line apply globally to all file systems. A line that starts with `fs =` must precede directives that are specific to a particular file system. Directives specific to a particular file system override global directives.

The following example `samfs.cmd` file sets the low and high water marks for disk cache utilization and specifies individualized parameters for two file systems:

```
low = 50
high = 75
       fs = samfs1
high = 65
writebehind = 512
readahead = 1024
       fs = samfs5
       partial = 64
```

The directives in the `samfs.cmd` file serve as defaults and override any default system settings, but arguments to the `mount`(1M) command override any directives in this file. Entries in the `/etc/vfstab` file also override directives specified in the `samfs.cmd` file.

For information on which directives can be entered in the `samfs.cmd` file, see the `samfs.cmd`(4) man page. For information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

To Unmount a File System

You can use the Sun Solaris `umount`(1M) command to unmount Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems.

On Sun SAM-FS and Sun SAM-QFS file systems, you must issue commands to stop the archiver prior to unmounting the file system. The following procedure shows you how to idle the archiver and unmount the file system. You do not need to idle the archiver if you are using a Sun QFS file system.

1. **Issue a `samcmd aridle fs.`*fsname* command for the file system. (Optional)**

   Perform this step if you are unmounting a Sun SAM-FS or Sun SAM-QFS file system.

   ```
   # samcmd aridle fs.samqfs2
   ```

   This step in the procedure cleanly halts the archiving for file system `samqfs2`. Specifically, it allows archiving operations to halt at a logical place before stopping the daemons.

2. **Issue a `samd stop` command. (Optional)**

   Perform this step if you are unmounting a Sun SAM-FS or Sun SAM-QFS file system.

   ```
   # samd stop
   ```

3. **Unmount the file system.**

   ```
   # umount /samqfs
   ```

   Several conditions can be present in a file system at unmounting time, so you might need to issue the `umount`(1M) command a second time. If the file system still does not unmount, use `unshare`(1M), `fuser`(1M), or other commands in conjunction with the `umount`(1M) command. Unmounting procedures are also described in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

To Check File System Integrity

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems write validation records in all records critical to file system operations: directories, indirect blocks, and inodes. If corruption is detected while searching a directory, an EDOM error is returned, and the directory is not processed. If an indirect block is not valid, an ENOCSI error is returned, and the file is not processed. The following list summarizes these error indicators:

Error	Sun Solaris Meaning	Sun QFS, Sun SAM-FS, Sun SAM-QFS Meaning
EDOM	Argument is out of domain	Values in validation records are out of range.
ENOCSI	No CSI structure available	Links between structures are invalid.

In addition, inodes are validated and cross checked with directories.

You should monitor the log file specified in `/etc/syslog.conf` for the preceding errors. You should watch the `/var/adm/messages` file for device errors. If a discrepancy is noted, the file system should be unmounted and checked using the `samfsck`(1M) command. You can send output from `samfsck`(1M) to both your screen and to a file by using it in conjunction with the `tee`(1) command, as follows.

C shell:

```
# samfsck –V family_set_name |& tee file
```

Bourne shell:

```
hostname# samfsck –V family_set_name 2>&1 | tee file
```

Nonfatal errors returned by `samfsck`(1M) are preceded by `NOTICE`. Nonfatal errors are lost blocks and orphans. The file system is still consistent if `NOTICE` errors are returned. These nonfatal errors can be repaired during a convenient, scheduled maintenance outage.

Fatal errors are preceded by `ALERT`. These errors include duplicate blocks, invalid directories, and invalid indirect blocks. The file system is not consistent if these errors occur. Notify Sun Microsystems if the `ALERT` errors cannot be explained by a hardware malfunction.

For more information on the `samfsck`(1M) and `tee`(1) commands, see the `samfsck`(1M) and `tee`(1) man pages.

To Repair a File System

If the `samfsck`(1M) command detects file system corruption by returning `ALERT` messages, the reason for the corruption should be determined. If hardware is faulty, it should be repaired prior to repairing the file system. Then the file system should be repaired by specifying the `–F` and `-V` options to the `samfsck`(1M) command, as follows:

```
# samfsck -F –V family_set_name
```

The `samfsck`(1M) command should be run when a file system is not mounted.

To Preserve Information for an Upgrade

If you are about to add or change disks, controllers, or other equipment in your environment, it can be difficult to correct or regenerate all the file system descriptions in the `mcf` file. The `samfsconfig`(1M) command can help you by generating information about your file system and file system components after making these changes.

The `samfsconfig`(1M) command examines devices and determines if any of them have Sun QFS, Sun SAM-FS, or Sun SAM-QFS superblocks on them. Using information from the discovered superblocks, it aggregates the devices into a format similar to an `mcf` file. You can save this format and edit it to recreate a damaged, missing, or incorrect `mcf` file.

This command returns information about each device that you specify and writes this information to `stdout`. The command can retrieve the family set number of the base device (the file system itself), the file system type (`ma` or `ms`), and whether the file system is a Sun QFS shared file system.

Irregularities are flagged with one of the following:

- A pound sign (#). This indicates incomplete family set information.
- A greater-than sign (>). This indicates that more than one device name refers to a particular file system element.

If necessary, this command's output can be used to regenerate the file system portions of your `mcf` file if your system is reconfigured or experiences a disaster. The following examples show output from the `samfsconfig`(1M) command.

Example 1

In this example, the system administrator has put a list of device names into a file. These device names are not accounted for in the environment. The system administrator wants to examine only these devices for Sun QFS, Sun SAM-FS, and Sun SAM-QFS family sets. The results show some old fragments of family sets and several complete instances.

CODE EXAMPLE 4-1 Output From `samfsconfig`(1M) Command

```
mn# samfsconfig -v ‘cat /tmp/dev_files‘
Device ’/dev/dsk/c0t0d0s0’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s1’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s3’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s4’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s5’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s6’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t0d0s7’ doesn’t have a SAM-FS superblock (SBLK).
Couldn’t open ’/dev/dsk/c0t1d0s0’;  errno=5.
Couldn’t open ’/dev/dsk/c0t1d0s1’;  errno=5.
Device ’/dev/dsk/c0t1d0s3’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c0t1d0s4’ has a SAM-FS superblock.
Device ’/dev/dsk/c0t1d0s5’ has a SAM-FS superblock.
Device ’/dev/dsk/c0t1d0s6’ has a SAM-FS superblock.
Couldn’t open ’/dev/dsk/c0t1d0s7’;  errno=5.
Couldn’t open ’/dev/dsk/c0t6d0s0’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s1’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s3’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s4’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s5’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s6’;  errno=16.
Couldn’t open ’/dev/dsk/c0t6d0s7’;  errno=16.
Couldn’t open ’/dev/dsk/c1t0d0s3’;  errno=5.
Couldn’t open ’/dev/dsk/c1t0d0s4’;  errno=5.
Couldn’t open ’/dev/dsk/c1t0d0s5’;  errno=5.
Device ’/dev/dsk/c1t0d0s6’ doesn’t have a SAM-FS superblock (SBLK).
Couldn’t open ’/dev/dsk/c1t0d0s7’;  errno=5.
Couldn’t open ’/dev/dsk/c1t1d0s0’;  errno=2.
Couldn’t open ’/dev/dsk/c1t2d0s3’;  errno=5.
Couldn’t open ’/dev/dsk/c1t2d0s4’;  errno=5.
Couldn’t open ’/dev/dsk/c1t2d0s5’;  errno=5.
Device ’/dev/dsk/c1t2d0s6’ doesn’t have a SAM-FS superblock (SBLK).
Couldn’t open ’/dev/dsk/c1t2d0s7’;  errno=5.
Could not read from device ’/dev/dsk/c1t3d0s0’; errno=5.
Couldn’t open ’/dev/dsk/c1t4d0s3’;  errno=5.
Couldn’t open ’/dev/dsk/c1t4d0s4’;  errno=5.
Couldn’t open ’/dev/dsk/c1t4d0s5’;  errno=5.
Device ’/dev/dsk/c1t4d0s6’ doesn’t have a SAM-FS superblock (SBLK).
```

CODE EXAMPLE 4-1 Output From `samfsconfig`(1M) Command *(Continued)*

```
Device ’/dev/dsk/c1t4d0s7’ doesn’t have a SAM-FS superblock (SBLK).
Couldn’t open ’/dev/dsk/c1t5d0s3’;  errno=5.
Couldn’t open ’/dev/dsk/c1t5d0s4’;  errno=5.
Couldn’t open ’/dev/dsk/c1t5d0s5’;  errno=5.
Device ’/dev/dsk/c1t5d0s6’ doesn’t have a SAM-FS superblock (SBLK).
Couldn’t open ’/dev/dsk/c1t5d0s7’;  errno=5.
Device ’/dev/dsk/c3t0d0s0’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c3t0d0s1’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c3t0d0s3’ has a SAM-FS superblock.
Device ’/dev/dsk/c3t0d0s4’ has a SAM-FS superblock.
Couldn’t open ’/dev/dsk/c3t0d0s7’;  errno=5.
Device ’/dev/dsk/c3t1d0s0’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c3t1d0s1’ doesn’t have a SAM-FS superblock (SBLK).
Device ’/dev/dsk/c3t1d0s3’ has a SAM-FS superblock.
Device ’/dev/dsk/c3t1d0s4’ has a SAM-FS superblock.
Couldn’t open ’/dev/dsk/c3t1d0s7’;  errno=5.
Device ’/dev/dsk/c4t0d0s0’ has a SAM-FS superblock.
Could not read from device ’/dev/dsk/c4t0d0s1’; errno=5.
Could not read from device ’/dev/dsk/c4t0d0s3’; errno=5.
Could not read from device ’/dev/dsk/c4t0d0s4’; errno=5.
Could not read from device ’/dev/dsk/c4t0d0s5’; errno=5.
Device ’/dev/dsk/c4t0d0s6’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t0d0s7’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t1d0s0’ has a SAM-FS superblock.
Could not read from device ’/dev/dsk/c4t1d0s1’; errno=5.
Could not read from device ’/dev/dsk/c4t1d0s3’; errno=5.
Could not read from device ’/dev/dsk/c4t1d0s4’; errno=5.
Could not read from device ’/dev/dsk/c4t1d0s5’; errno=5.
Device ’/dev/dsk/c4t1d0s6’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t1d0s7’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t2d0s0’ has a SAM-FS superblock.
Could not read from device ’/dev/dsk/c4t2d0s1’; errno=5.
Could not read from device ’/dev/dsk/c4t2d0s3’; errno=5.
Could not read from device ’/dev/dsk/c4t2d0s4’; errno=5.
Could not read from device ’/dev/dsk/c4t2d0s5’; errno=5.
Device ’/dev/dsk/c4t2d0s6’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t2d0s7’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t3d0s0’ has a SAM-FS superblock.
Could not read from device ’/dev/dsk/c4t3d0s1’; errno=5.
Could not read from device ’/dev/dsk/c4t3d0s3’; errno=5.
Could not read from device ’/dev/dsk/c4t3d0s4’; errno=5.
Could not read from device ’/dev/dsk/c4t3d0s5’; errno=5.
Device ’/dev/dsk/c4t3d0s6’ has a SAM-FS superblock.
Device ’/dev/dsk/c4t3d0s7’ has a SAM-FS superblock.
19 SAM-FS devices found.
#
# Family Set ’samfs2’ Created Mon Jun 25 10:37:52 2001
```

CODE EXAMPLE 4-1 Output From `samfsconfig`(1M) Command *(Continued)*

```
#
# Missing slices
# Ordinal 1
# /dev/dsk/c0t1d0s6    12    md   samfs2  -
#
# Family Set 'samfs1' Created Wed Jul 11 08:47:38 2001
#
# Missing slices
# Ordinal 1
# /dev/dsk/c0t1d0s4    12    md   samfs1  -
# Ordinal 2
# /dev/dsk/c0t1d0s5    13    md   samfs1  -
#
# Family Set 'samfs2' Created Sat Nov  3 17:22:44 2001
#
samfs2 ma 30 samfs2 - shared
/dev/dsk/c4t0d0s6    31    mm   samfs2  -
/dev/dsk/c4t1d0s6    32    mr   samfs2  -
/dev/dsk/c4t2d0s6    33    mr   samfs2  -
#
# Family Set 'qfs1' Created Wed Nov  7 15:16:19 2001
#
qfs1 ma 10 qfs1 -
/dev/dsk/c3t0d0s3    11    mm   qfs1  -
/dev/dsk/c3t0d0s4    12    g0   qfs1  -
/dev/dsk/c3t1d0s3    13    g0   qfs1  -
/dev/dsk/c3t1d0s4    14    g0   qfs1  -
#
# Family Set 'sharefsx' Created Wed Nov  7 16:55:19 2001
#
sharefsx ma 200 sharefsx - shared
/dev/dsk/c4t0d0s0    210    mm   sharefsx  -
/dev/dsk/c4t1d0s0    220    mr   sharefsx  -
/dev/dsk/c4t2d0s0    230    mr   sharefsx  -
/dev/dsk/c4t3d0s0    240    mr   sharefsx  -
#
# Family Set 'samfs5' Created Tue Nov 27 16:32:28 2001
#
samfs5 ma 80 samfs5 -
/dev/dsk/c4t3d0s6    82    mm   samfs5  -
/dev/dsk/c4t3d0s7    83    g0   samfs5  -
/dev/dsk/c4t0d0s7    84    g0   samfs5  -
/dev/dsk/c4t1d0s7    85    g1   samfs5  -
/dev/dsk/c4t2d0s7    86    g1   samfs5  -
```

Example 2

In this example, the devices flagged with a greater-than sign (>) are duplicated. The `s0` slice starts at the start of disk, as does the whole disk (`s2`) slice. This is the style of output obtained in a Sun Solaris 9 OE.

```
# samfsconfig /dev/dsk/c3t*
#
# Family Set 'shsam1' Created Wed Oct 17 14:57:29 2001
#
shsam1 160 ma shsam1 shared
> /dev/dsk/c3t50020F23000055A8d0s2    161    mm   shsam1  -
> /dev/dsk/c3t50020F23000055A8d0s0    161    mm   shsam1  -
/dev/dsk/c3t50020F23000055A8d0s1    162    mr   shsam1  -
> /dev/dsk/c3t50020F23000078F1d0s0    163    mr   shsam1  -
> /dev/dsk/c3t50020F23000078F1d0s2    163    mr   shsam1  -
/dev/dsk/c3t50020F23000078F1d0s1    164    mr   shsam1  -
```

Example 3

In this example, the whole disk slice (slice 2) is left off of the command line. This is the style of output obtained in a Sun Solaris 9 OE.

```
# samfsconfig /dev/dsk/c3t*s[013-7]
#
# Family Set 'shsam1' Created Wed Oct 17 14:57:29 2001
#
shsam1 160 ma shsam1 shared
/dev/dsk/c3t50020F23000055A8d0s0    161    mm   shsam1  -
/dev/dsk/c3t50020F23000055A8d0s1    162    mr   shsam1  -
/dev/dsk/c3t50020F23000078F1d0s0    163    mr   shsam1  -
/dev/dsk/c3t50020F23000078F1d0s1    164    mr   shsam1  -
```

For more information on this command, see the `samfsconfig`(1M) man page.

To Prepare for a Hardware Upgrade

Whether upgrading a server, adding a new tape drive, adding an automated library, or installing a different drive into an existing automated library, it is best to perform advance planning. This section is intended to prepare you for hardware upgrades to devices within your environment.

The following actions are recommended prior to the upgrade:

- Determine if the hardware addition or change requires a new license from Sun Microsystems.

  Examples of changes that do not require a license upgrade include adding memory and increasing disk cache. Examples of changes that require a license upgrade include adding more slots in an automated library and changing the model of your server.

- Read the hardware manufacturer's installation instructions carefully. Also read the documentation on adding hardware in your Sun Solaris system administrator documentation.
- The system should be quiet with no users logged in.
- Check the equipment ordinals between your old and new master configuration files. For information on the `mcf` file, see the `mcf`(4) man page.
- Decide whether or not the backup copies you have on hand are sufficient. For information on backing up your data and metadata, see the procedures described in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

  In a Sun QFS environment, the `qfsdump`(1M) command dump all data and metadata. For more information on this process, see the `qfsdump`(1M) man page.

  In Sun SAM-FS and Sun SAM-QFS environments, you can use the `samfsdump`(1M) command to dump all metadata. You must ensure that all files that need to be archived have an archive copy. Use the `sfind`(1) command on each Sun SAM-FS or Sun SAM-QFS file system to see which files do not have an archive copy. In the following example, `/sam` is the mount point.

```
# sfind /sam !-archived !-empty -type f -print
```

- In Sun SAM-FS and Sun SAM-QFS environments, insure that the archiver is in `wait` mode. The archiver must be in `wait` mode, and not running, during an upgrade.

  You can idle the archiver in one of the following ways:

- Insert a `wait` directive into the `/etc/opt/SUNWsamfs/archiver.cmd` file. For more information on the `wait` directive and the `archiver.cmd` file, see the `archiver.cmd`(4) man page.
- Use the `samu`(1M) operator utility.
- Issue the following command:

```
# samcmd aridle
```

For more information, see the `samcmd`(1M) man page.

To Add Disk Cache to a File System

At some point, you might want to add disk partitions or disk drives in order to increase the disk cache for a file system. This is accomplished by updating the `mcf` file and using the `samgrowfs`(1M) command. There is no need to reinitialize or restore the file system.

In Sun SAM-FS and Sun SAM-QFS environments, note that when adding disks or partitions, the equipment ordinal of the historian might be updated. The equipment ordinal of the historian is automatically generated by the system unless you specifically call it out. For more information, see the `historian`(7) man page.

1. **Unmount the file system you want to expand.**

   For information on unmounting a file system, see "To Unmount a File System" on page 69.

2. **If you want to rename the file system during this procedure, use the `samfsck`(1M) command with its `–R` and `-F` options to rename the file system. (Optional)**

   For more information on this command, see the `samfsck`(1M) man page.

3. **Edit the `/etc/opt/SUNWsamfs/mcf` file.**

   You can configure up to 252 disk partitions in a file system. New partitions must be added after the existing disk partitions. Save the changes, and quit the editor.

   To increase the size of a Sun QFS file system, at least one new metadata partition must be added. Metadata partitions require an Equipment Type of `mm`. Zero or more data partitions can be added.

   Do not change the equipment identifier name in the `/etc/opt/SUNWsamfs/mcf` file. If the name in the `mcf` file does not match the name in the superblock, the file systems can no longer be mounted. Instead, the following message is logged in `/var/adm/messages`:

```
WARNING SAM-FS superblock equipment identifier <id>s on eq <eq>
does not match <id> in mcf
```

4. **Enter the `sam-fsd`(1M) command to check for errors in the `mcf` file.**

   For more information, see the `sam-fsd`(1M) man page.

```
# sam-fsd
```

   If the `sam-fsd`(1M) command output indicates that there are errors in the `mcf` file, fix them before proceeding to the next step in this procedure.

5. **Enter the `samd config` command to reinitialize the `mcf` file.**

```
# samd config
```

   For more information on the `samd`(1M) command, see the `samd`(1M) man page.

6. **Enter the `samgrowfs`(1M) command on the file system that is being expanded.**

   For example:

```
# samgrowfs samfs1
```

   If you renamed your file system, run the `samgrowfs`(1M) command on the new name. For more information on this command, see the `samgrowfs`(1M) man page.

7. **Mount the file system.**

To Replace Disks in a File System

At some point, you might want to perform the following tasks:

- Change disks or partitions
- Add disks or partitions
- Remove disks or partitions

To accomplish these tasks, you need to back up and recreate the file system by following the steps in this procedure.

1. **Back up all site-customized system files and configuration files.**

   Depending on your software, these files can include `mcf`, `archiver.cmd`, `defaults.conf`, `samfs.cmd`, `inquiry.conf`, and so on. Back up these files for all file systems in your Sun QFS, Sun SAM-FS and Sun SAM-QFS environments. Also make sure that you have backup copies of files in the `/etc/opt/SUNWsamfs` directory, files in the `/var/opt/SUNWsamfs` directory, library catalogs, the historian, and any parameter files for network-attached automated libraries.

   In Sun SAM-FS and Sun SAM-QFS environments, if you do not know the names and locations of your catalog files, examine the `mcf` file with `vi`(1) or another viewing command and find the first `rb` entry in the `mcf` file. That entry contains the name of the library catalog file. If this is not present, the default location is `/var/opt/SUNWsamfs/catalog`.

2. **Ensure that each file system to be modified is backed up.**

   The file systems should be backed up regularly according to your site's policies. This is described as the last step in the installation procedure. If you are comfortable with the backup files that already exist for your file systems, there is no need to back them up again now. If, however, you need to back up your file systems to preserve information created since the last dump file was created, do so now. For information on how to create a dump file, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

   Note that if you are using the Sun SAM-FS or Sun SAM-QFS file system, the `samfsdump`(1M) command issues warnings when creating the dump file if it encounters unarchived files in the file system. If warnings are issued, these files need to be archived before unmounting the file systems.

3. **Unmount the file system.**

   For information on unmounting a file system, see "To Unmount a File System" on page 69.

4. **If you want to rename the file system during this procedure, use the** `samfsck`**(1M) command with its** `–R` **and** `-F` **options to rename the file system. (Optional)**

   For more information on this command, see the `samfsck`(1M) man page.

5. **Edit the** `/etc/opt/SUNWsamfs/mcf` **file.**

   You can configure up to 252 disk partitions in a file system. Edit the `mcf` file to add or delete disks or partitions. New partitions must be added after existing disk partitions. Save the changes, and quit the editor.

   To increase the size of a Sun QFS file system, at least one new metadata partition must be added. Metadata partitions require an Equipment Type of `mm`. Zero or more data partitions can be added.

   Do not change the Equipment Identifier name in the `/etc/opt/SUNWsamfs/mcf` file. If the name in the `mcf` file does not match the name in the superblock, the file systems can no longer be mounted. Instead, the following message is logged in `/var/adm/messages`:

```
WARNING SAM-FS superblock equipment identifier <id>s on eq <eq>
does not match <id> in mcf
```

6. **Enter the** `sam-fsd`**(1M) command to check for errors in the** `mcf` **file.**

   For more information, see the `sam-fsd`(1M) man page.

```
# sam-fsd
```

   If the `sam-fsd`(1M) command output indicates that there are errors in the `mcf` file, fix them before proceeding to the next step in this procedure.

7. **Enter the following command to reinitialize the** `mcf` **file:**

```
# samd config
```

   For more information on the `samd` command, see the `samd`(1M) man page.

8. **Use the** `sammkfs`**(1M) command to make a new file system.**

   For example, the following command creates `samfs10`.

```
# sammkfs samfs10
```

9. **Mount the file system.**

10. **Use the** `cd`**(1) command to change to the mount point of the file system.**

11. **Use the `samfsrestore`(1M) or `qfsrestore`(1M) command to restore each file system using the dump file you had or using the dump file created in Step 2.**

    For information on using these commands, see the `samfsdump`(1M) or `qfsdump`(1M) man pages, or see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Disaster Recovery Guide*.

12. **Use the `restore.sh`(1M) script to stage back all files that had been online.**

    For example:

    ```
    # restore.sh stage_file
    ```

    For more information on the `restore.sh`(1M) script, see the `restore.sh`(1M) man page.

To Upgrade a Host System

When it comes time to upgrade the host system being used for the file system, you should take the following into account:

- It is wise to move to the new host while the existing host is still in operation. This allows time to install, configure, and test the new hardware platform with your applications.
- Moving to a new host system is equivalent to installing the Sun QFS, Sun SAM-FS, or Sun SAM-QFS software for the first time. In Sun SAM-FS and Sun SAM-QFS environments, you need to reinstall the software and update the configuration files (specifically the `mcf` file, the `/kernel/drv/st.conf` file, and the `/etc/opt/SUNWsamfs/inquiry.conf` file). In addition, you need to copy your existing `archiver.cmd` and `defaults.conf` files to the new system, configure system logging, and so on.

  You can use the installation instructions in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* when re-installing the software.
- The license key needs to be updated. License keys are tied to the CPU host ID. Replacing the system requires a new license.
- Before powering down the old host system, decide whether or not the backup copies you have on hand are sufficient. You might need to create new dump files at this time. A dump file is used to recreate the file system on the new server. For more information on creating a dump file, see the `qfsdump`(1M) or `samfsdump`(1M) man pages or see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

To Upgrade Your Sun Solaris OE in a Sun SAM-FS or Sun SAM-QFS Environment

Many of the steps involved in upgrading your Sun Solaris level are identical to the steps involved in upgrading your Sun SAM-FS or Sun SAM-QFS environment. Some of the steps in this procedure reference procedures in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

1. **Obtain the software upgrade.**

   The Sun SAM-FS and Sun SAM-QFS software supports various levels of the Sun Solaris OE. You should not reinstall your old Sun SAM-FS or Sun SAM-QFS software onto your newly upgraded Sun Solaris OE.

   In addition, depending on the revision level currently installed and the level to which you are upgrading, you might need a new software license.

   Contact your ASP or Sun Microsystems to obtain new copies of the software and to determine whether or not your site needs a new license.

2. **Back up all site-customized system files and configuration files.**

   These files include `mcf`, `archiver.cmd`, `defaults.conf`, `samfs.cmd`, `inquiry.conf`, and so on. Back up these files for all file systems in your Sun SAM-FS and Sun SAM-QFS environments.

   Also make sure that you have backup copies of files in the `/etc/opt/SUNWsamfs` directory, files in the `/var/opt/SUNWsamfs` directory, library catalogs, the historian, and any parameter files for network-attached automated libraries.

   If you do not know the names and locations of your catalog files, examine the `mcf` file with `vi`(1) or another viewing command and find the first `rb` entry in the `mcf` file. That entry contains the name of the library catalog file. If this is not present, the default location is `/var/opt/SUNWsamfs/catalog`.

3. **Ensure that each file system affected is backed up.**

   The file systems should be backed up regularly according to your site's policies. This is described as the last step in the installation procedure. If you are comfortable with the backup files that already exist for your file systems, there is no need to back them up again now. If, however, you need to back up your file systems to preserve information created since the last dump file was created, do so now.

   Note that if you are using the Sun SAM-FS or Sun SAM-QFS file system, the `samfsdump`(1M) command issues warnings when creating the dump file if it encounters unarchived files in the file system. If warnings are issued, these files need to be archived before unmounting the file systems.

4. **Unmount the file systems.**

   For information on unmounting a file system, see "To Unmount a File System" on page 69.

5. **Remove existing Sun SAM-FS or Sun SAM-QFS software.**

   Use the `pkgrm`(1M) command to remove the existing software. You must remove all existing Sun SAM-FS and Sun SAM-QFS packages before installing the new packages or the new operating system level.

   For example, the following command removes the `SUNWsamtp` and the `SUNWsamfs` packages in a Sun SAM-FS or Sun SAM-QFS environment. The `SUNWsamfs` package must be removed last. Note that the `SUNWsamtp` package is an optional tools package, and it might not be installed on your system. The `pkgrm`(1M) command is as follows:

   ```
   # pkgrm SUNWsamtp SUNWsamfs
   ```

   The information in this step assumes that you are removing software packages at the 4.0 release level or later. The software package names changed as of the 4.0 releases. If you have software packages on your system that were released prior to the 4.0 releases, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* for information on removing them.

6. **Upgrade the Sun Solaris OE.**

   Install the new Sun Solaris revision using the Sun Solaris upgrade procedures for the OE level you are installing.

7. **Add the `SUNWsamfs` package.**

   The Sun SAM-FS and Sun SAM-QFS software package, `SUNWsamfs`, uses the Sun Solaris packaging utilities for adding and deleting software. As such, you must be logged in as superuser (root) to make changes to software packages. The `pkgadd`(1M) command prompts you to confirm various actions necessary to upgrade the Sun SAM-FS and Sun SAM-QFS package.

   On the CD-ROM, the Sun SAM-FS and Sun SAM-QFS package resides in the `/cdrom/cdrom0` directory.

   Run the `pkgadd`(1M) command, as follows, to upgrade the package, answering `yes` to each question:

   ```
   # pkgadd -d SUNWsamfs
   ```

During the installation, the system detects the presence of conflicting files and prompts you to indicate whether or not you want to continue with the installation. You can go to another window and copy the files you wish to save to an alternate location.

8. **Update the license keys. (Optional)**

   Depending on the Sun SAM-FS and Sun SAM-QFS software revision you had, and the revision to which you are upgrading, you might need to obtain new license keys for your Sun SAM-FS or Sun SAM-QFS software. Contact your ASP or Sun Microsystems for help on determining if you need a new license.

   If you are upgrading from a release prior to 4.0, you need to place a new license key in the following file:

   ```
   /etc/opt/SUNWsamfs/LICENSE.4.0
   ```

   For more information, see the licensing information in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

9. **Mount the file system(s). (Optional)**

   You must perform this step if you have not modified the `/etc/vfstab` file to have `yes`.

   Use the `mount`(1M) command to mount the file systems and continue operation with the upgraded Sun SAM-FS or Sun SAM-QFS software.

   In the following example, `samfs1` is the name of the file system to be mounted.

   ```
   # mount samfs1
   ```

To Upgrade Your Sun Solaris OE in a Sun QFS Environment

Many of the steps involved in upgrading your Sun Solaris level are identical to the steps involved in upgrading your Sun QFS environment. Some of the steps in this procedure reference procedures in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide.*

1. **Obtain the software upgrade.**

Sun QFS software supports various levels of the Sun Solaris OE. You should not reinstall your old Sun QFS software onto your newly upgraded Sun Solaris system.

In addition, depending on the revision level currently installed and the level to which you are upgrading, you may need a new Sun QFS license.

Contact your ASP or Sun Microsystems to obtain new copies of the software and to determine whether or not your site needs a new license.

2. **Back up all site-customized system files and configuration files.**

   These files include `mcf`, `defaults.conf`, `samfs.cmd`, and so on. Back up these files for all file systems in your Sun QFS environment. Also make sure that you have backup copies of files in the `/etc/opt/SUNWsamfs` directory.

3. **Ensure that each file system affected is backed up.**

   The file systems should be backed up regularly according to your site's policies. This is described as the last step in the installation procedure. If you are comfortable with the backup files that already exist for your file systems, there is no need to back them up again now. If, however, you need to back up your file systems to preserve information created since the last dump file was created, do so now. For information on how to create a dump file, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

4. **Unmount the file systems.**

   Unmount the file systems using the procedure described earlier in this chapter in the subsection called "To Unmount a File System" on page 69.

5. **Remove existing Sun QFS software.**

   Use the `pkgrm`(1M) command to remove the existing software. You must remove the existing Sun QFS package before installing the new package or the new operating system level.

   For example, the following command removes the `SUNWqfs` package in a Sun QFS environment:

   ```
   # pkgrm SUNWqfs
   ```

   The information in this step assumes that you are removing a software package at the 4.0 release level or later. The software package names changed as of the 4.0 releases. If you have a software package on your system that was released prior to the 4.0 releases, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* for information on removing them.

6. **Upgrade the Sun Solaris OE.**

Install the new Sun Solaris revision using the Sun Solaris upgrade procedures for the OE level you are installing.

7. **Add the package.**

   The Sun QFS software package uses the Sun Solaris packaging utilities for adding and deleting software. As such, you must be logged in as superuser (root) to make changes to software packages. The `pkgadd`(1M) command prompts you to confirm various actions necessary to upgrade the Sun QFS package.

   On the CD-ROM, the Sun QFS package resides in the `/cdrom/cdrom0` directory.

   Run the `pkgadd`(1M) command to upgrade the package, answering `yes to` each question:

   ```
   # pkgadd -d SUNWqfs
   ```

   During the installation, the system detects the presence of conflicting files and prompts you to indicate whether or not you want to continue with the installation. You can go to another window and copy the files you wish to save to an alternate location.

8. **Update the license keys. (Optional)**

   Depending on the Sun QFS software revision you had, and the revision to which you are upgrading, you might need to obtain new license keys for your Sun QFS software. Contact your ASP or Sun Microsystems for help on determining if you need a new license.

   If you are upgrading from a Sun QFS release prior to 4.0, you need to place a new license key in the following file:

   ```
   /etc/opt/SUNWsamfs/LICENSE.4.0
   ```

   For more information, see the licensing information in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

9. **Update the `mcf` file. (Optional)**

   If device names have changed, it might be necessary to update the `mcf` file to match the new device names. Veryify the new device names, and then follow the procedure in “To Initialize or Reinitialize an mcf or defaults.conf File” on page 62.

10. **Mount the file system(s). (Optional)**

    Perform this step if you have not modified the `/etc/vfstab` file to have `yes`.

Use the procedure described in “To Mount a File System” on page 66. Continue operation with the upgraded Sun QFS software.

CHAPTER **5**

Sun QFS Shared File System

A Sun QFS shared file system is a distributed file system that can be mounted on Solaris host systems. In a Sun QFS shared file system environment, one Solaris host acts as the metadata server for the file system, and additional hosts can be configured as clients. More than one host can be configured as a potential metadata server, but only one system can be configured as the metadata server at any one time. There is no limit to the number of Sun QFS shared file system mount points.

The advantage of the Sun QFS shared file system is that file data is passed directly from the Fibre Channel disks to the hosts. Data travels via local path I/O (also known as *direct access I/O*). This is in contrast to NFS, which transfers data over the network.

Note – The Sun QFS Shared File System is supported in only the Sun Solaris 8 and 9 operating environments (OEs).

This chapter describes how to configure and maintain the Sun QFS shared file system, and it specifically includes the following sections:

- "Overview" on page 90
- "Configuring the Sun QFS Shared File System" on page 92
- "Mounting and Unmounting Sun QFS Shared File Systems" on page 118
- "Adding and Removing a Client Host" on page 119
- "Changing the Metadata Server" on page 123
- "Daemons" on page 131
- "Mount Options in a Sun QFS Shared File System" on page 132
- "Mount Semantics in a Sun QFS Shared File System" on page 136
- "File Locking in a Sun QFS Shared File System" on page 137
- "Troubleshooting a Failed or Hung sammkfs(1M) or mount(1M) Command" on page 137

Overview

The Sun QFS shared file system can be configured in either a Sun QFS or a Sun SAM-QFS environment, as follows:

- If configured in a Sun QFS environment, no archiving or staging occurs, so no network connection to archive media is necessary. This chapter addresses archive media frequently, and if you are running in a Sun QFS standalone environment, you can ignore this information.
- If configured in a Sun SAM-QFS environment, each host that can become the metadata server needs to be connected to the same archive media repository. The archive media can consist of a library with removable media devices (tape or magneto-optical drives). If disk archiving is implemented, the archive media can consist of one or more file systems. The archive media must be specified in the `mcf` file or in the `diskvols.conf` file on each host that can become a metadata server.

  In a Sun SAM-QFS environment, the active metadata server is the only host upon which the staging (`sam-stagerd`) and archiving (`sam-archiverd`) daemons are active. The metadata server is designated as the server from which all file requests are staged.

This chapter describes how to configure and maintain a Sun QFS shared file system. It assumes that you have installed the Sun QFS or Sun SAM-QFS software on the host systems according to the instructions in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

Note – The Sun QFS shared file system cannot be configured in a Sun SAM-FS (an `ms` file system) environment.

FIGURE 5-1 illustrates a Sun QFS shared file system configuration in a Sun SAM-QFS environment.

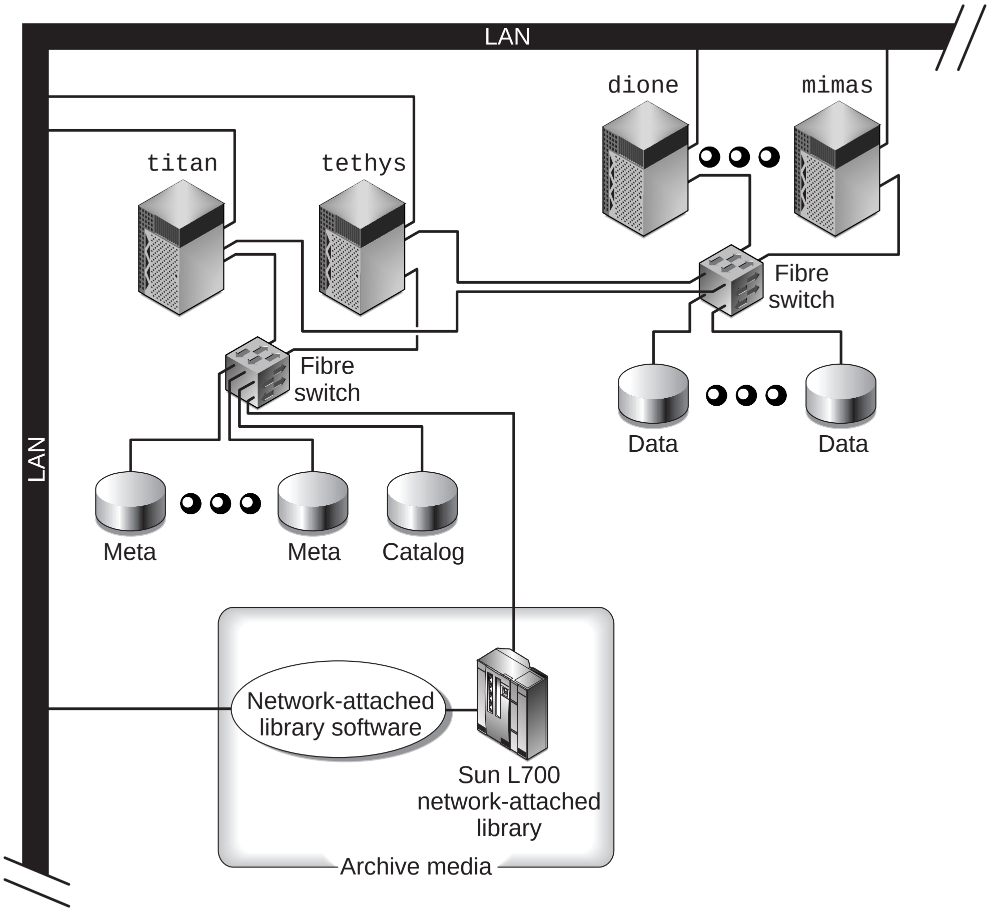

FIGURE 5-1 Sun QFS Shared File System Configuration in a Sun SAM-QFS Environment

FIGURE 5-1 shows four network-attached hosts: `titan`, `tethys`, `dione`, and `mimas`. The `tethys`, `dione`, and `mimas` hosts are the clients, and `titan` is the current metadata server. The `titan` and `tethys` hosts are potential metadata servers.

The archive media consists of a network-attached library and tape drives that are fibre-attached to `titan` and `tethys`. In addition, the archive media catalog resides in a file system that is mounted on the current metadata server, `titan`.

Metadata travels to and from the clients to the metadata server over the network. The metadata server makes all modifications to the name space, and this keeps the metadata consistent. The metadata server also provides the locking capability, the block allocation, and the block deallocation.

Several metadata disks are connected to `titan` and `tethys`, and these disks can only be accessed by the potential metadata servers. If `titan` were unavailable, the metadata server could failover to `tethys`, and the library, tape drives, and catalog could be accessed by `tethys` as part of the Sun QFS shared file system. The data disks are connected to all four hosts by a Fibre Channel connection.

The examples in this chapter use the preceding configuration several times to explain aspects of the Sun QFS shared file system.

Configuring the Sun QFS Shared File System

The following subsections describe the process for creating a Sun QFS shared file system. You can initialize (using the `sammkfs`(1M) command) the Sun QFS shared file system on the metadata server that already has the Sun QFS or Sun SAM-QFS package installed and operational. No additional software is required.

Note – The Sun QFS shared file system (multiwriter capability) is licensed separately from the Sun QFS and Sun SAM-QFS file systems. Contact your Sun sales representative for information on enabling the Sun QFS shared file system.

The procedures in this process assume that you have the Sun QFS or Sun SAM-QFS package installed and configured correctly on all Solaris systems that are to be part of the Sun QFS shared file system. For information on the Sun QFS and Sun SAM-QFS installation process, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

The configuration process consists of several procedures. The following configuration procedures must be performed in the order in which they appear:

- "To Review the Configuration Requirements" on page 93
- "To Configure the Shared Hosts" on page 94
- "To Configure the Metadata Server" on page 97
- "To Configure a Client Host" on page 106
- "To Enable Access to Archive Media (Optional)" on page 115
- "To Enable Access to the Media Catalog (Optional)" on page 115

▼ To Review the Configuration Requirements

Ensure that the following requirements have been met prior to configuring the Sun QFS shared file system:

- There must be at least one Solaris metadata server. To use this file system effectively in a failover (high availability) environment, there must be at least two Solaris systems that can become metadata servers.
- All Solaris OE systems in the Sun QFS shared file system must be based on a SPARC processor.
- The Solaris systems to be configured in the Sun QFS shared file system must be connected by a network.
- The Solaris systems to be included in the Sun QFS shared file system must have either a Sun QFS or Sun SAM-QFS software package installed upon them.
- All Sun QFS or Sun SAM-QFS software installed on the Solaris systems in the Sun QFS shared file system must be at the same release level. This ensures that all Solaris systems in a Sun QFS shared file system have identical over-the-wire protocol versions. If these levels do not match, the system generates the following message when mounting is attempted:

```
SAM-FS: client client package version x mismatch, should be y.
```

  The preceding message is written to the metadata server's `/var/adm/messages` file.

- Your Sun QFS or Sun SAM-QFS systems must be licensed for the Sun QFS shared file system. This is a separate license. Contact your Sun sales representative for information on obtaining a license for the Sun QFS shared file system.
- In a Sun SAM-QFS environment, the storage and archive management software must be known to be operational prior to the configuration of the Sun QFS shared file system.
- If you want to be able to change the metadata server, such as in a Sun SAM-QFS failover environment, the following requirements must be met:
  - Sun Solaris systems to be configured as potential metadata servers must be attached through a storage area network (such as Sun SAN 3.0 or later) or through a network attachment to the library and/or mount points that contain the archive media repository. This enables the other potential metadata servers in the Sun QFS shared file system to be able to access the archive images.
  - The media catalog should reside in a file system that can be accessed from the metadata server and from all potential metadata servers.
- If there is only one Solaris metadata server in the Sun QFS shared file system, it can be attached to its archive media through a SCSI connection.

- Online data storage devices must be directly accessible to all hosts. All online metadata storage devices must be directly accessible to all potential metadata server hosts.

In addition to the preceding requirements, the following are configuration recommendations with regard to metadata:

- It is recommended that a Sun QFS shared file system have multiple metadata (`mm`) partitions. This spreads out metadata I/O and improves file system throughput.
- It is recommended that a Sun QFS shared file system use a private metadata network so typical user traffic does not interfere with metadata traffic. A switch-based (not hub-based) network is recommended for this.

▼ To Configure the Shared Hosts

You can use the following procedure to do the initial configuration work for one metadata server and one or more client hosts in a Sun QFS shared file system.

1. **As superuser, log into each Sun Solaris system to be configured as a shared host in the Sun QFS shared file system.**

   You must have `root` permission to complete the steps in this procedure.

2. **Issue the `pkginfo`(1M) command and examine its output to make sure that a Sun QFS or Sun SAM-QFS package is installed on each host.**

   Each shared host must have either the `SUNWqfs` or the `SUNWsamfs` package installed upon it.

   Example 1. On a system with the Sun QFS package installed, the following output is obtained showing the needed `SUNWqfs` package:

   CODE EXAMPLE 5-1 `pkginfo`(1M) Command Example on a Sun QFS File System

```
# pkginfo | grep SUNWqfs
system SUNWqfs Sun QFS Solaris 2.8
```

   Example 2. On a system with the Sun SAM-QFS package installed, the following output is obtained showing the needed `SUNWsamfs` package:

   CODE EXAMPLE 5-2 `pkginfo`(1M) Command Example on a Sun SAM-QFS File System

```
# pkginfo | grep SUNWsamfs
system SUNWsamfs Sun SAM-FS and Sun SAM-QFS software Solaris 2.8
```

3. **Issue the `format`(1M) command and examine its output.**

   Make sure that the metadata disk partitions configured for the Sun QFS shared file system mount point are connected to the potential metadata servers. Also make sure that the data disk partitions configured for the Sun QFS shared file system are connected to the potential metadata servers and to all the client hosts in this file system.

   For example, CODE EXAMPLE 5-3 shows the `format`(1M) command output for `titan`. There is one meta disk on controller 1, and there are four data disks on controller 3.

CODE EXAMPLE 5-3 `format` (1M) Command Output on `titan`

```
titan<28>format
Searching for disks...done


AVAILABLE DISK SELECTIONS:
       0. c1t0d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>
          /pci@8,600000/SUNW,qlc@4/fp@0,0/ssd@w2100002037e9c296,0
       1. c2t2100002037E2C5DAd0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>
          /pci@8,600000/SUNW,qlc@4/fp@0,0/ssd@w2100002037e2c5da,0
       2. c3t50020F23000065EEd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@8,600000/SUNW,qlc@1/fp@0,0/ssd@w50020f23000065ee,0
       3. c3t50020F2300005D22d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@8,600000/SUNW,qlc@1/fp@0,0/ssd@w50020f2300005d22,0
       4. c3t50020F2300006099d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@8,600000/SUNW,qlc@1/fp@0,0/ssd@w50020f2300006099,0
       5. c3t50020F230000651Cd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@8,600000/SUNW,qlc@1/fp@0,0/ssd@w50020f230000651c,0
```

CODE EXAMPLE 5-4 shows the `format`(1M) command output for `tethys`. There is one meta disk on controller 2, and there are four data disks on controller 7.

CODE EXAMPLE 5-4 `format` (1M) Command Output on `tethys`

```
tethys<1>format
Searching for disks...done


AVAILABLE DISK SELECTIONS:
       0. c0t1d0 <IBM-DNES-318350Y-SA60 cyl 11112 alt 2 hd 10 sec 320>
          /pci@1f,4000/scsi@3/sd@1,0
       1. c2t2100002037E9C296d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>
          /pci@8,600000/SUNW,qlc@4/fp@0,0/ssd@w2100002037e9c296,0
       2. c7t50020F23000065EEd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,ifp@5/ssd@w50020f23000065ee,0
```

CODE EXAMPLE 5-4 `format` (1M) Command Output on `tethys`

```
       3. c7t50020F2300005D22d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,ifp@5/ssd@w50020f2300005d22,0
       4. c7t50020F2300006099d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,ifp@5/ssd@w50020f2300006099,0
       5. c7t50020F230000651Cd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,ifp@5/ssd@w50020f230000651c,0
```

Note the following in the `format`(1M) command's output from `tethys`:

- As the preceding `format`(1M) command output from `tethys` shows, the data disks on `titan`'s controller 3 are the same disks as `tethys'` controller 7. You can verify this by looking at the world-wide name, which is the last component in the device name. For `titan`'s target 3 disk, the world wide name is `ssd@w50020f230000651c,0`. This is the same name as controller 7, target 3 on `tethys`.
- For `titan`'s metadata disk, the world wide name is `ssd@w2100002037e9c296,0`. This is the same metadata disk as `tethys'` controller 2, target 0.

CODE EXAMPLE 5-5 shows the `format`(1M) command's output for `mimas`. This shows four data disks on controller 1 and no meta disks.

CODE EXAMPLE 5-5 `format` (1M) Command Output on `mimas`

```
mimas<9>format
Searching for disks...done


AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <SUN18G cyl 7506 alt 2 hd 19 sec 248>
          /pci@1f,4000/scsi@3/sd@0,0
       1. c1t50020F23000065EEd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,qlc@4/fp@0,0/ssd@w50020f23000065ee,0
       2. c1t50020F2300005D22d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,qlc@4/fp@0,0/ssd@w50020f2300005d22,0
       3. c1t50020F2300006099d0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,qlc@4/fp@0,0/ssd@w50020f2300006099,0
       4. c1t50020F230000651Cd0 <SUN-T300-0116 cyl 34901 alt 2 hd 128 sec 256>
          /pci@1f,4000/SUNW,qlc@4/fp@0,0/ssd@w50020f230000651c,0
```

The `format`(1M) command's output on `mimas` shows that the data disks on `titan`'s controller 3 are the same disks as `mimas'` controller 1. You can verify this by looking at the world-wide name, which is the last component in the device name. For `titan`'s target 3 disk, the world wide name is `ssd@w50020f230000651c,0`. This is the same name as controller 1, target 3 on `mimas`.

Note – All the data disk partitions must be connected and accessible from all the hosts who are to share this file system. All the disk partitions, for both data and metadata, must be connected and accessible to all potential metadata servers. You can use the `format`(1M) command to verify these connections.

---

4. **Verify that all the hosts have the same user and group IDs.**

   If you are not running the Network Information Name service (NIS), make sure that all `/etc/passwd` and all `/etc/group` files are identical. If you are running NIS, the `/etc/passwd` and `/etc/group` files should already be identical.

   For more information on this, see the `nis+`(1) man page.

5. **Set up the network time daemon command,** `xntpd`**(8), to synchronize the times on all the hosts.**

   The clocks of the metadata server and all client hosts must be synchronized during Sun QFS shared file system operations. For more information, see the `xntpd`(8) man page.

▼ To Configure the Metadata Server

You configure one metadata server in a single Sun QFS shared file system.

1. **As superuser, log into the system to be used as the primary metadata server.**

   You must have `root` permission to complete the steps in this procedure.

2. **Back up all site-customized system files and configuration files. (Optional)**

   If you are creating a new file system as a Sun QFS shared file system, you do not need to complete this step.

   Depending on your software, these files can include `mcf`, `archiver.cmd`, `defaults.conf`, `samfs.cmd`, `inquiry.conf`, and so on. Back up these files for all file systems in your Sun SAM-QFS environment. Also make sure that you have backup copies of files in the `/etc/opt/SUNWsamfs` directory, files in the `/var/opt/SUNWsamfs` directory, library catalogs, the historian, and any parameter files for network-attached automated libraries.

   In Sun SAM-QFS environments, if you do not know the names and locations of your catalog files, examine the `mcf` file with `vi`(1) or another viewing command and find the entries for the automated libraries. The path to each library's catalog files is in the Additional Parameters field or, if the Additional Parameters field is blank, the system uses the default path of `/var/opt/SUNWsamfs/catalog/`*catalog_name*. For more information on catalog file locations, see the `mcf`(4) man page.

3. **Ensure that each file system to be modified is backed up. (Optional)**

   If you are creating a new file system as a Sun QFS shared file system, you do not need to complete this step.

   If you want to move files from an existing Sun QFS or Sun SAM-QFS file system into a new Sun QFS shared file system, make sure that your file systems are backed up. The file systems should be backed up regularly according to your site's policies. This is described as the last step in the installation procedure. If you are comfortable with the backup files that already exist for your file systems, there is no need to back them up again now. If, however, you need to back up your file systems to preserve information created since the last dump file was created, do so now. For information on how to create a dump file, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

   To back up a Sun QFS file system, use the `qfsdump`(1M) command, which dumps both data and metadata. To back up a Sun SAM-QFS file system, use the `samfsdump`(1M) command. Note that the `samfsdump`(1M) command issues warnings when creating the dump file if it encounters unarchived files in the file system. If warnings are issued, these files need to be archived before unmounting the file systems.

4. **Modify the `mcf` file on the metadata server to include the Sun QFS shared file system.**

   In the `mcf` file of a metadata server, the only difference between a Sun QFS shared file system and a Sun QFS file system that is not shared is the presence of the `shared` keyword in the Additional Parameters field of the name line of a Sun QFS shared file system. For information on creating an `mcf` file for a QFS or SAM-QFS file system, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS File System Administrator's Guide*.

   If you are adding the Sun QFS shared file system as an additional file system, use `vi`(1) or another editor to create the necessary entries in the `mcf` file to define a Sun QFS shared file system. Make sure to include the `shared` keyword in the Additional Parameters field of the file system name line.

   If you are converting an existing Sun QFS or Sun SAM-QFS file system to a Sun QFS shared file system, use `vi`(1) or another editor to insert the `shared` keyword in the Additional Parameters field of the file system name line.

Note – If Sun QFS, Sun SAM-FS, or Sun SAM-QFS file systems are already operational on the Sun QFS shared file system's metadata server or on any of the client host systems, you must select a Family Set name that does not conflict with existing Family Set names on any host that will be included in the Sun QFS shared file system.

The following mcf file fragment defines several disks for use in the Sun QFS shared file system, and it shows the shared keyword in the Additional Parameters field on the file system name line:

CODE EXAMPLE 5-6 Sun QFS Shared File System mcf File Example for titan

```
# Equipment                         Eq     Eq    Family      Dev     Addl
# Identifier                        Ord    Ty    Set         Stat    Params
                                           pe
------------                        ---    --    ---         ----    ------
                                           --                -
sharefs1                            10     ma    sharefs1    on      shared
/dev/dsk/c1t2100002037E9C296d0s6    11     mm    sharefs1    on
/dev/dsk/c3t50020F2300005D22d0s6    12     mr    sharefs1    on
/dev/dsk/c3t50020F2300006099d0s6    13     mr    sharefs1    on
/dev/dsk/c3t50020F230000651Cd0s6    14     mr    sharefs1    on
```

5. **Create the hosts file on the metadata server.**

   Using vi(1) or another editor, create an ASCII file that contains configuration information pertaining to all hosts in the Sun QFS shared file system. The ASCII hosts file defines all the hosts that can share the file system family set name.

   Hosts files must reside in /etc/opt/SUNWsamfs/hosts.*fs_name*, where *fs_name* is the family set name of the Sun QFS shared file system. Comments are permitted in the hosts file. Comment lines must begin with a pound character (#). Characters to the right of the pound character are ignored.

TABLE 5-1 shows the fields in the hosts file.

TABLE 5-1 Hosts File Fields

Field Number	Content
1	The Host Name field. This field must contain an alphanumeric host name. It defines the Sun QFS shared file system hosts. This field can be created by using the output from the `hostname`(1) command.
2	The Host IP Addresses field. This field must contain a comma-separated list of host IP addresses. This field can be created by using the output received from the `ifconfig`(1M) command with its `-a` option. The individual addresses can be specified in one of the following ways: • Dotted-decimal IP address form. • IP version 6 hexadecimal address form. • As a symbolic name that the local domain name service (DNS) can resolve to a particular host interface. The metadata server uses this field to determine whether a host is allowed to connect to the Sun QFS shared file system. If the metadata server receives a connect attempt from any interface not listed in this field, it rejects the connection attempt. Conversely, care should be used in adding elements here because the metadata server accepts any host with an IP address that matches an address in this field. The client hosts use this field to determine the metadata server interface(s) to use when attempting to connect to the metadata server. Each host evaluates the addresses from left to right, and the connection is made using the first responding address in the list.
3	The Server Priority field. This field must contain either a dash character (`-`) or an integer ranging from 0 through *n*. The `-` and the `0` are equivalent. An integer number 1 indicates that this row defines a sever as the primary metadata host. Only one host can be assigned a metadata server priority of `1`, and this is the host designated as the metadata server whenever possible. The metadata server processes all the metadata modification for the file system. At any one time there is at most one metadata server host, and that metadata server supports archiving, staging, releasing, and recycling for the Sun SAM-QFS file system. An integer number of 2, 3, or greater indicates that these rows define servers that are designated as alternate metadata servers if the priority `1` server is unavailable or down. If the metadata server priority is `-` or `0`, that host is not eligible to be a metadata server.
4	Reserved for future use by Sun Microsystems. Must contain dash (`-`) characters.
5	The Server Host field. This field can contain either a blank or the `server` keyword in the row that defines the active metadata server. Only one row in the hosts file can contain the `server` keyword. All other rows must be blank.

The hosts file is read and manipulated by the system. You can use the samsharefs(1M) command to examine metadata server and client host information on a running system.

Example. CODE EXAMPLE 5-7 is an example hosts file that shows four hosts.

CODE EXAMPLE 5-7 Sun QFS Shared File System Hosts File Example

```
# File /etc/opt/SUNWsamfs/hosts.sharefs1
# Host   Host IP                          Server   Not  Server
# Name   Addresses                        Priority Used Host
# ----   -------------------------------- -------- ---- -----
titan    172.16.0.129,titan.xyzco.com     1        -    server
tethys   172.16.0.130,tethys.xyzco.com    2        -
mimas    mimas.xyzco.com                  -        -
dione    dione.xyzco.com                  -        -
```

CODE EXAMPLE 5-7 shows a hosts file that contains fields of information and comment lines for the sharefs1 file system. In this example, the Server Priority field contains the number 1 in the Server Priority field to define the primary metadata server as titan. If titan is down, the next metadata server is tethys, and the number 2 in this field indicates this secondary priority. Note that neither dione nor mimas can ever be a metadata server.

6. **Send a HUP signal to the** sam-fsd **daemon on the metadata server host.**

   The HUP is needed to inform the sam-fsd daemon of the configuration changes. For example:

```
# pkill -HUP sam-fsd
```

7. **Use the** sammkfs**(1M) command to initialize the file system and make the file system as a Sun QFS shared file system.**

   Enter the sammkfs(1M) command at the system prompt. Use the -S and -a options, and specify the family set name for the file system. The -S options specifies that the file system be a Sun QFS shared file system. The -a option specifies the disk allocation unit. Also specify the family set name of the file system.

   Format:

```
sammkfs -S -a allocation_unit fs_name
```

The arguments to the preceding format are as follows:

TABLE 5-2 sammkfs(1M) Command Arguments

Argument	Meaning
allocation_unit	Specifies the number of bytes, in units of 1024 (1-kilobyte) blocks, to be allocated to a Disk Allocation Unit (DAU). The specified *allocation_unit* must be a multiple of 8 kilobytes. For more information, see the sammkfs(1M) man page.
fs_name	Family set name of the file system as defined in the mcf file.

For more information on the sammkfs(1M) command, see the sammkfs(1M) man page. For example, the following sammkfs(1M) command can be used to make the Sun QFS shared file system file system and identify it as shared:

```
# sammkfs -S -a 512 sharefs1
```

If the shared keyword appears in the mcf file, the file system must have been initialized as a shared file system by using the -S option to the sammkfs(1M) command. You cannot mount a file system as shared if it was not initialized as shared.

8. **Set the port number for this Sun QFS shared file system's family set name.**

   There must be one port per file system. You must specify a unique port number for each file system. The port name is samsock. followed by the family set name of the Sun QFS shared file system. You can set this either in the /etc/inet/services file or, if you are using NIS, in the /etc/yp/src/services file.

   To set the port number in /etc/inet/services, add a line in this file that is similar to the following for the sharefs1 file system:

```
samsock.sharefs1    7105/tcp     # Sun QFS sharefs1 port number
```

   To set the port number in /etc/yp/src/services, add a line in this file that is similar to the following:

```
samsock.sharefs1    7105/tcp     # Sun QFS sharefs1 port number
```

If you set the port number in /etc/yp/src/services, verify that samsock.*fs_name* exists on the metadata server and on all the client hosts by entering the following command:

CODE EXAMPLE 5-8 Verifying the Port Number

```
# ypcat services -x | grep samsock
samsock.sharefs1 7105/tcp        # Sun QFS sharefs1 port number
```

If it does not exist, verify services is enabled on all the shared hosts by entering the following command:

CODE EXAMPLE 5-9 Verifying the Port Number

```
# ypwhich -m | grep services
services.byservicename earth
services.byname earth
```

9. **Send a HUP signal to** /usr/sbin/inetd**.**

   The inetd system software needs to reread the /etc/inet/services file. To accomplish this, enter the following command:

```
# pkill -HUP inetd
```

10. **Send a HUP signal to the** sam-fsd **daemon on the metadata server host. (Optional)**

    If you completed Step 9 by sending a HUP signal, you need to perform this step.

    The HUP is needed to inform the sam-fsd daemon of the configuration changes. For example:

```
# pkill -HUP sam-fsd
```

11. **Verify that the** sam-sharefsd **daemon is running for this file system.**

    To accomplish this, enter the following command:

```
# ps -ef | grep sam-sharefsd
```

CODE EXAMPLE 5-10 shows the output from the ps(1) command.

CODE EXAMPLE 5-10 Output from the ps(1) Command

```
root 26167 26158  0 18:35:20 ?        0:00 sam-sharefsd sharefs1
root 27808 27018  0 10:48:46 pts/21   0:00 grep sam-sharefsd
```

CODE EXAMPLE 5-10 shows that the `sam-sharefsd` daemon is active for the `sharefs1` file system. If this is the case for your system, you can proceed to the next step in this procedure. If, however, the output returned on your system does not show that the `sam-sharefsd` daemon is active for your Sun QFS shared file system, you need to perform some diagnostic procedures. For information on these procedures, see "Recovering a Hung mount(1M) Command" on page 144.

12. **Make the mount point for the new Sun QFS shared file system. (Optional)**

    If your mount point already exists, you do not need to complete this step.

    If you need to create a mount point, however, use the `mkdir`(1) command to make the directory for the mount point. For example:

```
# mkdir /sharefs1
```

13. **Issue the** `chmod`**(1M) command to give the mount point the** `755` **set of permissions.**

    For example:

```
# chmod 755 /sharefs1
```

    The permissions must be the same on all participant hosts. `755` is suggested as the initial permission set. After mounting the file systems, the `root` directory's permissions override this setting.

14. **Modify the** `/etc/vfstab` **file.**

    You must have an entry in the `/etc/vfstab` file for the Sun QFS shared file system.

    If you want the Sun QFS shared file system to automatically mount at boot, modify the `/etc/vfstab` file and put `yes` in the mount at boot field. If you put `yes`, Sun Microsystems recommends also adding the `bg` mount option in the mount parameters field. The `bg` mount option mounts the file system in the background if the metadata server is not responding.

If you do not want to mount this system automatically at boot time, put `no` in the mount at boot field. In either case, `shared` is a required entry in the mount parameters field. For example:

CODE EXAMPLE 5-11 `/etc/vfstab` File Example

```
# File /etc/vfstab
# FS name   FS to fsck    Mnt pt        FS type    fsck    Mt@boot   Mt params
#                                                  pass
sharefs1    -             /sharefs1     samfs      -       yes       shared,bg
```

15. **Use the `mount`(1M) command to mount the Sun QFS shared file system on the metadata server.**

    For failover purposes, the mount options should be the same on the metadata server and all potential metadata servers. For example, you can create a `samfs.cmd`(4) file containing mount options and copy it to all the hosts.

    For more information on mounting Sun QFS shared file systems, see “Mount Options in a Sun QFS Shared File System” on page 132 or see the `mount_samfs`(1M) man page.

16. **Use the `cd`(1) command to change to the directory that contains the mount point. (Optional)**

    If you have dumped file data using `qfsdump`(1M) or `samfsdump`(1M), use the `cd`(1) command to change to the mount point for the new Sun QFS shared file system. This is the location to which file data will be restored.

17. **Use the `qfsrestore`(1M) or `samfsrestore`(1M) commands to restore file system data. (Optional)**

    If you are creating a new file system that is a Sun QFS shared file system, you do not need to complete this step.

If you dumped existing file system data into a dump file earlier in this procedure, however, use the qfsrestore(1M) or samfsrestore(1M) commands to restore the data. For more information on restoring file systems, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Disaster Recovery Guide*.

Example 1. To restore from a Sun QFS file system, change to the directory that contains the mount point for the file system and issue the qfsrestore(1M) command. The following example shows restoring the files from the backup file named qfs1.dump:

CODE EXAMPLE 5-12 qfsrestore(1M) Example

```
# cd /sharefs1
# qfsrestore -T -f /save/qfs/qfs1.dump
```

Example 2. To restore from a Sun SAM-QFS file system, change to the directory that contains the mount point for the file system and issue the samfsrestore(1M) command. The following example shows restoring the metadata from the backup file named samqfs1.dump into the sharefs1 Sun QFS shared file system:

CODE EXAMPLE 5-13 samfsrestore(1M) Example

```
# cd /sharefs1
# samfsrestore -T -f /save/samqfs/samqfs1.dump
```

▼ To Configure a Client Host

You can configure multiple client hosts in a Sun QFS shared file system.

1. **As superuser, log into one of the client hosts.**

2. **Use the format(1M) command to verify the presence of client host disks.**

   For more information on this step, see how the format(1M) command is used in "To Configure the Shared Hosts" on page 94.

3. **Update the mcf file on the client host.**

   Use vi(1) or another editor to edit the mcf file on one of the client host systems. The mcf file must be updated on all client hosts to be included in the Sun QFS shared file system. The file system and disk declaration information must have the same data for the Family Set name, Equipment Ordinal, and Equipment Type as the configuration on the metadata server. The mcf files on the client hosts must also include the shared keyword. The device names, however, can change as controller assignments can change from host to host.

The samfsconfig(1M) command generates configuration information that can help you to identify the devices included in the Sun QFS shared file system. A separate samfsconfig(1M) command must be entered on each client host. Note that the controller number might not be the same controller number as on the metadata server because the controller numbers are assigned by each client host.

Example 1. The following example shows how the samfsconfig(1M) command is used to retrieve device information for family set sharefs1 on client tethys. Note that tethys is a potential metadata server, so it is connected to the same metadata disks as titan.

CODE EXAMPLE 5-14 samfsconfig(1M) Command Example on tethys

```
tethys# samfsconfig /dev/dsk/*
#
# Family Set 'sharefs1' Created Wed Jun 27 19:33:50 2001
#
sharefs1                            10     ma      sharefs1   on       shared
/dev/dsk/c2t2100002037E9C296d0s6    11     mm      sharefs1   on
/dev/dsk/c7t50020F2300005D22d0s6    12     mr      sharefs1   on
/dev/dsk/c7t50020F2300006099d0s6    13     mr      sharefs1   on
/dev/dsk/c7t50020F230000651Cd0s6    14     mr      sharefs1   on
```

Edit the mcf file on client host tethys by copying the last five lines of output from the samfsconfig(1M) command into the mcf file on client host tethys. Verify the following:

- Each Device State field must be set to on.
- The shared keyword must appear in the Additional Parameters field for the file system name.

The resulting mcf file is as follows:

CODE EXAMPLE 5-15 mcf File for sharefs1 Client Host tethys

```
# Equipment                         Eq     Eq      Family     Dev      Add
# Identifier                        Ord    Type    Set        State    Params
# ----------                        ---    ----    ------     -----    ------
sharefs1                            10     ma      sharefs1   on       shared
/dev/dsk/c2t2100002037E9C296d0s6    11     mm      sharefs1   on
/dev/dsk/c7t50020F2300005D22d0s6    12     mr      sharefs1   on
/dev/dsk/c7t50020F2300006099d0s6    13     mr      sharefs1   on
/dev/dsk/c7t50020F230000651Cd0s6    14     mr      sharefs1   on
```

In CODE EXAMPLE 5-15, note that the Equipment Ordinal numbers match those of the example mcf file for metadata server titan. These Equipment Ordinal numbers must not already be in use on client host tethys or any other client host.

Example 2. The following example shows how the samfsconfig(1M) command is used to retrieve device information for family set sharefs1 on client host mimas. Note that mimas can never become a metadata server, and it is not connected to the metadata disks.

CODE EXAMPLE 5-16 samfsconfig(1M) Command Example on mimas

```
mimas# samfsconfig /dev/dsk/*
#
# Family Set 'sharefs1' Created Wed Jun 27 19:33:50 2001
#
# Missing slices
# Ordinal 0
# /dev/dsk/c1t50020F2300005D22d0s6   12    mr   sharefs1   on
# /dev/dsk/c1t50020F2300006099d0s6   13    mr   sharefs1   on
# /dev/dsk/c1t50020F230000651Cd0s6   14    mr   sharefs1   on
```

In the output from the samfsconfig(1M) command on mimas, note that Ordinal 0, which is the metadata disk, is not present. Because devices are missing, the samfsconfig(1M) command comments out the elements of the file system and omits the file system Family Set declaration line. Make the following types of edits to the mcf file:

- Create a file system Family Set declaration line, beginning with sharefs1, in the mcf file for client host mimas. Enter the shared keyword into the Additional Parameters field of the file system Family Set declaration line.
- Create one or more nodev lines for each missing Equipment Ordinal. For these lines, the keyword nodev must appear in the Equipment Identifier field for each inaccessible device. In this example, you create a device entry in the mcf file named nodev to represent the missing metadata disk.
- Ensure that each Device State field is set to on.
- Uncomment the device lines.

CODE EXAMPLE 5-17 shows the resulting mcf file for mimas.

CODE EXAMPLE 5-17 mcf File for Client Host mimas

```
# The mcf File For mimas
# Equipment                          Eq     Eq      Family      Device    Addl
# Identifier                         Ord    Type    Set         State     Params
------------                         ---    ----    ---         -----     ------
sharefs1                             10     ma      sharefs1    on        shared
nodev                                11     mm      sharefs1    on
/dev/dsk/c1t50020F2300005D22d0s6     12     mr      sharefs1    on
/dev/dsk/c1t50020F2300006099d0s6     13     mr      sharefs1    on
/dev/dsk/c1t50020F230000651Cd0s6     14     mr      sharefs1    on
```

4. **Send a HUP signal to the `sam-fsd` daemon on the metadata server host.**

   The HUP is needed to inform the `sam-fsd` daemon of the configuration changes. For example:

   ```
   # pkill -HUP sam-fsd
   ```

5. **Create the local hosts configuration file on the client host. (Optional)**

   You might want to perform this step if your Sun QFS shared file system host systems have multiple host interfaces. You can use this file to specify how file system traffic should flow over public and private networks in your environment.

   Using `vi`(1) or another editor, create an ASCII local hosts configuration file that defines the host interfaces that the metadata server and the client hosts can use when accessing the file system. The local hosts configuration file must reside in the following location:

   ```
   /etc/opt/SUNWsamfs/hosts.fs_name.local
   ```

   In this path, *fs_name* must be the family set name of the Sun QFS shared file system.

   Comments are permitted in the local host configuration file. Comment lines must begin with a pound character (#). Characters to the right of the pound character are ignored.

TABLE 5-3 shows the fields in the local hosts configuration file.

TABLE 5-3 Local Hosts Configuration File Fields

Field Number	Content
1	The Host Name field. This field must contain the alphanumeric name of a metadata server or potential metadata server that is part of the Sun QFS shared file system.
2	The Host Interfaces field. This field must contain a comma-separated list of host interface addresses. This field can be created by using the output received from the `ifconfig`(1M) command with its `-a` option. The individual interfaces can be specified in one of the following ways: • Dotted-decimal IP address form. • IP version 6 hexadecimal address form. • As a symbolic name that the local domain name service (DNS) can resolve to a particular host interface. Each host uses this field to determine whether a host will try to connect to the specified host interface. The system evaluates the addresses from left to right, and the connection is made using the first responding address in the list that is also included in the system hosts file.

In a Sun QFS shared file system, each client host obtains the list of metadata server IP addresses from the metadata server host.

The metadata server and the client hosts use both the `/etc/opt/SUNWsamfs/hosts.`*fs_name* file on the metadata server and the `hosts.`*fsname*`.local` file on each client host(if it exists) to determine the host interface to use when accessing the file system. This process is as follows (note that *client*, as in *network client*, is used to refer to both client hosts and the metadata server host in the following process):

1. The client obtains the list of metadata server host IP interfaces from the file system's on-disk host file. To examine this file, issue the `samsharefs`(1M) command from the metadata server or from a potential metadata server.

2. The client searches its files for a `hosts.`*fsname*`.local` file. Depending on the outcome of the search, one of the following courses of action is taken:

   a. If a `hosts.`*fsname*`.local` file does not exist, the client attempts to connect, in turn, to each address in the system hosts configuration file until it succeeds in connecting.

   b. If the `hosts.`*fsname*`.local` file exists, the client performs the following tasks:

      i. The client compares the list of addresses for the metadata server from both the `/etc/opt/SUNWsamfs/hosts.`*fs_name* file on the metadata server and the `hosts.`*fsname*`.local` file.

ii. It builds a list of addresses that are present only in both places, and then it attempts to connect to each of these addresses, in turn, until it succeeds in connecting to the server. If the order of the addresses differs in these files, the client uses the ordering in the `hosts.`*fsname*`.local` file.

Example. This example expands on the example that was already begun in this chapter. CODE EXAMPLE 5-7 shows the hosts file for this configuration. FIGURE 5-2 shows the interfaces to these systems.

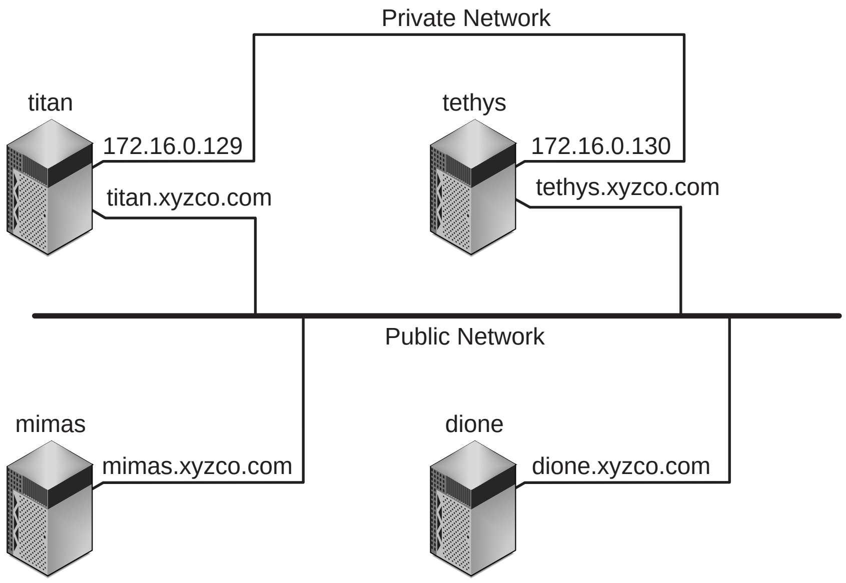

FIGURE 5-2 Network Interfaces

Systems `titan` and `tethys` share a private network connection with interfaces `172.16.0.129` and `172.16.0.130`. To guarantee that `titan` and `tethys` always communicate over their private network connection, the system administrator has created identical copies of `/etc/opt/SUNWsamfs/hosts.sharefs1.local` on each system. CODE EXAMPLE 5-18 shows the information in these files.

CODE EXAMPLE 5-18 File on Both `titan` and `tethys`

```
# This is file /etc/opt/SUNWsamfs/hosts.sharefs1.local
# Host Name    Host Interfaces
# ---------    ---------------
titan          172.16.0.129
tethys         172.16.0.130
```

Systems mimas and dione are not on the private network. To guarantee that they connect to titan and tethys through titan's and tethys' public interfaces, and never attempt to connect to titan's or tethys' unreachable private interfaces, the system administrator has created identical copies of /etc/opt/SUNWsamfs/hosts.sharefs1.local on mimas and dione. CODE EXAMPLE 5-19 shows the information in these files.

CODE EXAMPLE 5-19 File on Both mimas and dione

```
# This is file /etc/opt/SUNWsamfs/hosts.sharefs1.local
# Host Name    Host Interfaces
# ----------   --------------
titan          titan.xyzco.com
tethys         tethys.xyzco.com
```

6. **Set the port number for the client host.**

   If you are using NIS this was accomplished in a previous step and you do not need to complete this step.

   If you are not using NIS, you must complete this step by specifying the unique port for the file system in the /etc/inet/services file on the client host. To accomplish this, add a line into the /etc/inet/services file that is identical to the line added in Step 8 of "To Configure the Metadata Server" on page 97.

   For example:

```
samsock.sharefs1    7105/tcp     # SAM sharefs1 port number
```

7. **Send a HUP signal to /usr/sbin/inetd. (Optional)**

   If you are using NIS, you do not need to complete this step.

   The inetd system software needs to reread the /etc/inet/services file. To accomplish this, enter the following command:

```
# pkill -HUP inetd
```

8. **Send a HUP signal to the sam-fsd daemon on the client host.**

   The HUP is needed to inform the sam-fsd daemon of the configuration changes. For example:

```
# pkill -HUP sam-fsd
```

9. **Verify that the** `sam-sharefsd` **daemon is running for this file system.**

   To accomplish this, enter the following command:

```
# ps -ef | grep sam-sharefsd
```

   CODE EXAMPLE 5-10 shows the output from the `ps`(1) command.

   CODE EXAMPLE 5-20 Output from the `ps`(1) Command

```
root 26167 26158  0 18:35:20 ?        0:00 sam-sharefsd sharefs1
root 27808 27018  0 10:48:46 pts/21   0:00 grep sam-sharefsd
```

   CODE EXAMPLE 5-10 shows that the `sam-sharefsd` daemon is active for the `sharefs1` file system. If this is the case for your system, you can proceed to the next step in this procedure. If, however, the output returned on your system does not show that the `sam-sharefsd` daemon is active for your Sun QFS shared file system, you need to perform some diagnostic procedures. For information on these procedures, see "Recovering a Hung mount(1M) Command" on page 144.

10. **Make the mount point for the new Sun QFS shared file system. (Optional)**

   If your mount point already exists, you do not need to complete this step.

   If you need to create a mount point, however, use the `mkdir`(1) command to make the directory for the mount point. For example:

```
# mkdir /sharefs1
```

11. **Issue the** `chmod`**(1M) command to give the mount point the** `755` **set of permissions.**

   For example:

```
# chmod 755 /sharefs1
```

   The permissions must be the same on all participant hosts. `755` is suggested as the initial permission set. After mounting the file systems, the `root` directory's permissions override this setting.

12. **Modify the** `/etc/vfstab` **file.**

   You must have an entry in the `/etc/vfstab` file for the Sun QFS shared file system.

If you want the Sun QFS shared file system to automatically mount at boot, modify the /etc/vfstab file and put yes in the mount at boot field. If you put yes, Sun Microsystems recommends also adding the bg mount option in the mount parameters field. The bg mount option mounts the file system in the background if the metadata server is not responding.

If you do not want to mount this system automatically at boot time, put no in the mount at boot field. In either case, shared is a required entry in the mount parameters field. For example:

CODE EXAMPLE 5-21 /etc/vfstab File Example

```
# File /etc/vfstab
# FS name   FS to fsck    Mnt pt       FS type    fsck    Mt@boot   Mt params
#                                                 pass
sharefs1    -             /sharefs1    samfs      -       yes       shared,bg
```

13. **Issue the df(1M) command on the metadata server to verify that the file system is mounted on the metadata server.**

```
metadata_server# df -k
```

14. **From the client host, issue the mount(1M) command to mount the Sun QFS shared file system on the client host.**

    For failover purposes, the mount options should be the same on the metadata server and all potential metadata servers. For example, you can create a samfs.cmd(4) file containing mount options and copy it to all the hosts.

    For more information on mounting Sun QFS shared file systems, see “Mount Options in a Sun QFS Shared File System” on page 132 or see the mount_samfs(1M) man page.

    For example:

```
client_host# mount /sharefs1
```

15. **Repeat the steps in this procedure for each client host.**

▼ To Enable Access to Archive Media (Optional)

If your Sun QFS shared file system is implemented in a Sun SAM-QFS environment, the file system can access information stored on cartridges in a library. This procedure explains how to ensure that the data on these cartridges is accessible to the metadata server and the client hosts in a Sun QFS shared file system.

If your Sun QFS shared file system is implemented in a Sun QFS environment, you can omit this procedure.

1. **Add library and drive devices to the `mcf` file on the potential metadata servers. (Optional)**

   In a Sun SAM-QFS environment, you can configure a library and drives in the `mcf` file for all the potential metadata servers. If you are using disk archiving in this environment, you must configure a `diskvols.conf` file.

   For information on configuring a library or enabling disk archiving, see the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

2. **Send a HUP signal to the `sam-fsd` daemon on all the potential metadata servers.**

   The HUP is needed to inform the `sam-fsd` daemon of the configuration change. For example:

```
# pkill -HUP sam-fsd
```

▼ To Enable Access to the Media Catalog (Optional)

If your Sun QFS shared file system is implemented in a Sun SAM-QFS environment and you want to be able to change metadata servers, you must put your media catalog in a location that can be accessed by both the metadata server and all potential metadata servers.

If your Sun QFS shared file system is implemented in a Sun QFS environment, you can omit this procedure.

1. **Select a file system for the catalog that is accessible from the preferred metadata server and all potential metadata servers.**

   The media catalog must reside on a shared storage device that is accessible to each potential metadata server.

2. **Log in to the system that is to become the metadata server.**

3. **From the metadata server, mount the file system that is to contain the media catalog.**

```
titan # mount /catalog
```

4. **On the metadata server, edit the `mcf` file.**

   To ensure access for all hosts in a Sun QFS shared file system, make the following sets of edits:

   - Ensure that the Device State field is set to `on` for the libraries attached to the metadata server.
   - Ensure that you use the Additional Parameters field to designate the nondefault path to the library catalog that resides on the common shared storage device.

   The following example `mcf` file on `titan` shows the correct Device State and Additional Parameters field settings:

CODE EXAMPLE 5-22 Sun QFS Shared File System `mcf` File Example for `titan`

```
# titan mcf file (preferred metadata server)
# Equipment                          Eq     Eq    Family      Dv    Addl
# Identifier                         Ord    Ty    Set         Sa    Params
------------                         ---    --    ---         --    ------
                                                              -
sharefs1                             10     ma    sharefs1    on    shared
/dev/dsk/c1t2100002037E9C296d0s6     11     mm    sharefs1    on
/dev/dsk/c3t50020F2300005D22d0s6     12     mr    sharefs1    on
/dev/dsk/c3t50020F2300006099d0s6     13     mr    sharefs1    on
/dev/dsk/c3t50020F230000651Cd0s6     14     mr    sharefs1    on
#
/etc/opt/SUNWsamfs/L700              100    sk    L700        on    /catalog/L700
/drv/rmt/2cbn                        160    sg    L700        on
/drv/rmt/0cbn                        170    sg    L700        on
/drv/rmt/1cbn                        180    sg    L700        on
#
```

5. **On the metadata server, copy the `mcf` file to `mcf.on` and `mcf.off`.**

   The copies of the `mcf` file will be needed when you change metadata servers in a failover situation. At the end of this procedure there will be three mcf files on the metadata server and on each potential metadata server: `mcf`, `mcf.on`, and `mcf.off`. The `mcf` file is the only active file when the Sun QFS shared file system is in production. The `mcf.on` and `mcf.off` files are moved to `mcf` as needed when changing metadata servers.

To ensure catalog consistancy, the `mcf` file used when the file system is mounted must have the shared libraries configured in the `mcf` file with a Device State field set to `on` in the metadata server's `mcf` file. When you mount the Sun QFS shared file system for the first time, the `mcf` file you configured in the previous step is used. When you change the metadata server to a different server in the configuration, you will enable `mcf.off` on the old metadata server and enable `mcf.on` in the new metadata server. This is explained more thoroughly in subsequent steps.

For example:

```
titan# cp mcf mcf.on
titan# cp mcf mcf.off
```

6. **On the metadata server, edit `mcf.off` and change all the Device State field entries to `off` for all shared libraries and their drives.**

   This is the `mcf` file that will be enabled when you change metadata servers in a failover situation.

7. **On the metadata server, copy the `mcf`, `mcf.on`, and `mcf.off` files to all potential metadata servers.**

8. **On the metadata server, copy all configuration files to all potential metadata servers.**

   Most configuration files are optional, but if you have configured any of the following files, copy them to all potential metadata servers: `archiver.cmd`, `defaults.conf`, `diskvols.conf`, `ftp.cmd`, `inquiry.conf`, `preview.cmd`, `recycler.cmd`, `releaser.cmd`, `samfs.cmd`, and `stager.cmd`.

9. **On all potential metadata servers, copy file `mcf.off` to `mcf`.**

   This is the `mcf` file that will be used when the Sun QFS shared file system is mounted initially. All shared libraries must have their `Device State` fields set to `off` in the `mcf` files of the potential metadata servers.

10. **On all potential metadata servers, send a HUP signal to the `sam-fsd` daemon.**

    The HUP is needed to inform the `sam-fsd` daemon of the configuration change. For example:

```
# pkill -HUP sam-fsd
```

Caution – Be careful when updating the `mcf` file on the hosts included in a Sun QFS shared file system. If you create new file systems or add equipment, make sure to update the `mcf` files in all three locations on each host: `mcf`, `mcf.on`, and `mcf.off`.

Mounting and Unmounting Sun QFS Shared File Systems

When mounting or unmounting a Sun QFS shared file system, the order in which the Solaris OE is mounted or unmounted is important.

For failover purposes, the mount options should be the same on the metadata server and all potential metadata servers. For example, you can create a `samfs.cmd`(4) file containing mount options and copy it to all the hosts.

For more information on mounting Sun QFS shared file systems, see "Mount Options in a Sun QFS Shared File System" on page 132 or see the `mount_samfs`(1M) man page. For more information on mounting and unmounting file systems, see "Operations" on page 61.

▼ To Mount a Sun QFS Shared File System

The `mount`(1M) command mounts a Sun QFS shared file system in a Solaris OE. For more information on the `mount`(1M) command, see the `mount`(1M) man page.

1. **Log in as superuser (`root`) on the metadata server and on all the client hosts.**
2. **Use the `mount`(1M) command to mount the metadata server.**

   The metadata server must be mounted prior to mounting any client hosts.
3. **Use the `mount`(1M) command to mount the client hosts.**

   The order in which the client hosts are mounted is not important.

▼ To Unmount a Sun QFS Shared File System

The `umount`(1M) command unmounts a Sun QFS shared file system from a Solaris system. For more information on the `umount`(1M) command, see the `umount`(1M) man page.

1. **Log in as superuser (`root`) on the metadata server and on all the client hosts.**
2. **Use the `umount`(1M) command to unmount the client hosts.**

   The order in which the client hosts are unmounted is not important.
3. **Use the `umount`(1M) command to unmount the metadata server.**

   The metadata server should be unmounted only after unmounting all client hosts.

Several conditions can be present in a file system at unmounting time, so you might need to issue the `umount`(1M) command a second time. If the file system still does not unmount, use `unshare`(1M), `fuser`(1M), or other commands in conjunction with the `umount`(1M) command. Unmounting procedures are also described in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

Adding and Removing a Client Host

The following sections describe adding and removing client host systems:

- "To Add a Client Host" on page 119
- "To Remove a Client Host" on page 121

▼ To Add a Client Host

You can add a client host to a Sun QFS shared file system after the file system has been configured and mounted on all participants. The following subsections describe these procedures.

1. **Log in as superuser (`root`) on the metadata server.**
2. **Use the `samsharefs`(1M) command to retrieve the current Sun QFS shared file system information and write it to an editable file.**
   - If the Sun QFS shared file system is mounted, issue the `samsharefs`(1M) command on the current metadata server. For example:

```
# samsharefs sharefs1 > /etc/opt/SUNWsamfs/hosts.sharefs1
```

   - If the Sun QFS shared file system is unmounted, issue the `samsharefs`(1M) command with its `-R` option from the metadata server or from any of the potential metadata servers. For example:

```
# samsharefs -R sharefs1 > /etc/opt/SUNWsamfs/hosts.sharefs1
```

The `samsharefs`(1M) command can be issued only on the active metadata server or on client hosts configured as potential metadata servers. For more information, see the `samsharefs`(1M) man page.

Note – The hosts information can be changed on any potential metadata server when the file system is unmounted, so Sun Microsystems recommends that you always retrieve the hosts information to ensure that the hosts information is current.

3. **Use `vi(1)` or another editor to open the Sun QFS shared file system information file.**

   For example:

```
# vi /etc/opt/SUNWsamfs/hosts.sharefs1
```

CODE EXAMPLE 5-23 shows the information returned after the previous command is issued.

CODE EXAMPLE 5-23 `hosts.sharefs1` Prior to Editing

```
# File /etc/opt/SUNWsamfs/hosts.sharefs1
# Host   Host IP                          Server   Not  Server
# Name   Addresses                        Priority Used Host
# ----   -------------------------------- -------- ---- -----
titan    172.16.0.129,titan.xyzco.com     1        -    server
tethys   172.16.0.130,tethys.xyzco.com    2        -
mimas    mimas.xyzco.com                  -        -
dione    dione.xyzco.com                  -        -
```

4. **Use the editor to add a line for the new client host.**

   CODE EXAMPLE 5-24 shows the file after the line for `helene` has been added as the last line.

CODE EXAMPLE 5-24 `hosts.sharefs1` After Editing

```
# File /etc/opt/SUNWsamfs/hosts.sharefs1
# Host   Host IP                          Server   Not  Server
# Name   Addresses                        Priority Used Host
# ----   -------------------------------- -------- ---- -----
titan    172.16.0.129,titan.xyzco.com     1        -    server
tethys   172.16.0.130,tethys.xyzco.com    2        -
mimas    mimas.xyzco.com                  -        -
dione    dione.xyzco.com                  -        -
helene   helene.xyzco.com                 -        -
```

5. **Use the `samsharefs`(1M) command to update the current information in the binary file.**

   The options to use on this command, and the system from which it is issued, differ depending on whether or not the Sun QFS shared file system is mounted, as follows:

   - If the Sun QFS shared file system is mounted, issue the `samsharefs`(1M) command, using the `-u` option, from the current metadata server. For example:

```
# samsharefs -u sharefs1
```

   - If the Sun QFS shared file system is unmounted, issue the `samsharefs`(1M) command, using the `-R` and `-u` options, from the active metadata server or from any of the potential metadata servers. For example:

```
# samsharefs -R -u sharefs1
```

   The client host `helene` is now recognized.

6. **Follow the steps described in "To Configure a Client Host" on page 106.**

   Completing the task of adding a client host to a configured and mounted Sun QFS shared file system consists of following the steps described previously for configuring a client host.

▼ To Remove a Client Host

If the Sun QFS shared file system is unmounted, you can use the following procedure to delete a client host. This procedure includes a step for unmounting the Sun QFS shared file system.

1. **Log in as superuser (`root`) on the metadata server and on all the client hosts.**

2. **Use the `umount`(1M) command to unmount the Sun QFS shared file system on the first client host.**

   Repeat this step for all client hosts that have the Sun QFS shared file system mounted.

   For example:

```
client# umount sharefs1
```

3. **Use the `umount`(1M) command to unmount the Sun QFS shared file system on the metadata server.**

   For example:

```
metaserver# umount sharefs1
```

Tip – You can use the `samsharefs`(1M) command to verify that you are, indeed, logged into the metadata server or a client host.

---

4. **If you have not already done so, log in as superuser to the metadata server for the Sun QFS shared file system.**

5. **Use the `samsharefs`(1M) command to obtain the current configuration information.**

   The following example command writes current configuration information to file `/etc/opt/SUNWsamfs/hosts.sharefs1`:

```
# samsharefs -R sharefs1 > /etc/opt/SUNWsamfs/hosts.sharefs1
```

6. **Use `vi`(1) or another editor to open the Sun QFS shared file system information file.**

   For example:

```
# vi /etc/opt/SUNWsamfs/hosts.sharefs1
```

CODE EXAMPLE 5-25 shows the file prior to deleting the client host.

CODE EXAMPLE 5-25 `hosts.sharefs1` Prior to Deleting a Client Host

```
# File /etc/opt/SUNWsamfs/hosts.sharefs1
# Host   Host IP                          Server   Not  Server
# Name   Addresses                        Priority Used Host
# ----   -------------------------------- -------- ---- -----
titan    172.16.0.129,titan.xyzco.com     1        -    server
tethys   172.16.0.130,tethys.xyzco.com    2        -
mimas    mimas.xyzco.com                  -        -
dione    dione.xyzco.com                  -        -
helene   helene.xyzco.com                 -        -
```

7. **Use the editor to delete the client host(s) that are no longer to be supported.**

CODE EXAMPLE 5-26 shows the file after the line for `helene` has been deleted.

CODE EXAMPLE 5-26 `hosts.sharefs1` After Deleting a Client Host

```
# File /etc/opt/SUNWsamfs/hosts.sharefs1
# Host   Host IP                          Server   Not  Server
# Name   Addresses                        Priority Used Host
# ----   -------------------------------- -------- ---- -----
titan    172.16.0.129,titan.xyzco.com     1        -    server
tethys   172.16.0.130,tethys.xyzco.com    2        -
mimas    mimas.xyzco.com                  -        -
dione    dione.xyzco.com                  -        -
```

8. **Use the `samsharefs`(1M) command to update the current hosts information.**

   For example:

   ```
   # samsharefs -R -u sharefs1
   ```

   The host `helene` has been removed.

9. **Use the `samsharefs`(1M) command to display the current configuration.**

   For example:

   ```
   # samsharefs -R sharefs1
   ```

10. **Use the `mount`(1M) command to mount the Sun QFS shared file system on the metadata server.**

    For information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

11. **Use the `mount`(1M) command to mount the Sun QFS shared file system on the client hosts.**

    For information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

Changing the Metadata Server

Changing the metadata server enables you to perform a manual failover. The procedures in the following sections describe how to change the metadata server in a Sun QFS shared file system without using the automatic Membership Services feature of a software package such as Sun Cluster.

You can perform a manual failover if the metadata server goes down or becomes unavailable. Such a failover can also be performed if you want to change the metadata server or the potential metadata servers. For failover purposes, the mount options of the metadata server and all potential metadata servers should be the same.

Choose from one of the following procedures depending on the environment you are in and on whether the metadata server is available at the time the failover is being performed:

- "To Change the Metadata Server When the Metadata Server is Up (Sun QFS Environment)" on page 124
- "To Change the Metadata Server When the Metadata Server is Down (Sun QFS Environment)" on page 124
- "To Change the Metadata Server When the Metadata Server is Up (Sun SAM-QFS Environment)" on page 125
- "To Change the Metadata Server When the Metadata Server is Down (Sun SAM-QFS Environment)" on page 129

▼ To Change the Metadata Server When the Metadata Server is Up (Sun QFS Environment)

This procedure shows how to change the metadata server of a Sun QFS shared file system in a Sun QFS environment when the metadata server is up.

● **On the metadata server, issue the `samsharefs`(1M) command to declare the new metadata server.**

For example:

```
titan# samsharefs -s tethys sharefs1
```

▼ To Change the Metadata Server When the Metadata Server is Down (Sun QFS Environment)

This procedure shows how to change the metadata server of a Sun QFS shared file system in a Sun QFS environment when the metadata server is down.

1. **Ensure that the metadata server cannot restart without being rebooted.**

   For example, ensure that the server is powered down, rebooted, halted, or disconnected from the metadata disks.

2. **From the new (potential) metadata server, wait for at least the period of the maximum lease time and then issue the `samsharefs`(1M) command.**

   The wait is necessary because you must ensure that all client leases expire before the failover is performed. From the new metadata server, issue a command such as the following:

```
tethys# samsharefs -R -s tethys sharefs1
```

   If you are uncertain as to whether or not the lease time has expired, use the `samu`(1M) `N` display. For information on `samu`(1M), see "Using the samu(1M) Operator Utility" on page 153. For information on leases and their durations, see "Using Leases in a Sun QFS Shared File System: the rdlease=n, wrlease=n, aplease=n Options" on page 133.

Caution – If you use the `-R` option to the `samsharefs`(1M) command on a mounted file system to change the metadata server host, you must first stop, disable, and disconnect the active metadata server. Not doing so can cause file system corruption.

▼ To Change the Metadata Server When the Metadata Server is Up (Sun SAM-QFS Environment)

This procedure shows how to change the metadata server of a Sun QFS shared file system in a Sun SAM-QFS environment when the metadata server is up.

1. **Log into the metadata server.**

2. **Issue a `samcmd aridle fs.`*fsname* command from the metadata server for the shared file system.**

   For example:

```
titan# samcmd aridle fs.sharefs1
```

This step in the procedure cleanly halts the archiving for file system `sharefs1`. Specifically, it allows archiving operations to halt at a logical place before stopping the daemons.

3. **Issue a** `samd stop` **command.**

   This command stops all removable media activity.

   For example:

```
titan# samd stop
```

   You can verify that archiving has stopped by examining the `samu`(1M) `a` display. In CODE EXAMPLE 5-27, note that the last line indicates that the system is waiting for `arrun`. This message indicates that archiving has stopped gracefully for file system `sharefs1`.

CODE EXAMPLE 5-27 `samu`(1M) `a` Display

```
Archiver status              samu   4.0.work Wed Jul 24 10:10:06


sam-archiverd:  Idle


sam-arfind:  sqfs1 mounted at /sharefs1
Waiting for :arrun fs.sharefs1
```

4. **Use the** `cp`**(1) command to enable the** `mcf.off` **file.**

   The `mcf.off` file has all Device State fields set to `off` for all shared libraries and their drives. For example:

```
titan# cp /etc/opt/SUNWsamfs/mcf.off /etc/opt/SUNWsamfs/mcf
```

5. **Evaluate the** `diskvols.conf` **file on the metadata server. (Optional)**

   Perform this step if you are using disk archiving.

   Depending on how you have enabled disk archiving, you might need to change your `diskvols.conf` files to point to different client or server systems. For information on the `diskvols.conf` file, see *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

6. **Use the** `fuser`**(1M) command to kill any processes that are busy in the catalog's file system.**

For example:

```
titan# fuser -c -k /catalog
```

7. **Unmount the file system that contains the catalog.**

   For example:

```
titan# umount /catalog
```

8. **Send a HUP signal to the `sam-fsd` daemon.**

   The HUP is needed to inform the `sam-fsd` daemon of the configuration change.

   For example:

```
titan# pkill -HUP sam-fsd
```

9. **Log in to the potential metadata server.**

   This is the host that will be the new metadata server after this procedure is completed.

10. **Issue a `samcmd aridle fs.`*fsname* command from the potential metadata server for the shared file system.**

    For example:

```
tethys# samcmd aridle fs.sharefs1
```

11. **Issue a `samd stop` command.**

    This command stops all removable media activity.

    For example:

```
tethys# samd stop
```

12. **Use the `cp`(1) command to enable the `mcf.on` file.**

The `mcf.on` file has all Device State fields set to `on` for all shared libraries and their drives. For example:

```
tethys# cp /etc/opt/SUNWsamfs/mcf.on /etc/opt/SUNWsamfs/mcf
```

13. **Evaluate the `diskvols.conf` file on the new metadata server. (Optional)**

    Perform this step if you are using disk archiving.

    Depending on how you have enabled disk archiving, you might need to change your `diskvols.conf` files to point to different client or server systems. For information on the `diskvols.conf` file, see *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

14. **On the new metadata server, use the `mount`(1M) command to mount the file system containing the Sun SAM-QFS media catalog.**

    For example:

    ```
    tethys# mount /catalog
    ```

15. **On the new metadata server, issue the `samsharefs`(1M) command to declare the new metadata server.**

    For example:

    ```
    tethys# samsharefs -s tethys sharefs1
    ```

16. **Display the `/var/adm/message` file and search for the message that indicates that the failover completed.**

    The message you are searching for is as follows:

    ```
    Jul 10 12:46:10 titan samfs: [ID 949561 kern.notice] NOTICE:
    SAM-FS: Failed over to server tethys; filesystem samfs64, active
    operations = 0.
    ```

17. **Send a HUP signal to the `sam-fsd` daemon.**

    The HUP is needed to inform the `sam-fsd` daemon of the configuration change.

    For example:

    ```
    tethys# pkill -HUP sam-fsd
    ```

18. **Use the `samd`(1M) command to restart the storage and archive manager on the new metadata server.**

    For example:

```
tethys# samd start
```

19. **Use the `samcmd arrun` command to start the archvier.**

    For example:

```
tethys# samcmd arrun fs.sharefs1
```

▼ To Change the Metadata Server When the Metadata Server is Down (Sun SAM-QFS Environment)

This procedure shows how to change the metadata server of a Sun QFS shared file system in a Sun SAM-QFS environment when the metadata server is down.

1. **Ensure that old metadata server cannot restart without being rebooted.**

   For example, ensure that the server is powered down, halted, or disconnected from the metadata disks.

2. **Log in to the potential metadata server.**

   This is the host that will be the new metadata server after this procedure is completed.

3. **Issue a `samd stop` command.**

   This command stops all removable media activity.

   For example:

```
tethys# samd stop
```

4. **Use the `cp`(1) command to enable the `mcf.on` file.**

The `mcf.on` file has all Device State fields set to `on` for all shared libraries and their drives. For example:

```
tethys# cp /etc/opt/SUNWsamfs/mcf.on /etc/opt/SUNWsamfs/mcf
```

5. **Evaluate the `diskvols.conf` file on the new metadata server. (Optional)**

   Perform this step if you are using disk archiving.

   Depending on how you have enabled disk archiving, you might need to change your `diskvols.conf` files to point to different client or server systems. For information on the `diskvols.conf` file, see *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

6. **Use the `mount`(1M) command to mount the file system containing the Sun SAM-QFS archiver catalog on the new metadata server.**

   For example, assuming that the catalog is stored in a UFS file system in file system `/catalog`, the following command mounts the file system:

```
tethys# mount /catalog
```

7. **From the new metadata server, wait for at least the period of the maximum lease time and then issue the `samsharefs`(1M) command.**

   The wait is necessary because you must ensure that all client leases expire before the failover is performed. From the new metadata server, issue a command such as the following:

```
tethys# samsharefs -R -s tethys sharefs1
```

   If you are uncertain as to whether or not the lease time has expired, use the `samu`(1M) `N` display. For information on `samu`(1M), see “Using the samu(1M) Operator Utility” on page 153. For information on leases and their durations, see “Using Leases in a Sun QFS Shared File System: the rdlease=n, wrlease=n, aplease=n Options” on page 133.

Caution – If you use the `-R` option to the `samsharefs`(1M) command on a mounted file system to change the metadata server host, you must first stop, disable, and disconnect the active metadata server. Not doing so can cause file system corruption.

8. **Send a HUP signal to the `sam-fsd` daemon.**

   The HUP is needed to inform the `sam-fsd` daemon of the configuration change.

For example:

```
tethys# pkill -HUP sam-fsd
```

9. **Use the** `samd`**(1M) command to restart the storage and archive manager on the new metadata server.**

   Perform this step if you are using a Sun QFS shared file system in a Sun SAM-QFS environment. For example

```
tethys# samd start
```

10. **Use the** `samcmd arrun` **command to start the archvier.**

    For example:

```
tethys# samcmd arrun fs.sharefs1
```

Daemons

In a Sun QFS shared file system, a `sam-fsd` daemon is always active. In addition, one `sam-sharefsd` daemon is active for each mount point configured in the Sun QFS shared file system.

When a `sam-fsd` daemon recognizes a Sun QFS shared file system, it starts a shared file system daemon (`sam-sharefsd`). BSD sockets are used to communicate between the server and client hosts. All clients that connect to the metadata server are validated against the hosts file.

One Sun QFS shared file system daemon is started for each Sun QFS shared file system shared mount point on each client host. This daemon establishes a connection to the metadata server. The `sam-sharedfsd` on the metadata server opens a listener socket on the port associated with this file system. The shared file system port is defined in either the `/etc/inet/services` file or the `/etc/yp/src/services` file (if you are using NIS) as `samsock.`*fs_name*.

All metadata operations, block allocation and deallocation, file locking, and record locking are performed on the metadata server file system. The `sam-sharefsd` daemon does not keep any information. Hence, it can be killed and restarted without causing any consistency problems for the file system.

Mount Options in a Sun QFS Shared File System

The Sun QFS shared file system can be mounted with several mount options. This chapter describes many options within the context of their roles. Other options, however, are useful only in certain situations. This section describes the mount options that can be used for special purposes.

Most mount options can be specified on the `mount`(1M) command, in the `/etc/vfstab` file, or in the `samfs.cmd`(4) file. For example, the following `/etc/vfstab` file includes `mount`(1M) options for a Sun QFS shared file system:

```
sharefs1 - /sfs samfs - no shared,mh_write
```

The following sections summarize the mount options available to you in a Sun QFS shared file system. For more information on any of these mount options, see the `mount_samfs`(1M) man page or see the cross references mentioned in their descriptions.

Mounting in the Background: the `bg` Option

The `bg` mount option specifies that if the first mount operation fails, subsequent attempts at mounting should occur in the background. By default, `bg` is not in effect, and mount attempts continue in the foreground.

Reattempting a File System Mount: the `retry` Option

The `retry` mount option specifies the number of times that the system should attempt to mount a file system. The default is 10000.

Declaring a Sun QFS Shared File System: the `shared` Option

The `shared` mount option declares a file system to be a Sun QFS shared file system. This option must be specified in the `/etc/vfstab` file in order for the file system to be mounted as a Sun QFS shared file system. The presence of this option in a `samfs.cmd`(4) file or on the `mount`(1M) command does not causes an error condition, but it does not mount the file system as a Sun QFS shared file system.

For more information on how to use this option, see "To Configure the Metadata Server" on page 97 or see "To Configure a Client Host" on page 106.

Tuning Allocation Sizes: the `minallocsz=`*n* and `maxallocsz=`*n* Options

The `-o minallocsz=`*n* and `-o maxallocsz=`*n* options to the `mount`(1M) command specify an amount of space, in kilobytes. If a file is growing, the metadata server allocates blocks when an append lease is granted. The size of this allocation is specified by the `-o minallocsz=`*n* option. The metadata server can increase the size of the block allocation depending on the application's access patterns up to, but not exceeding, the `-o maxallocsz=`*n* option's setting.

These `mount`(1M) options can be specified on the `mount`(1M) command line, in the `/etc/vfstab` file, or in the `samfs.cmd` file.

Using Leases in a Sun QFS Shared File System: the `rdlease=`*n*, `wrlease=`*n*, `aplease=`*n* Options

A *lease* grants a shared host permission to perform an operation on a file for as long as the lease is valid. The metadata server issues leases to each shared host including itself. The leases are renewed as necessary to permit continued file operations. The possible file operations are as follows:

- A read lease enables existing file data to be read.
- A write lease enables existing file data to be overwritten.
- An append lease enables a file's size to be extended and enables newly allocated blocks to be written.

A shared host can continue to update leases for as long as necessary. The lease is tranparent to the end user. TABLE 5-4 shows the mount options that enable you to specify the duration of each lease type.

TABLE 5-4 Lease-related `mount`(1M) Options

Option	Action
`-o rdlease=`*n*	This option specifies the maximum amount of time, in seconds, for the read lease.
`-o wrlease=`*n*	This option specifies the maximum amount of time, in seconds, for the write lease.
`-o aplease=`*n*	This option specifies the maximum amount of time, in seconds, for the append lease.

All three leases enable you to specify an n such that $15 \leq n \leq 600$. The default time for each lease is 30 seconds. A file cannot be truncated or removed if a lease is in effect. For more information on setting these leases, see the `mount_samfs`(1M) man page.

If you change the metadata server because the current metadata server is down, you must add the lease time to the failover time because all leases must expire before an alternate metadata server can assume control. The high availability or cluster software must not mount a new metadata server until all leases have expired. For information on how to add the lease time to the failover time, see your high availability documentation.

Setting a small lease time causes more traffic between the client hosts and the metadata server because the lease must be renewed after it has expired.

Enabling Multiple Host Reads and Writes: the `mh_write` Option

By default, in a Sun QFS shared file system, multiple hosts can read the same file at the same time, and if no host is writing to that file, I/O can be paged on all hosts. Only one host can append to a file at any one time.

If `mh_write` is specified as a mount option on the metadata server host, the Sun QFS shared file system enables simultaneous reads and writes to the same file from multiple hosts. If `mh_write` is not specified on the metadata server host, only one host can write to a file at any one time.

The `mh_write` option controls write access to the same file from multiple hosts. By default, `mh_write` is disabled, and only one host is permitted to have write access to a file at any one time. The length of that time period is determined by the duration of the wrlease mount option. If the Sun QFS shared file system is mounted on the metadata server with the mh_write option enabled, simultaneous reads and writes to the same file can occur from multiple hosts.

TABLE 5-5 describes how file access from multiple hosts is affected depending on whether the mh_write is enabled on the metadata server.

TABLE 5-5 File Access Based on the mh_write Option

mh_write **Not Enabled on the Metadata Server**	mh_write **Enabled on the Metadata Server**
Multiple reader hosts allowed. Can use paged I/O.	Multiple reader hosts allowed. Can use paged I/O.
Only one writer host is allowed. Can use paged I/O. All other hosts wait.	Multiple reader and/or writer hosts allowed. If any writer hosts exist, all I/O is direct.
Only one append host. All other hosts wait.	Only one append host is allowed. All other hosts can read and/or write. If any writer hosts exist, all I/O is direct.

For more information on mh_write, see the mount_samfs(1M) man page.

Setting the Number of Concurrent Threads: the nstreams=*n* Option

The nstreams=*n* mount option sets the number of concurrent threads for the Sun QFS shared file system. By default, nstreams=16. This means, for example, that under default settings, up to 16 operations can be processed simultaneously, and the 17th operation commences only after an operation has finished. The nstreams=*n* mount option can be adjusted based on the Sun QFS shared file system's activity. For *n*, specify a value such that $4 \le n \le 256$.

Retaining Cached Attributes: the meta_timeo=*n* Option

The meta_timeo=*n* mount option determines how long the system waits between checks on the metadata information. By default, the system refreshes metadata information every 15 seconds. This means, for example, that an ls(1) command entered in a Sun QFS shared file system with several newly created files might not return information on all the files until 15 seconds had passed. For *n*, specify a value such that $0 \le n \le 60$.

Specifying Striped Allocation: the `stripe` Option

By default, data files in the Sun QFS shared file system are allocated using the round-robin file allocation method. To specify that file data be striped across disks, you can specify the `stripe` mount option on the metadata host and all potential metadata hosts. Note that by default, unshared file systems allocate file data using the striped method.

In a round-robin allocation, files are created in a round-robin fashion on each slice or striped group. This causes the maximum performance for one file to be the speed of a slice or striped group. For more information on file allocation methods, see "File System Design" on page 11.

Specifying the Frequency With Which Metadata is Written: the `sync_meta=`*n* Option

The `sync_meta=`*n* option can be set to `sync_meta=1` or `sync_meta=0`.

By default, `sync_meta=1` and a Sun QFS shared file system writes file metadata to disk every time the metadata changes. This slows data performance, but it ensures data consistency. This is the setting that must be in effect if failover capability is required.

If you set `sync_meta=0`, the Sun QFS shared file system writes the metadata to a buffer before writing it to disk. This delayed write delivers higher performance, but it decreases data consistency after an unscheduled machine interruption.

Mount Semantics in a Sun QFS Shared File System

The behavior of the Sun QFS shared file system is that of an interruptable hard connection. Each client tries repeatedly to communicate with the metadata server, even if the server is unavailable. If the metadata server is not responding, any user can teminate the communication attempt by pressing CTRL-C. If the communication attempt is not terminated, the client persists until the communication is successful.

The system generates the following messages to describe status conditions:

- `SAM-FS: Shared server is not responding.`

This message is also generated if the client `sam-sharefsd` daemon is not active or if the server `sam-sharefsd` daemon is not active.

When the server responds, the following message is returned:

```
SAM-FS: Shared server is responding.
```

- `SAM-FS: Shared server is not mounted.`

When the Sun QFS shared file system mounts on the server, the following message is returned:

```
SAM-FS: Shared server is mounted.
```

File Locking in a Sun QFS Shared File System

Mandatory locks are not supported. An EACCES error is returned if the mandatory lock is set. Advisory locks are supported. For more information on advisory locks, see the `fcntl`(2) system call.

Troubleshooting a Failed or Hung `sammkfs`(1M) or `mount`(1M) Command

The following sections describe what to do when a `sammkfs`(1M) or `mount`(1M) command fails or when a `mount`(1M) command hangs.

The procedures in this section can be performed on client hosts and can also be performed on the server. Commands that can be executed only on the metadata server are preceded with a `server#` prompt.

Recovering a Failed sammkfs(1M) Command

If the sammkfs(1M) command returns an error or messages indicating that an unexpected set of devices are to be initialized, you need to perform this procedure. It includes steps for verifying and reinitializing the mcf(4) file.

▼ To Verify and Reinitialize the mcf(4) File

1. **Use the sam-fsd(1M) command to verify the mcf(4) file.**

   For example:

   ```
   # sam-fsd
   ```

   Examine the output from the sam-fsd(1M) command.

2. **Edit the mcf(4) file to resolve the diagnostic issues. (Optional)**

   Perform this step if the output from the sam-fsd(1M) command indicates that there are errors in the /etc/opt/SUNWsamfs/mcf file.

3. **Issue the sam-fsd(1M) command again to verify the mcf(4) file.**

   Repeat Step 1, Step 2, and Step 3 of this process until the output from the sam-fsd(1M) command indicates that the mcf(4) file is correct.

4. **Send a HUP signal to the sam-fsd daemon.**

   The HUP is needed to inform the sam-fsd daemon of the configuration change.

   For example:

   ```
   # pkill -HUP sam-fsd
   ```

Recovering a Failed mount(1M) Command

A mount(1M) command can fail for several reasons. This section describes some actions you can take to remedy a mount problem. If the mount(1M) command hangs, rather than fails, see "Recovering a Hung mount(1M) Command" on page 144.

Some failed mount(1M) behaviors and their remedies are as follows:

- If the mount(1M) command fails with a `Shared server is not mounted` message generated on a client, determine the server host and mount the file system on the metadata server.

- If the mount command fails with a message indicating that there is a mismatch between the file system and the `mcf`(4) file, ensure the following:
  - That the `mcf`(4) file is syntactically valid. For more information, see “To Verify and Reinitialize the mcf(4) File” on page 138.
  - That recent changes to the `mcf`(4) file are valid and have been enacted. For more information, see “To Verify and Reinitialize the mcf(4) File” on page 138.
  - That the `mcf`(4) file matches the server’s `mcf`(4) file with device names or controller numbers adjusted for any differences on the client. You can use the `samfsconfig`(1M) command to diagnose some of these problems. For more information on using the `samfsconfig`(1M) command, see “To Use the samfsconfig(1M) Command” on page 142.
- If the `mount`(1M) command fails for other reasons, use the procedures described in the following sections to verify the system characteristics that must be in place in order for the `mount`(1M) command to be successful. These procedures are as follows:
  - “To Verify that the File System can be Mounted” on page 139
  - “To Use the samfsinfo(1M) and samsharefs(1M) Commands” on page 140
  - “To Use the samfsconfig(1M) Command” on page 142

▼ To Verify that the File System can be Mounted

The following procedure shows you what to verify if the `mount`(1M) command fails:

1. **Ensure that the mount point directory is present.**

For example, you can issue the `ls`(1) command, as follows:

```
ls -ld mountpoint
```

where

mountpoint	The name of the Sun SAM-QFS shared file system’s mount point.

When you examine the `ls`(1) command’s output, make sure that the output shows a directory with access mode 755. In other words, the codes should read `rwxr-xr-x`. CODE EXAMPLE 5-28 shows example output.

CODE EXAMPLE 5-28 Access Mode Values

```
# ls -ld /sharefs1
drwxr-xr-x   2 root     sys          512 Mar 19 10:46 /sharefs1
```

If the access is not at this level, enter the following chmod(1) command:

```
chmod 755 mountpoint
```

2. **Ensure that there is an entry for the file system in the /etc/vfstab file.**

   For example, the following /etc/vfstab file shows an entry for the shared file system named sharefs1:

CODE EXAMPLE 5-29 Example /etc/vfstab File

```
# File /etc/vfstab
# FS name  FS to fsck  Mnt pt FS type  fsck pass  Mt@boot  Mt params
sharefs1    -          /sharefs1 samfs -          yes      shared,bg
```

3. **Ensure that the shared flag is present in the Mount Parameters field of the shared file system's entry in the /etc/vfstab file.**

4. **Ensure that the mount point directory is not shared out for NFS use.**

   If the mount point is shared, use the unshare(1M) command to unshare it. For example:

```
# unshare mountpoint
```

If none of the preceding steps expose errors, perform "To Use the samfsinfo(1M) and samsharefs(1M) Commands" on page 140. This procedure verifies that the file system has been created and that the shared hosts file is correctly initialized.

▼ To Use the samfsinfo(1M) and samsharefs(1M) Commands

1. **Enter the samfsinfo(1M) command on the server.**

   This command has the following format:

```
samfsinfo filesystem
```

where

filesystem	The name of the Sun SAM-QFS shared file system as specified in the mcf(4) file.

For example:

CODE EXAMPLE 5-30 `samfsinfo`(1M) Command Example

```
titan-server# samfsinfo sharefs1
samfsinfo: filesystem sharefs1 is mounted.
name:     sharefs1       version:     2    shared
time:     Mon Apr 29 15:12:18 2002
count:    3
capacity:      10d84000          DAU:          64
space:         10180400
meta capacity: 009fe200          meta DAU:     16
meta space:    009f6c60
ord  eq   capacity      space   device
1    11   086c0000   080c39b0   /dev/dsk/c1t2100002037E9C296d0s6
2    12   086c4000   080bca50   /dev/dsk/c3t50020F2300005D22d0s6
3    13   086c4000   080a9650   /dev/dsk/c3t50020F2300006099d0s6
4    14   086c4000   08600000   /dev/dsk/c3t50020F230000651Cd0s6
```

The output from CODE EXAMPLE 5-30 shows a `shared` keyword in the following line:

```
name:     sharefs1       version:     2    shared
```

Also note the list of file system devices, ordinals, and equipment numbers that appear after the following line:

```
ord  eq   capacity      space   device
```

These numbers should correspond to the devices in the file system's `mcf`(4) entry.

2. **Enter the `samsharefs`(1M) command on the server.**

This command has the following format:

```
samsharefs -R filesystem
```

where

filesystem	The name of the Sun SAM-QFS shared file system as specified in the `mcf`(4) file.

For example:

CODE EXAMPLE 5-31 samsharefs(1M) Command Example

```
titan-server# samsharefs -R sharefs1
#
# Host file for family set ’sharefs1’
#
# Version: 3    Generation: 50    Count: 4
# Server = host 0/titan, length = 216
#
titan 173.26.2.129,titan.foo.com 1 - server
tethys 173.26.2.130,tethys.foo.com 2 -
dione dione.foo.com 0 -
mimas mimas.foo.com 0 -
```

The following information pertains to the diagnostic output from the `samfsinfo`(1M) or `samsharefs`(1M) commands.

- If either command issues diagnostics or error messages, resolve them. Ensure that the output from the `samfsinfo`(1M) command includes the `shared` keyword.
- These commands can be executed on alternate server hosts and client hosts that have no `nodev` devices in the host’s `mcf` entry for the file system.

If the `samfsinfo`(1M) and `samsharefs`(1M) commands do not expose irregularities, perform “To Use the samfsconfig(1M) Command” on page 142.

▼ To Use the `samfsconfig`(1M) Command

On clients with `nodev` device entries in the `mcf` file for the file system, the entire file system might not be accessible, and the shared hosts file might not be directly accessible. You can use the `samfsconfig`(1M) command to determine whether the shared file system’s data partitions are accessible.

- **Issue the samfsconfig(1M) command.**

The `samfsconfig`(1M) command has the following format:

```
samfsconfig list_of_devices
```

where

list_of_devices	List of devices from the file system entry in the `mcf`(4) file. Use a space to separate multiple devices in the list.

Example 1. CODE EXAMPLE 5-32 shows the `samfsconfig`(1M) command being used on a host that does not have a `nodev` entry in its `mcf` file. CODE EXAMPLE 5-32 shows the `mcf` file for the host `tethys`.

CODE EXAMPLE 5-32 `samfsconfig`(1M) Command Example Without `nodev` Entries

```
tethys# cat /etc/opt/SUNWsamfs/mcf
sharefs1                        10  ma   sharefs1    on  shared
/dev/dsk/c1t2100002037E9C296d0s6 11  mm   sharefs1    -
/dev/dsk/c3t50020F2300005D22d0s6 12  mr   sharefs1    -
/dev/dsk/c3t50020F2300006099d0s6 13  mr   sharefs1    -
/dev/dsk/c3t50020F230000651Cd0s6 14  mr   sharefs1    -

tethys# samfsconfig /dev/dsk/c1t2100002037E9C296d0s6
/dev/dsk/c3t50020F2300005D22d0s6 /dev/dsk/c3t50020F2300006099d0s6
/dev/dsk/c3t50020F230000651Cd0s6
#
# Family Set ’sharefs1’ Created Mon Apr 29 15:12:18 2002
#
sharefs1                          10    ma   sharefs1  - shared
/dev/dsk/c1t2100002037E9C296d0s6   11    mm   sharefs1  -
/dev/dsk/c3t50020F2300005D22d0s6   12    mr   sharefs1  -
/dev/dsk/c3t50020F2300006099d0s6   13    mr   sharefs1  -
/dev/dsk/c3t50020F230000651Cd0s6   14    mr   sharefs1  -
```

Example 2. CODE EXAMPLE 5-33 shows the `samfsconfig`(1M) command being used on a host that has a `nodev` entry in its `mcf` file.

CODE EXAMPLE 5-33 `samfsconfig`(1M) Command Example With `nodev` Entries

```
dione# cat /etc/opt/SUNWsamfs/mcf
sharefs1                          10    ma   sharefs1  on  shared
nodev                              11    mm   sharefs1  -
/dev/dsk/c4t50020F23000055A8d0s3   12    mr   sharefs1  -
/dev/dsk/c4t50020F23000055A8d0s4   13    mr   sharefs1  -
/dev/dsk/c4t50020F23000055A8d0s5   14    mr   sharefs1  -

dione# samfsconfig /dev/dsk/c4t50020F23000055A8d0s3
/dev/dsk/c4t50020F23000055A8d0s4 /dev/dsk/c4t50020F23000055A8d0s5
#
# Family Set ’sharefs1’ Created Mon Apr 29 15:12:18 2002
#
# Missing slices
# Ordinal 1
# /dev/dsk/c4t50020F23000055A8d0s3    12    mr   sharefs1  -
```

CODE EXAMPLE 5-33 `samfsconfig`(1M) Command Example With `nodev` Entries

```
# Ordinal 2
# /dev/dsk/c4t50020F23000055A8d0s4    13    mr   sharefs1  -
# Ordinal 2
# /dev/dsk/c4t50020F23000055A8d0s5    14    mr   sharefs1  -
```

For examples 1 and 2, you need to verify that the output lists all slices from the file system other than the metadata (`mm`) devices as belonging to the file system. This is the case for example 2.

Recovering a Hung `mount`(1M) Command

If the `mount`(1M) command hangs, follow the procedure in this section. You have a hung `mount`(1M) command if, for example, the `mount`(1M) command fails with a connection error or with a `Server not responding` message that does not resolve itself within 30 seconds.

The most typical remedy for a hung `mount`(1M) command is presented first. If that does not work, perform the subsequent procedures.

▼ To Verify Network Connections

The `netstat`(1M) command verifies that the `sam-sharefsd` daemon's network connections are correctly configured.

1. **Enter the `netstat`(1M) command on the server.**

   The format of this command is as follows:

```
netstat -a | grep samsock.filesystem
```

For example, the following command is entered on server `titan`:

CODE EXAMPLE 5-34 `netstat`(1M) Example on the Server

```
titan-server# netstat -a | grep samsock.sharefs1
      *.samsock.sharefs1 *.*         0     0 24576  0 LISTEN
titan.32891  titan.samsock.sharefs1 32768  0 32768  0 ESTABLISHED
titan.samsock.sharefs1  titan.32891 32768  0 32768  0 ESTABLISHED
titan.samsock.sharefs1 tethys.32884 24820  0 24820  0 ESTABLISHED
titan.samsock.sharefs1  dione.35299 24820  0 24820  0 ESTABLISHED
     *.samsock.sharefs1    *.*        0    0 24576  0 LISTEN
```

2. **Verify the output from the `netstat`(1M) command on the server.**

Verify that at least three output lines, including one `LISTEN` and two `ESTABLISHED` entries are present. There should be one extra `ESTABLISHED` entry for each client that is configured and running whether or not it is mounted.

3. **Enter the `netstat`(1M) command on the client.**

   Use the format of the `netstat`(1M) command as shown in Step 1.

   For example, the following command is entered on client `dione`:

CODE EXAMPLE 5-35 `netstat`(1M) Command on the Client

```
dione-client# netstat -a | grep samsock.sharefs1
dione.35299  titan.samsock.sharefs1  24820  0  24820  0 ESTABLISHED
```

Verify that one line is present, including an `ESTABLISHED` connection. There should be no `LISTEN` lines. If an `ESTABLISHED` connection is not reported, perform one or more of the following procedures:

- “To Verify that the Client Can Reach the Server (Optional)” on page 145
- “To Verify that the Server Can Reach the Client (Optional)” on page 147
- “To Verify Service Name Availability (Optional)” on page 148
- “To Examine the sam-sharefsd Trace Log (Optional)” on page 150

▼ To Verify that the Client Can Reach the Server (Optional)

Perform these steps if using the procedure described in “To Verify Network Connections” on page 144 did not show an `ESTABLISHED` connection.

1. **Use the `samsharefs`(1M) command to verify the hosts file on the server.**

   Use the `-R` option in the following format:

```
samsharefs -R filesystem
```

where

filesystem	The name of the Sun SAM-QFS shared file system as specified in the `mcf`(4) file.

For example:

CODE EXAMPLE 5-36 shamsharefs(1M) -R Command

```
titan-server# samsharefs -R sharefs1
#
# Host file for family set ’sharefs1’
#
# Version: 3    Generation: 50    Count: 4
# Server = host 0/titan, length = 216
#
titan 173.26.2.129,titan.foo.com 1 - server
tethys 173.26.2.130,tethys.foo.com 2 -
dione dione.foo.com 0 -
mimas mimas.foo.com 0 -
```

2. **Save this output.**

   If the steps in this procedure fail, you need this output for use in subsequent procedures.

3. **Verify that the output matches expectations.**

   If the command fails, verify that the file system was created. In this case it is likely that the file system was never created or that the initial hosts configuration files have not been created. For information on configuring these files, see the procedures earlier in this chapter. The configuration process involves editing the existing `mcf`(4) file, reinitializing the `mcf`(4) file, and configuring the hosts files.

   The `samsharefs`(1M) command can be executed on alternate server hosts and client hosts that have no `nodev` devices listed in the host’s `mcf`(4) entry for the file system.

4. **Find the row containing the server’s name in the first column.**

5. **From the client, use the `ping`(1M) command on each entry from the second column of `samsharefs`(1M) output to verify that the server can be reached.**

   This command has the following format:

```
ping servername
```

where

servername	The name of the server as shown in the second column of the `samsharefs`(1M) command’s output.

For example:

CODE EXAMPLE 5-37 Using `ping`(1M) on Systems Named in `samsharefs`(1M) Output

```
dione-client# ping 173.26.2.129
ICMP Host Unreachable from gateway dione (131.116.7.218)
for icmp from dione (131.116.7.218) to 173.26.2.129
dione-client# ping titan.foo.com
titan.foo.com is alive
```

6. **From the client, examine the `hosts.`*filesystem*`.local` file. (Optional)**

   Perform this step if the `ping`(1M) command revealed unreachable hosts.

   If there is more than one entry in the second column of `samsharefs`(1M) output, and if some of the entries are not reachable, ensure that the reachable entries for the entries you want the shared file system to use are present and that it or they are also present in the `/etc/opt/SUNWsamfs/hosts.`*filesystem*`.local` file entry for the server. Ensure that the unreachable hosts are not entered in these places.

   If the `sam-sharefsd` daemon attempts to connect to unreachable server interfaces, there can be substantial delays in its connecting to the server after installation, rebooting, or file system host reconfiguration.

   For example:

   CODE EXAMPLE 5-38 Examining the `hosts.`*filesystem*`.local` File

```
dione-client# cat /etc/opt/SUNWsamfs/hosts.sharefs1.local
titan       titan.foo.com            # ! 173.26.2.129
tethys      tethys.foo.com           # ! 173.26.2.130
```

7. **Enable the correct server interfaces. (Optional)**

   If the `ping`(1M) command revealed that there were no reachable server interfaces, then either the server network interfaces must be configured and initialized for typical operations, or you must use the `samsharefs`(1M) command to update the interface names in the hosts file so they match the actual names.

▼ To Verify that the Server Can Reach the Client (Optional)

Perform these steps if the procedure in “To Verify Network Connections” on page 144 did not show an `ESTABLISHED` connection.

1. **Obtain `samsharefs`(1M) output.**

   This can be the output generated in “To Verify that the Client Can Reach the Server (Optional)” on page 145, or you can generate it again using the initial steps in that procedure.

2. **Find the row containing the client’s name in the first column.**

3. **On the client, run the `hostname`(1M) command and ensure that the output matches the name in the first column of `samsharefs`(1M) output.**

   For example:

   CODE EXAMPLE 5-39 `hostname`(1M) Output

   ```
   dione-client# hostname
   dione
   ```

4. **Use the `ping`(1M) command on the server on each entry from the second column to verify that the client can be reached. (Optional)**

   Perform this step if the `hostname`(1M) command output matched the name in the second column of `samsharefs`(1M) output.

   For example:

   CODE EXAMPLE 5-40 `ping`(1M) Output

   ```
   titan-server# ping dione.foo.com
   dione.foo.com is alive
   ```

   It is not necessary that every entry in the row’s column be reachable, but all interfaces that you wish any potential server to accept connections from must be present in the column.

5. **Enable the correct client interfaces. (Optional)**

   If the `ping`(1M) command revealed that there were no reachable client interfaces, then either the client network interfaces must be configured and initialized for typical operations, or you must use the `samsharefs`(1M) command to update the interface names in the hosts file so they match the actual names

▼ To Verify Service Name Availability (Optional)

Perform this step if the procedure in “To Verify Network Connections” on page 144 did not show an `ESTABLISHED` connection.

1. **Obtain `samsharefs`(1M) output.**

   This can be the output generated in "To Verify that the Client Can Reach the Server (Optional)" on page 145, or you can generate it again using the initial steps in that procedure.

2. **Find the row containing the server's name in the second column.**

3. **Use `telnet`(1) to verify that the service name required by the file system is recognized.**

   This command has the following format:

```
telnet server samsock.filesystem
```

   where

server	The server name as taken from the second field of the server's line in the shared hosts file.
filesystem	The name of the Sun SAM-QFS shared file system as specified in the `mcf`(4) file.

   For example:

CODE EXAMPLE 5-41 `telnet`(1) Command Output

```
dione-client# telnet titan.foo.com samsock.sharefs1
Trying 131.116.7.203...
Connected to titan.foo.com.
Escape character is '^]'.
Connection closed by foreign host.
```

4. **Examine the `telnet`(1) command output.**

   If the `telnet`(1) command output indicates that it cannot connect, ensure one of the following:

   - That `samsock.`*filesystem* is listed in the `/etc/inet/services` file and that `inetd` has been sent a HUP signal.
   - That `samsock.`*filesystem* is included in your NIS, NIS+, or LDAP database and is distributed to the hosts of the shared file system.

   If the `telnet`(1) command connects successfully, you can disconnect by holding down the CTRL key while pressing the right bracket (]) key, and then quit typing when you get the `telnet` prompt. Alternatively, the connection should time out in 15 seconds or so.

▼ To Examine the `sam-sharefsd` Trace Log (Optional)

The following procedures can resolve `mount`(1M) problems:

- "To Verify Network Connections" on page 144
- "To Verify that the Client Can Reach the Server (Optional)" on page 145
- "To Verify that the Server Can Reach the Client (Optional)" on page 147
- "To Verify Service Name Availability (Optional)" on page 148

If none of the preceding procedures resolved the problem, perform the steps in this section.

1. **Verify the presence of file** `/var/opt/SUNWsamfs/trace/sam-sharefsd`**.**

   If this file is not present, or if it shows no recent modifications, proceed to the next step.

2. **Edit file** `/etc/opt/SUNWsamfs/defaults.conf` **and add lines to enable** `sam-sharefsd` **tracing. (Optional)**

   Perform this step if Step 1 indicates that file `/var/opt/SUNWsamfs/trace/sam-sharefsd` does not exist or if the file shows no recent modifications.

   Add the following lines to `/etc/opt/SUNWsamfs/defaults.conf`:

```
trace
sam-sharefsd.options = all
endtrace
```

   Alternatively, you could also add the following line to the trace section of your `defaults.conf` file if one already exists.

```
sam-sharefsd.options = all
```

   After tracing has been enabled, reinitialize the `defaults.conf` file by entering the following command:

```
# pkill -HUP samfsd
```

3. **Examine the last few dozen lines of the trace file for hints.**

For example:

CODE EXAMPLE 5-42 Trace File

```
dione# tail -20 /var/opt/SUNWsamfs/trace/sam-sharefsd
2002-05-13 11:23:19 shf-sharefs1[5659]: FS sharefs1: **** shared fs daemon
exited for Host dione
2002-05-13 11:23:29 shf-sharefs1[5663]: FS sharefs1: shared file system daemon
started
2002-05-13 11:23:29 shf-sharefs1[5663]: FS sharefs1: Host dione
2002-05-13 11:23:31 shf-sharefs1[5663]: FS sharefs1: filesystem is mounted
2002-05-13 11:23:33 shf-sharefs1[5663]: FS sharefs1: client dione; server =
titan
2002-05-13 11:23:33 shf-sharefs1[5663]: FS sharefs1: Set Client (Server
titan/1).
2002-05-13 11:23:35 shf-sharefs1[5663]: FS sharefs1: client connected to
titan/titan.foo.com
2002-05-13 11:23:35 shf-sharefs1[5663]: FS sharefs1: SetClientSocket dione
2002-05-13 11:23:50 shf-sharefs1[5663]: OS call error: FS sharefs1:
syscall[SC_client_rdsock] failed: I/O error
2002-05-13 11:23:50 shf-sharefs1[5663]: ClientRdSocket kill Main
2002-05-13 11:23:50 shf-sharefs1[5663]: FS sharefs1: signal 2 received:
Interrupt
2002-05-13 11:23:50 shf-sharefs1[5663]: FS sharefs1: ClientRdSocket died titan:
I/O error
2002-05-13 11:23:50 shf-sharefs1[5663]: FS sharefs1: **** shared fs daemon
exited for Host dione
2002-05-13 11:24:00 shf-sharefs1[5665]: FS sharefs1: shared file system daemon
started
2002-05-13 11:24:00 shf-sharefs1[5665]: FS sharefs1: Host dione
2002-05-13 11:24:03 shf-sharefs1[5665]: FS sharefs1: filesystem is mounted
2002-05-13 11:24:03 shf-sharefs1[5665]: FS sharefs1: client dione; server =
titan
2002-05-13 11:24:03 shf-sharefs1[5665]: FS sharefs1: Set Client (Server
titan/1).
2002-05-13 11:24:05 shf-sharefs1[5665]: FS sharefs1: client connected to
titan/titan.foo.com
2002-05-13 11:24:05 shf-sharefs1[5665]: FS sharefs1: SetClientSocket dione
```

CHAPTER 6

Using the samu(1M) Operator Utility

This chapter provides instructions for controlling the devices configured within your Sun QFS, Sun SAM-FS, and Sun SAM-QFS environment through the `samu`(1M) operator utility. Not all `samu`(1M) displays are supported in a Sun QFS environment, but the three types of environments are described in this chapter for the sake of completeness.

The following topics are presented:

Overview

The `samu`(1M) operator utility requires a display terminal that displays a minimum of 24 lines by 80 characters wide. The utility includes the following features:

- Displays that enable you to monitor Sun QFS, Sun SAM-FS, and Sun SAM-QFS devices and file system activity
- Commands that enable you to select displays, set display options, control access to and the activity of devices, and take snapshots of display windows

The display windows shown in this chapter are representative examples. The exact format and amount of information displayed on your terminal can be different depending on your terminal model and the devices configured in your Sun QFS, Sun SAM-FS, or Sun SAM-QFS environment.

The operations that can be performed from within samu(1M) can also be performed by using the samcmd(1M) command. For more information on samcmd(1M), see the samcmd(1M) man page.

The following sections describe how to start and stop samu(1M), interact with the utility, access the help windows, and view operator displays.

▼ To Invoke samu(1M)

1. **To start samu(1M), enter the samu(1M) command from the UNIX command line as follows:**

```
# samu
```

   The system starts samu(1M) and shows the help display.

2. **Press CTRL-f to move to the next help screen, which shows the keys that control the displays.**

   The samu(1M) command accepts options on its command line. These options include those for selection of its initial display. For more information on the samu(1M) command line options, see the samu(1M) man page.

Note – samu(1M), like the vi(1) editor, is based on the curses(3X) library routine. You must have your terminal type defined correctly before invoking samu(1M).

▼ To Stop samu(1M)

● **To exit samu(1M), enter one of the following:**

- Press the q key
- Enter :q

The samu(1M) operator utility exits and returns you to the command shell.

Interacting with samu(1M)

Interacting with samu(1M) is similar to interacting with the UNIX vi(1) editor with respect to paging forward or backward, entering commands, refreshing the display, and quitting the utility.

While viewing an operator display, you can use the keys described in TABLE 6-1 to control the display. The exact function of these keys depends on the display being viewed at the time. For information on display-specific key operations, see the `samu`(1M) man page.

TABLE 6-1 `samu`(1M) Display Control Key Sequences

Key	Function	Display
CTRL-b	Previous file system	`:a,a`
	Page backward	`c,h,o,p,s,t,u,v,w`
CTRL-d	Half-page forward	`c,p,s,u,w`
	Next robot catalog	`v`
	Page forward (top portion)	`h`
	Page forward (bottom portion)	`a`
CTRL-f	Next file system	`:a,a`
	Page forward	`c,h,o,p,s,t,u,v,w`
CTRL-k	Select (manual, robotic, both, priority)	`p`
	Advance sort key	`v`
	Toggle path display	`n,u,w`
CTRL-u	Half-page backward	`c,p,s,u,w`
	Previous robot catalog	`v`
	Page backward (top portion)	`h`
	Page backward (bottom portion)	`a`
CTRL-i	Detailed, 2-line display format	`v`
1-7	Select sort key, as follows: • 1 sorts by slot. • 2 sorts by count. • 3 sorts by usage. • 4 sorts by VSN. • 5 sorts by access time. • 6 sorts by barcode. • 7 sorts by label time.	`v`
/	Search for VSN	`v`
%	Search for barcode	`v`

Command and display error messages are displayed on the last line of the display window. If a command error occurs, automatic display refreshing halts until the next operator action.

Entering a Device

Each device included in the Sun QFS, Sun SAM-FS, or Sun SAM-QFS environment is assigned an Equipment Ordinal (for example, 10) in the `mcf` file. Many `samu`(1M) commands reference a specific device.

Example 1. The syntax for the `:off` command is as follows:

`:off` *eq*

For the *eq*, enter the Equipment Ordinal for the device you are trying to address.

Example 2. At certain times, `samu`(1M) prompts for a device to be entered. When you access the Robot Catalog Display (described later in this chapter), you are prompted to enter a robot Equipment Ordinal:

```
Enter robot:
```

At the prompt, enter the Equipment Ordinal, or enter a carriage return to select the previous device used.

Getting Online Help

When you start `samu`(1M), the system automatically displays the first help screen. This help screen differs depending on whether you have a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system. There are five pages of help screens, but this manual shows only the first. Subsequent help screens show `samu`(1M) commands.

For the Sun SAM-FS and Sun SAM-QFS file systems, CODE EXAMPLE 6-1 shows the initial help screen.

CODE EXAMPLE 6-1 Sun SAM-FS and Sun SAM-QFS `samu`(1M) Initial Help Screen

```
Help information         page 1/5   samu 4.0-x Thu Oct 11 13:22:30

Displays:
    a Archiver status                v Robot catalog
    c Device configuration           w Pending stage queue
    d Daemon trace controls          C Memory
    f File systems                   F Optical disk label
```

CODE EXAMPLE 6-1 Sun SAM-FS and Sun SAM-QFS samu(1M) Initial Help Screen

```
    h Help information                  I Inode
    l License information               J Preview shared memory
    m Mass storage status               L Shared memory tables
    n Staging status                    M Shared memory
    o Optical disk status               N File System Parameters
    p Removable media load requests     R SAM-Remote
    r Removable media                   S Sector data
    s Device status                     T SCSI sense data
    t Tape drive status                 U Device table
    u Staging queue

more (ctrl-f)
```

For the Sun QFS file system, CODE EXAMPLE 6-2 shows the initial help screen.

CODE EXAMPLE 6-2 Sun QFS samu(1M) Initial Help Screen

```
Help information         page 1/5   samu 4.0-x Thu Oct 11 13:58:20

Displays:
    d Daemon trace controls             m Mass storage status
    f File systems                      C Memory
    h Help information                  I Inode
    l License information               N File System Parameters

more (ctrl-f)
```

To move forward or backward from one screen to the next, enter the following key sequence:

- Press CTRL-f to page the display forward.
- Press CTRL-b to page the display backward to previous pages.

You can return to the help display at any time by pressing the h key.

Note – This manual does not describe the uppercase samu(1M) displays (A, C, F, I, J, L, M, N, R, S, T, and U) because they are designed to be used at a customer site only with the assistance of a member of the technical support staff.

Operator Displays

You can view the samu(1M) operator displays by pressing the key corresponding to each display. The lowercase keys a through w display operational information.

For displays that overflow the screen area, the word more appears on the bottom of the screen display, indicating that the display contains additional information. CODE EXAMPLE 6-3 contains the word more, indicating that more information appears on subsequent screens.

CODE EXAMPLE 6-3 samu(1M) Screen That Indicates More Text Can Be Obtained

```
xb54  54  exb8505  pt03  0  yes  2  0  on
lt55  55  dlt2000  pt02  1  yes  4  0  on  ml65
hp56  56  hpc1716  pt01  1  yes  3  0  on  hp70
hp57  57  hpc1716  pt01  1  yes  4  0  on  hp70
more
```

If samu(1M) prompts you to enter a device, enter its associated Equipment Ordinal. Equipment Ordinals for all devices are shown in the configuration display (c). To control all displays, use the control keys.

The following sections describe the operator displays. Examples are provided, and when necessary, displays are followed by a table describing the fields displayed.

(a) - Archiver Status Display

The archiver display shows the status of the archiver on a per-file-system basis.

Sample Display

CODE EXAMPLE 6-4 shows activity and statistics for a single file system.

CODE EXAMPLE 6-4 samu(1M) a Display

```
Archiver status samu 4.0.x Fri Jan 04 14:08:45

sam-archiverd:  Archiving files

sam-arfind:  samfs1 mounted at /sam1
Sleeping until Fri Jan 04 14:10:26 2002

sam-arcopy:  samfs1 arset0.2.9360 mo.opt06a
Copying file testdir0/filewh
```

Field Descriptions

To view the archiver detail display, enter `:a` *filesystem*. TABLE 6-2 shows the fields in the detail display.

TABLE 6-2 samu(1M) a Display Field Descriptions

Field	Description
`samfs1 mounted at`	Mount point.
`regular files`	Number of regular files and size.
`offline files`	Number of offline files and size.
`archdone files`	Number of `archdone` files and size. Indicates that the archiver has completed processing and can perform no further processing for `archdone` files. However, note that `archdone` files have not been archived.
`copy1`	Number of files and total size for archive copy 1.
`copy2`	Number of files and total size for archive copy 2.
`copy3`	Number of files and total size for archive copy 3.

TABLE 6-2 `samu`(1M) `a` Display Field Descriptions *(Continued)*

Field	Description
`copy4`	Number of files and total size for archive copy 4.
`Directories`	Number of directories and total size.
`sleeping until`	Indicates when archiver runs again.

(c) - Device Configuration Display

The configuration display shows your configuration's connectivity. To view the configuration display, press the c key.

Sample Display

CODE EXAMPLE 6-5 shows the device configuration display.

CODE EXAMPLE 6-5 samu(1M) c Display

```
Device configuration:      samu 4.0.x Thu Oct 11 13:10:23

ty  eq  state  device_name       fs family_set
ae  60  on     /dev/samst/c0t0u0 60 m160
at  61  on     /dev/rmt/0cbn     60 m160
at  62  on     /dev/rmt/1cbn     60 m160
at  63  on     /dev/rmt/3cbn     60 m160
at  64  on     /dev/rmt/4cbn     60 m160
hy  65  on     historian         65
```

Field Descriptions

TABLE 6-3 shows the field descriptions for this display.

TABLE 6-3 samu(1M) c Display Field Descriptions

Field	Description
`ty`	Device type.
`eq`	Equipment Ordinal of the device (unique number defined in the master configuration file).
`state`	Current operating state of the device. Valid device states are as follows: • `on`—The device is available for access. • `ro`—The device is available for read-only access. • `off`—The device is not available for access. • `down`—The device is available only for maintenance access. • `idle`—The device is not available for new connections. Operations in progress continue until completion.
`device_name`	Path to the device.
`fs`	Family set Equipment Ordinal.
`family_set`	Name of the storage family set or library to which the device belongs.

(d) - Daemon Trace Controls Display

The daemon trace controls display shows the events being traced as specified in the `defaults.conf` file. For more information on enabling trace files, see the `defaults.conf`(4) man page.

Sample Display

CODE EXAMPLE 6-6 shows trace file information. It includes information on the daemons being traced, the paths to the trace files, the events being traced, and information on the size and age of the trace files.

CODE EXAMPLE 6-6 `samu`(1M) `d` Display

```
Daemon trace controls       samu   4.0.5816 Fri Jan 18 10:42:02



sam-archiverd  /var/opt/SUNWsamfs/trace/archiver
               cust err misc files date module
               size    0    age 0

sam-catserverd /var/opt/SUNWsamfs/trace/catserver
               cust err fatal ipc misc proc queue ftp debug date module
               size    0    age 0

sam-fsd        /var/opt/SUNWsamfs/trace/fsd
               cust err fatal ipc misc proc queue ftp debug date module
               size    0    age 0

sam-ftpd       /var/opt/SUNWsamfs/trace/ftp
               cust err fatal ipc misc proc queue ftp debug date module
               size    0    age 0

sam-recycler   /var/opt/SUNWsamfs/trace/recycler
               cust err fatal ipc misc proc queue ftp debug date module
               size    0    age 0

sam-sharefsd   off



sam-stagerd    /var/opt/SUNWsamfs/trace/stager
               cust err misc proc files debug date module
               size    0    age 0
```

(f) - File Systems Display

The file systems display shows the components of your Sun QFS, Sun SAM-FS, or Sun SAM-QFS file systems. To view the file systems display, press the f key.

Sample Display

CODE EXAMPLE 6-7 shows the file systems display.

CODE EXAMPLE 6-7 samu(1M) f Display

```
File systems                       samu   4.0.x Thu Oct 11 13:12:07

ty eq state       device_name      status high low mountpoint server
ms  1    on            samfs1 m----2----d  80% 70%    /samfs1
md 11    on /dev/dsk/c2t5d0s5
md 12    on /dev/dsk/c2t6d0s5
```

Field Descriptions

TABLE 6-4 shows the field descriptions for this display.

TABLE 6-4 samu(1M) f Display Field Descriptions

Field	Description
ty	Device type.
eq	Equipment Ordinal of the device (unique number defined in the master configuration file).
state	Current operating state of the device. Valid device states are as follows: • on—The device is available for access. • ro—The device is available for read-only access. • off—The device is not available for access. • down—The device is available only for maintenance access. • idle—The device is not available for new operations. Operations in progress continue until completion.
device_name	File system name or path to the device.
status	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
high	High disk usage threshold percentage.

TABLE 6-4 `samu`(1M) `f` Display Field Descriptions *(Continued)*

Field	Description
`low`	Low disk usage threshold percentage.
`mountpoint`	Mount point of the file system.
`server`	Name of the host system upon which the file system is mounted.

(l) - License Display

The license display shows the licenses and expiration dates for Sun QFS, Sun SAM-FS, and Sun SAM-QFS software. To view the configuration display, press the l key.

Sample Display

CODE EXAMPLE 6-8 shows an example of a license display.

CODE EXAMPLE 6-8 samu(1M) l Display

```
License Information samu 4.0.x Thu Oct 11 13:13:11

hostid = xxxxxxx

License never expires
Remote sam server feature enabled
Remote sam client feature enabled
Migration toolkit feature enabled
Fast file system feature enabled
Data base feature enabled
Direct media access feature enabled
Shared SAN filesystem support enabled
Segment feature enabled
Robot type ADIC 100 Library is present and licensed
    100 at slots present and licensed
Robot type DLT Tape Library is licensed
    100 lt slots licensed
Robot type IBM 3570 Changer is licensed
    100 i7 slots licensed
Robot type IBM 3584 Library is licensed
    100 li slots licensed
```

The sample display shows license information for a Sun SAM-FS file system. The license information is derived from the license keys in the following file:

```
/etc/opt/SUNWsamfs/LICENSE.4.0
```

The following information is displayed for the system:

- Expiration information
- Host ID
- Sun QFS, Sun SAM-FS, and Sun SAM-QFS products and features enabled

- Equipment/media combinations

(m) - Mass-Storage Status Display

The mass-storage status display shows the status of mass-storage file systems and their member drives. To view the mass-storage status display, press the m key.

Sample Display

CODE EXAMPLE 6-9 shows how member drives are indented one space and appear directly below the file system to which they belong.

CODE EXAMPLE 6-9 samu(1M) m Display

```
Mass storage status                     samu 4.0.x Thu Oct 11 13:13:42

ty   eq  status       use  state  ord  capacity    free    ra  part  high  low
ms    1  m----2----d  21%  on            8.402G  6.644G  1024    16   80%  70%
 md  11               21%  on       0    4.251G  3.372G
 md  12               21%  on       1    4.151G  3.272G
```

Field Descriptions

TABLE 6-5 shows the field descriptions for this display.

TABLE 6-5 samu(1M) m Display Field Descriptions

Field	Description
ty	Device type.
eq	Equipment Ordinal of the mass-storage device.
status	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
use	Percentage of disk space in use.
state	Current operating state of the mass-storage device.
ord	Ordinal number of the disk device within the storage family set.
capacity	Number of 1024-byte blocks of usable space on the disk.
free	Number of 1024-byte blocks of disk space available.
ra	Readahead size in kilobytes.

TABLE 6-5 `samu`(1M) `m` Display Field Descriptions *(Continued)*

Field	Description
`part`	Partial stage size in kilobytes.
`high`	High disk usage threshold percentage.
`low`	Low disk usage threshold percentage.

(n) - Staging Status Display

The staging status display shows the status of the stager for all media. To view the staging status display, press the n key. To view the status for a specific device type, enter :n *media*, where *media* is the media type.

Sample Display

CODE EXAMPLE 6-10 samu(1M) n Display

```
Staging status                      samu   4.0.x Thu Oct 11 13:14:23

Log output to:

Stage request: at.000004
Copying file /samfs1/testdir3/fileia

Stage request: at.000002
Copying file /samfs1/testdir1/fileei

Stage request: at.000003
Positioning for file /samfs1/testdir2/fileaa
```

(o) - Optical Disk Status Display

The optical disk status display shows the status of all optical disk drives configured within the Sun SAM-FS or Sun SAM-QFS environment. To view the optical disk status display, enter `:o`.

Sample Display

CODE EXAMPLE 6-11 `samu`(1M) `o` Display

```
Optical disk status             samu    4.0.x Thu Oct 11 13:15:40

ty  eq  status      act  use  state  vsn
mo  35  --l---wo-r    1  29%  ready  oper2
```

Field Descriptions

TABLE 6-6 shows the field descriptions for this display.

TABLE 6-6 `samu`(1M) `o` Display Field Descriptions

Field	Description
`ty`	Device type.
`eq`	Equipment Ordinal of the optical disk.
`status`	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
`act`	Activity count.
`use`	Percentage of cartridge space used.
`state`	Current operating state of the optical disk. Valid device states are as follows: • `ready`—The device is on, and the disk is loaded in the transport; available for access. • `notrdy`—The device is on, but no disk is present in the transport. • `idle`—The device is not available for new connections. Operations in progress continue until completion. • `off`—The device is not available for access. • `down`—The device is available only for maintenance access.
`vsn`	Volume serial name assigned to the optical disk, or the keyword `nolabel` if the volume is not labeled.

(p) - Removable Media Load Requests Display

The removable media load requests display lists information on pending load requests for removable media. You can select either a specific type of media, such as DLT tape, or a family of media, such as tape. The priority display lists the priority in the preview queue rather than the user, and sorts the entries by priority.

Mount requests are displayed in three formats: both manual and robotic requests, manual requests only, or robotics requests only.

Enter only `:p` to display mount requests for all removable devices currently selected.

Enter `:p` *media_type* to display mount requests for devices of a given removable media type.

To select either the manual/robot display or the priority display, press the CTRL-k key sequence.

Sample Display 1

CODE EXAMPLE 6-12 samu(1M) p Display 1

```
Removable media mount requests all  both  samu  4.0.x Fri Feb 9 11:21:42
                                                      count: 1
count  type  pid  user  rb   flags    wait count  vsn
   0   1t    473  root  40  Wb-f---  0:00        TAPE0
```

Sample Display 2

CODE EXAMPLE 6-13 samu(1M) p Display 2

```
Removable media load requests all   priority samu   4.0.x Mon Apr 26 21:44:27
License: License never expires.                         count: 3

index type pid  priority  rb  flags   wait  count  vsn
   0  i7   0        3007  70 ---f---  0:00        TAPE5
   2  i7   0           0  70 ---f---  0:00        TAPE1
  99  i7   1383   -49607  70 W--f---  0:06        TAPE14
```

Field Descriptions

TABLE 6-7 shows the field descriptions for this display.

TABLE 6-7 `samu`(1M) `p` Display Field Descriptions

Field	Description
`index`	Index number in the preview table.
`type`	Device type code assigned to the removable media.
`pid`	UNIX process identifier. A process identifier of 1 indicates NFS access.
`user`	Name assigned to the user requesting the load.
`priority`	Priority of the request.
`rb`	Equipment Ordinal of the robot in which the requested VSN resides.
`flags`	Flags for the device. See TABLE 6-8.
`wait`	The elapsed time since the mount request was received.
`count`	The number of requests for this VSN, if it is a stage.
`vsn`	Volume serial name of the volume.

Flags

TABLE 6-8 shows the flags.

TABLE 6-8 Flags Field for `samu`(1M) `p` Display

Field	Description
`W------`	Write access requested
`-b-----`	Entry is busy
`--C----`	Clear VSN requested
`---f---`	File system requested
`-----S-`	Flip side already mounted
`------s`	Stage request flag

(r) - Removable Media Status Display

The removable media status display enables you to monitor the activity on removable media devices such as tape drives. You can monitor either a specific type of device, such as video tape, or a family of devices such as all tape devices.

To view the status for all removable media devices, enter `:r`. To view the status for a specific device, enter `:r` *dt*, where *dt* is the device.

Sample Display

CODE EXAMPLE 6-14 samu(1M) r Display

```
Removable media status: all    samu 4.0.x    Thu Oct 11 13:17:06

ty  eq  status     act  use  state  vsn
at  61  --l----o-r   1  73%  ready  000002
       0x541 blocks transferred
at  62  --l----o-r   1  70%  ready  000004
       0x7da blocks transferred
at  63  --l----o-r   1  90%  ready  000003
       0x2a0 blocks transferred
at  64  --l------r   0  54%  ready  000001
       idle
```

Field Descriptions

TABLE 6-9 shows the field descriptions for this display.

TABLE 6-9 samu(1M) r Display Field Descriptions

Field	Description
`ty`	Device type.
`eq`	Equipment Ordinal of the drive.
`status`	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
`act`	Activity count.

TABLE 6-9 samu(1M) r Display Field Descriptions

Field	Description
use	Percentage of cartridge space used (optical disk only).
state	Current operating state of the removable media. Valid device states are as follows: • ready—The device is on, and the disk or tape is loaded in the transport; available for access. • notrdy—The device is on, but no disk or tape is present in the transport. • idle—The device is not available for new connections. Operations in progress continue until completion. • off—The device is not available for access. • down—The device is available only for maintenance access.
vsn	Volume serial name assigned to the volume, or the keyword nolabel if the volume is not labeled. Blank if no volume is present in the transport, or device is off.

(s) - Device Status Display

The device status display shows the status for all devices configured within the Sun SAM-FS or Sun SAM-QFS environment. To view the device status summary display, enter `:s`.

Sample Display

CODE EXAMPLE 6-15 samu(1M) s Display

```
Device status               samu  4.0.x       Thu Oct 11 13:18:18

ty  eq  state  device_name       fs  status       pos
ae  60  on     /dev/samst/c0t0u0  60  m--------r
     move complete
at  61  on     /dev/rmt/0cbn      60  --l----o-r
     0x70d blocks transferred
at  62  on     /dev/rmt/1cbn      60  --l----o-r
     0x986 blocks transferred
at  63  on     /dev/rmt/3cbn      60  --l----o-r
     0x46d blocks transferred
at  64  on     /dev/rmt/4cbn      60  --l------r
     idle
hy  65  on     historian          65  ----------
```

Field Descriptions

TABLE 6-10 shows the field descriptions for this display.

TABLE 6-10 samu(1M) s Display Field Descriptions

Field	Description
`ty`	Device type.
`eq`	Equipment ordinal of the device.
`state`	Current operating state of the device.
`device_name`	Path to the device. For file system devices, this is the file system name.
`fs`	Equipment Ordinal of the family, set to which the device belongs.
`status`	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
`pos`	Device position.

(t) - Tape Drive Status Display

The tape drive status display shows the status of all tape drives configured within the Sun SAM-FS or Sun SAM-QFS environment. To view the tape status display, press the t key.

Sample Display

CODE EXAMPLE 6-16 samu(1M) t Display

```
Tape drive status              samu   4.0.x Thu Oct 11 13:18:48

ty  eq  status      act  use  state  vsn
at  61  --l----o-r    1  73%  ready  000002
       0x7b7 blocks transferred
at  62  --l----o-r    1  70%  ready  000004
       0xa35 blocks transferred
at  63  --l----o-r    1  90%  ready  000003
       0x518 blocks transferred
at  64  --l----o-r    1  54%  ready  000001
       0x20 blocks transferred
```

Field Descriptions

TABLE 6-11 shows the field descriptions for this display.

TABLE 6-11 samu(1M) t Display Field Descriptions

Field	Description
ty	Device type.
eq	Equipment Ordinal of the drive.
status	Device status. For a description of status codes, see “Operator Display Status Codes” on page 184.
act	Activity count.

TABLE 6-11 samu(1M) t Display Field Descriptions

Field	Description
use	Percentage of cartridge space used (optical disk only).
state	Current operating state of the removable media. Valid device states are as follows: • ready—The device is on and the disk or tape is loaded in the transport; available for access. • notrdy—The device is on but no disk or tape is present in the transport. • idle—The device is not available for new connections. Operations in progress continue until completion. • off—The device is not available for access. • down—The device is available only for maintenance access.
vsn	Volume serial name assigned to the volume, or the keyword nolabel if volume is not labeled. Blank if no volume is present in the transport, or device is off.

(u) - Staging Queue Display

The samu utility's u display lists all files in the staging queue. To select this display, type u. Press the CTRL-k key sequence to list the file path name on the second line of each entry.

Sample Display

CODE EXAMPLE 6-17 samu(1M) u Display

```
Staging queue by media type: all samu 4.0.x Thu Oct 11 13:19:34
volumes 2 files 827

ty    length  fseq   ino  position  offset  vsn

at    1.674M     1  2513     389d4  7e70b   000004
at    1.875M     1  2640     389d4  7f470   000004
at    1.643M     1  1536     389d4  80372   000004
at    1.063M     1   248     389d4  81099   000004
at  562.037k     1   595     389d4  8191b   000004
at    1.000M     1   142     389d4  81d81   000004
at    1.264M     1   442     389d4  82582   000004
at  599.014k     1  2237     389d4  82fa0   000004
at  816.685k     1  2435     389d4  83450   000004
at    1.429M     1  2701     389d4  83ab3   000004
at    1.752M     1   439     389d4  84623   000004
at    1.089M     1   565     389d4  85428   000004
at  975.326k     1   121     389d4  85ce1   000004
at    1.014M     1    28     389d4  86481   000004
at  683.581k     1   419     389d4  86c9f   000004
at    1.562M     1  1608     389d4  871f8   000004
   more
```

Field Descriptions

TABLE 6-12 shows the field descriptions for this display.

TABLE 6-12 samu(1M) u Display Field Descriptions

Field	Description
ty	Device type.
length	File length.
fseq	File system equipment number.
ino	The inode number.
position	The position (in decimal format) of the archive file on the specific medium.
offset	Offset of the archive file on the specific medium.
vsn	Volume serial name of the volume.

(v) - Robot Catalog Display

The robot catalog display shows the location and VSN of all disks or tapes currently cataloged in the robot. To view the library VSN catalog display, press the v key. If the operator utility prompts for a robot name, enter either the device name or an Equipment Ordinal. A null entry displays the last library shown. For a list of all device names and Equipment Ordinals, view the configuration display by pressing the c key.

The CTRL-k key sequence changes the sorting key for this display. The CTRL-i key sequence changes to a two-line display that shows the times and barcodes. Pressing the CTRL-i key sequence a second time displays volume reservation information on the second line.

Sample Display

CODE EXAMPLE 6-18 samu(1M) v Display

```
Robot VSN catalog by slot : eq 60 samu 4.0.x Thu Oct 11 13:20:04
                                                    count 32
slot       access time  count  use  flags         ty  vsn

   0  none               70    0%  -il-oCb-----  at  CLN005
   1  2001/10/11 08:31   10   90%  -il---b-----  at  000003
   2  2001/10/11 13:07   17   73%  -il---b-----  at  000002
   3  2001/10/11 12:48   16   70%  -il---b-----  at  000004
   4  2001/10/11 12:55   30   54%  -il---b-----  at  000001
   5  none                0    0%  -il-o-b-----  at  000005
   6  none                0    0%  -il-o-b-----  at  000044
   7
  13  2001/10/11 13:05   61    0%  -il-o-b-----  at  000033
```

Field Descriptions

TABLE 6-13 shows the field descriptions for this display.

TABLE 6-13 samu(1M) v Display Field Descriptions

Field	Description
Robot VSN catalog	Name of the specified robot and time the display refreshed.
count	Number of slots in library.
slot	Slot number within the specified library.

TABLE 6-13 `samu`(1M) `v` Display Field Descriptions *(Continued)*

Field	Description
`access time`	Time the optical disk was last accessed.
`count`	Number of accesses to this volume since the last audit operation.
`use`	Percentage of space used for the volume.
`flags`	Flags for the device. See TABLE 6-14 for information on the flags.
`ty`	Device type.
`vsn`	Volume serial name of the volume.

Flags

In some cases, more than one flag can occur in a field, and one flag overrides the other. TABLE 6-14 shows the flags from the `flags` field from TABLE 6-13.

TABLE 6-14 Flags Field for `samu`(1M) `v` Display

Flags	Description
`A-----------`	Volume needs audit.
`-i----------`	Slot in use.
`--l---------`	Labeled. Overrides `N`.
`--N---------`	Unlabeled. This volume is foreign to the Sun SAM-FS or Sun SAM-QFS environment.
`---E--------`	Media error. Set when the Sun SAM-FS or Sun SAM-QFS software detects a write error on a cartridge.
`----o-------`	Slot occupied.
`-----C------`	Volume is a cleaning tape. Overrides `p`.
`-----p------`	Priority VSN.
`------b-----`	Barcode detected.
`-------W----`	Write protect. Set when the physical write protection mechanism is enabled on a cartridge.
`--------R---`	Read only.
`---------c--`	Recycle.
`----------d-`	Duplicate VSN. Overrides `U`.
`----------U-`	Volume unavailable.
`-----------X`	Export slot.

(w) - Pending Stage Queue

The pending stage queue display shows queued stage requests for which the volumes have not yet been loaded. Press the CTRL-k key sequence to list the path name on the second line of each entry.

Sample Display

CODE EXAMPLE 6-19 samu(1M) w Display

```
Pending stage queue by media type: all     samu     4.0.x Thu Oct 11 13:20:27
                                                      volumes 1 files 13

ty     length  fseq  ino  position  offset  vsn

at     1.383M     1   42     3a786    271b  000002
at     1.479M     1   56     3a786    5139  000002
at  1018.406k     1   60     3a786    6550  000002
at     1.000M     1   65     3a786    7475  000002
at     1.528M     1   80     3a786    99be  000002
at     1.763M     1   92     3a786    ce57  000002
at     1.749M     1  123     3a786   11ece  000002
at   556.559k     1  157     3a786   1532f  000002
at   658.970k     1  186     3a786   17705  000002
at   863.380k     1  251     3a786   1dd58  000002
at     1.268M     1  281     3a786   1f2b7  000002
at     1.797M     1  324     3a786   23dfa  000002
at     1.144M     1  401     3a786   2bb6d  000002
```

Field Descriptions

TABLE 6-15 shows the field descriptions for this display.

TABLE 6-15 samu(1M) w Display Field Descriptions

Field	Description
ty	Device type.
length	File length.
fseq	File system Equipment Ordinal.
ino	The inode number.

TABLE 6-15 samu(1M) w Display Field Descriptions *(Continued)*

Field	Description
position	The position (in decimal format) of the archive file on the specific medium.
offset	Offset of the archive file on the specific medium.
vsn	Volume serial name of the volume.

Operator Display Status Codes

The operator displays have different status codes for the removable media device displays versus the file system displays. The following sections describe these displays.

Removable Media Device Display Status Codes

The `c`, `o`, `r`, `s`, and `t` operator displays show status codes for removable media devices. Status codes are displayed in a 10-position format, reading from left (position 1) to right (position 10).

The status codes in this subsection do not apply to the `samu`(1M) `f`, `m`, and `v` displays. For information on the status codes for the `f` and `m` displays, see "File System Display Status Codes" on page 185. For information on the status codes for the `v` display, see "(v) - Robot Catalog Display" on page 180.

TABLE 6-16 defines the valid status codes for each position.

TABLE 6-16 Removable Media Device Display Status Codes

Status Bit	Meaning for a Device
`s---------`	Media is being scanned.
`M---------`	Maintenance mode.
`-E--------`	Device received an unrecoverable error in scanning.
`-a--------`	Device is in audit mode.
`--l-------`	Media has a label.
`--N-------`	Foreign media.
`---I------`	Waiting for device to idle.
`---A------`	Needs operator attention.
`----C-----`	Needs cleaning.
`----U-----`	Unload has been requested.
`-----R----`	Device is reserved.
`------w---`	A process is writing on the media.
`-------o--`	Device is open.
`--------P-`	Device is positioning (tape only).
`--------F-`	For robots, all storage slots occupied. For tape and magneto optical drives, media is full.

TABLE 6-16 Removable Media Device Display Status Codes *(Continued)*

Status Bit	Meaning for a Device
`---------R`	Device is ready and the media is read-only.
`---------r`	Device is spun up and ready.
`---------p`	Device is present.
`---------W`	Device is write protected.

File System Display Status Codes

The `f` and `m` operator displays show status codes for file systems. Status codes are displayed in an 11-position format, reading from left (position 1) to right (position 11).

The status codes in this section do not apply to the `samu`(1M) `c`, `o`, `r`, `s`, `t`, or `v` displays. For information on the status codes for the `c`, `o`, `r`, `s`, and `t` displays, see "Removable Media Device Display Status Codes" on page 184. For information on the status codes for the `v` display, see "(v) - Robot Catalog Display" on page 180.

TABLE 6-17 defines the valid status codes for each position.

TABLE 6-17 File System Display Status Codes

Status Bit	Meaning for a File System
`m----------`	File system is currently mounted.
`M----------`	File system is being mounted.
`-u---------`	File system is being unmounted.
`--A--------`	File system data is being archived.
`---R-------`	File system data is being released.
`----S------`	File system data is being staged.
`-----1-----`	Sun SAM-FS or Sun SAM-QFS file system version 1.
`-----2-----`	Sun SAM-FS or Sun SAM-QFS file system version 2.
`------C----`	Sun QFS shared file system.
`-------W---`	Single writer.
`--------R--`	Multireader.
`---------r-`	`mr` devices.
`----------d`	`md` devices.

Operator Display Device States

The c, m, o, r, s, and t operator displays show device state codes. These codes represent the current access state for the device.

You can use samu(1M) to change the state of a device. The following examples show a typical progression to change a drive's state from down to on and from on to down:

- **Example 1**. The following progression can be used to change a device state from down to on:

  down -> off -> [unavail] -> on

  In this progression, the brackets indicate that it is not necessary to pass through the unavail state.

- **Example 2**. The following progression can be used to change a device state from on to down:

  on -> [idle] -> [unavail] -> off -> down

  In this progression, the brackets indicate that it is not necessary to pass through the idle or unavail states.

TABLE 6-18 defines the valid state codes.

TABLE 6-18 Operator Display Device States

Device State	Description
on	The device is available for access. For certain displays, this state may be superseded by the states ready or notrdy.
ro	The device is available for read-only access. Like on, this state can be superseded for certain displays by ready or notrdy.
off	The device is not available for access. For tape and optical disk drives, possible reasons for the device to be in the off state include the following: • Cleaning was requested, but no cleaning cartridge was found in the automated library. • The cleaning cartridge cannot be loaded or unloaded from the drive. • Initialization found the drive status to be full, and attempts to clear the drive failed. • The system was unable to clear a cartridge from a drive. • Opening the drive for I/O failed during spin-up. • An error other than NOT READY was received when spinning the drive down for unloading. • Opening the standard tape driver on the drive failed during spin up.
down	The device is available for maintenance access only.

TABLE 6-18 Operator Display Device States *(Continued)*

Device State	Description
`idle`	The device is not available for new connections. Operations in progress continue until completion.
`ready`	The device is on and the disk or tape loaded in the transport is available for access.
`notrdy`	The device is on, but no disk or tape is present in the transport.
`unavail`	The device is unavailable for access and cannot be used for automatic Sun SAM-FS or Sun SAM-QFS operations. You can continue to use the load and unload commands for placing and removing media from the device while it is in the unavail state.

Operator Commands

This section describes the following types of operator commands:

- "Archiver Commands" on page 188
- "Device Commands" on page 189
- "Display Control Commands" on page 190
- "File System Commands" on page 191
- "Robot Commands" on page 193
- "Miscellaneous Commands" on page 194

Note – If you want to enter any operator commands from the Sun Solaris operating environment (OE) command line, you must use them as arguments to the `samcmd`(1M) command. For more information on the `samcmd`(1M) command, see the `samcmd`(1M) man page.

Each `samu`(1M) command is prefaced with a colon (:) when it is entered to designate that a command line command is being entered and not a series of hot keys.

Archiver Commands

TABLE 6-19 shows the archiver commands and their actions.

TABLE 6-19 Archiver Command Actions

Command	Action
`aridle`	Stops all archiving at the next convenient point. For example, at the end of the current `tar`(1) file for `sam-arcopy` operations. This command can be used, for example, to stop all archiving activity for all file systems prior to unmounting the file systems.
`arrestart`	Interrupts the archiver and restarts the archiver. This action occurs regardless of the state of the archiver. Therefore, `arrestart` should be used with caution. Some copy operations to archive media might not complete and must be repeated. This wastes space on the media.
`arrun`	Causes the archiver to begin archiving. This command overrides any existing global `wait` command in the `archiver.cmd` file.
`arstop`	Stops all archiving immediately.

The formats for the archiver commands are as follows:

```
:aridle [ dk | rm | fs.fsname ]
:arrestart
:arrun [ dk | rm | fs.fsname ]
:arstop [ dk | rm | fs.fsname ]
```

The arguments to these commands are optional. If no arguments are specified, all file systems are affected. If arguments are specified, the command takes action based on the type of archive file specified (`dk` or `rm`) and the file system specified. TABLE 6-20 shows the archiver command arguments.

TABLE 6-20 Archiver Command Arguments

Argument	Description
`dk`	Specifies that this command pertains to disk archive files.
`rm`	Specifies that this command pertains to removable media files.
`fs.`*fsname*	Specifies that this command pertains to a specific file system. Enter a file system name for *fsname*.

Device Commands

TABLE 6-21 shows the device commands and their actions.

TABLE 6-21 Device Command Actions

Command	**Action**
`devlog`	Sets device-logging options.
`down`	Terminates operation on device *eq*.
`idle`	Restricts access to device *eq* by preventing new connections to the device. Existing operations continue until completion.
`off`	Logically turns off device *eq*.
`on`	Logically turns on device *eq*.
`unavail`	Selects device *eq* and makes it unavailable for use with the Sun SAM-FS or Sun SAM-QFS file system.
`unload`	Unloads the mounted media for the specified removable media device *eq*. For magazine devices, the `unload` command unloads the mounted cartridge and ejects the magazine.

The formats for the device control commands are as follows:

```
:devlog eq [ option ...]
:down eq
:idle eq
:off eq
:on eq
:unavail eq
:unload eq
```

TABLE 6-22 shows the device command arguments.

TABLE 6-22 Device Command Arguments

Argument	**Description**
eq	The Equipment Ordinal of a device in the `mcf` file.
option	Zero or more event types. Possible event types are as follows: `all`, `date`, `default`, `detail`, `err`, `event`, `label`, `mig`, `module`, `msg`, `none`, `retry`, `stage`, `syserr`, and `time`. For information on these options, see the `defaults.conf`(4) man page.

Display Control Commands

TABLE 6-23 shows the display control commands and their actions.

TABLE 6-23 Display Control Command Actions

Command	**Action**
`:a` [*filesystem*]	Displays the archiver status.
`:n` [*media*]	Selects the media type for the removable media I/O activity display.
`:p` [*media*]	Selects the media type for the mount requests display.
`:q`	Causes the samu operator utility to exit.
`:r` [*media*]	Selects the device type for the removable media status display.
`:refresh` *i*	Sets the time interval for refreshing the display window and enables display refreshing. The CTRL-r key sequence toggles display refreshing on and off.
`:u` [*media*]	Displays the stage queue. This pertains to currently mounted volumes.
`:v` [*eq*]	Selects the library VSN catalog for display. To view the VSNs in the historian catalog, enter the keyword `historian` in place of *eq*.
`:w` [*media*]	Displays the prestage queue. This pertains to volumes that are not yet mounted.

The formats for the display control commands are as follows:

```
:a [ filesystem ]
:n [ media ]
:p [ media ]
:q
:r [ media ]
:refresh i
:u [ media ]
:v [ eq ]
:w [ media ]
```

The brackets around the arguments to these commands show that in many cases, the arguments are optional. The arguments to many of these commands narrow the `samu`(1M) display output to a specific file system, media type, or Equipment Ordinal. If no arguments are specified, the command displays information for all file systems, media types, and Equipment Ordinals that are currently selected or configured.

TABLE 6-24 shows the display control command arguments.

TABLE 6-24 Display Control Command Arguments

Argument	Description
filesystem	Specifies the name of a Sun SAM-FS or Sun SAM-QFS file system. If the *filesystem* argument is specified, the archiver status display shows the number of regular files; the number of offline files; the number of archived files; the number of archive copies and directories; file systems; mount points; inode activity; and interval. If the *filesystem* argument is not specified, the archiver status display shows the name of the file system and mount point, scans for inode activity, and lists the next time the archiver will scan the file system.
media	Specifies a media type. For a list of supported media types, see the `mcf`(4) man page. The keyword all can also be specified to represent all media types or removable media devices.
eq	The Equipment Ordinal of a device in the `mcf` file.
i	The time interval in seconds.

File System Commands

The `:meta_timeo` *eq interval* Command

The `metatimeo` command sets the Sun QFS shared file system metadata cache time out value.

For *eq*, specify the Equipment Ordinal of the file system.

For *interval*, specify an interval in seconds. The default *interval* is 15. After this interval expires, the client host systems obtain a new copy of the metadata information from the metadata server host.

The `:notrace` *eq* Command

The `notrace` command disables tracing.

For *eq*, specify the Equipment Ordinal of the file system.

The `:partial` *eq size* Command

The `partial` command sets the number of kilobytes to leave online after release of the file.

For *eq*, specify the Equipment Ordinal for the file system.

For *size*, specify the number of kilobytes to leave online. The default *size* is 16.

The `:readahead` *eq contig* Command

The `readahead` command specifies the maximum number of bytes that can be read ahead by the file system.

For *eq*, specify the Equipment Ordinal for the file system.

For *contig*, specify units of 1-kilobyte blocks. This must be an integer such that 1 < *contig* < 8192. The *contig* specified is truncated to a multiple of 8 kilobytes. The default *contig* is `8` (131072 bytes).

For example, the following command sets the maximum contiguous block size to 262,144 bytes for the file system defined as Equipment Ordinal `3`:

```
:readahead 3 256
```

This value can also be configured in the `samfs.cmd` file by specifying the `readahead` directive. For more information, see the `samfs.cmd`(4) man page.

The `:thresh` *eq high low* Command

The `thresh` command sets the high and low thresholds for a file system to control file archiving.

For *eq*, specify the Equipment Ordinal of the storage family set.

For *high*, specify the high threshold.

For *low*, specify the low threshold.

For example, the following command sets the high threshold to 50 percent and the low threshold to 40 percent for the storage family set whose file system Equipment Ordinal is 10:

```
:thresh 10 50 40
```

The `:trace` *eq* Command

The `trace` command enables tracing for a file system.

For *eq*, specify the Equipment Ordinal of a file system.

The `:writebehind` *eq contig* Command

The `writebehind` command specifies the maximum number of bytes that can be written behind by a file system.

For *eq*, specify the Equipment Ordinal for a file system.

For *contig*, specify units of 1-kilobyte blocks. This must be an integer such that 1 < *contig* < 8192. The default *contig* is 8 (131072 bytes).

For example, the following command sets the maximum contiguous block size to 262,144 bytes for the file system defined as Equipment Ordinal `50`:

```
:writebehind 50 256
```

This value can also be configured in the `samfs.cmd` file by specifying the `writebehind` directive. For more information, see the `samfs.cmd`(4) man page.

Robot Commands

The `:audit [ -e ]` *eq* [`:`*slot* [`:`*side*]] Commands

The `audit` command causes the specified robotic device to mount each volume, read the VSN, and rebuild the library catalog.

For *eq*, specify the Equipment Ordinal of a robotic device.

The `:export` *eq*`:`*slot* and `:export` *mt*`.`*vsn* Commands

The `export` command causes the specified robotic device to export a volume to the mail slot. The volume is identified by its slot position within the robot.

- If exporting by Equipment Ordinal and slot number, the specified robotic device loads the volume into a drive. For *eq*, specify the Equipment Ordinal or device name. For *slot*, specify the slot number containing the volume you want to load.

- If exporting by logical identifier, the specified robotic device to mounts a labeled volume in to a drive. For *mt*, specify the media type; for information on valid media types, see the `mcf`(4) man page. For *vsn*, specify the volume to mount.

The `:import` *eq* Command

The `import` command causes the specified robotic device to allow you to add a cartridge. For *eq*, specify the Equipment Ordinal of the robotic device.

The `:load` *eq*`:`*slot* [`:`*side*] and `:load` *mt*`.`*vsn* Commands

The `load` command allows you to load by either a physical or a logical identifier, as follows:

- If loading by Equipment Ordinal and slot number, the specified robotic device loads the volume into a drive. For *eq*, specify the Equipment Ordinal or device name. For *slot*, specify the slot number containing the volume you want to load.
- If loading by logical identifier, the specified robotic device to load mounts a labeled volume in to a drive. For *mt*, specify the media type; for information on valid media types, see the `mcf`(4) man page. For *vsn*, specify the volume to mount.

Miscellaneous Commands

The `:clear` *vsn* [*index*] Command

The `clear` command clears the specified VSN from the removable media mount requests display (see "(p) - Removable Media Load Requests Display" on page 171). Any process waiting for the VSN mount is aborted. If *index* is specified, *index* is the decimal ordinal of the VSN in the removable media display.

The `:dtrace` Commands

The `dtrace` commands are as follows:

- `:dtrace` *daemon_name* `on`
- `:dtrace` *daemon_name* `off`
- `:dtrace` *daemon_name*`.`*variable value*

The `dtrace` commands specify various tracing options. TABLE 6-25 shows the tracing control command arguments.

TABLE 6-25 Tracing Command Arguments

Argument	Description
daemon_name	Specify the keyword `all` or a process name. If the keyword `all` is specified, the tracing command affects all daemons. If one of the following process names is specified, the tracing command affects that process only: `sam-archiverd`, `sam-catserverd`, `sam-fsd`, `sam-ftpd`, `sam-recycler`, `sam-sharefsd`, and `sam-stagerd`. One of the keywords `on` or `off` can be specified after a process name. If `on` or `off` are specified, tracing is turned off or on for all processes specified.
variable value	Many different *variable* and *value* arguments can be specified. The `defaults.conf`(4) man page contains comprehensive information on these arguments. Specify one of the following *variable* and *value* combinations: • `file` *value*. For *value*, specify the name of a file to which trace files can be written. This can be a full path name. • `options` *value*. For *value*, specify a space-separated list of trace options. • `age` *value*. For *age*, specify the trace file rotation age. • `size` *value*. For *value*, specify the size of the trace file at which rotation will begin.

The `:mount` *mntpt* Command

The `mount` command selects a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system.

The `:open` *eq* Command

The `open` command enables access to the specified disk device. This command must be issued before you can use the `read` command, disk sector display (`S`), or file label display (`F`). *eq* is the Equipment Ordinal.

The `:read` *addr* Command

The `read` command reads the specified sector from the currently opened disk device. You must open the device before it can be read. For *addr*, specify the hexadecimal sector address.

The `:snap` [*filename*] Command

The `snap` command sends a snapshot of a display window to *filename*, which is the name of a file to receive the display information.

To aid in problem reporting, you can take a snapshot of all the `samu`(1M) utility's displays. Each new snapshot is appended to the `snapshots` file. The default file is `snapshots` in the current working directory. The file can be printed, examined using `vi`(1), or faxed to Sun Microsystems customer support staff.

The `:!` *shell_command* Command

The `!` command allows you to run a shell command without leaving the `samu` operator utility.

CHAPTER 7

File System Quotas

File system quotas control the amount of online disk space that can be consumed by a specific user, a group of users, or an admin set in a file system. An *admin set* is a site-determined group of users.

Quotas can help control the size of a file system by limiting the amount of disk space and the number of inodes for each user. Quotas can be especially useful on file systems that contain user home directories. After quotas are enabled, you can monitor usage and adjust the quotas as needs change.

This chapter describes the following topics:

- Overview
- Enabling quotas
- Checking quotas
- Changing and removing quotas

Overview

File system quotas can be set on a user, group, or a site-defined admin set basis. You, the system administrator, can set limits as to the number of files and the number of blocks.

A file system provides a user with blocks for data and inodes for files. Each file uses one inode, and file data is stored in a disk allocation unit (DAU). DAU sizes are determined at the time the file system is created. Quotas account for disk usage in multiples of 512 bytes.

The following sections provide background information on using quotas. These sections are as follows:

- “Quotas and Archive Media” on page 198

- “Disk Blocks and File Limits” on page 198
- “Soft Limits and Hard Limits” on page 199
- “Types of Quotas, Quota Files, and Quota Records” on page 200

Quotas and Archive Media

This chapter describes how to use and set file system quotas in Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems. Be aware that quotas are for disk file systems only. Quotas are not applicable to archive media. As such, there are limitations to the quota facility when used in Sun SAM-FS or Sun SAM-QFS configurations.

Example 1. The `stage`(1) command brings data online from archive media. It is possible for a user quota to be exceeded when the `stage`(1) command is invoked at the system level in the following way:

```
# stage -r *
```

Example 2. A user quota is observed when a user issues the `stage`(1) command with the `-w` option in the following way:

```
# stage -w *
```

In example 2, files are staged until the user’s quota is met. After that time, no more files are staged.

Disk Blocks and File Limits

It is possible for a user to exceed his or her inode quota, without using any blocks, by creating all empty files. It is also possible for a user to use only one inode and still exceed his or her block quota by creating a file that is large enough to consume all data blocks in the user’s quota.

File system quotas are expressed in terms of the number of 512-byte blocks that a user can allocate. However, disk space is allocated to user files in terms of DAUs. The DAU setting is specified by the `-a` *allocation_unit* option to the `sammkfs`(1M) command. It is preferable to set a block quota to a multiple of the file system DAU. If this is not the case, users will be able to allocate only up to the block count, rounded down to the nearest DAU.

TABLE 7-1 shows the terms that are used extensively in this chapter's quota documentation.

TABLE 7-1 Quota Terminology

Term	**Definition**
grace period	The amount of time that can elapse during which a user is allowed to create files and/or allocate storage after users reach their soft limit.
soft limit	For disk quotas, a threshold limit on file system resources (blocks and inodes) that a user can temporarily exceed. Exceeding the soft limit starts a timer. When a user exceeds the soft limit for the specified time (the grace period), no further system resources can be allocated until the user reduces file system use below the soft limit.
hard limit	For disk quotas, a maximum limit on file system resources (blocks and inodes) that users cannot exceed.
quota	The amount of system resources that a user is allowed to consume.
timer	A facility for tracking the time elapsed after a user reaches a soft limit. When it reaches the grace period, a hard limit is imposed on the user.

Soft Limits and Hard Limits

You can set both soft and hard limits. A hard limit specifies a fixed amount of system resources available for use, and the system never allows a user to exceed this limit. A soft limit specifies a level of system resource use that can be exceeded temporarily. The soft limits are always set lower than the hard limits. If a new user attempts to allocate resources beyond his or her hard limit, the operation is aborted. In this case, the operation (typically a `write`(2) or `creat`(2)) fails and generates an `EDQUOT` error.

After a user exceeds a soft limit, a timer starts, and the user enters a grace period. While the timer is ticking, the user is allowed to operate above the soft limit but cannot exceed the hard limit. Once the user goes below the soft limit, the timer gets reset. If the grace period ends and the timer stops without the user having gone below the soft limit, the soft limit is then enforced as a hard limit.

For example, assume that a user has a soft limit of 10,000 blocks and a hard limit of 12,000 blocks. If the user's block usage exceeds 10,000 blocks and the timer exceeds the grace period, this user is no longer able to allocate more disk blocks on that file system until his or her usage drops below the 10,000-block soft limit.

You, the administrator, can use the `samquota`(1M) command to see the timer value. The `squota`(1) command is a user version of the `samquota`(1M) command. The `squota`(1) user command contains options that a user can specify to obtain information on quotas that pertain to them.

Types of Quotas, Quota Files, and Quota Records

Quotas can be set according to user ID, group ID, or an administrator's site-specific grouping. This site-specific grouping is called an *admin set ID*. An admin set ID could be used, for example, to identify a collection of users working on a project for which file system quotas are imposed.

Quotas are enabled when the following two events have occurred:

- The file system is mounted with the `–o quota` option on the `mount`(1M) command or with the `quota` mount option in the `/etc/vfstab` or `samfs.cmd` file.
- The system detects the presence of one or more quota files in the file system's root directory.

Each quota file contains a sequence of records. Record zero is the record for the system administrator's quotas. The system administrator's resource usage is accumulated in record zero. System administrator quotas are never enforced, but the system administrator's record can be edited and used as a template for subsequent records in the quota file. Record one is the record in the quota file for user one, group one, or admin set ID one, depending on the type of quota file. Record one and all subsequent records can be edited in order to set different quotas for different users. TABLE 7-2 shows the quota file names and the quotas they enable in `/root`.

TABLE 7-2 Quota File Names

Quota File Name in `/root` Directory	Quota Type
`.quota_u`	UID (system user ID)
`.quota_g`	GID (system group ID)
`.quota_a`	AID (system admin set ID)

Default quota limits for users can be set by editing record zero in the quota file and allowing the values in record zero to be used as the initial quota settings for all other users. By default, if user quota limits have not been set specifically, the values in record zero are used.

Enabling Quotas

Quotas are enabled through a process that includes creating quota files and using various quota commands. This procedure is described in more detail later in this section, but generally, enabling quotas involves editing system files, creating quota files, and entering various quota commands.

TABLE 7-3 shows the commands used when manipulating quotas.

TABLE 7-3 Quota Commands

Command	Description
`squota`(1)	Displays quota statistics for a user. This is a subset of the `samquota`(1M) command for administrators.
`samchaid`(1M)	Changes file admin set ID attributes.
`samquota`(1M)	Displays quota statistics for a user, group, or admin set. This command also enables an administrator to edit quota records.
`samquotastat`(1M)	Reports which, if any, quotas are active on a file system.

When it is run, the `samfsck`(1M) command checks the file system to make sure that usage values recorded in the quota files match the actual file system usage totals. If they do not match, the `samfsck`(1M) command issues notices, and it updates all existing, incorrect quota records if a file system repair is performed.

The following sections provide more details on how to configure a file system to use quotas and how to enable quotas.

Guidelines for Setting Up Quotas

Before you enable quotas, you should determine how much disk space and how many inodes to allocate to each user. If you want to be sure that the total file system space is never exceeded, you can divide the total size of the file system between the number of users. For example, if three users share a 100-megabyte slice and have equal disk space needs, you could allocate 33 megabytes to each. In environments in which not all users are likely to push their limits, you might want to set individual quotas so that they add up to more than the total size of the file system. For example, if three users share a 100-megabyte slice, you could allocate 40 megabytes to each.

The two quota commands are as follows:

- The `squota`(1) command is for end users. It enables them to retrieve quota information for themselves on a user, group, or admin set basis.

- The samquota(1M) command is for system administrators. It enables you to retrieve quota information or to set quotas. The –U, -G, and –A options on the samquota(1M) command determine whether the command is being used for a user, a group, or an admin set. For example:

```
# samquota –U janet /mount_point   #Prints a user quota
# samquota –G pubs /mount_point    #Prints a group quota
# samquota –A 99 /mount_point      #Prints an admin set quota
```

To Configure a New File System to Use Quotas

The following procedure shows how to configure a new file system to use quotas. This procedure applies if you are creating a new file system at this time and no files currently reside in the file system.

To configure an existing file system to use quotas, see "To Configure an Existing File System to Use Quotas" on page 205.

1. **Use the su(1) command to become superuser.**
2. **Create the file system.**

   To create the file system, either follow the steps outlined in the *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide* or use the examples in "Configuration Examples" on page 52 to guide you through creating the mcf file, creating the mount point, initializing the file system, and so on.
3. **Use the mount(1M) command to mount the file system.**

   Mount the file system using the mount(1M) command, as follows:

```
# mount /qfs1
```

4. **Use the dd(1M) command to create the quota file(s).**

   The arguments to this command differ depending on the type of quota you are creating, as follows:

   To create admin set quotas, use the following command:

```
# dd if=/dev/zero of=/qfs1/.quota_a bs=4096 count=1
```

To create group quotas, use the following command:

```
# dd if=/dev/zero of=/qfs1/.quota_g bs=4096 count=1
```

To create user quotas, use the following command:

```
# dd if=/dev/zero of=/qfs1/.quota_u bs=4096 count=1
```

For more information on the `dd`(1M) command, see the `dd`(1M) man page.

5. **Use the `umount`(1M) command to unmount the file system.**

   Unmount the file system in which the quota files have been created using the `umount`(1M) command. For example:

   ```
   # umount /qfs1
   ```

   The file system needs to be unmounted so it can be remounted and have its quota files read at mount time. For more information on the `umount`(1M) command, see the `umount`(1M) man page.

6. **Edit the `/etc/vfstab` or `samfs.cmd` file. (Optional)**

   Quotas are enabled at mount time by one of the following:

   - By using the `–o quota` option to the `mount`(1M) command
   - By editing the `/etc/vfstab` file or the `samfs.cmd` file and adding the quota mount option. For more information on the `samfs.cmd` file, see the `samfs.cmd`(4) man page.

   If you want to mount the file system with quotas enabled every time you issue the `mount`(1M) command, consider performing this step. It eliminates the need to include the `–o quota` mount option on the `mount`(1M) command every time the file system is mounted.

   For example, you can edit the `/etc/vfstab` file and add `quota` to the mount options field for each file system for which quotas are to be enabled. The following file has been edited to be compatible with quotas:

   ```
   # /etc/vfstab
   # device    device   mount  FS    fsck  mount    mount
   # to mount  to fsck  point  type  pass  at boot  options
   # --------  -------  -----  ----  ----  -------  -------
   qfs1        -        /qfs1  samfs  -    yes      stripe=0,quota
   ```

7. **Use the `samfsck`(1M) command to perform a file system check.**

   Run the `samfsck`(1M) command on the file system. For example, the following command performs a file system check. The `-F` option reinitializes the `samfs.cmd` file.

```
# samfsck -F qfs1
```

8. **Use the `mount`(1M) command to remount the file system.**

   Mount the file system in which the quota files have been created using the `mount`(1M) command. Whether or not you need to include the `–o quota` option depends on your `/etc/vfstab` or `samfs.cmd` file, as follows:

   - If you have edited the `/etc/vfstab` or `samfs.cmf` file to include the `quota` mount option, do not use the `–o quota` option on the `mount`(1M) command. Enter the `mount`(1M) command without the `–o quota` option, as follows:

```
# mount /qfs1
```

   - If the `/etc/vfstab` or `samfs.cmd` file does not include `quota` as a mount option, include the `–o quota` mount option on the `mount`(1M) command, as follows:

```
# mount –o quota /qfs1
```

Caution – Sun Microsystems recommends that you include the `quota` mount option in the `/etc/vfstab` or `samfs.cmd` file. If the file system is mounted without quotas enabled, and blocks or files are allocated or freed, the quota records become inconsistent with actual usages. Putting the `quota` option in the `/etc/vfstab` or `samfs.cmd` file helps to avoid this potential problem.

If a file system with quotas is mounted and run without the `quota` mount option, run `samfsck`(1M) with its `-F` option to update the quota file usage counts before again remounting the file system with quotas enabled.

For more information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

9. **Use the `samquota`(1M) command to set quotas for users, groups, or admin sets.**

Use the samquota(1M) command to set quotas for users, groups, or admin sets. Subsequent sections in this chapter provide procedures and show examples of this process. For more information on the samquota(1M) command, see the samquota(1M) man page.

▼ To Configure an Existing File System to Use Quotas

This procedure applies if you are creating quotas for a file system that is already populated with files.

If you are configuring a new file system to use quotas, see "To Configure a New File System to Use Quotas" on page 202.

1. **Use the su(1) command to become superuser.**
2. **Use the mount(1M) command to ensure that the file system is mounted.**

   Examine the /etc/mnttab file using the mount(1M) command with no arguments, as follows:

   ```
   # mount
   ```

3. **Use the cd(1) command to change to the root directory.**

   Change to the root directory of the file system for which quotas are to be enabled. For example:

   ```
   # cd /oldfs1
   ```

4. **Verify that quotas do not already exist on the file system.**

   From the root directory, use the ls(1) command's -a option to retrieve the list of files in this directory. You cannot enable quotas on a file system if quotas are already enabled for this file system.

   If any of the following files are present, quotas are, or previously have been, enabled for this file system: .quota_u, .quota_g, .quota_a.

5. **Use the dd(1M) command to create the quota file(s).**

Create the quota files for the type(s) of quota(s) you wish to enforce. Determine the highest existing ID numbers of the types of quotas you wish to enforce. Make the initial, zero quota files large enough to hold the record for those IDs; each quota file record requires 128 bytes.

Example 1. If you want to enable admin set quotas, and the highest admin set ID in use on the file system is 1024, the calculation is as follows:

- (1024+1)*128 = 131200
- 131200 / 4096 = 32.031...

Use the following command:

```
# dd if=/dev/zero of=/oldfs1/.quota_a bs=4096 count=33
```

Example 2. If you want to enable group quotas, and group IDs up to 2000 are in use, the calculation is as follows:

- (2000+1)*128 = 256128
- 256128 / 4096 = 62.531...

Use the following command:

```
# dd if=/dev/zero of=/oldfs1/.quota_g bs=4096 count=63
```

Example 3. If you want to enable user ID quotas, and user IDs up to 4799 are in use, the calculation is as follows:

- (4799+1)*128 = 1228800
- 1228800 / 4096 = 300.0

Use the following command:

```
# dd if=/dev/zero of=/oldfs1/.quota_u bs=4096 count=300
```

For more information on the `dd`(1M) command, see the `dd`(1M) man page.

6. **Use the `umount`(1M) command to unmount the file system.**

Unmount the file system in which the quota files have been created using the `umount`(1M) command. For example:

```
# umount /oldfs1
```

The file system needs to be unmounted so it can be remounted and have its quota files read at mount time. For more information on unmounting a file system, see “To Unmount a File System” on page 69.

7. **Edit the `/etc/vfstab` or `samfs.cmd` file. (Optional)**

   Quotas are enabled at mount time by one of the following:

   - By using the `–o quota` option to the `mount`(1M) command
   - By editing the `/etc/vfstab` file or the `samfs.cmd` file and adding the `quota` mount option. For more information on the `samfs.cmd` file, see the `samfs.cmd`(4) man page.

   If you want to mount the file system with quotas enabled every time you issue the `mount`(1M) command, consider performing this step. It eliminates the need to include the `–o quota` mount option on the `mount`(1M) command every time the file system is mounted.

   For example, you can edit the `/etc/vfstab` file and add `quota` to the mount options field for each file system for which quotas are to be enabled. The following file has been edited to be compatible with quotas:

```
# /etc/vfstab
# device    dev to  mount    FS     fsck  mount    mount
# to mount  fsck    point    type   pass  at boot  options
# --------  ------  -----    ----   ----  -------  -------
oldfs1      -       /oldfs1  samfs  -     yes      stripe=0,quota
```

8. **Use the `samfsck`(1M) command to perform a file system check.**

   Use the `samfsck`(1M) command’s `–F` option to perform a file system check. The `samfsck`(1M) command updates the quota files with correct, current usage information. Note, however, that it updates only records already allocated in the quota files. For example:

```
# samfsck –F /oldfs1
```

9. **Use the `mount`(1M) command to mount the file system.**

   Mount the file system using the `mount`(1M) command. Whether or not you need to include the `–o quota` option depends on your `/etc/vfstab` or `samfs.cmd` file, as follows:

- If you have edited the /etc/vfstab or samfs.cmd file to include the quota mount option, do not use the –o mount option on the mount(1M) command. Enter the mount(1M) command without the –o mount option, as follows:

```
# mount /oldfs1
```

- If the quota mount option is not present in the /etc/vfstab or samfs.cmd file, use the –o quota option to the mount(1M) command when you mount this file system, as follows:

```
# mount –o quota /oldfs1
```

Caution – Sun Microsystems recommends that you include the quota mount option in the /etc/vfstab or samfs.cmd file. If the file system is mounted without quotas enabled, and blocks or files are allocated or freed, the quota records become inconsistent with actual usages. Putting the quota option in the /etc/vfstab file helps to avoid this potential problem.

If a file system with quotas is mounted and run without the quota mount option, run samfsck(1M) with its -F option to update the quota file usage counts before again remounting the file system with quotas enabled.

For more information on the mount(1M) command, see the mount_samfs(1M) man page.

10. **Use the samquota(1M) command to set quotas for users, groups, or admin sets.**

    Subsequent sections in this chapter provide procedures and show examples of this process. For more information on the samquota(1M) command, see the samquota(1M) man page.

▼ To Assign Admin Set IDs to Directories and Files

1. **Use the su(1) command to become superuser.**
2. **Set the admin IDs.**

   Use the samchaid(1M) command to change the admin set IDs for the directory or file, as follows:

- To set IDs for a file or directory, specify the directory name or path. For example:

```
# samchaid 100 admin.dir
```

- To set IDs for a directory tree, use the –R and (if necessary) the –h options. The –R option specifies a recursive operation, and the –h option changes links, not targets. For example:

```
# samchaid –R -h 22 /qfs1/joe /qfs1/nancee
```

For more information on the samchaid(1M) command, see the samchaid(1M) man page.

Infinite Quotas and Zero Quotas

There are two kinds of special quotas: infinite quotas and zero quotas. These types of quotas are as follows:

- **Infinite quotas.** Users with infinite quotas are never denied access to any available file system resource.

  Infinite quotas can be set on a user, group, or admin set basis by setting both the hard block and hard file limits to zero. For example, the following command sets an infinite quota.

```
# samquota –U fred -b 0:s –f 0:h /qfs1
```

- **Zero Quotas.** Users with zero quotas cannot allocate any file system resources.

  A zero quota is assumed if the hard block or file limits are lower than the soft block or file limits. The following command sets these values.

```
# samquota –U fred –b 2:s –b 1:h /qfs1
```

If the system determines that any set of quota values are not valid, quota values are treated as if there were a zero quota. The samquota(1M) command reports this when it is run. If a user's soft limit is larger than the user's hard limit, the system denies any request for quota resources.

The file system treats infinite quotas and zero quotas as special quotas. Infinite and zero quota values can be set into record zero of the user, group, or admin set ID quota files, and from there they can become the default values for new users, groups, or admin set IDs.

▼ To Set Infinite Quotas

You can use the `samquota`(1M) command to set infinite quotas for particular users, groups, or admin set IDs by setting zero values for all hard and soft limits. For example:

```
# samquota -G turtles -b 0:s,h -f 0:s,h /qfs1
# samquota -G turtles /qfs1
                            Limits
       Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19        0        0
Blocks group   101     74992        0        0
Grace period                    1w
---> Infinite quotas in effect.
```

▼ To Set Zero Quotas

You can use the `samquota`(1M) command to set zero quotas to any inconsistent set of values. For example, this can be accomplished by setting any soft limit to a value that is larger than its corresponding hard limit, as follows:

```
# samquota -G turtles -b 1:s -b 0:h -f 1:s -f 0:h /qfs1
# samquota -G turtles
                             Limits
       Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19!       1        0
Blocks group   101     74992!       1        0
Grace period                    1w
---> Quota values inconsistent; zero quotas in effect.
```

▼ To Enable Default Quota Values for Users, Groups, or Admin Sets

You can use the samquota(1M) command to enable a default quota for a user, group, or admin set. This is accomplished by setting default limits into user, group, or admin set zero (0).

For example, the following samquota(1M) command sets default quotas for all admin set IDs:

```
# samquota -A 0 -b 12000:s -b 15000:h -f 1000:s -f 1200:h -t 1w /qfs1
```

On first reference, the preceding command sets any user's uninitialized admin set quota limits as follows:

- The soft block limit is set to 12000 blocks.
- The hard block limit is set to 15000 blocks.
- The soft file limit is set to 1000 files.
- The hard file limit is set to 1200 files.
- The grace period is set to one week.

Similar default quotas can be set for users or groups by specifying `-U 0` or `-G 0`, respectively, in place of `-A 0`.

For more information on the samquota(1M) command, see the samquota(1M) man page.

▼ To Enable Limits for Particular Users, Groups, or Admin Sets

You can use the samquota(1M) command to enable a set of limits for a particular user, group, or admin set. For example, the following commands enable various limits:

```
# samquota -U joe -b 15000:s -b 20000:h -f 500:s -f 750:h -t 3d /qfs1
# samquota -G proj -b 15000:s -b 20000:h -f 500:s -f 750:h -t 3d /qfs1
# samquota -A 7 -b 15000:s -b 20000:h -f 500:s -f 750:h -t 3d /qfs1
```

For more information on the samquota(1M) command, see the samquota(1M) man page.

Checking Quotas

After you have enabled disk and inode quotas, you can check quotas for individual users who exceed their quotas. The `samquota`(1M) command is an administrator command that generates a quota report on an individual user, group, or admin set. The `squota`(1) command is a user command that enables users to check their own individual quotas. TABLE 7-4 shows commands you can use to check quotas.

TABLE 7-4 Commands for Checking Quotas

Command	Task
`squota`(1)	This is a user command. It displays user quotas and other information specific to a single user. For more information, see the `squota`(1) man page.
`samquota`(1M)	This is an administrator command. It displays user, group, and admin set quotas, and it displays current disk use. This command also displays information about users who are exceeding their quotas. For more information, see the `samquota`(1M) man page.

▼ To Check for Exceeded Quotas

The following procedure shows how to check quotas.

1. **Use the `su`(1) command to become superuser.**
2. **Use the `samquota`(1M) command to display the quotas in effect.**

   Use the `samquota`(1M) command in one of the following ways to display quotas for mounted file systems in which quotas are enabled.

   a. **To display user quotas, specify the following command:**

```
# samquota –U userID [ file ]
```

For *userID*, specify the numeric user ID or user name of the user whose quotas are being examined.

Example 1. The following command retrieves user fred's quota statistics in the qfs1 file system on the server and displays output indicating that this user is not exceeding his quota:

```
# samquota -U fred /qfs1
                             Limits
        Type    ID    In Use    Soft     Hard
/qfs1
Files   user 28482      240      10000      12000
Blocks  user 28482      7540     1000000000 1200000000
Grace period                    1d
```

Example 2. The following command retrieves user gloria's quota statistics in all Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems and displays output indicating that this user is exceeding her quota. Note the plus sign (+) in the Blocks row of the output. The plus sign would appear in the Files row, too, if the soft quota limit were being exceeded for files.

```
# samquota -U gloria
                              Limits
       Type    ID    In Use     Soft     Hard
/qfs1
Files  user    101       26       500      750
Blocks user    101       42024+   40000    50000
Grace period                    1w
---> Warning:  online soft limits to be enforced in 6d21h36m45s
```

If a hard limit has been exceeded, or if the soft limit has been exceeded and the grace period has expired, the offending In Use field is marked with an asterisk character (*). If a quota record's limits are determined to be inconsistent, (for example if a soft limit is larger than a hard limit), an exclamation point is used to mark the field, and all allocation operations are prevented.

TABLE 7-5 shows the fields in the samquota(1M) output.

TABLE 7-5 samquota(1M) Output Fields

Field Name	Content
In Use	Current block usage.
Soft	Soft block limit.
Hard	Hard block limit.
Grace Period	Amount of time the user is allowed to exceed the soft limit.

b. To display group quotas, specify the following command:

```
# samquota –G groupID [ file ]
```

For *groupID*, specify the numeric group ID or the group name for the group of users whose quotas are being examined. For example, the following command retrieves user quota statistics for the group `turtles` in the `qfs3` file system:

```
# samquota –G turtles /qfs3
```

c. To display admin set quotas, specify the following command:

```
# samquota –A adminsetID [ file ]
```

For *adminsetID*, specify the numeric admin set ID of the site-specific administrator set whose quotas are being examined. For example, the following command retrieves user quota statistics for the admin set 457 in all Sun QFS, Sun SAM-FS and Sun SAM-QFS file systems:

```
# samquota –A 457 /qfs3
```

Changing and Removing Quotas

You can change quotas to adjust the amount of disk space or number of inodes allocated to users. You can also remove quotas from users or from an entire file system. The following sections describe how to change and remove quotas. The topics are as follows:

- "To Change the Grace Period" on page 215
- "To Change the Grace Period Expiration" on page 216
- "To Disable Quotas" on page 219
- "To Remove a File System's Quotas" on page 221
- "To Correct Quotas" on page 222

▼ To Change the Grace Period

You can use the samquota(1M) command to change the soft time limit grace period. This procedure does not affect the expiration time of those currently over their soft limit.

1. **Use the samquota(1M) command to retrieve quota statistics.**

   The samquota(1M) command can be used on a user, group, or admin set basis. The format is as follows:

```
# samquota –U userID [ file ]
# samquota –G groupID [ file ]
# samquota –A adminsetID [ file ]
```

   TABLE 7-6 shows the arguments to these commands.

TABLE 7-6 samquota(1M) Command Arguments

Argument	Description
userID	Specify the numeric user ID or user name of the user whose quotas are being changed.
groupID	Specify the numeric group ID or the group name for the group of users whose quotas are being changed.
adminsetID	Specify the numeric admin set ID of the site-specific administrator set whose quotas are being changed.
file	Specify a specific file system for the selected user, group, or admin set. The *file* argument can also be the name of any file in the file system. Typically, *file* is the name of the root directory of the file system.

2. **Examine the output from the samquota(1M) command.**

   Examine the output and determine what the new limits should be.

3. **Use the samquota(1M) command to change the limits.**

   Use the samquota(1M) command to change the soft time limit grace period. The format of this command is as follows:

```
# samquota –U userID –t interval file
# samquota –G groupID –t interval file
# samquota –A adminID –t interval file
```

In the preceding format, *interval* specifies the interval to use for the grace period. Specify an integer number for *interval* to indicate the quantity, and then specify a unit multiplier, if desired. By default, the unit multiplier is `s` to indicate that the interval is being specified in seconds. You can also specify `w` (for weeks), `d` (for days), `h` (for hours), or `m` (for minutes).

Example. Assume that you want to change the grace period for user `28482`. You enter the following `samquota`(1M) command:

```
# samquota -U 28482 /qfs1
```

This command generates the following information:

```
 Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files   user 28482         0    10000    12000
Blocks  user 28482         0 1000000000 1200000000
Grace period                    3d
```

You enter the following command to lower the soft time limits:

```
# samquota -U 28482 -t 1d /qfs1
```

Enter another `samquota`(1M) command to ensure that the time limits are reset:

```
# samquota -U 28482 /qfs1
Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files   user 28482         0    10000    12000
Blocks  user 28482         0 1000000000 1200000000
Grace period                    1d
```

▼ To Change the Grace Period Expiration

If a user has exceeded their soft quota limit, changing the grace period itself does not modify the expiration timer of any grace periods that have already started. If the grace period is already in effect, you can use the `samquota`(1M) command to modify the grace period in one of the following ways:

- **Clear the grace period.** The next time the user allocates a file or block (and is still over a soft limit), the grace period timer is reset to the grace period and starts counting down.
- **Reset the grace period.** When an expiration period is reset, the timer is reset to the present grace period, which starts counting down immediately.
- **Set the grace period to a value.** The timer is set to a value and it starts counting down immediately from that value. There are no restrictions on this value. The value can be larger than the grace period.
- **Expire the grace period.** The timer is set to expire immediately.

Example. The following command retrieves information on group `turtles` and shows that this group is over its soft limit:

```
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19     1000     1200
Blocks group   101     74992+   60000    75000
Grace period                    1w
---> Warning:  online soft limits to be enforced in 5d23h51m9s
```

The following commands clear the timer, so it starts counting the next time a user in group `turtles` attempts to allocate a block or file in `/qfs1`:

```
# samquota -G turtles -x clear /qfs1
Setting In-Use Field:  continue? y
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19     1000     1200
Blocks group   101     74992+   60000    75000
Grace period                    1w
```

The following commands reset the grace period:

```
# samquota -G turtles -x reset /qfs1
Setting In-Use Field:  continue? y
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19     1000     1200
Blocks group   101     74992+   60000    75000
Grace period                    1w
---> Warning:  online soft limits to be enforced in 6d23h59m54s
```

The following command expires the grace period:

```
# samquota -G turtles -x expire /qfs1
Setting In-Use Field:  continue? y
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19     1000     1200
Blocks group   101     74992*   60000    75000
Grace period                    1w
---> Online soft limits under enforcement (since 10s ago)
```

The following command sets a very long expiration period:

```
# samquota -G turtles -x 52w /qfs1
Setting In-Use Field:  continue? y
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101        19     1000     1200
Blocks group   101     74992+   60000    75000
Grace period                    1w
---> Warning:  online soft limits to be enforced in 51w6d23h59m56s
```

▼ To Disable Quotas

The following procedure shows how to disable quotas for a user, group, or admin set.

1. **Use the `su`(1) command to become superuser.**

2. **Obtain, save, and examine current quota information.**

   Use the `samquota`(1M) command to retrieve current quota information and write it to a backup file. The following example obtains quota information for a group quota on the group `turtles`:

```
# samquota –G turtles –e /qfs1 | & tee restore.quota.turtles
# Type  ID
#                  Limits
#            soft            hard
# Files
# Blocks
# Grace Periods
#
samquota -G 101 \
      -f      500:s -f      750:h \
      -b    10000:s -b    12000:h \
                 -t  1w   /qfs1
```

   To obtain quota information on a user quota, specify the `-U` *userID* option in place of the `–G` option. To obtain quota information on an admin set quota, specify the `–A` *adminID* option in place of the `–G` option.

3. **Use the `samquota`(1M) command to set soft and hard quotas to zero.**

   Use the `samquota`(1M) command to reset the quotas to invalid. The following command sets the quotas for group `turtles` to zero:

```
# samquota –G turtles –b 2:s –b 1:h /qfs1
```

   To zero the quotas for users or admin sets, specify the `-U` *userID* or `–A` *adminID* options in place of the `–G` option.

4. **Use the `samquota`(1M) command to verify your changes.**

Use the samquota(1M) command to verify that the quota has been correctly changed. The following example obtains quota information for a group quota of the group turtles:

```
# samquota –G turtles /qfs1
```

Enter the following command to change the soft and hard limits for the group:

```
# samquota -G turtles -b 2:s -b 1:h /qfs1
```

Enter the following command to verify the changed quotas:

```
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101         1!     500      750
Blocks group   101         8!       2        1
Grace period                    1w
---> Quota values inconsistent; zero quotas in effect.
```

In the preceding output, a zero quota is in effect. Note the exclamation point characters (!) to indicate the over quota condition in the output.

5. **Use the sh(1) and samquota(1M) commands to restore the group's quota.**

For example, enter the following commands to restore and verify the changed quotas:

```
# sh restore.quota.turtles
# samquota -G turtles /qfs1
                                    Limits
        Type    ID    In Use     Soft     Hard
/qfs1
Files  group   101         1      500      750
Blocks group   101         8    40000    50000
Grace period                    1w
```

To perform this operation on a user quota, specify the -U *userID* option in place of the –G option. To perform this operation on an admin set quota, specify the –A *adminID* option in place of the –G option.

▼ To Remove a File System's Quotas

To remove or disable quotas for a file system, you need to remove quota specifications from the mount process. The following procedure shows how to disable quotas for a file system.

1. **Use the `su`(1) command to become superuser.**

2. **Remove the `quota` mount option from the `/etc/vfstab` or `samfs.cmd` file.**

   Use a viewer, such as `vi`(1) or `cat`(1) to examine the `/etc/vfstab` or `samfs.cmd` file for the presence of the `quota` mount option.

   If this mount option is present, edit the file and remove the `quota` mount option.

3. **Use the `umount`(1M) command to unmount the file system.**

   If the file system is mounted, use the `umount`(1M) command to unmount the file system.

   For example:

   ```
   # umount /myfs
   ```

   If you have difficulty unmounting the file system, see "To Unmount a File System" on page 69.

4. **Use the `mount`(1M) command to remount the file system.**

   For example:

   ```
   # mount /myfs
   ```

5. **Dispense with the quota files.**

   If you expect to reinstate the quota feature at a later date, do not destroy the quota files. To preserve the quota files and reinstate quotas at a later date, unmount the file system, run the `samfsck`(1M) command with its `-F` option on the file system, and remount the file system again with the `quota` mount option. The `quota` mount option can be specified in either the `/etc/vfstab` file or in the `samfs.cmd` file as a mount option, or it can be specified on the `mount`(1M) command with the `–o quota` option.

If you do not expect to reinstate the quota feature at a later date, or if you want to reclaim the space consumed by the quota files, use the rm(1) command to remove the .quota_u, .quota_g, and .quota_a files. For example:

```
# rm /myfs/.quota_u
```

▼ To Correct Quotas

1. **Use the su(1) command to become superuser.**
2. **Use the umount(1M) command to unmount the file system.**

   If the file system is mounted, use the umount(1M) command to unmount the file system.

   For example:

   ```
   # umount /myfs
   ```

   If you have difficulty unmounting the file system, see “To Unmount a File System” on page 69.

3. **Use the samfsck(1M) command to perform a file system check.**

   Use the samfsck(1M) command’s –F option to perform a file system check. The samfsck(1M) command updates the quota files with correct, current usage information. Note, however, that it updates only records already allocated in the quota files. For example:

   ```
   # samfsck –F /myfs
   ```

4. **Use the mount(1M) command to remount the file system.**

   For example:

   ```
   # mount /myfs
   ```

CHAPTER 8

Advanced Topics

This chapter discusses advanced topics that are beyond the scope of basic system administration and usage. These topics are as follows:

Striping the `.inodes` File

This topic is applicable to Sun QFS and Sun SAM-FS file system only.

The Sun QFS and Sun SAM-QFS `.inodes` files are allocated in 16-kilobyte blocks as needed. An inode uses 512 bytes. In a Sun QFS or Sun SAM-QFS file system, the metadata devices (device type `mm`) are striped by default at the 16-kilobyte DAU level. This means that the first 32 inodes would be created on the first metadata device, then the next 32 inodes would be created on the next metadata device.

The stripe specification is taken from the `-o mm_stripe=`*n* option on the `mount`(1M) command. By default, one 16-kilobyte DAU is written to a meta device until it is full, then it switches to the next one. To use this feature, more than one `mm` device must be defined for the file system. This feature can be disabled by specifying `-o mm_stripe=0`.

For more information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

Daemons and Processes

All Sun QFS, Sun SAM-FS and Sun SAM-QFS daemons are named in the form `sam-`*daemon_name*`d`, which is `sam-`, followed by the daemon name, and followed by the lowercase letter `d`. This convention allows the daemons to be identified easily. Processes are named in a similar manner; the difference is that they do not end in the lowercase letter `d`. TABLE 8-1 shows some of the daemons and processes that can be running on your system (others, such as `sam-genericd` and `sam-catserverd`, might also be running depending on system activities).

TABLE 8-1 Daemons and Processes

Process	Description
`sam-archiverd`	Automatically archives Sun SAM-FS and Sun SAM-QFS files. This process runs as long as the Sun SAM-FS or Sun SAM-QFS file system is mounted.
`sam-fsd`	Master daemon.
`sam-ftpd`	Transfers data between multiple Sun SAM-FS or Sun SAM-QFS host systems.
`sam-robotsd`	Starts and monitors automated library media changer control daemons.
`sam-scannerd`	Monitors all manually mounted removable media devices. The scanner periodically checks each device for inserted archive media cartridges.
`sam-releaser`	Attempts to release disk space occupied by previously archived files on Sun SAM-FS or Sun SAM-QFS file systems until a low water mark is reached. The releaser is started automatically when a high water mark is reached on disk cache and stops when it has finished releasing files. This is a process, not a daemon.

TABLE 8-1 Daemons and Processes *(Continued)*

Process	Description
`sam-stagealld`	Controls the associative staging of Sun SAM-FS and Sun SAM-QFS files.
`sam-stagerd`	Controls the staging of Sun SAM-FS and Sun SAM-QFS files.
`sam-rpcd`	Controls the remote procedure call (RPC) application programmer interface (API) server process.

When running Sun QFS, Sun SAM-FS, or Sun SAM-QFS, the `sam-fsd` daemon is started by `init` as part of `/etc/inittab` processing. It is started at `init` levels 2 and 3. It should restart automatically in case of kill or failure.

When running Sun SAM-FS or Sun SAM-QFS, the `sam-fsd` daemon creates the following processes:

- `sam-archiverd`. The `sam-archiverd` daemon starts the `sam-arcopy` and the `sam-arfind` processes.
- `sam-catserverd`. Issuing a `samd stop` command stops this daemon.
- `sam-ftpd`.
- `sam-initd`.
- `sam-robotsd`. Issuing a `samd stop` command stops this daemon.
- `sam-scannerd`. Issuing a `samd stop` command stops this daemon.
- `sam-sharefsd`. One of these is created for each Sun QFS shared file system.
- `sam-stagealld`.
- `sam-stagerd`.

Trace Files

Several Sun QFS, Sun SAM-FS and Sun SAM-QFS processes can write messages to trace files. These messages contain information about the state and progress of the work performed by the daemons. The messages are primarily used by Sun Microsystems staff members to to improve performance and diagnose problems. The message content and format are subject to change from release to release.

Trace files can be used in debugging. Typically, trace files are not written. You can enable trace files for Sun SAM-FS and Sun SAM-QFS software by editing the `defaults.conf` file. You can enable tracing for all processes, or you can enable tracing for individual processes. The following processes can be traced:

- `sam-archiverd`

- `sam-catserverd`
- `sam-fsd`
- `sam-ftpd`
- `sam-recycler`
- `sam-sharefsd`
- `sam-stagerd`

By default, the trace files are written to `/var/opt/SUNWsamfs/trace`. In that directory, the trace files are named for the processes (`archiver`, `catserver`, `fsd`, `ftpd`, `recycler`, `sharefsd`, and `stager`). You can change the names of the trace files by specifying directives in the `defaults.conf` configuration file. For information on the `defaults.conf` file, see the `defaults.conf`(4) man page.

Trace File Content

Trace file messages contain the time and source of the message. The messages are produced by events in the processes. The events can be selected using directives in the `defaults.conf` file.

The default events are as follows:

- Customer notification syslog or notify file messages
- Nonfatal program errors
- Fatal syslog messages
- Process initiation and completion
- Other miscellaneous events

The following events can also be traced:

- Memory allocations
- Interprocess communication
- File actions
- Operator messages
- Queue contents when changed
- Other miscellaneous events

The default message elements (program name, process id (pid) and time) are always included and cannot be excluded. Optionally, the messages can also contain the following elements:

- The date. (The time is always included.)
- The source file name and line number.

- The event type.

Trace File Rotation

To prevent the trace files from growing indefinitely, the `sam-fsd` daemon monitors the size of the trace files and periodically executes the following script:

```
/opt/SUNWsamfs/sbin/trace_rotate.sh
```

This script moves the trace files to sequentially numbered copies. You can modify this script to suit your operation. Alternatively, you can provide this function using `cron`(1) or some other facility.

If `/opt/SUNWsamfs/sbin/trace_rotate.sh` does not exist, the `sam-fsd` daemon performs no action.

Determining Which Processes Are Being Traced

To determine which processes are being traced currently, enter the `sam-fsd`(1M) command at the command line. CODE EXAMPLE 8-1 shows the output from this command.

CODE EXAMPLE 8-1 Output From the `sam-fsd`(1M) Command

```
# sam-fsd
Trace file controls:
sam-archiverd /var/opt/SUNWsamfs/trace/archiver
              cust err misc files date module
              size    0    age 0
sam-catserverd /var/opt/SUNWsamfs/trace/catserver
              cust err fatal ipc misc proc queue ftp debug date
module
              size    0    age 0
sam-fsd       /var/opt/SUNWsamfs/trace/fsd
              cust err fatal ipc misc proc queue ftp debug date
module
              size    0    age 0
sam-ftpd      /var/opt/SUNWsamfs/trace/ftp
              cust err fatal ipc misc proc queue ftp debug date
module
              size    0    age 0
sam-recycler  /var/opt/SUNWsamfs/trace/recycler
```

CODE EXAMPLE 8-1 Output From the sam-fsd(1M) Command

```
# sam-fsd
              cust err fatal ipc misc proc queue ftp debug date
module
              size    0    age 0
sam-sharefsd  off

sam-stagerd   /var/opt/SUNWsamfs/trace/stager
              cust err misc proc files debug date module
              size    0    age 0
Would stop sam-archiverd()
Would stop sam-ftpd()
Would stop sam-stagealld()
Would stop sam-stagerd()
Would stop sam-initd()
```

For more information on enabling trace files, see the defaults.conf(4) man page and the sam-fsd(1M) man page.

Using the setfa(1) Command to Set File Attributes

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems allow end users to set performance attributes for files and directories. These performance features can be enabled by applications on a per-file or per-directory basis. The following sections describe how the application programmer can use these features to select file attributes for files and directories, to preallocate file space, to specify the allocation method for the file, and to specify the disk stripe width.

Selecting File Attributes for Files and Directories

File attributes are set using the setfa(1) command. The setfa(1) command sets attributes on a new or existing file. The file is created if it does not already exist.

Attributes can be set on a directory as well as a file. When using setfa(1) with a directory, files and directories created within that directory inherit the attributes set in the original directory. To reset attributes on a file or directory to the default, use the -d (default) option. When the -d option is used, attributes are first reset to the default and then other attributes are processed.

Preallocating File Space

An end user can preallocate space for a file. This space is associated with a file so that no other files in the file system can use the disk addresses allocated to this file. Preallocation ensures that space is available for a given file, which avoids a file system full condition, and that this space is allocated sequentially as defined by the file system. Preallocation is assigned at the time of the request rather than when the data is actually written to disk.

Note that space can be wasted when preallocating files. If the file size is less than the allocation amount, the kernel allocates space to the file from the current file size up to the allocation amount. When the file is closed, space below the allocation amount is not freed.

A file is preallocated by using the `setfa`(1) command with the `–l` (lowercase letter L) option and specifying the file length in bytes (`b`), kilobytes (`k`), megabytes (`m`), or gigabytes (`g`).

For example, to preallocate a 1-gigabyte file named `/qfs/file_alloc`, enter the following:

```
# setfa –l 1g /qfs/file_alloc
```

After space for a file has been preallocated, truncating a file to 0 length or removing the file returns all space allocated for a file. There is no way to return only part of a file’s preallocated space to the file system. In addition, if a file is preallocated in this manner, there is no way to extend the file beyond its preallocated size in future operations.

Selecting a File Allocation Method and Stripe Width

By default, a file created uses the allocation method and stripe width specified at mount time (see the `mount_samfs`(1M) man page). However, an end user might want to use a different allocation scheme for a file or directory of files, and this can be accomplished by using the `setfa`(1) command with the `–s` (stripe) option.

The allocation method can be either round-robined or striped. The `–s` option determines the allocation method and the stripe width, and TABLE 8-2 shows the effect of this option.

TABLE 8-2 File Allocations and Stripe Widths

-s stripe	Allocation Method	Stripe Width	Explanation
0	Round-robin	n/a	The file is allocated on one device until that device has no space.
1-255	Striped	1-255 DAUs	The file stripes across all disk devices with this number of DAUs per disk

The following example shows how a file can be created explicitly by specifying a round-robined allocation method. The command also preallocates 100 megabytes of space for a file called `/qfs/100MB.rrobin`:

```
# setfa –s 0 –l 100m /qfs/100MB.rrobin
```

The next example shows how a file can be created explicitly by specifying a striped allocation method with a stripe width of 64 DAUs. Preallocation is not used.

```
# setfa –s 64 /qfs/file.stripe
```

Selecting a Striped Group Device

Striped group devices are supported for Sun QFS and Sun SAM-QFS file systems only.

A user can specify that a file begin allocation on a particular striped group. If the file allocation method is round-robin, the file is allocated on the designated stripe group.

For example, the following `setfa`(1) commands specify that `file1` and `file2` be independently spread across two different striped groups:

```
# setfa -g0 –s0 file1
# setfa -g1 –s0 file2
```

This capability is particularly important for applications that must achieve levels of performance that approach raw device speeds. For more information, see the `setfa`(1) man page.

Accommodating Large Files

When manipulating very large files, pay careful attention to the size of disk cache available on the system. If you try to write a file that is larger than your disk cache, behavior differs depending on the type of file system you are using, as follows:

- If you are using the Sun QFS file system, the system returns an `ENOSPC` error.
- If you are using the Sun SAM-FS or Sun SAM-QFS file system, the program blocks, waiting for space that might never exist, because there is not enough disk space available to handle such requests.

  If you are operating within a Sun SAM-FS or Sun SAM-QFS environment and if your application requires writing a file that is larger than the disk cache, you can segment the file using the `segment`(1) command. For more information on the `segment`(1) command, see the `segment`(1) man page or see the *Sun SAM-FS and Sun SAM-QFS Storage and Archive Management Guide*.

Caution – Even though the Sun SAM-FS and Sun SAM-QFS file systems do not use the `tar`(1) command to read or write data onto cartridges, data appears on the cartridges in the industry standard `tar`(1) format. This is done for compatibility reasons. In addition, this practice allows users to read cartridges even when the Sun SAM-FS or Sun SAM-QFS file system is not available.

The `star`(1) command can be used to restore data on any UNIX system. A Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system does not have to be mounted, but the `star`(1) command binary (which is part of the software package) must be installed. For more information on disaster recovery, see *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Disaster Recovery Guide*.

Multireader File System

The multireader file system is single-writer, multireader file system. The `writer` and `reader` mount options that enable the multireader file system can be specificed on Sun QFS file systems only. The mount options are described in this section and on the `mount_samfs`(1M) man page.

The multireader file system is mounted on the single writer host by specifying the `–o writer` option on the `mount`(1M) command. The host system with the `writer` mount option is the only host system that is allowed to write to the file system. The `writer` host system updates the file system. You must ensure that only one host in a multireader file system has the file system mounted with the `writer` mount option enabled. If `–o writer` is specified, directories are written through to disk at each change and files are written through to disk at close.

Caution – The multireader file system can become corrupted if more than one writer host has the file system mounted at one time. It is the site's responsibility to insure that this situation does not occur.

A multireader file system is mounted on one or more reader hosts by specifying the `–o reader` option on the `mount`(1M) command. There is no limit to the number of host systems that can have the multireader file system mounted as a reader.

A major difference between the multireader file system and Sun QFS shared file system is that the multireader host reads metadata from the disk, and the client hosts of a Sun QFS shared file system read metadata over the network.

Note – You cannot mount or use a Sun QFS shared file system with the `writer` or `reader` mount options enabled. For more information, see "Sun QFS Shared File System" on page 89.

You need to ensure that any potential metadata server in a multireader file system has access to the device definitions that describe the `ma` device. Copy the lines from the `mcf` file that resides on the primary metadata server host to the `mcf` files on the alternate metadata servers.

In a multireader file system environment, the Sun QFS software ensures that all servers that access the same file system can always access the current environment. When the writer closes a file, the Sun QFS file system writes all information for that file to disk immediately. A `reader` host can access a file after the file is closed by the writer. These and other steps ensure that no host system in a multireader file system ever risks getting into an out-of-sync condition with the file system.

By default, the metadata information on a reader host is invalidated and refreshed every 30 seconds. You can use the `–o invalid=`*n* option to the `mount`(1M) command to specify a refresh rate between 0 seconds and 60 seconds. If the refresh rate is set to a small value, the Sun QFS file system reads the directory and other metadata information more frequently. More frequent updates result in more overhead for the system and can affect performance.

Note – Prior to the Sun QFS 4.0 release, the `writer` and `reader` mount options were implemented as `shared_writer` and `shared_reader` options, respectively. As of the 4.0 release, these options are implemented as the `writer` and `reader` options. The `shared_writer` and `shared_reader` syntax is supported in the 4.0 release for backward compatibility. For more information on the multireader file system, see the `mount_samfs`(1M) man page.

Using the SAN-QFS File System

The SAN-QFS file system allows multiple users to access the same data at full disk speeds. This product can be especially useful for database, data streaming, web page service, or any application that demands high performance, shared-disk access in a heterogeneous environment.

The SAN-QFS file system can be used in conjunction with fiber-attached devices in a Storage Area Network (SAN). The SAN-QFS file system enables high-speed access to data using Sun QFS software and software such as Tivoli SANergy File Sharing software. To use the SAN-QFS file system, you must have both Sun QFS 4.0 release and the Tivoli SANergy File Sharing 2.2.3 software installed. For information on other levels of Sun QFS and Tivoli SANergy File sharing that are supported, contact your Sun Microsystems sales representative.

Note – In environments that include only Sun Solaris operating environment (OE) systems, Sun Microsystems recommends that you use the Sun QFS shared file system described in "Sun QFS Shared File System" on page 89.

The following sections describe other aspects of the SAN-QFS file system:

- "To Enable the SAN-QFS File System" on page 233
- "Releasing SANergy File Holds" on page 235
- "Expanding SAN-QFS File Systems" on page 235
- "SAN-QFS Shared File System and Sun QFS Shared File System Comparison" on page 235

▼ To Enable the SAN-QFS File System

1. **Verify your environment.**

Verify that the following conditions are present:

- The Sun QFS file system must be tested and fully operational.
- You must have Tivoli SANergy File Sharing 2.2.3 software.

2. **Use the `mount`(1M) command to mount the file system on your server.**

3. **Enable NFS access.**

   Enable NFS access to client hosts by using the following command:

```
# share qfs_file_system_name
```

   In the preceding format, *qfs_file_system_name* is the name of your Sun QFS file system. For example, `qfs1`. For more information on the `share`(1M) command, see the `share`(1M) or `share_nfs`(1M) man pages.

4. **Edit the file system table (`/etc/dfs/dfstab`) on the server to enable access at boot time. (Optional)**

   Perform this step if you want to automatically enable this access at boot time.

5. **Edit the `/etc/vfstab` file on each client and add the file system.**

   Add the *qfs_file_system_name* from Step 3 to the table.

   For example, you can edit the `/etc/vfstab` file and add a line similar to the following:

```
server:/qfs1  -  /qfs1  samfs  -  yes  stripe=1
```

   For more information on editing the `/etc/vfstab` file, see *Sun QFS, Sun SAM-FS, and Sun SAM-QFS Installation and Configuration Guide*.

6. **Use the `mount`(1M) command to mount the Sun QFS file system.**

   Use the `mount`(1M) command to mount the Sun QFS file system on each client. For example:

```
client# mount qfs1
```

   Enter one `mount`(1M) command per client. For more information on the `mount`(1M) command, see the `mount`(1M) or the `mount_samfs`(1M) man pages.

7. **Configure the Tivoli SANergy File Sharing software.**

Use the `config`(1M) command (in `/opt/SANergy/config`) to invoke the SANergy configuration tool. The SANergy configuration tool has a graphical user interface. Provide the information requested at each step in its process. For more information on this tool, see your Tivoli SANergy documentation.

Releasing SANergy File Holds

The `samunhold`(1M) command can be used to release SANergy file holds. If holds are present in a file system, the holds are described in messages written to console messages and to `/var/adm/messages` when you attempt to unmount the file system.

It is preferable to allow SANergy File Sharing to clean up its holds, but in an emergency, or in case of a SANergy File Sharing system failure, you can use the `samunhold`(1M) command to avoid a reboot.

For more information on this command, see the `samunhold`(1M) man page.

Expanding SAN-QFS File Systems

You can use the `samgrowfs`(1M) command to increase the size of a SAN-QFS file system. To perform this task, follow the procedures described in “To Add Disk Cache to a File System” on page 78. When using this procedure, be aware that the line-by-line device order in the `mcf` file must match the order of the devices listed in the file system’s superblock. The devices listed in the file system’s superblock are numbered in the order encountered in the `mcf` file (when created).

When the `samgrowfs`(1M) command is issued, the devices that had been in the `mcf` file prior to issuing the `samgrowfs`(1M) command keep their position in the superblock. New devices are written to subsequent entries in the order encountered.

If this new order does not match the order in the superblock, the SAN-QFS file system cannot be fused.

SAN-QFS Shared File System and Sun QFS Shared File System Comparison

The SAN-QFS file system and the Sun QFS shared file system are both shared file systems with the following similarities:

- Both can stage files.

- Both are useful in data capture environments in which it is desirable that the primary file system host not be responsible for writing the data.
- Both are advantageous in environments where there is contention for writing files.

These file systems differ in the following areas:

TABLE 8-3 SAN-QFS Shared File System Versus Sun QFS Shared File System

SAN-QFS File System	Sun QFS Shared File System
Does not use the natural metadata and incurs additional latency in opening files.	Uses natural metadata.
Preferred in heterogeneous computing environments (that is, when not all hosts are Sun systems).	Preferred in homogeneous Sun Solaris OEs.
Useful in environments where multiple hosts must be able to write data.	Multiple hosts can write. Preferred when multiple hosts must write to the same file at the same time.
User mode implementation.	Kernel mode implementation with strong security.

I/O Performance

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support paged I/O and direct I/O. The following sections describe these types of I/O and explain how to enable the ability to switch between the I/O types automatically.

Paged I/O

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems support both paged and direct I/O. Paged I/O (also called buffered or cached I/O) is selected by default.

Direct I/O

Direct I/O is a process by which data is transferred directly between the user's buffer and the disk. This means that much less time is spent in the system. For performance purposes, direct I/O should be specified only for large, block-aligned, sequential I/O.

The `setfa`(1) command and the `sam_setfa`(3) library routine both have a `–D` option that sets the direct I/O attribute for a file and/or directory. If applied to a directory, the direct I/O attribute is inherited by any files and directories created in that directory. After the `–D` option is set, the file uses direct I/O.

You can also select direct I/O for a file by using the Sun Solaris Operating environment (OE) `directio`(3C) function call. If you use the function call to enable direct I/O, it is a temporary setting. The setting lasts only while the file is active.

To enable direct I/O on a file system basis, see the `–o forcedirectio` option on the `mount`(1M) command; put the `forcedirectio` keyword in the mount option column of the `/etc/vfstab` file; or use it as a directive in the `samfs.cmd` file.

For more information, see the `setfa`(1), `sam_setfa`(3), `directio`(3C), `samfs.cmd`(4) , and `mount_samfs`(1M) man pages.

I/O Switching

The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems also support automatic I/O switching. I/O switching is a process by which you can specify that a certain amount of paged I/O should occur before the system switches to direct I/O. This automatic, direct I/O switching allows the system to perform a site-defined amount of consecutive I/O operations and then automatically switch from paged I/O to direct I/O. By default, paged I/O is performed, and I/O switching is disabled.

I/O switching should reduce page cache usage on large I/O operations. To enable this, use the `dio_wr_consec` and `dio_rd_consec` parameters as directives in the `samfs.cmd` file or as options to the `mount`(1M) command.

For more information on these options, see the `mount_samfs`(1M) or `samfs.cmd`(4) man pages.

Increasing Large File Transfer Performance

Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems are tuned to work with a mix of file sizes. You can increase the performance of disk file transfers for large files by enabling file system settings.

Note – Sun Microsystems recommends that you experiment with performance tuning outside of a production environment. Tuning these variables incorrectly can have unexpected effects on the overall system.

If your site has a Sun Enterprise Services (SES) support contract, please inform SES if you change performance tuning parameters.

---

1. **Set the maximum device read/write directive.**

   The `maxphys` parameter in the Sun Solaris `/etc/system` file controls the maximum number of bytes that a device driver reads or writes at any one time. The default value for the `maxphys` parameter can differ depending on the level of your Sun Solaris OE, but it is typically around 128 kilobytes. In this step, you set `maxphys` to 8 megabytes.

```
set maxphys = 0x800000
```

2. **Set the SCSI disk maximum transfer parameter.**

   The `sd` driver enables large transfers for a specific file by looking for the `sd_max_xfer_size` definition in the `/kernel/drv/sd.conf` file. If it is not defined, it uses the value defined in the `sd` device driver definition, `sd_max_xfer_size`, which is 1024*1024 bytes.

   To enable and encourage large transfers, add the following line at the end of the `/kernel/drv/sd.conf` file.

```
sd_max_xfer_size=0x800000;
```

3. **Set the fibre disk maximum transfer parameter.**

The ssd driver enables large transfers for a specific file by looking for the ssd_max_xfer_size definition in the /kernel/drv/ssd.conf file. If it is not defined, it uses the value defined in the ssd device driver definition, ssd_max_xfer_size, which is 1024*1024 bytes.

Add the following line at the end of the /kernel/drv/ssd.conf file:

```
ssd_max_xfer_size=0x800000;
```

4. **Reboot the system.**

5. **Set the** writebehind **parameter.**

   This step affects paged I/O only.

   The writebehind parameter specifies the number of bytes that are written behind by the file system when performing paged I/O on a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system. Matching the writebehind value to a multiple of the RAID's read-modify-write value can increase performance.

   This parameter is specified in units of kilobytes and is truncated to an 8-kilobyte multiple. If set, this parameter is ignored when direct I/O is performed. The default writebehind value is 512 kilobytes. This value favors large-block, sequential I/O.

   Set the writebehind size to a multiple of the RAID 5 stripe size for both hardware and software RAID 5. The RAID 5 stripe size is the number of data disks multiplied by the configured stripe width.

   For example, assume that you configure a RAID 5 device with 3 data disks plus 1 parity disk (3+1) with a stripe width of 16 kilobytes. The writebehind value should be 48 kilobytes, 96 kilobytes, or some other multiple, to avoid the overhead of the read-modify-write RAID 5 parity generation.

   For Sun QFS and Sun SAM-QFS file systems, the DAU (sammkfs -a option) should also be a multiple of the RAID 5 stripe size. This allocation ensures that the blocks are contiguous.

   You should test the system performance after resetting the writebehind size. The following example shows testing timings of disk writes:

```
# timex dd if=/dev/zero of=/sam/myfile bs=256k count=2048
```

   The writebehind parameter can be set from a mount option, from within the samfs.cmd file, from within the /etc/vfstab file, or from a command within the samu(1M) utility. For information on enabling this from a mount option, see the

-o writebehind=*n* option on the mount_samfs(1M) man page. For information on enabling this from the samfs.cmd file, see the samfs.cmd(4) man page. For information on enabling this from within samu(1M), see the samu(1M) man page.

6. **Set the readahead parameter.**

   This step affects paged I/O only.

   The readahead parameter specifies the number of bytes that are read ahead by the file system when performing paged I/O on a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system. This parameter is specified in units of kilobytes and is truncated to an 8-kilobyte multiple. If set, this parameter is ignored when direct I/O is performed.

   Increasing the size of the readahead parameter increases the performance of large file transfers, but only to a point. You should test the performance of the system after resetting the readahead size until you see no more improvement in transfer rates. The following is an example method of testing timings on disk reads:

```
# timex dd if=/sam/myfile of=/dev/null bs=256k
```

   The readahead parameter should be set to a size that increases the I/O performance for paged I/O. Also note that too large a readahead size can hurt performance. You should test various readahead sizes for your environment. It is important to consider the amount of memory and number of concurrent streams when you set the readahead value. Setting the readahead value multiplied by the number of streams to a value that is greater than memory can cause page thrashing.

   The default readahead is 1024 kilobytes. This value favors large-block, sequential I/O. For short-block, random I/O applications, readahead should be set to the typical request size. Database applications do their own readahead, so for these applications, set readahead to 0.

   The readahead setting can be enabled from either a mount option, from within the samfs.cmd file, from within the /etc/vfstab file, or from a command within the samu(1M) utility. For information on enabling this from a mount option, see the -o readahead=*n* option on the mount_samfs(1M) man page. For information on enabling this from the samfs.cmd file, see the samfs.cmd(4) man page. For information on enabling this from within samu(1M), see the samu(1M) man page.

7. **Set the stripe width.**

   The -o stripe=*n* option on the mount(1M) command specifies the stripe width for the file system. The stripe width is based on the disk allocation unit (DAU) size. The *n* argument specifies that *n* * DAU bytes are written to one device before switching to the next device. The DAU size is set when the file system is initialized by the sammkfs(1M) command's -a option.

If `–o stripe=0` is set, files are allocated to file system devices using the round-robin allocation method. Each file is created on the next device. Each file is completely allocated on this device until that device is full. Round robin is the preferred setting for a multistream environment. If `–o stripe=`*n* is set to an integer greater than 0, files are allocated to file system devices using the stripe method. To determine the appropriate `–o stripe=`*n* setting, try varying the setting and taking performance readings. Striping is the preferred setting for turnkey applications with a required bandwidth.

The stripe width can also be set from the `/etc/vfstab` file or from the `samfs.cmd` file. Options on the `mount`(1M) command override settings in the `/etc/vfstab` file. Settings in the `/etc/vfstab` file override directives in the `samfs.cmd` file.

For more information on the `mount`(1M) command, see the `mount_samfs`(1M) man page. For more information on the `samfs.cmd` file, see the `samfs.cmd`(4) man page.

Qwrite

The Qwrite capability can be enabled in Sun QFS and Sun SAM-QFS environments.

By default, the Sun QFS and Sun SAM-QFS file systems disable simultaneous reads and writes to the same file. This is the mode defined by the UNIX `vnode` interface standard, which gives exclusive access to only one write while other writers and readers must wait. Qwrite enables simultaneous reads and writes to the same file from different threads.

The Qwrite feature can be used in database applications to enable multiple simultaneous transactions to the same file. Database applications typically manage large files and issue simultaneous reads and writes to the same file. Unfortunately, each system call to a file acquires and releases a read/write lock inside the kernel. This lock prevents overlapped (or simultaneous) operations to the same file. If the application itself implements file locking mechanisms, the kernel locking mechanism impedes performance by unnecessarily serializing I/O.

Qwrite can be enabled in the `/etc/vfstab` file, in the `samfs.cmd` file, and as a mount option. The `–o qwrite` option on the `mount`(1M) command bypasses the file system locking mechanisms (except for applications accessing the file system through NFS) and lets the application control data access. If `qwrite` is specified, the file system enables simultaneous reads and writes to the same file from different threads. This option improves I/O performance by queuing multiple requests at the drive level.

The following example uses the mount(1M) command to enable Qwrite on a database file system:

```
# mount –F samfs –o qwrite /db
```

For more information on this feature, see the qwrite directive on the samfs.cmd(4) man page or the –o qwrite option on the mount_samfs(1M) man page.

Setting the Write Throttle

By default, the Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems set the -o wr_throttle=*n* option to the mount(1M) command to 16 megabytes. The -o wr_throttle=*n* option limits the number of outstanding write bytes for one file to *n*.

If a file has *n* write bytes outstanding, an application that attempts to write to that file is suspended until enough bytes have completed the I/O to allow the application to be resumed.

If your site has thousands of streams, such as thousands of NFS-shared workstations accessing the file system, you can tune the -o wr_throttle=*n* option in order to avoid memory stales. Generally, the number of streams multiplied by the *n* argument to the -o wr_throttle=*n* option should be less than the total size of the host system's memory minus the memory needs of the Solaris OE. In other words:

```
number_of_streams * n < total_memory - Solaris OE memory needs
```

For turnkey applications, you might want to use a size larger than the default 16 megabytes because this keeps more pages in memory.

Setting the Flush-Behind Rate

Two mount parameters control the flush-behind rate for pages written sequentially and stage pages. The flush_behind and stage_flush_behind mount parameters are read from the samfs.cmd file, the /etc/vfstab file, or from the mount(1M) command.

The `flush_behind=`*n* mount parameter sets the maximum flush-behind value. Modified pages that are being written sequentially are written to disk asynchronously to help the Sun Solaris VM layer keep pages clean. To enable this feature, set *n* to be an integer, $16 \le n \le 8192$. By default, *n* is set to 0, which disables this feature. The *n* argument is specified in kilobyte units.

The `stage_flush_behind=`*n* mount parameter sets the maximum stage flush-behind value. Stage pages that are being staged are written to disk asynchronously to help the Sun Solaris VM layer keep pages clean. To enable this feature, set *n* to be an integer such that, $16 \le n \le 8192$. By default, *n* is set to 0, which disables this feature. The *n* argument is specified in kilobyte units.

For more information on these mount parameters, see the `mount_samfs`(1M) man page or the `samfs.cmd`(4) man page.

Glossary

A

addressable storage	The storage space encompassing online, nearline, offsite, and offline storage that is user-referenced through a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system.
archive media	The media to which an archive file is written. Archive media can be removable tape or magneto-optical cartridges in a library. In addition, archive media can be a mount point on another system.
archive storage	Copies of file data that have been created on archive media.
archiver	The archive program that automatically controls the copying of files to removable cartridges.
audit (full)	The process of loading cartridges to verify their VSNs. For magneto-optical cartridges, the capacity and space information is determined and entered into the automated library's catalog.
automated library	A robotically controlled device designed to automatically load and unload removable media cartridges without operator intervention. An automated library contains one or more drives and a transport mechanism that moves cartridges to and from the storage slots and the drives.

B

backup storage	A snapshot of a collection of files for the purpose of preventing inadvertent loss. A backup includes both the file's attributes and associated data.
block allocation map	A bitmap representing each available block of storage on a disk and indicating whether the block is in use or free.
block size	See DAU.

C

cartridge	A physical entity that contains media for recording data. A tape or optical disk. Sometimes referred to as *a piece of media*, *a volume*, or *the medium*.
catalog	A record of the VSNs in an automated library. There is one catalog for each automated library, and at a site, there is one historian for all automated libraries.
client-server	The model of interaction in a distributed system in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called the client. The program satisfying the response is called the server.
connection	The path between two protocol modules that provides reliable stream delivery service. A TCP connection extends from a TCP module on one machine to a TCP module on the other.

D

data device	For a Sun QFS, Sun SAM-FS, or Sun SAM-QFS file system, a device or group of devices upon which file data is stored.

DAU (disk allocation unit) The basic unit of online storage. Also called block size.

The Sun SAM-FS and Sun SAM-QFS file systems support both a small and a large DAU. The small DAU is 4 kilobytes (2^{14} or 4096 bytes). The large DAU is 16, 32, or 64 kilobytes. The available DAU size pairs are 4/16, 4/32, and 4/64.

In addition, the Sun QFS and Sun SAM-QFS file systems support a fully adjustable DAU, sized from 16 kilobytes through 65,528 kilobytes. The DAU you specify must be a multiple of 8 kilobytes.

device logging A configurable feature that provides device-specific error information used to analyze device problems.

device scanner Software within the Sun SAM-FS or Sun SAM-QFS file system that periodically monitors the presence of all manually mounted removable devices and that detects the presence of mounted cartridges that can be requested by a user or other process.

direct access A file attribute (stage never) designating that a nearline file can be accessed directly from the archive media and need not be retrieved to disk cache.

direct-attached library An automated library connected directly to a server using a SCSI interface. A SCSI attached library is controlled directly by the Sun SAM-FS or Sun SAM-QFS software by using the SCSI standard for automated libraries.

direct I/O An attribute used for large block-aligned sequential I/O. The `setfa`(1) command's `-D` option is the direct I/O option. It sets the direct I/O attribute for a file or directory. If applied to a directory, the direct I/O attribute is inherited.

directory A file data structure that points to other files and directories within the file system.

disk allocation unit See DAU.

disk buffer When using Sun SAM-Remote software, the disk buffer is a buffer on the server system that is used when archiving data from the client to the server.

disk cache The disk-resident portion of the Sun SAM-FS and Sun SAM-QFS file system software. It is used to create and manage data files between online disk cache and archive media. Individual disk partitions or an entire disk can be used as disk cache.

disk space thresholds An administrator-defined amount of disk space that is available to a user. This defines the range of desirable disk cache utilization. The high threshold indicates the maximum level of disk cache utilization. The low threshold indicates the minimum level of disk cache utilization. The releaser controls disk cache utilization based on these predefined disk space thresholds.

disk striping	The process of recording a file across several disks, thereby improving access performance and increasing overall storage capacity. Also see entries for striping.
drive	A mechanism for transferring data to and from a removable media volume.

E

Ethernet	A local-area, packet-switched network technology. Originally designed for coaxial cable, it is now found running over shielded, twisted-pair cable. Ethernet is a 10- or 100-megabytes-per-second LAN.
extent array	The array within a file's inode that defines where each data block assigned to the file is located on the disk.

F

family device set	See family set.
family set	A storage device that is represented by a group of independent physical devices, such as a collection of disks or the drives within an automated library. Also see disk cache family set.
FDDI	Fiber distributed data interface. A 100-megabytes-per-second fiber-optic LAN.
fibre channel	The ANSI standard that specifies high-speed serial communication between devices. Fibre channel is used as one of the bus architectures in SCSI-3.
fibre-distributed data interface	See FDDI.
file system	A hierarchical collection of files and directories.
file system specific directives	Archiver and releaser directives that follow global directives, are specific to a particular file system, and begin with `fs =`. File system specific directives apply until the next `fs =` directive line or until the end of file is encountered. If multiple directives affect a file system, the file system-specific directives override the global directives.
FTP	File Transfer Protocol. An internet protocol for transferring files between two hosts over a TCP/IP network.

G

global directives	Archiver and releaser directives that apply to all file systems and that appear before the first `fs =` line.
grace period	For disk quotas, this is the amount of time that can elapse during which a user is allowed to create files and/or allocate storage after a user reaches their soft limit.

H

hard limit	For disk quotas, a maximum limit on file system resources (blocks and inodes) that users cannot exceed.

I

indirect block	A disk block that contains a list of storage blocks. The Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems have up to three levels of indirect blocks. A first-level indirect block contains a list of blocks used for data storage. A second-level indirect block contains a list of first-level indirect blocks. A third-level indirect block contains a list of second-level indirect blocks.
inode	Index node. A data structure used by the file system to describe a file. An inode describes all the attributes associated with a file other than the name. The attributes include ownership, access, permission, size, and the file location on the disk system.
inode file	A special file (`.inodes`) on the file system that contains the inode structures for all files resident in the file system. All Sun QFS, Sun SAM-FS, and Sun SAM-QFS inodes are 512 bytes long. The inode file is a metadata file, which is separated from file data in the Sun QFS and Sun SAM-QFS file systems.

K

kernel	The central controlling program that provides basic system facilities. The UNIX kernel creates and manages processes, provides functions to access the file system, provides general security, and supplies communication facilities.

L

LAN	Local area network.
lease	In a Sun QFS shared file system, a lease grants a client host permission to perform an operation on a file for as long as the lease is valid. The metadata server issues leases to each client host. The leases are renewed as necessary to permit continued file operations.
library	See automated library.
library catalog	See catalog.
LUN	Logical unit number.

M

mcf	Master configuration file. The file that is read at initialization time that defines the relationships between the devices (the topology) within a Sun QFS, Sun SAM-FS, and Sun SAM-QFS environment.
media	Tape or optical disk cartridges.
media recycling	The process of recycling or reusing archive media with low use (that is, archive media with few active files).
metadata	Data about data. Metadata is the index information needed to locate the exact data position of a file on a disk. It consists of information about files, directories, access control lists, symbolic links, removable media, segmented files, and the indexes of segmented files. Metadata must be protected because if data is lost, the metadata that locates the data must be restored before the lost data can be retrieved.

metadata device A separate device (for example, a solid-state disk or mirrored device) upon which Sun QFS and Sun SAM-QFS file system metadata is stored. Separating file data from metadata can increase performance. In the `mcf` file, a metadata device is declared as an `mm` device within an `ma` file system.

mirror writing The process of maintaining two copies of a file on disjointed sets of disks to prevent loss from a single disk failure.

mount point The directory on which a file system is mounted.

multireader file system The Sun QFS multireader file system is a single-writer, multireader capability that enables you to specify a file system that can be mounted on multiple hosts. Multiple hosts can read the file system, but only one host can write to the file system. Multiple readers are specified with the `–o reader` option on the `mount`(1M) command. The single-writer host is specified with the `–o writer` option on the `mount`(1M) command. For more information on the `mount`(1M) command, see the `mount_samfs`(1M) man page.

N

name space The metadata portion of a collection of files that identifies the file, its attributes, and its storage locations.

nearline storage Removable media storage that requires robotic mounting before it can be accessed. Nearline storage is usually less expensive than online storage, but it incurs a somewhat longer access time.

network-attached automated library A library, such as those from StorageTek, ADIC/Grau, IBM, or Sony, that is controlled using a software package supplied by the vendor. The Sun SAM-FS and Sun SAM-QFS file systems interface with the vendor software using a Sun SAM-FS or Sun SAM-QFS media changer daemon designed specifically for the automated library.

NFS Network file system. A Sun distributed file system that provides transparent access to remote file systems on heterogeneous networks.

NIS The SunOS 4.0 (minimum) Network Information Service. A distributed network database containing key information about the systems and the users on the network. The NIS database is stored on the master server and all the slave servers.

O

offline storage	Storage that requires operator intervention for loading.
offsite storage	Storage that is remote from the server and is used for disaster recovery.
online storage	Storage that is immediately available (for example, disk cache storage).

P

partition	A portion of a device or a side of a magneto-optical cartridge.
preallocation	The process of reserving a contiguous amount of space on the disk cache for writing a file. This ensures that the space is contiguous. Preallocation can be performed only on zero-sized files. That is, the `setfa -l` command can be specified only for a file that is size zero. For more information, see the `setfa`(1) man page.
prioritizing preview requests	Assigning priority to archive and stage requests that cannot be immediately satisfied.
pseudo device	A software subsystem or driver with no associated hardware.

Q

quota	The amount of system resources that a user is allowed to consume. Quotas are not supported for removable media or disk archive resources.

R

RAID	Redundant array of inexpensive/independent disks. A disk technology that uses several independent disks to reliably store files. It can protect against data loss from a single disk failure, can provide a fault-tolerant disk environment, and can provide higher throughput than individual disks.

recycler A Sun SAM-FS and Sun SAM-QFS utility that reclaims space on cartridges that is occupied by expired archive copies.

release priority A method of calculating the release priority of a file within a file system by multiplying various weights by the corresponding file properties and then summing the results.

releaser A Sun SAM-FS and Sun SAM-QFS component that identifies archived files and releases their disk cache copies, thus making more disk cache space available. The releaser automatically regulates the amount of online disk storage to high and low thresholds.

remote procedure calls See RPC.

removable media file A special type of user file that can be accessed directly from where it resides on a removable media cartridge, such as magnetic tape or optical disk cartridge. also used for writing archive and stage file data.

robot The portion of an automated library that moves cartridges between storage slots and drives. Also called a transport.

round robin A data access method in which entire files are written to logical disks in a sequential fashion. When a single file is written to disk, the entire file is written to the first logical disk. The second file is written to the next logical disk, and so on. The size of each file determines the size of the I/O.

By default, Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems implement striped data access unless striped groups are present. Files are round-robined if round robin access is specified. If the file system contains mismatched striped groups, striping is not supported and round robin is forced.

Also see glossary entries for disk striping and striping.

RPC Remote procedure calls. The underlying data exchange mechanism used by NFS to implement custom network data servers.

S

samfsdump A program that creates a control structure dump and copies all the control structure information for a given group of files. It is analogous to the UNIX `tar`(1) utility, but it does not generally copy file data.

samfsrestore A program that restores inode and directory information from a control structure dump.

SCSI	Small Computer System Interface. An electrical communication specification commonly used for peripheral devices such as disk and tape drives and automated libraries.
small computer system interface	See SCSI.
soft limit	For disk quotas, a threshold limit on file system resources (blocks and inodes) that you can temporarily exceed. Exceeding the soft limit starts a timer. When you exceed the soft limit for the specified time (default is one week), no further system resources can be allocated until you reduce file system use to a level below the soft limit.
staging	The process of copying a nearline or offline file from archive storage back to online storage.
storage family set	A set of disks that are collectively represented by a single disk family device.
storage slots	Locations inside an automated library in which cartridges are stored when not being used in a drive. If the library is direct-attached, the contents of the storage slots are kept in the automated library's catalog.
stripe size	The number of disk allocation units (DAUs) to allocate before moving to the next device of a stripe. If `stripe=0`, the file system uses round-robin access, not striped access.
striped group	A collection of devices within a Sun QFS or Sun SAM-QFS file system and defined in the `mcf` file as one (usually two) or more `g`*XXX* devices. Striped groups are treated as one logical device and are always striped with a size equal to the disk allocation unit (DAU). You can specify up to 128 striped groups within a file system, but you can specify no more than 252 total devices.
striping	A data access method in which files are simultaneously written to logical disks in an interlaced fashion. All Sun QFS, Sun SAM-FS, and Sun SAM-QFS file systems enable you to declare either striped or round robin access for each individual file system. The Sun QFS and Sun SAM-QFS file systems enable you to declare striped groups within each file system. Also see the glossary entry for round robin.
Sun SAM-FS	The Sun Storage and Archive Manager File System. The Sun SAM-FS software controls the access to all files stored and all devices configured in the master configuration file (`mcf`).
Sun SAM-QFS	The Sun SAM-QFS software combines the Sun Storage and Archive Manager with the Sun QFS file system. Sun SAM-QFS offers a high-speed, standard UNIX file system interface to users and administrators in conjunction with the storage and archive management utilities. It uses many of the commands available in the Sun SAM-FS command set as well as standard UNIX file system commands.

Sun SAM-Remote client	A Sun SAM-Remote client is a Sun SAM-FS or Sun SAM-QFS system that establishes a Sun SAM-Remote client daemon that contains a number of pseudodevices. It might or might not have its own library devices. The client depends on a Sun SAM-Remote server for archive media for one or more archive copies.
Sun SAM-Remote server	The Sun SAM-Remote server is both a full-capacity Sun SAM-FS or Sun SAM-QFS storage management server and a Sun SAM-Remote server daemon that defines libraries to be shared among Sun SAM-Remote clients.
superblock	A data structure in the file system that defines the basic parameters of the file system. It is written to all partitions in the storage family set and identifies the partition's membership in the set.

T

tar	Tape archive. A standard file/data recording format used by the Sun SAM-FS and Sun SAM-QFS software for archive images.
TCP/IP	Transmission Control Protocol/Internet Protocol. The internet protocols responsible for host-to-host addressing and routing, packet delivery (IP), and reliable delivery of data between application points (TCP).
thresholds	A mechanism for defining the desirable available storage window for online storage. Thresholds set the storage goals for the releaser. Also see disk space thresholds.
timer	Quota software that keeps track of the time elapsed between a user reaching a soft limit and a hard limit being imposed on the user.

V

volume	A named area on a cartridge for sharing data. A cartridge has one or more volumes. Double-sided cartridges have two volumes, one on each side.
volume overflow	A capability that enables the system to span a single file over multiple volumes. Volume overflow is useful for sites using very large files that exceed the capacity of their individual cartridges.

VSN	Volume serial name. If you are archiving to removable media cartridges, the VSN is a logical identifier for magnetic tape and optical disk that is written in the volume label. If you are archiving to disk cache, this is the unique name for the disk archive set.

W

WORM	Write once read many. A storage classification for media that can be written only once but read many times.

Index