StorageTek Storage Archive Manager

and StorageTek QFS Software Maintenance and Administration Guide

Release 5.4

E42064-02

January 2015

StorageTek Storage Archive Manager and StorageTek QFS Software Maintenance and Administration Guide, Release 5.4

E42064-02

Primary Author: Robert Craig Johnson

Contents

6 Backing Up the Configuration and File Systems

7 Migrating to New Storage Media

A Glossary of Equipment Types

B Media Status Flags

C Mount Options in a Shared File System

D Configuration Directives for Archiving

E Product Accessibility Features

Glossary

Index

Preface

This document addresses the needs of system administrators, storage and network administrators, and service engineers who are tasked with monitoring, managing, and maintaining SAM-QFS file systems and archiving solutions.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Prerequisites for Using this Document

This document assumes that you are already familiar with SAM-QFS Solaris operating system, storage, and network administration. Please refer to the Solaris documentation and man pages and to storage hardware documentation for information on relevant tasks, commands, and procedures.

Conventions

The following textual conventions are used in this document:

- *Italic* type represents book titles and emphasis.
- `Monospace` type represents commands and text displayed in a terminal window and the contents of configuration files, shell scripts, and source code files.
- **`Monospace bold`** type represents user inputs and significant changes to commandline output, terminal displays, or file contents. It may also be used to emphasize particularly relevant parts of a file or display.
- ***`Monospace bold oblique`*** type represents variable inputs and outputs in a terminal display or file.
- *`Monospace oblique`* type represents other variables in a terminal display or file.
- `. . .` (three-dot ellipsis marks) represent file contents or command output that is not relevant to the example and has thus been omitted for brevity or clarity.

- `\` (a backslash) at the end of a line in examples escapes the line break so that the following line is part of the same command.
- `[ - ]` (brackets surrounding values separated by a hyphen) delimit value ranges.
- `[ ]` (brackets) in command syntax descriptions indicate optional parameters.
- `root@solaris:~#` and `[`*`hostname`*`]:root@solaris:~#` represent Solaris command shell prompts.
- `[root@Linux ~]#` represents Linux command shell prompts.

Available Documentation

The *StorageTek Storage Archive Manager and StorageTek QFS Software Maintenance and Administration Guide* is part of the multivolume *SAM-QFS Customer Documentation Library*), available from `docs.oracle.com`.

For Solaris operating system documentation, see the *Oracle Solaris 11.1 Information Library*, also available from `docs.oracle.com`.

1

Maintaining SAM-QFS Solutions

Throughout the planning and deployment process, we have stressed that QFS file systems and Storage Archive Manager software are designed to hide the complexities of performance optimization, data protection, and archiving behind a simple, UNIX file-system interface. Users, applications, and, for the most part, administrators should be able to treat a fully optimized, SAM-QFS archiving system implemented on a mix of disk arrays and tape libraries as if it were an ordinary UFS file system on a single, local disk. Once installed and configured, SAM-QFS software should automatically manage your data and storage resources in the most efficient and reliable way possible, with minimal human intervention.

That said, as with any UNIX file system, there are still monitoring and periodic management tasks that need to be carried out. This book outlines these activities.

2

Monitoring File System Operations

Correctly configured SAM-QFS file systems need little routine administrative intervention. You do, however, need to monitor the system for signs of abnormalities. In general, you monitor two things: availability and utilization.

Availability is conceptually straightforward and easy to monitor. When key components such as a host system, network interface, file system, or storage subsystem become unavailable, core functionality is abruptly lost or degraded, and alerts are displayed in the administrative interfaces and logs.

Utilization issues are more subtle and require more judgment on your part. When usage is excessive, lack of resources, such as storage media, can stop the archiving process as effectively as a component failure, even though the system is functioning normally. Utilization levels that presage imminent trouble in one organization may be perfectly consistent with years of trouble-free operation in another. So recognizing trends and rates is crucial when you are monitoring utilization. A resource that is 80% used is fine if utilization grows 1% per year but a crisis if growth exceeds 1% every week.

SAM-QFS provides three monitoring interfaces:

- SAM-QFS Manager
- `samu`
- Log and Trace Files

Each has its strengths, depending on your working style and habits

SAM-QFS Manager

SAM-QFS Manager is a browser-based graphical user interface that lets administrators monitor and control all aspects of file-system operations. The SAM-QFS Manager browser interface pages are divided into three sections:

- Banner
- Navigation tree
- Content pane

The banner displays the name of the application and status information, such as the last time data was updated in the browser interface, the name and role of the user that is currently logged in, the name of the management station that is hosting the SAM-QFS Manager software, and the current number and type of unacknowledged faults.

The navigation tree on the left side of the interface contains the server menu and a hierarchical listing of the available displays. Clicking a link in the navigation tree causes the corresponding display to appear in the content pane.

The Monitoring node of the navigation tree is your principle monitoring resource. You can list and sort all faults detected on monitored file systems and equipment. You can configure automatic email alerts. You can list all jobs that are currently running. The Monitoring node also contains a link to the monitoring Dashboard, a popup window that alerts you to problems and provides quick links to summary information on the following areas of concern:

- Daemons
- File Systems
- Archive Media Utilization
- Tape Libraries
- Library Drives
- Volume Pending Load Request
- Unusable Volumes
- Archiving Copy Queue
- Staging Queue

The Metrics and Reports node of the navigation tree provides a comprehensive range of status and utilization reports as well as a link to System Details. The System Details page lets you quickly review the SAM-QFS configuration and provides quick, convenient access to the log and trace files.

A comprehensive help system fully documents the use of the SAM-QFS Manager.

samu

The `samu` operator utility is a text-based, menu-driven configuration and management interface that you launch from the command line. It is a convenient, lightweight way to monitor SAM-QFS devices, file system activity, and error messages.

The `samu` utility is in some respects similar to the UNIX `vi` editor. You select displays, set display options, navigate within and between displays, enter commands, refresh displays, and quit the utility using similar control key sequences. The last line of each display window displays error messages. Displays refresh automatically unless an error occurs, in which case the display halts until the operator takes further action. When desired, you can take snapshots of display windows for later reference.

The `h` command opens help screens that list all of the keyboard short cuts, commands. and parameters. You can also consult the `samu` man page and the *StorageTek Storage Archive Manager and StorageTek QFS* `samu` *Command Reference* (*SAM-QFS Customer Documentation Library*, `docs.oracle.com`) for additional information.

The following is a typical `samu` monitoring display:

```
Archiver status                 samu      5.4     12:24:10 Mar 19 2014

sam-archiverd: Waiting for resources

sam-arfind:  samma1 mounted at /samma1
Files waiting to start 0    schedule 70,524    archiving 0
Monitoring file system activity.
```

```
sam-arfind:  DISKVOL1 mounted at /diskvols/DISKVOL1
Files waiting to start 0    schedule 0    archiving 0
Monitoring file system activity.

                              samu on samqfshost1
```

Log and Trace Files

The SAM-QFS software performs comprehensive logging and, when configured, tracing. So you may wish to monitor the following files, particularly when problems arise:

- `/var/adm/messages`
- `/var/adm/sam-log`
- `/var/opt/SUNWsamfs/trace/` (holds trace files for the daemons and processes)
- `/var/opt/SUNWsamfs/devlog/` (holds logs for the devices configured in the `/etc/opt/SUNWsamfs/mcf` file)
- `/var/opt/SUNWsamfs/archiver.log`
- `/var/opt/SUNWsamfs/stager.log`
- `/var/opt/SUNWsamfs/recycler.log`
- additional archiving logs specific to file systems (if configured).

For information on configuring logging and tracing, see the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide* (*SAM-QFS Customer Documentation Library*, `docs.oracle.com`).

3

Managing SAM-QFS File Systems

This chapter covers file-system maintenance and reconfiguration tasks. The first section, Managing SAM-QFS File Systems, addresses maintenance of all SAM-QFS file systems, archiving and non-archiving, shared and unshared/standalone. The second section, Managing SAM-QFS Shared File Systems, deals with special considerations that affect shared file systems.

Managing SAM-QFS File Systems

This section outlines the following tasks:

- Administering File System Quotas
- Controlling Archiving and Staging Operations
- Renaming File Systems
- Repairing File Systems
- Adding Devices to File Systems
- Removing Data Devices from a File System

Administering File System Quotas

Set file system quotas to control the online and total storage space that a given user or collection of users can consume within the file system. You can set quotas by user ID, group ID, or by an administrator-defined *admin set ID* that groups users by common characteristics, such as participation in a particular project that involves users from several groups and spans multiple directories and files.

You enable quotas by mounting a file system with the `quota` mount option (set by default) and disable quotas by mounting it with the `noquota` mount option. You define the quotas by placing one or more *quota files* in the file system root directory: `.quota_u`, `.quota_g`, and `.quota_a`, which set quotas for users, groups, and admin sets, respectively. The first record in each file, record `0`, sets the default values. Subsequent records set values specific to particular users, groups, or admin sets.

Quotas allocate usable file-system space, not simply storage space. They thus set upper limits on both the number of 512-byte blocks allocated on the media and the number of inodes allocated in the file system. The block count measures the storage space per se. The inode count measures the resources available for accessing that storage. A single file that used a great many blocks of storage space but only one inode might thus take up the same amount of file-system space as a great many empty, zero-length files that take up many inodes and no blocks.

Each quota can include both *soft* and *hard limits*. A hard limit defines the maximum file-system resources that all of a given owner's files can use temporarily. A soft limit defines the maximum file-system resources that the owner's files can use indefinitely. Resource usage can grow to amounts that lie between the soft and hard limits only for brief intervals, as defined by the *grace period* in the quota.

This section describes the following administrative tasks:

- Characterize the Storage Requirements of Users, Groups, and Organizational Units
- Create Admin Sets for Projects and for Directories Used by Multiple Groups
- Configure a New File System to Use Quotas
- Configure an Existing File System to Use Quotas
- Set Quotas for Groups, Projects, Directories, and Users
- Repair Inconsistent Quotas
- Check Quotas
- Temporarily Extend or Cancel Grace Periods
- Stop New Resource Allocations

Characterize the Storage Requirements of Users, Groups, and Organizational Units

To set sustainable quotas, you have to set limits that accommodate user requirements and you have to set them in a way that is both manageable and scalable. So, before setting quotas, estimate the storage requirements of your users. To keep the process manageable, start by classifying user requirements as broadly as possible, so that you address the greatest number of requirements with the smallest amount of administrative effort. You can then specially assess a small number of user requirements that do not fit in to the broader categories. The results will provide the broad outlines of the quotas and types of limits that you will set.

The approach outlined below starts by identifying the file-system requirements of access-control groups, since most organizations already define these groups. Then it defines special sets of users whose needs do not align well with those of the standard groups. Then and only then does it begin to address any requirements that are unique to individual users. Proceed as follows:

1. Since your existing access-control groups already gather together users with similar resource requirements, start by defining the average storage requirements of any groups that will use the file system. Estimate both the *average amount of storage space* used (in 512-kilobyte blocks) and the *average number of files* stored, which is equivalent to the average number of inodes used.

   Since group members typically have similar organizational roles and work responsibilities, they frequently need access to the same directories and files and generally make similar demands on storage. In the example, we identify three groups that will use the file system `/samqfs1`: `dev` (Product Development), `cit` (Corporate Information Technologies), and `pgmt` (Program Management). We list the groups, the number of members in each, and their average individual and group requirements in a simple spreadsheet:

Group	Users	Average Blocks Per User	Average Files Per User	Average Blocks/Group	Average Files/Group
dev	30	67108864	500	2013265920	15000

Group	Users	Average Blocks Per User	Average Files Per User	Average Blocks/Group	Average Files/Group
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
Total Blocks/ Files (Average)					

2. Next, carry out the same calculations for the *maximum amount of storage space* and the *maximum number of files* that group members will store at any given time. Record the results.

   In the example, we record the results in a new spreadsheet:

Group	Users	Maximum Blocks Per User	Maximum Files Per User	Maximum Blocks/Group	Maximum Files/Group
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
Total Blocks/ Files (Maximum)					

3. Now identify any sets of users that belong to different groups but share distinct storage requirements that cannot be not addressed on the basis of group membership. Make the same estimates and carry out the same calculations for each identified organization as you did for each access-control group.

   In the example, we identify two company projects that will require storage allocations, code named `portal` and `lockbox`. Members of the engineering, marketing, compliance, test, and documentation groups will work together on these projects and will use the same directories and many of the same files. So we add them to our requirements spreadsheets:

Group	Users	Average Blocks Per User	Average Files Per User	Average Blocks/Group	Average Files/Group
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000
Total Blocks/ Files (Average)					

Group	Users	Maximum Blocks Per User	Maximum Files Per User	Maximum Blocks/Group	Maximum Files/Group
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
portal	10	37748736	700	377487360	7000
lockbox	12	45613056	600	547356672	7200

Group	Users	Maximum Blocks Per User	Maximum Files Per User	Maximum Blocks/Group	Maximum Files/Group
Total Blocks/ Files (Maximum)					

4. Now identify any individual users whose requirements have not yet been addressed. Make the same estimates and carry out the same calculations for each user as you did for each access-control group and non-group organization.

   Where possible, address user requirements collectively, so that polices are uniform and management overhead is at a minimum. However, when individual requirements are unique, you need to address them individually. In the example, we identify jr23547 in the pgmt group as a user whose special responsibilities require special storage allocations. So we add him to our requirements spreadsheets:

Group	Users Per Set	Average Blocks Per User	Average Files Per User	Average Blocks	Average Files
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000
jr23547	1	10485760	600	10485760	600
Total Blocks/ Files (Average)					

Group	Users	Maximum Blocks Per User	Maximum Files Per User	Maximum Blocks/Group	Maximum Files/Group
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
portal	10	37748736	700	377487360	7000
lockbox	12	45613056	600	547356672	7200
jr23547	1	100663296	2000	100663296	2000
Total Blocks/ Files (Maximum)					

5. Finally, calculate the average and maximum blocks and files that all users require.

Group	Users	Average Blocks Per User	Average Files Per User	Average Blocks/Group	Average Files/Group
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000

Group	Users	Average Blocks Per User	Average Files Per User	Average Blocks/Group	Average Files/Group
`jr23547`	1	10485760	600	10485760	600
Total Blocks/ Files (Average)				2998927360	27550

Group	Users	Maximum Blocks Per User	Maximum Files Per User	Maximum Blocks/Group	Maximum Files/Group
`dev`	30	100663296	1000	3019898880	30000
`cit`	15	15728640	100	235929600	1500
`pmgt`	6	31457280	400	188743680	2400
`portal`	10	37748736	700	377487360	7000
`lockbox`	12	45613056	600	547356672	7200
`jr23547`	1	100663296	2000	100663296	2000
Total Blocks/ Files (Average)				4470079488	50100

6. If you need to administer project-based quotas or other quotas that cannot be defined by access-control group and user IDs, Create Admin Sets for Projects and for Directories Used by Multiple Groups.

7. Otherwise, if you are setting quotas on an empty, newly created file system, go to Configure a New File System to Use Quotas.

8. Or if you are setting quotas on a file system that already holds files, go to "Configure an Existing File System to Use Quotas" on page 3-8.

Create Admin Sets for Projects and for Directories Used by Multiple Groups

An admin set is a directory hierarchy or an individual directory or file that is identified for quota purposes by an *admin set ID*. All files created with a specified admin set ID or in a directory with a specified admin set ID have the same quotas, regardless of the user or group IDs that actually own the files. To define admin sets, proceed as follows:

1. Log in to the file-system server as `root`.

   In the example, the server is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. If you are using an admin set to configure storage quotas for a new project or team, create a new directory somewhere within the file system for this project or team.

   In the example, we create the directory in the `/samqfs1` file system and name it `portalproject/` for the project of the same name

   ```
   [server1]root@solaris:~# mkdir /samqfs1/portalproject
   ```

3. Assign an admin set ID to the directory or file on which you need to set a quota. Use the command `samchaid [-fhR]` *`admin-set-id directory-or-file-name`*, where:

   - `-f` forces the assignment and does not report errors.
   - `-h` assigns the admin set ID to symbolic links. Without this option, the group of the file referenced by the symbolic link is changed.

- `-R` assigns the admin set ID recursively to subdirectories and files.
- *`admin-set-id`* is a unique integer value.
- *`directory-or-file-name`* is the name of the directory or file to which you are assigning the admin set ID.

In the example, we assign the admin ID **1** to the directory `/samqfs1/portalproject/` and all of its subdirectories and files.

```
[server1]root@solaris:~# samchaid -R 1 /samqfs1/portalproject/
```

4. You can check the assignment, if desired. Use the `command sls -D` *`directory-path`*, where the `-D` specifies a detailed SAM-QFS directory listing for files and directories in *`directory-path`*:

```
[server1]root@solaris:~# sls -D /samqfs1/
/portalproject:
  mode: drwxr-xr-x    links:   2  owner: root        group: root
  length:      4096  admin id:    1  inode:     1047.1
  project: user.root(1)
  access:      Feb 24 12:49  modification: Feb 24 12:44
  changed:     Feb 24 12:49  attributes:   Feb 24 12:44
  creation:    Feb 24 12:44  residence:    Feb 24 12:44
```

5. If you are setting quotas on an empty, newly created file system, go to Configure a New File System to Use Quotas.

6. If you are setting quotas on a file system that already holds files, go to "Configure an Existing File System to Use Quotas" on page 3-8.

Configure a New File System to Use Quotas

Use this procedure if you are creating a new file system and no files currently reside in the file system.

1. Log in to the file-system server as `root`.

   In the example, the server is named `server2`:

```
[server2]root@solaris:~#
```

2. If the new file system is not currently mounted, mount it before proceeding.

3. If you have to set up quotas for groups, create a group quota file, `.quota_g`, in the file-system root directory. Use the Solaris command `dd if=/dev/zero of=`*`mountpoint`*`/.quota_g bs=4096 count=`*`number-blocks`*, where:

   - `if=/dev/zero` specifies null characters from the UNIX special file `/dev/zero` as the input.
   - `of=`*`mountpoint`*`/.quota_g` specifies the output file, where *`mountpoint`* is the mount point directory for the file system.
   - `bs=4096` sets the block size for the write to `4096` bytes.
   - `count=`*`number-blocks`* specifies the number of blocks to write. This value depends on the number of records that the file will hold. There is one 128-byte record for each specified quota, so one block can accommodate 32 records.

   In the example, we create the group quota file for the file system `newsamfs` mounted at `/newsamfs`. During the requirements-gathering phase, we identified three groups that need quotas on the file system, `dev`, `cit`, and `pgmt`. We do not anticipate adding any other group quotas, so we size the file at one block:

```
[server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_g bs=4096 count=1
```

4. If you have to set up quotas for admin sets, create an admin set quota file, `.quota_a`, in the file-system root directory. Use the Solaris command `dd if=/dev/zero of=`*`mountpoint`*`/.quota_a bs=4096`, where:

   - *`mountpoint`* is the mount point directory for the file system.
   - `.quota_a` is the name of the output file.
   - `4096` is the block size for the write in bytes.
   - *`number-blocks`* is the number of blocks to write.

   In the example, we create the admin sets quota file for the file system `newsamfs` mounted at `/newsamfs`. During the requirements-gathering phase, we identified two projects that need quotas on the file system, `portal` (admin set ID `1`) and `lockbox` (admin set ID `2`). We do not anticipate adding any other admin set quotas, so we size the file at one block:

```
[server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_a bs=4096 count=1
```

5. If you have to set up quotas for users, create a user quota file, `.quota_u`, in the file-system root directory. Use the Solaris command `dd if=/dev/zero of=`*`mountpoint`*`/.quota_u bs=4096 count=`*`number-blocks`*, where:

   - *`mountpoint`* is the mount point directory for the file system.
   - `.quota_u` is the name of the output file.
   - `4096` is the block size for the write in bytes.
   - *`number-blocks`* is the number of blocks to write.

   In the example, we create the user quota file for the file system `newsamfs` mounted at `/newsamfs`. During the requirements-gathering phase, we identified one user that needed specific quotas on the file system, `jr23547`. We do not anticipate adding any other individual user quotas, so we size the file at one block:

```
[server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_u bs=4096 count=1
```

6. Unmount the file system.

   You must unmount the file system before you can remount it and enable the quota files.

```
[server2]root@solaris:~# umount /newsamfs
```

7. Perform a file system check.

```
[server2]root@solaris:~# samfsck -F newsamfs
```

8. Remount the file system.

   The system enables quotas when it detects one or more quota files in the root directory of the file system.

   You do not need to include the `quota` mount option in the `/etc/vfstab` or `samfs.cmd` file, because file systems are mounted with quotas enabled by default.

```
[server2]root@solaris:~# mount /newsamfs
```

9. Next, set or update quotas as needed. See "Set Quotas for Groups, Projects, Directories, and Users" on page 3-10.

Configure an Existing File System to Use Quotas

Use this procedure if you are creating quotas for a file system that already populated holds files.

1. Log in to the file-system server as `root`.

   In the example, the server is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Open the `/etc/vfstab` file in a text editor, and make sure that the `noquota` mount option has *not* been set.

   In the example, we open the file in the `vi` text editor. The `noquota` mount option has been set:

   ```
   [server1]root@solaris:~# vi /etc/vfstab
   #File
   #Device        Device   Mount              System  fsck  Mount    Mount
   #to Mount      to fsck  Point              Type    Pass  at Boot  Options
   #------------  -------  ---------------    ------  ----  -------  ------------
   /devices       -        /devices           devfs   -     no       -
   /proc          -        /proc              proc    -     no       -
   ...
   samqfs1        -        /samqfs1           samfs   -     no       noquota
   ```

3. If the `noquota` mount option has been set in the `/etc/vfstab` file, delete it and save the file.

   ```
   [server1]root@solaris:~# vi /etc/vfstab
   #File
   #Device        Device   Mount              System  fsck  Mount    Mount
   #to Mount      to fsck  Point              Type    Pass  at Boot  Options
   #------------  -------  ---------------    ------  ----  -------  ------------
   /devices       -        /devices           devfs   -     no       -
   /proc          -        /proc              proc    -     no       -
   ...
   samqfs1        -        /samqfs1           samfs   -     no       -
   :wq
   [server1]root@solaris:~#
   ```

4. Open the `/etc/opt/SUNWsamfs/samfs.cmd` file in a text editor, and make sure that the `noquota` mount option has not been set.

   In the example, we open the file in the `vi` text editor. The `noquota` mount option has not been set:

   ```
   [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samfs.cmd
   # These are the defaults.  To change the default behavior, uncomment the
   # appropriate line (remove the '#' character from the beginning of the line)
   # and change the value.
   #
   #inodes = 0
   #fs = samqfs1
   #  forcedirectio (default no forcedirectio)
   #  high = 80
   #  low = 70
   #  weight_size = 1.
   #  weight_age = 1.
   #  readahead = 128
   ...
   #  dio_wr_ill_min = 0
   ```

```
#  dio_wr_consec = 3
#  qwrite        (ma filesystem, default no qwrite)
#  shared_writer (ma filesystem, default no shared_writer)
#  shared_reader (ma filesystem, default no shared_reader)
```

5. If the noquota mount option has been set in the /etc/opt/SUNWsamfs/samfs.cmd file, delete it and save the file.

6. If you deleted the noquota mount option from the /etc/vfstab file and/or the /etc/opt/SUNWsamfs/samfs.cmd file, unmount the file system.

   When you remove a noquota mount option, you must unmount the file system so that you can remount it with quotas enabled.

   ```
   [server1]root@solaris:~# umount /samqfs1
   ```

7. If the file system is not currently mounted, mount it now.

   The file system must be mounted before you can enable quotas.

   ```
   [server1]root@solaris:~# mount /samqfs1
   ```

8. Change to the root directory of the file system and check for any existing quota files. Use the Solaris command ls -a and look for the files .quota_g, .quota_a, and/or .quota_u.

   In the example, no quota files currently exist.

   ```
   [server1]root@solaris:~# cd /samqfs1
   [server1]root@solaris:~# ls -a /samqfs1
   .              .archive        .fuid          .stage          portalproject
   ..             .domain         .inodes        lost+found
   ```

9. If quota files exist, do not modify them.

10. If you have to set up quotas for groups and the group quota file, .quota_g, does not already exist in the file-system root directory, create the file now. Use the Solaris command dd if=/dev/zero of=*mountpoint*/.quota_g bs=4096 count=*number-blocks*, where:

    - if=/dev/zero specifies null characters from the UNIX special file /dev/zeroas the input.
    - of=*mountpoint*/.quota_g specifies the output file, where *mountpoint* is the mount point directory for the file system.
    - bs=4096 sets the block size for the write to 4096 bytes.
    - count=*number-blocks* specifies the number of blocks to write. This value depends on the number of records that the file will hold. There is one 128-byte record for each specified quota, so one block can accommodate 32 records.

    In the example, we create the group quota file for the file system /samqfs1 mounted at /samqfs1. During the requirements-gathering phase, we identified three groups that need quotas on the file system, dev, cit, and pgmt. We do not anticipate adding any other group quotas, so we size the file at one block:

    ```
    [server1]root@solaris:~# dd if=/dev/zero of=/samqfs1/.quota_g bs=4096 count=1
    ```

11. If you have to set up quotas for admin sets and the admin sets quota file, .quota_a, does not already exist in the file-system root directory, create the file now. Use the Solaris command dd if=/dev/zero of=*mountpoint*/.quota_a bs=4096 count=*number-blocks*, where:

- *mountpoint* is the mount point directory for the file system.
- .quota_a is the name of the output file.
- 4096 is the block size for the write in bytes.
- *number-blocks* is the number of blocks to write.

In the example, we create the admin sets quota file for the file system /samqfs1 mounted at /samqfs1. During the requirements-gathering phase, we identified two projects that need quotas on the file system, portal (admin set ID 1) and lockbox (admin set ID 2). We do not anticipate adding any other admin set quotas, so we size the file at one block:

```
[server1]root@solaris:~# dd if=/dev/zero of=/samqfs1/.quota_a bs=4096 count=1
```

12. If you have to set up quotas for users and the user quota file, .quota_u, does not already exist in the file-system root directory, create the file now. Use the Solaris command dd if=/dev/zero of=*mountpoint*/.quota_u bs=4096 count=*number-blocks*, where:

- *mountpoint* is the mount point directory for the file system.
- .quota_u is the name of the output file.
- 4096 is the block size for the write in bytes.
- *number-blocks* is the number of blocks to write.

In the example, we create the user quota file for the file system /samqfs1 mounted at /samqfs1. During the requirements-gathering phase, we identified one user that needed specific quotas on the file system, jr23547. We do not anticipate adding any other individual user quotas, so we size the file at one block:

```
[server1]root@solaris:~# dd if=/dev/zero of=/samqfs1/.quota_u bs=4096 count=1
```

13. Unmount the file system.

You must unmount the file system before you can remount it and enable the quota files.

```
[server1]root@solaris:~# umount /samqfs1
```

14. Perform a file system check.

```
[server1]root@solaris:~# samfsck -F /samqfs1
```

15. Remount the file system.

The system enables quotas when it detects one or more quota files in the root directory of the file system.

You do not need to include the quota mount option in the /etc/vfstab or samfs.cmd file, because file systems are mounted with quotas enabled by default.

```
[server1]root@solaris:~# mount /samqfs1
```

16. Next, Set Quotas for Groups, Projects, Directories, and Users.

Set Quotas for Groups, Projects, Directories, and Users

You set new quotas and adjust existing ones using the samquota command. Follow the procedure below:

1. Once you have characterized storage requirements, decide on the appropriate quotas for each group, user, and non-group organization. Consider the following factors and make adjustments as necessary:

   - the size of the file system compared to the average and maximum number of blocks that all users require
   - the number of inodes in the file system compared to the average and maximum number of inodes that all users require
   - the numbers and types of users that are likely to be close to their maximum requirement at any given time

2. Log in to the file-system server as `root`.

   In the example, the server is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

3. Set limits for each group that requires them. Use the command `samquota -b` *`number-blocks`*`:`*`type`*`[:`*`scope`*`] -f` *`number-files`*`:`*`type`*`[:`*`scope`*`] -t` *`interval`*`[:`*`scope`*`] -G` *`groupID`* `[`*`directory-or-file`*`]`, where:

   - `-b` *`number-blocks`* sets the maximum number of 512-kilobyte blocks that can be stored in the file system to*`number-blocks`*, an integer (see the `samquota` man page for alternative ways of specifying size). A value of `0` (zero) specifies an unlimited number of blocks.
   - `:` is a field separator.
   - *`type`* specifies the kind of limit, either `h` for a hard limit or `s` for a soft limit.
   - *`scope`* (optional) identifies the type of storage that is subject to the limit. It can be either `o` for online (disk-cache) storage only or `t` for total storage, both online and in the offline archive (the default).
   - `-f` *`number-files`* sets the maximum number of files that can be stored in the file system to *`number-files`*, an integer. A value of `0` (zero) specifies an unlimited number of files.
   - `-t` *`number-seconds`* sets the *grace period*, the time during which soft limits can be exceeded, to *`number-seconds`*, an integer representing a number of seconds (see the `samquota` man page for alternative ways of specifying time).
   - `-G` *`groupID`* specifies a group name or integer identifier for the group. A value of `0` (zero) sets the default limits for all groups.
   - *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file on which you need to set a quota.

   In the example, we use our estimates from the requirements-gathering phase to set both hard and soft limits on both the amount of storage space in the `/samqfs1` file system that group `dev` can use and the numbers of files that it can store. We set the grace period to `4320` seconds (twelve hours) for online storage only:

   ```
   [server1]root@solaris:~# samquota -b 3019898880:h:t -f 30000:h:t -t 4320:o \
   -G dev /samqfs1
   [server1]root@solaris:~# samquota -b 2013265920:s:t -f 15000:s:t -t 4320 \
   -G dev /samqfs1
   ```

4. Set limits for each admin set that requires them. Use the command `samquota -b` *`number-blocks`*`:`*`type`*`[:`*`scope`*`] -f` *`number-files`*`:`*`type`*`[:`*`scope`*`] -t` *`interval`*`[:`*`scope`*`] -A` *`adminsetID`* `[`*`directory-or-file`*`]`, where `-A` *`adminsetID`* is the integer value that uniquely identifies the admin set.

Setting *adminsetID* to 0 (zero) sets the default limits for all admin sets.

In the example, we use our estimates from the requirements-gathering phase to set both hard and soft limits on both the amount of storage space in the /samqfs1 file system that the portal project (admin set ID 1) can use and the numbers of files that it can store. We set the grace period to 4320 seconds (twelve hours) for total storage used (the default scope):

```
[server1]root@solaris:~# samquota -b 377487360:h:t -f 7000:h:t -t 4320 \
-A 1 /samqfs1
[server1]root@solaris:~# samquota -b 314572800:s:t -f 4000:s:t -A 1 /samqfs1
```

5. Set limits for each individual user that requires them. Use the command samquota -b *number-blocks*:*type*[:*scope*] -f *number-files*:*type*[:*scope*] -t *interval*[:*scope*] -U *userID* [*directory-or-file*], where -U *userID* is a user name or integer identifier for the user.

   Setting *userID* to 0 (zero) sets the default limits for all users.

   In the example, we use our estimates from the requirements-gathering phase to set both hard and soft limits on both the amount of storage space in the /samqfs1 file system that user jr23547 can use and the numbers of files that jr23547 can store. We set the grace period to 120960 seconds (two weeks) for total storage used (the default scope):

```
[server1]root@solaris:~# samquota -b 100663296:h:t -f 2000:h:t -t 4320 \
-U jr23547 /samqfs1
[server1]root@solaris:~# samquota -b 10485760:s:t -f 600:s:t -t 4320 \
-U jr23547 /samqfs1
```

6. Stop here.

Repair Inconsistent Quotas

If you mount a SAM-QFS file system with the noquota mount option when there are quota files in the root directory, quota records become inconsistent with actual usage as blocks or files are allocated or freed. In this situation, proceed as follows:

1. Log in to the file-system server as root.

   In the example, the server is named server1:

```
[server1]root@solaris:~#
```

2. Unmount the affected file system.

   In the example, we unmount file system samfs2:

```
[server1]root@solaris:~# umount samfs2
[server1]root@solaris:~#
```

3. Open the /etc/vfstab file in a text editor, and make sure that the noquota mount option has not been set.

   In the example, we open the file in the vi text editor. The noquota mount option has been set:

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount            System  fsck  Mount    Mount
#to Mount      to fsck  Point            Type    Pass  at Boot  Options
#------------  -------  ---------------  ------  ----  -------  ------------
/devices       -        /devices         devfs   -     no       -
```

```
/proc          -        /proc             proc    -     no       -
...
samfs2         -        /samfs2           samfs   -     no       noquota
```

4. If the `noquota` mount option has been set in the `/etc/vfstab` file, delete it and save the file.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
samfs2         -        /samfs2           samfs   -     no       -
:wq
[server1]root@solaris:~#
```

5. Open the `/etc/opt/SUNWsamfs/samfs.cmd` file in a text editor, and make sure that the `noquota` mount option has not been set.

   In the example, we open the file in the `vi` text editor. The `noquota` mount option has not been set:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samfs.cmd
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
#
#inodes = 0
#fs = samqfs1
#  forcedirectio (default no forcedirectio)
#  high = 80
#  low = 70
#  weight_size = 1.
#  weight_age = 1.
#  readahead = 128
...
#  dio_wr_ill_min = 0
#  dio_wr_consec = 3
#  qwrite        (ma filesystem, default no qwrite)
#  shared_writer (ma filesystem, default no shared_writer)
#  shared_reader (ma filesystem, default no shared_reader)
```

6. If the `noquota` mount option has been set in the `/etc/opt/SUNWsamfs/samfs.cmd` file, delete it and save the file.

7. Repair the inconsistent quota records. Use the command `samfsck -F` *`family-set-name`*, where *`family-set-name`* is the family set name for the file system in the `/etc/opt/SUNWsamfs/mcf` file.

```
[server1]root@solaris:~# samfsck -F samfs2
```

8. Remount the file system.

   The system enables quotas when it detects one or more quota files in the root directory of the file system.

   You do not need to include the `quota` mount option in the `/etc/vfstab` or `samfs.cmd` file, because file systems are mounted with quotas enabled by default.

```
[server1]root@solaris:~# mount /samfs2
[server1]root@solaris:~#
```

9. Stop here.

Check Quotas

Both administrators and users can monitor quotas and resource usage. The `root` user can generate quota reports on users, groups, or admin sets with the `samquota` command. File-system users can check their own quotas using the `squota` command.

See the procedures below:

- Monitor Quotas as the File-System Administrator
- Monitor Your Own User Quota

Monitor Quotas as the File-System Administrator

1. Log in to the file-system server as `root`.

   In the example, the server is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. To display quota statistics for all groups, use the command `samquota -g [`*`directory-or-file`*`]`, where *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to see quotas.

   In the example, we request a report for the `/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -g /samqfs1
   ```

3. To display quota statistics for all admin sets, use the command `samquota -a [`*`directory-or-file`*`]`, where *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to see quotas.

   In the example, we request a report for the `/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -a /samqfs1
   ```

4. To display quota statistics for all users, use the command `samquota -u [`*`directory-or-file`*`]`, where *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to see quotas.

   In the example, we request a report for the `/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -u /samqfs1
   ```

5. To display quota statistics for a specific group, use the command `samquota -G` *`groupID`* `[`*`directory-or-file`*`]`, where *`groupID`* specifies a group name or integer identifier for the group and *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to see quotas.

   In the example, we request a report on quotas for the `dev` group in the`/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -G dev /samqfs1
   ```

6. To display quota statistics for a specific admin set, use the command `samquota -A` *`adminsetID`* `[`*`directory-or-file`*`]`, where *`adminsetID`* specifies an integer identifier for the admin set and *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to see quotas.

   In the example, we request a report on quotas for admin set `1` in the `/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -A 1 /samqfs1
   ```

7. To display quota statistics for a specific user, use the command `samquota -U` *`userID`* `[`*`directory-or-file`*`]`, where *`userID`* specifies a user name or integer identifier for the user and *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to see quotas.

   In the example, we request a report on the quotas for user `jr23547` in the`/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -U jr23547 /samqfs1
   ```

8. Stop here.

Monitor Your Own User Quota

1. Log in to a file-system host using your user ID.

   In the example, we log in to host `server1` as user `od447`:

   ```
   [server1]od447@solaris:~#
   ```

2. To display quota statistics for all groups, use the command `squota [`*`directory-or-file`*`]`, where *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to see quotas.

   In the example, we request a report for all file systems:

   ```
   [server1]od447@solaris:~# squota
                                            Limits
                Type    ID    In Use     Soft     Hard
        /samqfs1
        Files  group   101         1     1000     1200
        Blocks group   101         8    20000    30000
        Grace period                    25920
        No user quota entry.
   [server1]od447@solaris:~#
   ```

3. Stop here.

Temporarily Extend or Cancel Grace Periods

When you need to extend a grace period temporarily or when you need to cut a grace period short, you can do so:

- Extending a Grace Period by a Specified Amount
- Restarting the Grace Period
- End a Grace Period Early.

Extending a Grace Period by a Specified Amount If a group, user, or admin set has exceeded the specified soft limit for its quota and needs to remain above the soft limit temporarily but for a period that is longer than the current grace period allows, you can grant the extension as follows:

1. Log in to the file-system server as `root`.

   In the example, we log in to host `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Check the quota that requires an extension. Use the command `samquota -`*`quota-type ID`* `[`*`directory-or-file`*`]` where:

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to extend a grace period.

   In the example, the `dev` group is significantly over the soft limit and has only a couple of hours left in its grace period:

   ```
   [server1]root@solaris:~# samquota -G dev /samqfs1
                                    Online Limits                    Total Limits
           Type   ID    In Use       Soft      Hard       In Use     Soft  Hard
   /samqfs1
   Files  group  101       323      15000      30000         323    15000  30000
   Blocks group  101 3109330961 2013265920 3019898880 3109330961 2013265920
   3019898880
   Grace period                 4320                             4320
   ---> Warning:  soft limits to be enforced in 2h21m16s
   [server1]root@solaris:~#
   ```

3. Extend the grace period, if warranted. Use the command `samquota -`*`quota-type ID`* `-x` *`number-seconds`* `[`*`directory-or-file`*`]`, where:

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to extend a grace period.
   - *`number-seconds`* is an integer representing the number of seconds in the extension (see the `samquota` man page for alternative ways of specifying time).

   Enter `y` (yes) when prompted to continue.

   In the example, we extend the grace period for the `dev` group to `267840` seconds (31 days) for files in the `/samqfs1` file system:

   ```
   [server1]root@solaris:~# samquota -G dev -x 267840 /samqfs1
   Setting Grace Timer:  continue? y
   ```

   When we recheck the `dev` group quota, the grace period has been extended:

   ```
   [server1]root@solaris:~# samquota -G dev /samqfs1
                                    Online Limits                    Total Limits
           Type    ID    In Use       Soft      Hard   In Use       Soft   Hard
   /samqfs1
   Files  group   101       323      15000      30000     323      15000   30000
   Blocks group   101     43208 2013265920 3019898880   43208 2013265920
   ```

```
3019898880
Grace period                 267840                      267840
---> Warning:  soft limits to be enforced in 31d
[server1]root@solaris:~#
```

4. If a group, admin set, or user regularly needs extensions, re-evaluate storage requirements and/or consider increasing the grace period permanently. Use the procedure "Set Quotas for Groups, Projects, Directories, and Users" on page 3-10.

5. Stop here.

Restarting the Grace Period If a group, user, or admin set has exceeded the specified soft limit for its quota and cannot free space quickly enough to get below the soft limit before the current grace period expires, you can restart the grace period. Proceed as follows:

1. Log in to the file-system server as `root`.

   In the example, we log into host `server1`:

```
[server1]root@solaris:~#
```

2. Check the quota that requires an extension. Use the command `samquota -`*`quota-type ID`* `[`*`directory-or-file`*`]` where:

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to extend a grace period.

   In the example, the `cit` group is over the soft limit and has just over an hour left in its grace period:

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/samqfs1
Files  group  119       762         750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                 4320                           4320
---> Warning:  soft limits to be enforced in 1h11m23s
[server1]root@solaris:~#
```

3. To reset the grace period to its full starting size the next time that a file or block is allocated, *clear* the grace period timer. Use the command `samquota -`*`quota-type ID`* `-x clear [`*`directory-or-file`*`]`, where:

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to extend a grace period.

   Enter `y` (yes) when prompted to continue.

   In the example, we clear the grace-period timer for the `cit` group's quota on the `/samqfs1` file system.

```
[server1]root@solaris:~# samquota -G cit -x clear /samqfs1
```

```
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

When we recheck the `cit` group quota, a file has been allocated and the grace period has been reset to `12h`, 12 hours (`4320` seconds):

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                 Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/samqfs1
Files  group  119       763         750       1500        763      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                 4320                              4320
---> Warning:  soft limits to be enforced in 12h
[server1]root@solaris:~#
```

4. Alternatively, to reset the grace period to its full starting size immediately, *reset* the grace period timer. Use the command `samquota -`*`quota-type ID`*`-x reset [`*`directory-or-file`*`]`.

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is either the mount point directory for a specific file system or the specific directory or file for which you need to extend a grace period.

   Enter `y` (yes) when prompted to continue.

   In the example, we clear the grace-period timer for the `cit` group's quota on the `/samqfs1` file system.

```
[server1]root@solaris:~# samquota -G cit -x reset /samqfs1
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

   When we recheck the `cit` group quota, the grace period has been reset to `12h`, 12 hours (`4320` seconds):

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                 Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/samqfs1
Files  group  119       762         750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                 4320                              4320
---> Warning:  soft limits to be enforced in 12h
[server1]root@solaris:~#
```

5. Stop here.

End a Grace Period Early

1. Log in to the file-system server as `root`.

   In the example, we log into host `server1`:

```
[server1]root@solaris:~#
```

2. Check the grace period that you need to cut short. Use the command `samquota -`*`quota-type ID`*` [`*`directory-or-file`*`]` where:

- *quota-type ID* is G plus a group name or ID number, A plus an admin set ID number, or U plus a user name or ID number.
- *directory-or-file* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to extend a grace period.

In the example, the cit group is over the soft limit and has eleven hours left in its grace period, but we need to end the grace period early:

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                 Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use       Soft
Hard
/samqfs1
Files  group  119        822        750       1500        822        750
1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                 4320                              4320
---> Warning:  soft limits to be enforced in 11h
[server1]root@solaris:~#
```

3. *Expire* the grace period. Use the command samquota -*quota-type ID* -x expire [*directory-or-file*], where:

- *quota-type ID* is G plus a group name or ID number, A plus an admin set ID number, or U plus a user name or ID number.
- *directory-or-file* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to extend a grace period.

In the example, we expire the grace period for the cit group:

```
root@solaris:~# samquota -G cit -x reset /samqfs1
Setting Grace Timer:  continue? y
```

When we re-check quotas, soft limits for the cit group are being enforced as hard limits:

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                 Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/samqfs1
Files  group  119        762        750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                 4320                              4320
---> Online soft limits under enforcement (since 6s ago)
[server1]root@solaris:~#
```

4. Stop here.

Stop New Resource Allocations

You can inhibit file system resource allocations by creating inconsistent quota values. When the file system detects that quota values are not consistent for a user, group, or admin set, it prevents that user, group, or admin set from using any more system resources. So setting the hard limit for a quota lower than the corresponding soft limit stops further allocations. To use this technique, proceed as follows:

1. Log in to the file-system server as `root`.

   In the example, we log into host `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Back up the quota so that you can restore it later. Export the current configuration, and redirect the information to a file. Use the command `samquota` *`-quota-type ID`* `[`*`directory-or-file`*`] >` *`file`* where:

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to extend a grace period.
   - *`file`* is the name of the output file.

   In the example, we export the quota for the `cit` group to the file `restore.samqfs1.quota_g.cit` in the `root` user's home directory:

   ```
   [server1]root@solaris:~# samquota -G cit -e /samqfs1 > \
   /root/restore.samqfs1.quota_g.cit
   [server1]root@solaris:~#
   ```

3. Check the output. Use the Solaris command `more <` *`file`*, where *`file`* is the name of the output file.

   ```
   [server1]root@solaris:~# more < /root/restore.samqfs1.quota_g.cit
   # Type  ID
   #                  Online Limits                          Total   Limits
   #              soft            hard                    soft       hard
   # Files
   # Blocks
   # Grace Periods
   samquota -G 119 \
       -f       750:s:o  -f      1500:h:o        -f       750:s:t -f
   1500:h:t \
       -b 157286400:s:o  -b
   235929600:h:o        -b 157286400:s:t -b 235929600:h:t \
               -t 4320:o                               -t 4320:t
   [server1]root@solaris:~#
   ```

4. Set the hard limits for the quota to `0` (zero) and set the soft limits to `1` (or any non-zero value). Use the command `samquota` *`-quota-type ID`* `-f 1:s -f 0:h -b 1:s -b 0:h [`*`directory-or-file`*`]`.

   - *`quota-type ID`* is `G` plus a group name or ID number, `A` plus an admin set ID number, or `U` plus a user name or ID number.
   - *`directory-or-file`* (optional) is the mount point directory for a specific file system or a specific directory or file for which you need to extend a grace period.

   In the example, we make the quota settings for the `cit` group in the `/samqfs1` file system inconsistent, and thereby stop new resource allocations.

   ```
   [server1]root@solaris:~# samquota -G cit -f 1:s -f 0:h -b 1:s -b 0:h /samqfs1
   [server1]root@solaris:~#
   ```

When we check the quota for the `cit` group, zero quotas are in effect. The exclamation point characters (**!**) show all current use as over-quota, so no further allocations will be made:

```
[server1]root@solaris:~# samquota -G cit /samqfs1
Online Limits                Total Limits
Type   ID              In Use    Soft     Hard         In Use     Soft     Hard
/sam6
Files  group 119        822!       1        0         822!       1        0
Blocks group 119 3109330961!       1        0  3109330961!       1        0
Grace period                     4320                           4320
---> Quota values inconsistent; zero quotas in effect.
[server1]root@solaris:~#
```

5. When you are ready resume normal allocations by restoring the modified quota to its original state. Execute the backup file that you created as a shell script. Use the Solaris command `sh` *`file`*, where *`file`* is the name of the backup file.

   In the example, we restore the quota for the `cit` group by executing the file `/root/restore.samqfs1.quota_g.cit`

```
[server1]root@solaris:~# sh /root/restore.samqfs1.quota_g.cit
Setting Grace Timer:  continue? y
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

   When we check the quota, normal limits have been restored and allocations are no longer blocked:

```
[server1]root@solaris:~# samquota -G cit /samqfs1
                                 Online Limits                       Total Limits
        Type   ID    In Use         Soft       Hard     In Use     Soft  Hard
/samqfs1
Files  group  119       822          750       1500        822     750   1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
235929600
Grace period                     4320                            4320
---> Warning:  soft limits to be enforced in 11h
[server1]root@solaris:~#
```

6. Stop here.

Remove the Quotas for a File System

To remove or disable quotas for a file system, disable quotas in the mount process.

1. Log in to the file-system server as `root`.

   In the example, we log in to host `server1`:

```
[server1]root@solaris:~#
```

2. Open the `/etc/vfstab` file in a text editor, add the `noquota` mount option to the mount options column of the file system row, and save the file.

   In the example, we open the file in the `vi` text editor, and set the `noquota` mount option for the `samqfs1` file system:

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount            System  fsck  Mount    Mount
#to Mount      to fsck  Point            Type    Pass  at Boot  Options
#------------  -------  ---------------  ------  ----  -------  ------------
```

```
/devices        -       /devices        devfs   -    no      -
/proc           -       /proc           proc    -    no      -
...
samqfs1         -       /samqfs1        samfs   -    no      noquota
:wq
[server1]root@solaris:~#
```

3. If the file system is mounted, unmount it.

   You must unmount and then remount a file system so that the operating system reloads the /etc/vfstab file and makes the specified changes. In the example, we unmount the samqfs1 file system:

   ```
   [server1]root@solaris:~# umount samqfs1
   [server1]root@solaris:~#
   ```

4. Mount the file system.

   In the example, we mount the samqfs1 file system:

   ```
   [server1]root@solaris:~# mount samqfs1
   [server1]root@solaris:~#
   ```

5. If you expect to reinstate quotas later, leave the quota files in pace.

   When you are ready to reinstate quotas, you can simply unmount the file system, run the command samfsck -F on the file system, remove the noquota mount option, and then remount the file system.

6. If you do not expect to reinstate quotas or if you need to reclaim the space consumed by quota files, use the Solaris command rm to delete the files .quota_g, .quota_a, and/or .quota_u from the root directory of the file system.

   In the example, we remove all quota files from the /samqfs1 file system root directory:

   ```
   [server1]root@solaris:~# rm /samqfs1/.quota_g
   [server1]root@solaris:~# rm /samqfs1/.quota_a
   [server1]root@solaris:~# rm /samqfs1/.quota_u
   [server1]root@solaris:~#
   ```

7. Stop here.

Controlling Archiving and Staging Operations

In general, you manage archiving file systems in much the same was as you would non-archiving file systems. However, you must stop the archiving process before carrying out most file-system management tasks. When active, the archiving processes make changes to file-system's primary disk cache. So you must quiesce these processes before you do maintenance work on the disk cache. This section covers the following tasks:

- Idle Archiving and Staging Processes
- Stop Archiving and Staging Processes
- Restart Archiving and Staging Processes.

Idle Archiving and Staging Processes

1. Log in to the file system host as root.

   In the example, we log in to host server1:

```
[server1]root@solaris:~#
```

2. Idle all archiving processes. Use the command `samcmd aridle`.

   This command will allow current archiving and staging to complete, but will not start any new jobs:

   ```
   [server1]root@solaris:~# samcmd aridle
   [server1]root@solaris:~#
   ```

3. Idle all staging processes. Use the command `samcmd stidle`.

   This command will allow current archiving and staging to complete, but will not start any new jobs:

   ```
   [server1]root@solaris:~# samcmd stidle
   [server1]root@solaris:~#
   ```

4. Wait for active archiving jobs to complete. Check on the status of the archiving processes using the command `samcmd a`.

   When archiving processes are `Waiting for :arrun`, the archiving process is idle:

   ```
   [server1]root@solaris:~# samcmd a
   Archiver status samcmd     5.4 10:20:34 May 20 2014
   samcmd on samfs-mds
   sam-archiverd:  Waiting for :arrun
   sam-arfind: ...
   Waiting for :arrun
   ```

5. Wait for active staging jobs to complete. Check on the status of the staging processes using the command `samcmd u`.

   When staging processes are `Waiting for :strun`, the staging process is idle:

   ```
   [server1]root@solaris:~# samcmd u
   Staging queue samcmd      5.4  10:20:34 May 20 2014
   samcmd on solaris.demo.lan
   Staging queue by media type: all
   sam-stagerd:  Waiting for :strun
   root@solaris:~#
   ```

6. To fully quiesce the system, Stop Archiving and Staging Processes as well.

Stop Archiving and Staging Processes

1. If you have not already done so, Idle Archiving and Staging Processes.
2. If you have not already done so, log in to the file system host as `root`.

   In the example, we log in to host `server1`:

   ```
   [server1]root@solaris:~#
   ```

3. Idle all removable media drives before proceeding further. For each drive, use the command `samcmd` *`equipment-number`* `idle`, where *`equipment-number`* is the equipment ordinal number assigned to the drive in the `/etc/opt/SUNWsamfs/mcf` file.

   This command will allow current archiving and staging jobs to complete before turning drives `off`, but will not start any new work. In the example, we idle four drives, with ordinal numbers `801`, `802`, `803`, and `804`:

   ```
   [server1]root@solaris:~# samcmd 801 idle
   ```

```
[server1]root@solaris:~# samcmd 802 idle
[server1]root@solaris:~# samcmd 803 idle
[server1]root@solaris:~# samcmd 804 idle
[server1]root@solaris:~#
```

4. Wait for running jobs to complete.

   We can check on the status of the drives using the command `samcmd r`. When all drives are `notrdy` and `empty`, we are ready to proceed.

```
[server1]root@solaris:~# samcmd r
Removable media samcmd     5.4 18:37:09 Feb 17 2014
samcmd on samqfs1host
ty   eq   status      act  use  state  vsn
li  801   ---------p    0   0%  notrdy
          empty
li  802   ---------p    0   0%  notrdy
          empty
li  803   ---------p    0   0%  notrdy
          empty
li  804   ---------p    0   0%  notrdy
          empty
[server1]root@solaris:~#
```

5. When the archiver and stager processes are idle and the tape drives are all `notrdy`, stop the library-control daemon. Use the command `samd stop`.

```
[server1]root@solaris:~# samd stop
[server1]root@solaris:~#
```

6. Proceed with file-system maintenance.

7. When maintenance is complete, Restart Archiving and Staging Processes.

   When you restart operations, pending stages are reissued and archiving is resumed.

8. Stop here.

Restart Archiving and Staging Processes

When you are ready, resume normal, automatic operation, proceed a follows:

1. Log in to the file system host as `root`.

   In the example, we log in to host `server1`:

```
[server1]root@solaris:~#
```

2. Restart the SAM-QFS library-control daemon. Use the command `samd start`.

```
[server1]root@solaris:~# samd start
[server1]root@solaris:~#
```

3. Stop here.

Renaming File Systems

Renaming a file system is a two-step process. First you change the family set name for the file system by editing the `/etc/opt/SUNWsamfs/mcf` file. Then you have the `samfsck -R -F` command read the new name and update the superblock on the corresponding disk devices. To rename a file system, use the procedure below:

Rename a File System

1. Log in to the file-system server as `root`.

   In the example, we log in to host `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. If you are repairing an archiving file system, carry out the procedure "Idle Archiving and Staging Processes" on page 3-22 before proceeding further.

3. Unmount the file system that you need to rename.

   In the example, we unmount file system `samqfs1`:

   ```
   [server1]root@solaris:~# umount samqfs1
   ```

4. Open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and locate the file system that you need to rename.

   In the example, we use the `vi` editor. We need to change the name of the `samqfs1` file system:

   ```
   [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family        Device   Additional
   # Identifier         Ordinal    Type       Set           State    Parameters
   #-----------------   ---------  ---------  ------------  ------   ----------
   samqfs1              100        ms         samqfs1       on
    /dev/dsk/c1t3d0s3    101        md         samqfs1       on
    /dev/dsk/c1t4d0s5    102        md         samqfs1       on
   ```

5. In the fourth column of the file, change the family set name of the file system to the new value. You may also change the file-system equipment identifier in the first column, but do not change anything else. Save the file and close the editor.

   In the example, we change both the equipment identifier and the family set name of the file system from `samqfs1` to `samqfs-hpcc`:

   ```
   [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family        Device   Additional
   # Identifier         Ordinal    Type       Set           State    Parameters
   #-----------------   ---------  ---------  ------------  ------   ----------
   samqfs-hpcc          100        ms         samqfs-hpcc   on
    /dev/dsk/c1t3d0s3    101        md         samqfs-hpcc   on
    /dev/dsk/c1t4d0s5    102        md         samqfs-hpcc   on
   :wq
   root@solaris:~#
   ```

6. Rewrite the file-system super block to reflect the new family set name. Use the command `samfsck -R -F` *`family-set-name`*, where *`family-set-name`* is the family set name that you just specified in the `/etc/opt/SUNWsamfs/mcf` file.

   When issued with the `-R` and `-F` options, the `samfsck` command reads the new family set name and the corresponding disk-storage equipment identifiers from the `/etc/opt/SUNWsamfs/mcf` file. It then rewrites the super block on the specified disk devices with the new family set name. In the example, we run the command with the new `samqfs-hpcc` family set name:

   ```
   [server1]root@solaris:~# samfsck -R -F samqfs-hpcc
   ```

7. Open the `/etc/vfstab` file in a text editor, and locate the entry for the file system that you are renaming.

In the example, we open the file in the `vi` text editor. We need to change the `samqfs1` file system entry to use the new name:

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount              System  fsck  Mount    Mount
#to Mount      to fsck  Point              Type    Pass  at Boot  Options
#------------  -------  ---------------    ------  ----  -------  ------------
/devices       -        /devices           devfs   -     no       -
/proc          -        /proc              proc    -     no       -
...
samqfs1        -        /samqfs1           samfs   -     no       -
```

8. In the entry for the file system that you have renamed, change the file system name in the first column and the mount-point directory name in the third column (if required), and save the file.

   In the example, we change the name of the `samqfs1` file system to `samqfs-hpcc` and change the mount point to match:

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount              System  fsck  Mount    Mount
#to Mount      to fsck  Point              Type    Pass  at Boot  Options
#------------  -------  ---------------    ------  ----  -------  ------------
/devices       -        /devices           devfs   -     no       -
/proc          -        /proc              proc    -     no       -
...
samqfs-hpcc    -        /samqfs-hpcc       samfs   -     no       -
:wq
[server1]root@solaris:~#
```

9. Create the new mount-point directory for the new file system, if required, and set the access permissions for the mount point.

   Users must have execute (`x`) permission to change to the mount-point directory and access files in the mounted file system. In the example, we create the `/qfsms` mount-point directory and set permissions to `755` (`-rwxr-xr-x`):

```
[server1]root@solaris:~# mkdir /samqfs-hpcc
[server1]root@solaris:~# chmod 755 /samqfs-hpcc
[server1]root@solaris:~#
```

10. Check the `mcf` file for errors by running the `sam-fsd` command, and correct any that are detected.

    The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

```
[server1]root@solaris:~# sam-fsd
```

11. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly. Use the command `samd config`.

```
[server1]root@solaris:~# samd config
```

12. If `samd config` reports errors, correct them and re-issue the command until no errors are found.

13. Mount the file system.

    In the example, we use the new mount point directory:

```
[server1]root@solaris:~# mount /samqfs-hpcc
```

14. Stop here.

Repairing File Systems

When file systems report errors via samu, SAM-QFS Manager, or the /var/adm/sam-log file, follow the procedure below:

Repair a File System

1. Log in to the file-system server as root.

   In the example, we log in to host server1:

   ```
   [server1]root@solaris:~#
   ```

2. If you are repairing an archiving file system, carry out the procedure "Idle Archiving and Staging Processes" on page 3-22 before proceeding further.

3. Unmount the affected file system.

   You may need to try more than once if you are waiting for archiving to stop. In the example, we unmount file system samqfs1:

   ```
   [server1]root@solaris:~# umount samqfs1
   samfs umount: /samqfs1: is busy
   [server1]root@solaris:~# umount samqfs1
   [server1]root@solaris:~#
   ```

4. Repair the file system. Use the command samfsck -F -V *family-set-name*, where *family-set-name* is the family set name specified for the file system in the /etc/opt/SUNWsamfs/mcf file.

   It is often a good idea to save the repair results to a date-stamped file for later reference and for diagnostic purposes, when necessary. So in the example, we save the results by piping the samfsck output to the command tee /var/tmp/samfsck-FV.*family-set-name*.`date '+%Y%m%d.%H%M%S'`:

   ```
   [server1]root@solaris:~# samfsck -F -V samqfs1 | tee \
   /var/tmp/samfsck-FV.samqfs1. `date '+%Y%m%d.%H%M%S'`
   name:     /samqfs1       version:     2A
   First pass
   Second pass
   Third pass
   NOTICE: ino 2.2,  Repaired link count from 8 to 14
   Inodes processed: 123392
   total data kilobytes       = 1965952
   total data kilobytes free  = 1047680
   total meta kilobytes       = 131040
   total meta kilobytes free  = 65568
   INFO:  FS samma1 repaired:
           start:  May 19, 2014 10:57:13 AM MDT
           finish: May 19, 2014 10:57:37 AM MDT
   NOTICE: Reclaimed 70057984 bytes
   NOTICE: Reclaimed 9519104 meta bytes
   [server1]root@solaris:~#
   ```

5. Remount the file system.

   ```
   [server1]root@solaris:~# mount /samqfs1
   [server1]root@solaris:~#
   ```

6. Stop here.

Adding Devices to File Systems

Before you add devices to an existing file system, you should consider your requirements and your alternatives. Make sure that enlarging the existing file system is the best way to meet growing capacity requirements. If you need more physical storage space to accommodate new projects or user communities, creating one or more new SAM-QFS file systems may be a better choice. Multiple, smaller file systems will generally offer better performance than one much larger file system, and the smaller file systems may be easier to create and maintain.

Once you have decided that you need to enlarge a file system, take either of the following approaches:

- Add Devices to a Mounted File System (recommended)
- Add Devices to an Unmounted File System

Add Devices to a Mounted File System

Proceed as follows:

1. Log in to the file-system server as `root`.

   In the example, we log in to host `server1`:

```
[server1]root@solaris:~#
```

2. Open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and locate the file system that you need to enlarge.

   In the examples, we use the `vi` editor. We need to enlarge two file systems, the general-purpose `samqfsms` file system and the high-performance `samqfs2ma` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
samqfsms            100        ms         samqfsms   on
 /dev/dsk/c1t3d0s3   101        md         samqfsms   on
 /dev/dsk/c1t4d0s5   102        md         samqfsms   on
samqfs2ma           200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3   201        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5   202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5   203        md         samqfs2ma  on
```

3. If you are adding devices to a general-purpose `ms` file system, add additional data/metadata devices to the end of the file system definition in the `mcf` file. Then save the file, and close the editor.

   You can add up to 252 logical devices. In the example, we add two devices, `103` and `104`, to the `samqfsms` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
samqfsms            100        ms         samqfsms    on
 /dev/dsk/c1t3d0s3   101        md         samqfsms    on
 /dev/dsk/c1t4d0s5   102        md         samqfsms    on
 /dev/dsk/c1t3d0s7   103        md         samqfsms    on
 /dev/dsk/c1t4d0s7   104        md         samqfsms    on
:wq
[server1]root@solaris:~#
```

4. If you are adding devices to a high-performance `ma` file system, add data devices and one or more `mm` disk devices to the end of the file system definition in the `mcf` file. Then save the file, and close the editor.

   Always add new devices at the end of the list of existing devices. You can add up to 252, adding metadata devices proportionately as you add data devices. In the example, we add one `mm` metadata device, `204`, and two `md` data devices, `205` and `206`, to the `samqfs2ma` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
...
samqfs2ma           200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3   201        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5   202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5   203        md         samqfs2ma  on
 /dev/dsk/c1t5d0s6   204        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s7   205        md         samqfs2ma  on
 /dev/dsk/c1t4d0s7   206        md         samqfs2ma  on
:wq
[server1]root@solaris:~#
```

5. Check the `mcf` file for errors by running the `sam-fsd` command, and correct any that are detected.

   The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

```
[server1]root@solaris:~# sam-fsd
```

6. If the `sam-fsd` command finds an error in the `mcf` file, edit the file to correct the error and recheck as described in the preceding step.

   In the example below, `sam-fsd` reports an unspecified problem with a device:

```
[server1]root@solaris:~# sam-fsd
Problem in mcf file /etc/opt/SUNWsamfs/mcf for filesystem samqfsms
sam-fsd: Problem with file system devices.
```

   Usually, such errors are the result of inadvertent typing mistakes. Here, when we open the `mcf` file in an editor, we find that we have typed a letter `o` instead of a 0 in the equipment name for device `104`, the second new `md` device:

```
samqfsms                     100        ms         samqfsms    on
 /dev/dsk/c1t3d0s3           101        md         samqfsms    on
 /dev/dsk/c1t4d0s5           102        md         samqfsms    on
 /dev/dsk/c1t3d0s7           103        md         samqfsms    on
 /dev/dsk/c1t4dos7           104        md         samqfsms    on
               ^
```

7. If the `sam-fsd` command runs without error, the `mcf` file is correct. Proceed to the next step.

   The example is a partial listing of error-free output:

```
[server1]root@solaris:~# sam-fsd
Trace file controls:
```

```
sam-amld      /var/opt/SUNWsamfs/trace/sam-amld
              cust err fatal ipc misc proc date
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[server1]root@solaris:~#
```

8. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly. Use the command `samd config`.

```
[server1]root@solaris:~# samd config
```

9. Make sure that `samd config` has updated the SAM-QFS file system configuration to include the new devices. Use the command `samcmd f`.

   The devices should be in the `off` state. In the example, `samcmd f` shows the new devices, `103` and `104`, and both are `off`:

```
[server1]root@solaris:~# samcmd f
File systems samcmd     5.4 16:57:35 Feb 27 2014
samcmd on server1
ty      eq   state device_name       status      high low mountpoint server
ms     100   on    samqfsms          m----2-----  80% 70% /samqfsms
 md     101   on   /dev/dsk/c1t3d0s3
 md     102   on   /dev/dsk/c1t4d0s5
 md     103   off  /dev/dsk/c1t3d0s7
 md     104   off  /dev/dsk/c1t4d0s7
[server1]root@solaris:~#
```

10. Enable the newly added devices. For each device, use the command `samcmd add` *`equipment-number`*, where *`equipment-number`* is the equipment ordinal number assigned to the device in the `/etc/opt/SUNWsamfs/mcf` file.

    In the example, we enable new devices, `103` and `104`:

```
[server1]root@solaris:~# samcmd add 103
[server1]root@solaris:~# samcmd add 104
```

11. If you are adding devices to a shared file system, go to "Finish Configuring New Devices Added to a Shared File System" on page 3-31.

12. If you are adding devices to an unshared, standalone file system, make sure that the devices were added and are ready for use by the file system. Use the command `samcmd m`, check the results.

    When the device is in the `on` state, it has been added successfully and is ready to use. In the example, we have successfully added devices `103` and `104`:

```
[server1]root@solaris:~# samcmd f
Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
samcmd on server1
ty   eq  status       use state  ord  capacity      free  ra  part high low
ms   100  m----2----- 13% on           3.840G    3.588G  1M    16  80% 70%
 md   101             31% on       0  959.938M  834.250M
 md   102             13% on       1  959.938M  834.250M
 md   103              0% on       2  959.938M  959.938M
 md   104              0% on       3  959.938M  959.938M
[server1]root@solaris:~#
```

13. Stop here.

Finish Configuring New Devices Added to a Shared File System When you add devices to a shared file system, you must carry out a few more steps before the devices are configured on all file-system hosts. Proceed as follows:

1. Log in to the file-system metadata server host as `root`.

   In the example, the metadata server host is named `metadata-server`:

   ```
   [metadata-server]root@solaris:~#
   ```

2. Make sure that the new devices were added to the metadata server. Use the command `samcmd m`.

   When the device is in the `unavail` state, it has been added successfully but is not yet ready for use. In the example, we have successfully added devices `103` and `104`:

   ```
   [metadata-server]root@solaris:~# samcmd f
   Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
   samcmd on metadata-server
   ty      eq  status       use state  ord   capacity     free    ra  part high
   low
   ms     100  m----2-----  13% on           3.840G    3.588G    1M    16  80%
   70%
    md     101             31% on       0  959.938M  834.250M
    md     102             13% on       1  959.938M  834.250M
    md     103              0% unavail  2  959.938M  959.938M
    md     104              0% unavail  3  959.938M  959.938M
   [metadata-server]root@solaris:~#
   ```

3. Log in to each file-system client hosts as `root`.

   Remember to include potential metadata servers, since they are also clients. In the example, we need to log in to a potential metadata server, named `potential-metadata-server`, and two clients, `client1` and `client2Linux`. So we open three terminal windows and use secure shell (`ssh`):

   ```
   [metadata-server]root@solaris:~# ssh root@potential-metadata-server
   Password:
   [potential-metadata-server]root@solaris:~#
   [metadata-server]root@solaris:~# ssh root@client1
   Password:
   [client1]root@solaris:~#
   [metadata-server]root@solaris:~# ssh root@client2Linux
   Password:
   [client2Linux]:[root@Linux ~]#
   ```

4. If the client is a Linux client, unmount the shared file system.

   ```
   [client2Linux]:[root@Linux ~]# umount /samqfsms
   ```

5. On each client, open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and add the new devices to the end of the file system definition, just as you did on the server.

   In the example, we add devices `103` and `104` to the `mcf` file on `client1`:

   ```
   [client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family     Device   Additional
   # Identifier         Ordinal    Type       Set        State    Parameters
   #-----------------   ---------  ---------  ---------  ------   ----------
   samqfsms             100        ms         samqfsms    on      shared
    /dev/dsk/c1t3d0s3    101        md         samqfsms    on
    /dev/dsk/c1t4d0s5    102        md         samqfsms    on
    /dev/dsk/c1t3d0s7    103        md         samqfsms    on
   ```

```
 /dev/dsk/c1t4d0s7    104        md         samqfsms    on
:wq
[metadata-server]root@solaris:~#
```

6. On each client, check the `mcf` file for errors by running the `sam-fsd` command, and correct any that are detected.

```
[metadata-server]root@solaris:~# sam-fsd
```

7. On each client, tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly:

```
[metadata-server]root@solaris:~# samd config
```

8. If the client is a Linux client, mount the shared file system.

```
[client2Linux]:[root@Linux ~]# mount /samqfsms
```

9. Once all clients have been configured, return to the metadata server, and enable storage allocation on the new devices. For each device, use the command `samcmd alloc` *`equipment-number`*, where *`equipment-number`* is the equipment ordinal number assigned to the device in the `/etc/opt/SUNWsamfs/mcf` file.

   In the example, we enable storage allocation on devices `103` and `104`:

```
[metadata-server]root@solaris:~# samcmd alloc 103
[metadata-server]root@solaris:~# samcmd alloc 104
```

10. Finally, make sure that the devices are ready for use by the file system. Use the command `samcmd m`, and check the results.

    When the device is in the `on` state, it has been added successfully and is ready to use. In the example, we have successfully added devices `103` and `104`:

```
[metadata-server]root@solaris:~# samcmd f
Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
samcmd on metadata-server
ty      eq  status       use state  ord   capacity     free    ra  part high
low
ms     100  m----2-----  13% on           3.840G    3.588G    1M    16  80%
70%
 md     101              31% on       0  959.938M  834.250M
 md     102              13% on       1  959.938M  834.250M
 md     103               0% on       2  959.938M  959.938M
 md     104               0% on       3  959.938M  959.938M
[metadata-server]root@solaris:~#
```

11. Stop here.

Add Devices to an Unmounted File System

Proceed as follows:

1. Log in to the file-system server host as `root`.

   In the example, the metadata server host is named `server1`:

```
[server1]root@solaris:~#
```

2. Before you unmount an archiving file system, you must carry out the procedure "Idle Archiving and Staging Processes" on page 3-22.

3. Unmount the file system.

Do not proceed until you have unmounted the file system. In the example, we unmount file system `samqfs1`:

```
[server1]root@solaris:~# umount samqfs1
```

4. Open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and locate the file system that you need to enlarge.

   In the example, we use the `vi` editor. We need to enlarge the `samqfs1` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
samqfs1               100        ms         samqfs1    on
 /dev/dsk/c1t3d0s3     101        md         samqfs1    on
 /dev/dsk/c1t4d0s5     102        md         samqfs1    on
```

5. If you are adding devices to a high-performance `ma` file system, you must add metadata storage along with the data storage. Add enough additional `mm` disk devices to the file-system definition in the `mcf` file to store the metadata for the data devices that you add. Then save the file, and close the editor.

   You can add up to 252 logical devices. In the example, we add one `mm` metadata device to the `samqfs2ma` file system and two data devices to the `samqfs2ma` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
samqfs2ma             200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3     201        mm         samqfs2ma  on
 /dev/dsk/c1t5d0s6     204        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5     202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5     203        md         samqfs2ma  on
 /dev/dsk/c1t3d0s7     205        md         samqfs2ma  on
 /dev/dsk/c1t4dos7     206        md         samqfs2ma  on
:wq
[server1]root@solaris:~#
```

6. If you are adding devices to a general-purpose `ms` file system, add additional data/metadata devices to the file system definition in the `mcf` file. Then save the file, and close the editor.

   You can add up to 252 logical devices. In the example, we add two devices to the `samqfs1` file system:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
samqfs1               100        ms         samqfs1    on
 /dev/dsk/c1t3d0s3     101        md         samqfs1    on
 /dev/dsk/c1t4d0s5     102        md         samqfs1    on
 /dev/dsk/c1t3d0s7     103        md         samqfs1    on
 /dev/dsk/c1t4dos7     104        md         samqfs1    on
:wq
[server1]root@solaris:~#
```

7. Check the `mcf` file for errors by running the `sam-fsd` command, and correct any that are detected.

The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

```
[server1]root@solaris:~# sam-fsd
```

8. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly:

   ```
   root@solaris:~# samd config
   ```

9. Incorporate the new devices into file system. Use the command `samgrowfs` *`family-set-name`*, where *`family-set-name`* is the family set name specified for the file system in the `/etc/opt/SUNWsamfs/mcf` file.

   In the example, we grow the `samqfs1` file system:

   ```
   [server1]root@solaris:~# samgrowfs samqfs1
   ```

10. Remount the file system.

    ```
    [server1]root@solaris:~# mount /samqfs1
    ```

11. If you added devices to an archiving file system, restart the SAM-QFS library-management daemon. Use the command `samd start`.

    ```
    [server1]root@solaris:~# samd start
    ```

12. If you neglected to unmount the file system before making changes and if, consequently, the file system will not mount, restore the original `mcf` file by deleting references to the added devices. Then run `samd config` to restore the configuration, unmount the file system, and start over.

13. Stop here.

Removing Data Devices from a File System

When required, you can remove data devices from mounted SAM-QFS file systems. Typically this becomes necessary when you need to replace a failed unit or when you need to free up under-utilized devices for other uses. There are, however, some limitations.

You can only remove data devices. You cannot remove any devices used to hold metadata, since metadata defines the organization of the file system itself. This means that you can remove `md`, `mr`, and striped-group devices from high-performance `ma` file systems only. You cannot remove `mm` metadata devices from `ma` file systems. Nor can you remove `md` devices from general purpose `ms` file systems, since these devices store both data and metadata.

To remove devices, you must also have somewhere to move any valid data files that reside on the target device. This means that you cannot remove all the devices. One device must always remain available in the file system and it must have enough free capacity to hold all files residing on removed devices. So, if you need to remove a striped group, you must have another available striped group configured with an identical number of member devices.

To remove devices, proceed as follows:

- Make Sure that File-System Metadata and Data are Backed Up
- Remove Devices from a Mounted High-Performance File System.

Make Sure that File-System Metadata and Data are Backed Up

Carry out the following tasks:

- Run samexplorer
- Create a Recovery Point File for the File System.

Run `samexplorer`

1. Log in to the file-system server host as `root`.

   In the example, the metadata server host is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Create a `samexplorer` report. Use the command `samexplorer` *path/hostname.YYYYMMDD.hhmmz.tar.gz*, where:

   - *path* is the path to the chosen directory.
   - *hostname* is the name of the SAM-QFS file system host.
   - *YYYYMMDD.hhmmz* is a date and time stamp.

   By default, the file is called `/tmp/SAMreport.`*hostname.YYYYMMDD.hhmmz*`.tar.gz`. In the example, we use the directory `/zfs1/tmp/`, where `/zfs1` is a file system that has no components in common with the SAM-QFS file system:

   ```
   [server1]root@solaris:~# samexplorer \
   /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz

        Report name:
   /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
        Lines per file:  1000
        Output format:   tar.gz (default) Use -u for unarchived/uncompressed.

        Please wait.............................................
        Please wait.............................................
        Please wait......................................

        The following files should now be ftp'ed to your support provider
        as ftp type binary.

        /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
   ```

3. Create a Recovery Point File for the File System.

Create a Recovery Point File for the File System

1. Log in to the file-system server host as `root`.

   In the example, the metadata server host is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Select the location where the recovery point file will be stored. The selected location must share no devices with the file system that you are backing up and must have room to store an unusually large file.

   The devices that we intend to remove may contain files that have not been archived. Since such files exist only as single copies, we will have to create a recovery point file that stores at least some data as well as metadata. This can substantially increase the size of the recovery point file.

In the example, we create a subdirectory, `tmp/`, in a file system has no components in common with the SAM-QFS file system, `/zfs1`:

```
[server1]root@solaris:~# mkdir /zfs1/tmp/
[server1]root@solaris:~#
```

3. Change to the file system's root directory.

   In the example, we change to the mount-point directory **`/samqfs1`**:

   ```
   [server1]root@solaris:~# cd /samqfs1
   [server1]root@solaris:~#
   ```

4. Back up the file-system metadata and any unarchived data. Use the command `samfsdump -f -u` *`recovery-point`*, where *`recovery-point`* is the path and file name of the finished recovery point file.

   Note that the `-u` option adds the data portion of unarchived files to the recovery point. This can greatly increase the size of the file.

   In the example, we create a recovery point file for the `samqfs1` file system called `samqfs1-20140313.025215` in the directory `/zfs1/tmp/`. We check the result using the command `ls -l`:

   ```
   [server1]root@solaris:~# cd /samqfs1
   [server1]root@solaris:~# samfsdump -f \
   /zfs1/tmp/samqfs1-`date '+%Y%m%d.%H%M%S'` -T /samqfs1
   samfsdump statistics:
       Files:              10010
       Directories:        2
       Symbolic links:     0
       Resource files:     0
       Files as members of hard links :    0
       Files as first hard link :  0
       File segments:      0
       File archives:      10010
       Damaged files:      0
       Files with data:    0
       File  warnings:     0
       Errors:             0
       Unprocessed dirs:   0
       File data bytes:    0
   [server1]root@solaris:~# ls -l /zfs1/tmp/samqfs1*
   -rw-r--r-- 1 root other 5376517 Mar 13 02:52 /zfs1/tmp/samqfs1-20140313.025215
   [server1]root@solaris:~#
   ```

5. Now Remove Devices from a Mounted High-Performance File System.

Remove Devices from a Mounted High-Performance File System

You must remove devices one at a time. For each device, proceed as follows:

1. Log in to the file-system server host as `root`.

   In the example, the metadata server host is named `server1`:

   ```
   [server1]root@solaris:~#
   ```

2. Open the `/etc/opt/SUNWsamfs/mcf` file, and note the equipment ordinal number for the device that you need to remove.

In the example, we use the `vi` editor. We need to remove device `/dev/dsk/c1t4d0s7` from the equipment list for the `samqfs1` file system. The equipment ordinal number is `104`:

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   --------------
samqfs1              100        ms         samqfs1    on
 /dev/dsk/c1t3d0s3    101        md         samqfs1    on
 /dev/dsk/c1t4d0s5    102        md         samqfs1    on
 /dev/dsk/c1t3d0s7    103        md         samqfs1    on
 /dev/dsk/c1t4d0s7    104        md         samqfs1    on
:q
[server1]root@solaris:~#
```

3. Before you try to remove a device, make sure that the remaining devices in the file system can accept any files that have to be moved from the device that you intend to delete.

   - Make sure that the remaining devices have adequate capacity.
   - If the device is a striped group, make sure that the file system contains another striped group with an equivalent configuration.

     For example, if the striped group that you plan to remove has four equipment numbers, you must have another striped group that is in the ON state and has four equipment numbers.

4. Make sure that the file system that you plan to modify has a version 2A superblock. Use the command `samfsinfo` *`filesystem-name`*, where *`filesystem-name`* is the name of the file system.

   In the example, file system `samqfs1` uses a `version:2A` superblock:

```
[server1]root@solaris:~# /opt/SUNWsamfs/sbin/samfsinfo samqfs1
samfsinfo: filesystem samqfs1 is mounted.
name:     samqfs1       version:     2A
time:     Tuesday, June 28, 2011  6:07:36 AM MDT
feature:  Aligned Maps
count:    4
...
[server1]root@solaris:~#
```

5. If the file system does not have a version 2A superblock, stop here. You cannot remove devices while this file system is mounted.

6. If you are removing devices from a SAM-QFS archiving file system, release all archived files from the disk device that you are removing. Use the command `samcmd release` *`equipment-number`*, where *`equipment-number`* is the equipment ordinal number that identifies the device in the `/etc/opt/SUNWsamfs/mcf` file.

   If the device is a striped group, provide the equipment number of any device in the group.

   The SAM-QFS software changes the state of the specified device to `noalloc` (no allocations) so that no new files are stored on it, and starts releasing previously archived files. Once the device contains no unarchived files, the software removes the device from the file system configuration and changes its state to `off`.

   In the example, we release files from device `104` in the archiving file system `samqfs1`:

```
[server1]root@solaris:~# samcmd release 104
```

7. If you are removing a device from a SAM-QFS non-archiving file system, move all remaining valid files off the disk device that you are removing. Use the command `samcmd remove` *`equipment-number`*, where *`equipment-number`* is the equipment ordinal number that identifies the device in the `/etc/opt/SUNWsamfs/mcf` file.

   The SAM-QFS software changes the state of the specified device to `noalloc` (no allocations) so that no new files are stored on it, and starts moving files that contain valid data to the remaining devices in the file system. When all files have been moved, the software removes the device from the file system configuration and changes its state to `off`.

   In the example, we move files off of device `104`:

```
[server1]root@solaris:~# samcmd remove 104
```

8. Monitor the progress of the selected process, `samcmd remove` or `samcmd release`. Use the command `samcmd m` and/or watch the log file and `/var/opt/SUNWsamfs/trace/sam-shrink` file.

   The `release` process completes fairly quickly if all files have been archived, because it merely releases space associated with files that have been copied to archival media. Depending on the amount of data and the number of files, the `remove` process takes considerably longer because it must move files between disk devices.

```
[server1]root@solaris:~# samcmd m
ty     eq  status       use state  ord  capacity      free    ra  part high low
ms     100 m----2-----  27% on            3.691G    2.628G    1M    16  80% 70%
 md     101              27% on       0  959.938M  703.188M
 md     102              28% on       1  899.938M  646.625M
 md     103              13% on       2  959.938M  834.250M
 md     104               0% noalloc  3  959.938M  959.938M
[server1]root@solaris:~#
```

9. If you are using `samcmd release` and the target device does not enter the `off` state, there are unarchived files on the device. Wait for the archiver to run and archiving to complete. Then use the command `samcmd release` again. You can check on the progress of archiving by using the command `samcmd a`.

   The `release` process cannot free the disk space until unarchived files are archived.

```
[server1]root@solaris:~# samcmd a
Archiver status samcmd     5.4 14:12:14 Mar  1 2014
samcmd on server1
sam-archiverd:  Waiting for resources
sam-arfind:  samqfs1  mounted at /samqfs1
Files waiting to start          4   schedule          2   archiving          2
[server1]root@solaris:~#
```

10. If `samcmd release` fails because one or more unarchived files cannot be archived, move the unarchived files to another device. Use the command `samcmd remove` *`equipment-number`*, just as you would when removing devices from a non-archiving, standalone file system.

    In the example, we move files off of device `104`:

```
[server1]root@solaris:~# samcmd remove 104
```

11. Once the device state has been changed to `off`, open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, locate the file system, and update the equipment list to reflect the changes. Save the file and close the editor.

    In the example, `samcmd m` shows that `104` is `off`. So we use the `vi` editor to open the `mcf` file. We remove the entry for device `104` from the equipment list for the `samqfs1` file system and save our changes:

    ```
    [server1]root@solaris:~# samcmd m
    ty      eq  status       use state  ord  capacity      free    ra  part high
    low
    ms     100  m----2-----  27% on            3.691G    2.628G    1M    16  80%
    70%
     md     101              27% on       0  959.938M  703.188M
     md     102              28% on       1  899.938M  646.625M
     md     103              13% on       2  959.938M  834.250M
     md     104               0% off      3  959.938M  959.938M
    [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
    # Equipment          Equipment  Equipment  Family     Device   Additional
    # Identifier         Ordinal    Type       Set        State    Parameters
    #-----------------   ---------  ---------  ---------  ------   ---------
    samqfs1              100        ms         samqfs1    on
     /dev/dsk/c1t3d0s3    101        md         samqfs1    on
     /dev/dsk/c1t4d0s5    102        md         samqfs1    on
     /dev/dsk/c1t3d0s7    103        md         samqfs1    on
    :wq
    [server1]root@solaris:~#
    ```

12. Check the modified `mcf` file for errors by running the `sam-fsd` command, and correct any errors that are detected.

    The `sam-fsd` command will stop if it encounters an error:

    ```
    [server1]root@solaris:~# sam-fsd
    ```

13. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly:

    ```
    [server1]root@solaris:~# samd config
    ```

14. Stop here.

Managing SAM-QFS Shared File Systems

This section outlines the following tasks:

- Mounting and Unmounting Shared File Systems
- Changing the Host Configuration of a Shared File System
- Switching from the Active Metadata Server to a Potential Metadata Server
- Converting an Unshared File System to a Shared File System
- Converting a Shared File System to an Unshared File System.

Mounting and Unmounting Shared File Systems

When you mount or unmount a shared file system, the order in which you mount or unmount the metadata server and the clients 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 file that contains the mount options and copy that file to all of the hosts.

For more information about mounting shared file systems, see the mount_samfs man page.

Mount a Shared File System

1. Log in to the SAM-QFS metadata server and client hosts as root.

   In the example, we log in to the metadata server host for the sharefs file system, sharefs-mds. Then we open a terminal window for each client, sharefs-client1 and sharefs-client2 and use ssh (Secure Shell) to log in:

   ```
   [sharefs-mds]root@solaris:~# ssh root@sharefs-client1
   Password:
   [sharefs-client1]root@solaris:~#

   [sharefs-mds]root@solaris:~# ssh root@sharefs-client2
   Password:
   [sharefs-client2]root@solaris:~#
   ```

2. If the file system has an entry in the Solaris /etc/vfstab file, mount the shared file system on the metadata server host using the command mount *mountpoint*, where *mountpoint* is the mount point directory on the host's root file system.

   Always mount the file system on the metadata server host first, before mounting the file system on clients.

   In the example, the sharefs file system has the following entry in the /etc/vfstab file:

   ```
   sharefs    -         /sharefs  samfs    -     no       shared
   ```

   So we can mount the file system by supplying only the mount point parameter:

   ```
   [sharefs-mds]root@solaris:~# mount /sharefs
   [sharefs-mds]root@solaris:~#
   ```

3. If the file system does not have an entry in the Solaris /etc/vfstab file, mount the shared file system on the metadata server host using the command mount -F samfs -o shared *mountpoint*, where *mountpoint* is the mount point directory on the host's root file system.

   Always mount the file system on the metadata server host first, before mounting the file system on clients.

   In the example, the sharefs file system has no entry in the /etc/vfstab file:

   ```
   [sharefs-mds]root@solaris:~# mount -F samfs -o shared /sharefs
   [sharefs-mds]root@solaris:~#
   ```

4. If the file system has an entry in the Solaris /etc/vfstab file, mount the shared file system on each client host using the command mount *mountpoint*, where *mountpoint* is the mount point directory on the host's root file system.

   You can mount the file system on the client hosts in any order.

   ```
   [sharefs-client1]root@solaris:~# mount /sharefs
   [sharefs-client1]root@solaris:~#

   [sharefs-client2]root@solaris:~# mount /sharefs
   ```

```
[sharefs-client2]root@solaris:~#
```

5. If the file system does not have an entry in the Solaris `/etc/vfstab` file, mount the shared file system on each client host using the command `mount -F samfs -o shared` *`mountpoint`*, where *`mountpoint`* is the mount point directory on the host's root file system.

   You can mount the file system on the client hosts in any order.

```
[sharefs-client1]root@solaris:~# mount -F samfs -o shared /sharefs
[sharefs-client1]root@solaris:~#

[sharefs-client2]root@solaris:~# mount -F samfs -o shared /sharefs
[sharefs-client2]root@solaris:~#
```

6. Stop here.

Unmount a Shared File System

1. Log in to the SAM-QFS metadata server and client hosts as `root`.

   In the example, we log in to the metadata server host for the `sharefs` file system, `sharefs-mds`. Then we open a terminal window for each client, `sharefs-client1` and `sharefs-client2` and use `ssh` (Secure Shell) to log in:

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~#

[sharefs-mds]root@solaris:~# ssh root@sharefs-client2
Password:
[sharefs-client2]root@solaris:~#
```

2. If the file system is shared through NFS or SAMBA, unshare the file system before you unmount it. On the metadata server, use the command `unshare` *`mount-point`*, where *`mount-point`* is the mount point directory of the SAM-QFS file system.

```
[sharefs-mds]root@solaris:~# unshare /sharefs
[sharefs-mds]root@solaris:~#
```

3. Unmount the SAM-QFS shared file system from each client. Use the command `umount` *`mount-point`*, where *`mount-point`* is the mount point directory of the SAM-QFS file system.

   See the `umount_samfs` man page for further details. In the example, we unmount `/sharedqfs1` from our two clients, `sharefs-client1` and `sharefs-client2`:

```
[sharefs-client1]root@solaris:~# umount /sharefs
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#

[sharefs-client2]root@solaris:~# umount /sharefs
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

4. Unmount the SAM-QFS shared file system from the metadata server. Use the command `umount -o await_clients=`*`interval`* *`mount-point`*, where *`mount-point`* is the mount point directory of the SAM-QFS file system and *`interval`* is the delay in seconds specified by the `-o await_clients` option delays execution.

   When the `umount` command is issued on the metadata server of a SAM-QFS shared file system, the `-o await_clients` option makes `umount` wait the specified number of seconds so that clients have time to unmount the share. It has no effect if you unmount an unshared file system or issue the command on a SAM-QFS client. See the `umount_samfs` man page for further details.

In the example, we unmount the `/sharefs` file system from the server, allowing `60` seconds for clients to unmount:

```
[sharefs-mds]root@solaris:~# umount -o await_clients=60 /sharefs
[sharefs-mds]root@solaris:~#
```

5. Stop here.

Changing the Host Configuration of a Shared File System

This section provides instructions for configuring additional hosts as clients of a shared file system and for de-configuring existing clients. It includes the following sections:

- Configuring Additional File System Clients
- Removing a Host from a Shared File System Configuration
- Configuring Datamover Clients for Distributed Tape I/O
- Connecting Tape Drives Using Persistent Bindings.

Configuring Additional File System Clients

There are three parts to the process of adding a client host to a shared file system:

- First, you Add the Host Information to the Shared File System Configuration.
- Then you configure the shared file system on the host, using the procedure specific to the host operating system, either Configure the Shared File System on a Solaris Client or Configure the Shared File System on a Linux Client Host.
- Finally, you mount the shared file system on the host, using the procedure specific to the host operating system, either Mount the Shared File System on a Solaris Host or Mount the Shared File System on a Linux Client Host.

Add the Host Information to the Shared File System Configuration

1. Log in to the SAM-QFS metadata server as `root`.

   In the example, the SAM-QFS shared file system is `sharefs`, and the metadata server host is `sharefs-mds`:

   ```
   [sharefs-mds]root@solaris:~#
   ```

2. Back up the file `/etc/opt/SUNWsamfs/hosts.`*`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

   ```
   [sharefs-mds]root@solaris:~# mv /etc/opt/SUNWsamfs/hosts.sharefs \
   /etc/opt/SUNWsamfs/hosts.sharefs.bak
   ```

3. If the shared file system is mounted, run the command `samsharefs` *`filesystem`* from the active metadata server, redirecting output to a file, `/etc/opt/SUNWsamfs/hosts.`*`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

   The `samsharefs` command displays the host configuration for a SAM-QFS shared file system. Redirecting the output to a file creates a new hosts file.

   ```
   [sharefs-mds]root@solaris:~# samsharefs sharedqfs1 > \
   ```

```
/etc/opt/SUNWsamfs/hosts.sharedqfs1
```

4. If the shared file system is not mounted, run the command `samsharefs -R` *`filesystem`* from an active or potential metadata server, redirecting output to the file `/etc/opt/SUNWsamfs/hosts.`*`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

   The `samsharefs -R` command can only be run from an active or potential metadata server (see the `samsharefs` man page for more details). The `samsharefs` command displays the host configuration for a SAM-QFS shared file system. Redirecting the output to a file creates a new hosts file. In the example, we run the command from the metadata server `sharefs-mds`:

   ```
   [sharefs-mds]root@solaris:~# samsharefs -R sharedqfs1 \
   > /etc/opt/SUNWsamfs/hosts.sharedqfs1
   ```

5. Open the newly created hosts file in a text editor.

   In the example, we use the `vi` editor. The host configuration includes the active metadata server, `sharefs-mds`, one client that is also a potential metadata server, `sharefs-mds_alt`, and two other clients, `sharefs-client1` and `sharefs-client2`:

   ```
   [sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
   #                                             Server   On/  Additional
   #Host Name           Network Interface        Ordinal  Off  Parameters
   #------------------  ----------------------   -------  ---  ----------
   sharefs-mds          10.79.213.117            1        0    server
   sharefs-mds_alt      10.79.213.217            2        0
   sharefs-client1      10.79.213.133            0        0
   sharefs-client2      10.79.213.47             0        0
   ```

6. In the hosts file, add a line for the new client host, save the file, and close the editor.

   In the example, we add an entry for the host `sharefs-client3`:

   ```
   [sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
   #                                             Server   On/  Additional
   #Host Name           Network Interface        Ordinal  Off  Parameters
   #------------------  ----------------------   -------  ---  ----------
   sharefs-mds          10.79.213.117            1        0    server
   sharefs-mds_alt      10.79.213.217            2        0
   sharefs-client1      10.79.213.133            0        0
   sharefs-client2      10.79.213.47             0        0
   sharefs-client3      10.79.213.49             0        0
   :wq
   [sharefs-mds]root@solaris:~#
   ```

7. If the file system is mounted, update the file-system from the active metadata server. Use the command `samsharefs -u` *`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

   The `samsharefs` command re-reads the revised hosts file and updates the configuration:

   ```
   [sharefs-mds]root@solaris:~# samsharefs -u sharefs1
   ```

8. If the file system is not mounted, update the file-system from an active or potential metadata server. Use the command `samsharefs -R -u` *`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

The samsharefs command re-reads the revised hosts file and updates the configuration:

```
[sharefs-mds]root@solaris:~# samsharefs -R -u sharefs1
```

9. If you are adding a Solaris host as a client, go to "Configure the Shared File System on a Solaris Client".
10. If you are adding a Linux host as a client, go to "Configure the Shared File System on a Linux Client Host".

Configure the Shared File System on a Solaris Client

1. On the shared file-system client, log in as root.

   In the example, the SAM-QFS shared file system is sharefs, and the client host is sharefs-client1:

   ```
   [sharefs-client1]root@solaris:~#
   ```

2. In a terminal window, retrieve the configuration information for the shared file system. Use the command samfsconfig *device-path*, where *device-path* is the location where the command should start to search for file-system disk devices (such as /dev/dsk/* or /dev/zvol/dsk/rpool/*).

   ```
   [sharefs-client1]root@solaris:~# samfsconfig /dev/dsk/*
   ```

3. If the host has access to the metadata devices for the file system and is thus suitable for use as a potential metadata server, the samfsconfig output closely resembles the mcf file that you created on the file-system metadata server.

   In our example, host sharefs-client1 has access to the metadata devices (equipment type mm), so the command output shows the same equipment listed in the mcf file on the server, sharefs-mds. Only the host-assigned device controller numbers differ:

   ```
   [sharefs-client1]root@solaris:~# samfsconfig /dev/dsk/*
   # Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
   # Generation 0 Eq count 4 Eq meta count 1
   sharefs             300         ma         sharefs   -
   /dev/dsk/c1t0d0s0    301         mm         sharefs   -
   /dev/dsk/c1t3d0s0    302         mr         sharefs   -
   /dev/dsk/c1t3d0s1    303         mr         sharefs   -
   ```

4. If the host does not have access to the metadata devices for the file system, the samfsconfig command cannot find the metadata devices and thus cannot fit the SAM-QFS devices that it discovers into the file-system configuration. The command output lists Ordinal 0—the metadata device—under Missing Slices, fails to include the line that identifies the file-system family set, and comments out the listings for the data devices.

   In our example, host sharefs-client2 has access to the data devices only. So the samfsconfig output looks like this:

   ```
   [sharefs-client2]root@solaris:~# samfsconfig /dev/dsk/*
   # Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
   # Missing slices
   # Ordinal 0
   # /dev/dsk/c4t3d0s0    302         mr         sharefs   -
   # /dev/dsk/c4t3d0s1    303         mr         sharefs   -
   ```

5. Copy the entries for the shared file system from the `samfsconfig` output. Then, in a second window, open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and paste the copied entries into the file.

   In our first example, the host, `sharefs-client1`, has access to the metadata devices for the file system, so the `mcf` file starts out looking like this:

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------    ---------  ---------  ---------  ------   ---------------
sharefs              300        ma         sharefs    -
/dev/dsk/c1t0d0s0    301        mm         sharefs    -
/dev/dsk/c1t3d0s0    302        mr         sharefs    -
/dev/dsk/c1t3d0s1    303        mr         sharefs    -
```

   In the second example, the host, `sharefs-client2`, does not have access to the metadata devices for the file system, so the `mcf` file starts out looking like this:

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------    ---------  ---------  ---------  ------   ---------------
# /dev/dsk/c4t3d0s0    302         mr         sharefs   -
# /dev/dsk/c4t3d0s1    303         mr         sharefs   -
```

6. If the host has access to the metadata devices for the file system, add the `shared` parameter to the `Additional Parameters` field of the entry for the shared file system.

   In the first example, the host, `sharefs-client1`, has access to the metadata:

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------    ---------  ---------  ---------  ------   ---------------
sharefs              300        ma         sharefs    -        shared
/dev/dsk/c1t0d0s0    301        mm         sharefs    -
/dev/dsk/c1t3d0s0    302        mr         sharefs    -
/dev/dsk/c1t3d0s1    303        mr         sharefs    -
```

7. If the host does not have access to the metadata devices for the file-system, add a line for the shared file system and include the `shared` parameter

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------    ---------  ---------  ---------  ------   ---------------
sharefs              300        ma         sharefs    -        shared
# /dev/dsk/c4t3d0s0    302         mr         sharefs   -
# /dev/dsk/c4t3d0s1    303         mr         sharefs   -
```

8. If the host does not have access to the metadata devices for the file system, add a line for the metadata device. Set the `Equipment Identifier` field to `nodev` (*no device*) and set the remaining fields to exactly the same values as they have on the metadata server:

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------    ---------  ---------  ---------  ------   ---------------
sharefs              300         ma        sharefs    on       shared
```

```
nodev               301         mm        sharefs    on
# /dev/dsk/c4t3d0s0   302        mr         sharefs    -
# /dev/dsk/c4t3d0s1   303        mr         sharefs    -
```

9. If the host does not have access to the metadata devices for the file system, uncomment the entries for the data devices.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    on       shared
nodev               301        mm         sharefs    on
/dev/dsk/c4t3d0s0   302        mr         sharefs    -
/dev/dsk/c4t3d0s1   303        mr         sharefs    -
```

10. Make sure that the `Device State` field is set to `on` for all devices, save the `mcf` file, and close the editor.

    In our first example, the host, `sharefs-client1`, has access to the metadata devices for the file system, so the `mcf` file ends up looking like this:

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    on       shared
/dev/dsk/c1t0d0s0   301        mm         sharefs    on
/dev/dsk/c1t3d0s0   302        mr         sharefs    on
/dev/dsk/c1t3d0s1   303        mr         sharefs    on
:wq
[sharefs-client1]root@solaris:~#
```

    In the second example, the host, `sharefs-client2`, does not have access to the metadata devices for the file system, so the `mcf` file starts ends up like this:

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs               300        ma         sharefs    on       shared
nodev                 301        mm         sharefs    on
/dev/dsk/c4t3d0s0     302        mr         sharefs    on
/dev/dsk/c4t3d0s1     303        mr         sharefs    on
:wq
[sharefs-client2]root@solaris:~#
```

11. Check the `mcf` file for errors by running the `sam-fsd` command, and correct any errors found.

    The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error. In the example, we check the `mcf` file on `sharefs-client1`:

```
[sharefs-client1]root@solaris:~# sam-fsd
```

12. Next Mount the Shared File System on a Solaris Host.

Mount the Shared File System on a Solaris Host

1. On the shared file-system host, log in as `root`.

In the example, the SAM-QFS shared file system is sharefs, and the host is a client named sharefs-client1:

```
[sharefs-client1]root@solaris:~#
```

2. Back up the operating system's /etc/vfstab file.

   ```
   [sharefs-client1]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
   ```

3. Open the /etc/vfstab file in a text editor, and add a line for the shared file system.

   In the example, we open the file in the vi text editor and add a line for the sharefs family set device:

   ```
   [sharefs-client1]root@solaris:~# vi /etc/vfstab
   #File
   #Device    Device   Mount     System  fsck  Mount    Mount
   #to Mount  to fsck  Point     Type    Pass  at Boot  Options
   #--------  -------  --------  ------  ----  -------  ------------------------
   /devices   -        /devices  devfs   -     no       -
   /proc      -        /proc     proc    -     no       -
   ...
   sharefs    -        /sharefs  samfs   -     no
   ```

4. To mount the file system on the client as a shared file system, enter the shared option in the Mount Options column of the vfstab entry for the shared file system.

   If we wanted the current client to mount the shared file system sharefs read-only, we would edit the vfstab entry as shown in the example below:

   ```
   #File
   #Device    Device   Mount     System  fsck  Mount    Mount
   #to Mount  to fsck  Point     Type    Pass  at Boot  Options
   #--------  -------  --------  ------  ----  -------  ------------------------
   /devices   -        /devices  devfs   -     no       -
   /proc      -        /proc     proc    -     no       -
   ...
   sharefs    -        /sharefs  samfs   -     no       shared
   ```

5. Add any other desired mount options using commas as separators, and make any other desired changes to the /etc/vfstab file. Then save the /etc/vfstab file.

   In the example, we add no additional mount options:

   ```
   #File
   #Device    Device   Mount     System  fsck  Mount    Mount
   #to Mount  to fsck  Point     Type    Pass  at Boot  Options
   #--------  -------  --------  ------  ----  -------  -------------------------
   /devices   -        /devices  devfs   -     no       -
   /proc      -        /proc     proc    -     no       -
   ...
   sharefs    -        /sharefs  samfs   -     no       shared
   :wq
   [sharefs-client1]root@solaris:~#
   ```

6. Create the mount point specified in the /etc/vfstab file, and set the access permissions for the mount point.

   The mount-point permissions must be the same as on the metadata server and on all other clients. Users must have execute (x) permission to change to the mount-point directory and access files in the mounted file system. In the example, we create the /sharefs mount-point directory and set permissions to 755 (-rwxr-xr-x):

```
[sharefs-client1]root@solaris:~# mkdir /sharefs
[sharefs-client1]root@solaris:~# chmod 755 /sharefs
[sharefs-client1]root@solaris:~#
```

7. Mount the shared file system:

```
[sharefs-client1]root@solaris:~# mount /sharefs
[sharefs-client1]root@solaris:~#
```

8. If you are adding a potential metadata server host as a distributed tape I/O datamover, go to "Configuring Datamover Clients for Distributed Tape I/O" on page 3-53.
9. Stop here.

Configure the Shared File System on a Linux Client Host

1. On the Linux client, log in as root.

   In the example, the SAM-QFS shared file system is sharefs, and the host is a Linux client named sharefs-clientL:

```
[sharefs-clientL][root@Linux ~]#
```

2. In a terminal window, retrieve the configuration information for the shared file system using the samfsconfig *device-path* command, where *device-path* is the location where the command should start to search for file-system disk devices (such as /dev/*).

   Since Linux hosts do not have access to the metadata devices for the file system, the samfsconfig command cannot find the metadata devices and thus cannot fit the SAM-QFS devices that it discovers into the file-system configuration. The command output lists Ordinal 0—the metadata device—under Missing Slices, fails to include the line that identifies the file-system family set, and comments out the listings for the data devices.

   In our example, the samfsconfig output for Linux host sharefs-clientL looks like this:

```
[sharefs-clientL][root@Linux ~]# samfsconfig /dev/*
# Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
#
# Missing slices
# Ordinal 0
# /dev/sda4            302         mr         sharefs   -
# /dev/sda5            303         mr         sharefs   -
```

3. Copy the entries for the shared file system from the samfsconfig output. Then, in a second window, open the /etc/opt/SUNWsamfs/mcf file in a text editor, and paste the copied entries into the file.

   In the example, the mcf file for the Linux the host, sharefs-clientL, starts out looking like this:

```
[sharefs-clientL][root@Linux ~]# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
# /dev/sda4             302        mr         sharefs   -
# /dev/sda5             303        mr         sharefs   -
```

4. In the mcf file, insert a line for the shared file system, and include the shared parameter.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma         sharefs    -        shared
# /dev/sda4            302         mr         sharefs   -
# /dev/sda5            303         mr         sharefs   -
```

5. In the mcf file, insert lines for the file system's metadata devices. Since the Linux host does not have access to metadata devices, set the Equipment Identifier field to nodev (*no device*) and then set the remaining fields to exactly the same values as they have on the metadata server:

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300         ma        sharefs    on       shared
nodev                 301         mm        sharefs    on
# /dev/sda4            302         mr         sharefs   -
# /dev/sda5            303         mr         sharefs   -
```

6. In the mcf file, uncomment the entries for the Linux data devices.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma        sharefs     on       shared
nodev                 301        mm        sharefs     on
/dev/sda4             302        mr        sharefs     -
/dev/sda5             303        mr        sharefs     -
```

7. Make sure that the Device State field is set to on for all devices, and save the mcf file.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma         sharefs    on       shared
nodev                 301        mm         sharefs    on
/dev/sda4             302        mr         sharefs    on
/dev/sda5             303        mr         sharefs    on
:wq
[sharefs-clientL][root@Linux ~]#
```

8. Check the mcf file for errors by running the sam-fsd command, and correct any errors found.

   The sam-fsd is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error. In the example, we check the mcf file on the Linux client, sharefs-clientL:

```
[sharefs-clientL][root@Linux ~]# sam-fsd
```

9. Now, Mount the Shared File System on a Linux Client Host.

Mount the Shared File System on a Linux Client Host

1. On the Linux client, log in as root.

In the example, the SAM-QFS shared file system is `sharefs`, and the host is a Linux client named `sharefs-clientL`:

```
[sharefs-clientL][root@Linux ~]#
```

2. Back up the operating system's `/etc/fstab` file.

```
[sharefs-clientL][root@Linux ~]# cp /etc/fstab /etc/fstab.backup
```

3. Open the `/etc/fstab` file in a text editor, and start a line for the shared file system.

   In the example, we use the `vi` text editor and add a line for the `sharefs` family set device:

```
[sharefs-clientL][root@Linux ~]# vi /etc/fstab
#File
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs
```

4. In the fourth column of the file, add the mandatory `shared` mount option.

```
[sharefs-clientL][root@Linux ~]# vi /etc/fstab
#File
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared
```

5. In the fourth column of the file, add any other desired mount options using commas as separators.

   Linux clients support the following additional mount options:

   - `rw`, `ro`
   - `retry`
   - `meta_timeo`
   - `rdlease`, `wrlease`, `aplease`
   - `minallocsz`, `maxallocsz`
   - `noauto`, `auto`

   In the example, we add the option `noauto`:

```
#File
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared,noauto
```

6. Enter zero (`0`) in each of the two remaining columns in the file. Then save the `/etc/fstab` file.

```
#File
```

```
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared,noauto              0         0
:wq
[sharefs-clientL][root@Linux ~]#
```

7. Create the mount point specified in the `/etc/fstab` file, and set the access permissions for the mount point.

   The mount-point permissions must be the same as on the metadata server and on all other clients. Users must have execute (`x`) permission to change to the mount-point directory and access files in the mounted file system. In the example, we create the `/sharefs` mount-point directory and set permissions to `755` (`-rwxr-xr-x`):

   ```
   [sharefs-clientL][root@Linux ~]# mkdir /sharefs
   [sharefs-clientL][root@Linux ~]# chmod 755 /sharefs
   ```

8. Mount the shared file system. Use the command `mount` *`mountpoint`*, where *`mountpoint`* is the mount-point directory specified in the `/etc/fstab` file.

   As the example shows, the `mount` command generates a warning. This is normal and can be ignored:

   ```
   [sharefs-clientL][root@Linux ~]# mount /sharefs
   Warning: loading SUNWqfs will taint the kernel: SMI license
   See http://www.tux.org/lkml/#export-tainted for information
   about tainted modules. Module SUNWqfs loaded with warnings
   ```

9. Stop here.

Removing a Host from a Shared File System Configuration

Removing a host from a shared file system is simply a matter of the removing it from the server configuration, as described below (to fully deconfigure the host, uninstall the software and the configuration files):

Remove the Host from the File System Hosts File

1. Log in to the SAM-QFS metadata server as `root`.

   In the example, the SAM-QFS shared file system is `sharefs`, and the metadata server host is `sharefs-mds`:

   ```
   [sharefs-mds]root@solaris:~#
   ```

2. Log in to each client as `root`, and unmount the shared file system.

   Remember that potential metadata servers are themselves clients. In the example, we have three clients: `sharefs-client1`, `sharefs-client2`, and `sharefs-mds_alt`, a potential metadata server. For each client, we log in using `ssh`, unmount the file system `sharefs`, and close the `ssh` session:

   ```
   [sharefs-mds]root@solaris:~# ssh root@sharefs-client1
   Password:
   [sharefs-client1]root@solaris:~# umount sharefs
   [sharefs-client1]root@solaris:~# exit
   [sharefs-mds]root@solaris:~# ssh root@sharefs-client2
   Password:
   ```

```
[sharefs-client2]root@solaris:~# umount sharefs
[sharefs-client2]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
Password:
[sharefs-mds_alt]root@solaris:~# umount sharefs
root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

3. On the metadata server, unmount the shared file system.

```
[sharefs-mds]root@solaris:~# umount sharefs
```

4. On the metadata server, rename the file `/etc/opt/SUNWsamfs/hosts.`*`filesystem`* to `/etc/opt/SUNWsamfs/hosts.`*`filesystem`*`.bak`, where *`filesystem`* is the name of the file system from which you are removing the client host.

```
[sharefs-mds]root@solaris:~# mv /etc/opt/SUNWsamfs/hosts.sharefs \
/etc/opt/SUNWsamfs/hosts.sharefs.bak
```

5. Capture the current shared file system host configuration top a file. From the metadata server, run the command `samsharefs -R` *`filesystem`*, redirecting the output to the file `/etc/opt/SUNWsamfs/hosts.`*`filesystem`*, where *`filesystem`* is the name of the file system to which you are adding the client host.

   The `samsharefs` command displays the host configuration for the specified SAM-QFS shared file system. Redirecting the output to a file creates a new hosts file. In the example, we run the command from the metadata server `sharefs-mds`:

```
[sharefs-mds]root@solaris:~# samsharefs -R sharedqfs1 > /
/etc/opt/SUNWsamfs/hosts.sharedqfs1
```

6. Open the newly created hosts file in a text editor.

   In the example, we use the `vi` editor. We need to remove the client `sharefs-client3`:

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs-mds           10.79.213.117           1        0    server
sharefs-mds_alt       10.79.213.217           2        0
sharefs-client1       10.79.213.133           0        0
sharefs-client2       10.79.213.47            0        0
sharefs-client3       10.79.213.49            0        0
```

7. In the hosts file, delete the line that corresponds to the client host that you need to remove. Then save the file, and close the editor.

   In the example, we delete the entry for the host `sharefs-client3`:

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs-mds           10.79.213.117           1        0    server
sharefs-mds_alt       10.79.213.217           2        0
sharefs-client1       10.79.213.133           0        0
sharefs-client2       10.79.213.47            0        0
:wq
[sharefs-mds]root@solaris:~#
```

8. Update the file-system with the revised hosts file. From the metadata server, use the command `samsharefs -R -u` *`filesystem`*, where *`filesystem`* is the name of the file system from which you are removing the client host.

   ```
   [sharefs-mds]root@solaris:~# samsharefs -u sharefs
   ```

9. On the metadata server host, mount the shared file system.

   In the examples, the `/etc/vfstab` file contains an entry for the `sharefs` file system, so we use the simple mounting syntax (see the `mount_samfs` man page for full information):

   ```
   [sharefs-mds]root@solaris:~# mount sharefs
   ```

10. On the each client host, mount the shared file system.

    Remember that potential metadata servers are themselves clients. In the example, we have three clients: `sharefs-client1`, `sharefs-client2`, and `sharefs-mds_alt`, a potential metadata server. For each client, we log in using `ssh`, unmount the file system `sharefs`, and close the `ssh` session:

    ```
    [sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
    Password:
    [sharefs-mds_alt]root@solaris:~# mount sharefs
    sharefs-mds_alt]root@solaris:~# exit
    [sharefs-mds]root@solaris:~# ssh root@sharefs-client1
    Password:
    [sharefs-client1]root@solaris:~# mount sharefs
    sharefs-client1]root@solaris:~# exit
    [sharefs-mds]root@solaris:~# ssh root@sharefs-client2
    Password:
    [sharefs-client2]root@solaris:~# mount sharefs
    sharefs-client2]root@solaris:~# exit
    [sharefs-mds]root@solaris:~#
    ```

11. Stop here.

Configuring Datamover Clients for Distributed Tape I/O

Starting with SAM-QFS Release 5.4, any client of a shared archiving file system that runs on Solaris 11 or higher can attach tape drives and carry out tape I/O on behalf of the file system. Distributing tape I/O across these *datamover* hosts greatly reduces server overhead, improves file-system performance, and allows significantly more flexibility when scaling SAM-QFS implementations. As your archiving needs increase, you now have the option of either replacing SAM-QFS metadata servers with more powerful systems (vertical scaling) or spreading the load across more clients (horizontal scaling).

Configure the Datamover Client To configure a client for distributed tape I/O, proceed as follows:

1. Connect all devices that will be used for distributed I/O to the client.
2. If you have not already done so, carry out the procedure "Connecting Tape Drives Using Persistent Bindings" on page 3-56. Then return here.
3. Log in to the shared archiving file system's metadata server as `root`.

   In the example, the host name is `samsharefs-mds`:

   ```
   [samsharefs-mds]root@solaris:~#
   ```

4. Make sure that the metadata server is running SAM-QFS Solaris 11 or higher.

   ```
   [samsharefs-mds]root@solaris:~# uname -r
   5.11
   [samsharefs-mds]root@solaris:~#
   ```

5. Make sure that all clients that serve as datamovers are running SAM-QFS Solaris 11 or higher.

   In the example, we open a terminal window for each client host, `samsharefs-client1` and `samsharefs-client2`, and log in remotely using `ssh`. The log-in banner displays the Solaris version:

   ```
   [samsharefs-mds]root@solaris:~# ssh root@samsharefs-client1
   ...
   Oracle Corporation      SunOS 5.11      11.1    September 2013
   [samsharefs-client1]root@solaris:~#

   [samsharefs-mds]root@solaris:~# ssh root@samsharefs-client2
   ...
   Oracle Corporation      SunOS 5.11      11.1    September 2013
   [samsharefs-client2]root@solaris:~#
   ```

6. On the metadata server, open the file `/etc/opt/SUNWsamfs/defaults.conf` in a text editor, enable distributed I/O by uncommenting the line `distio =` and setting the value to `on`.

   By default, `distio` is `off` (disabled).

   In the example, we open the file in the `vi` editor and add the line:

   ```
   [samsharefs-mds]root@solaris:~#  vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults.  To change the default behavior, uncomment the
   # appropriate line (remove the '#' character from the beginning of the line)
   # and change the value.
   ...
   distio = on
   ```

7. Next, identify the device types that should participate in distributed I/O. To use device type *dev* with distributed I/O, add the line *dev*`_distio = on` to the `defaults.conf` file. To exclude device type *dev* from distributed I/O, add the line *dev*`_distio = off`. Save the file, and close the editor.

   By default, SAM-QFS T10000 drives and LTO drives are allowed to participate in distributed I/O (`ti_distio = on` and `li_distio = on`), while all other types are excluded. In the example, we exclude LTO drives:

   ```
   [samsharefs-mds]root@solaris:~#  vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults.  To change the default behavior, uncomment the
   # appropriate line (remove the '#' character from the beginning of the line)
   # and change the value.
   ...
   distio = on
   li_distio = off
   :wq
   [samsharefs-mds]root@solaris:~#
   ```

8. On each client that will serve as a datamover, edit the `defaults.conf` file so that it matches the file on the server.

   In the example, we edit the `defaults.conf` file on client `samsharefs-client1` using `vi`, save the file, and close the editor:

```
[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client1
Password:
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
...
distio = on
li_distio = off
:wq
[samsharefs-client1]root@solaris:~#
[samsharefs-mds]root@solaris:~#
```

9. On each client that will serve as a datamover, open the /etc/opt/SUNWsamfs/mcf file in a text editor. Add all of the tape devices that the metadata server is using for distributed tape I/O. Make sure that the device order and equipment numbers are identical to those in the mcf file on the metadata server.

   In the example, we edit the mcf file on client samsharefs-client1 using vi:

```
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
 /dev/rmt/60cbn           901       ti                    on
 /dev/rmt/61cbn           902       ti                    on
 /dev/rmt/62cbn           903       ti                    on
 /dev/rmt/63cbn           904       ti                    on
```

10. If the tape library listed in the /etc/opt/SUNWsamfs/mcf file on the metadata server is configured on the client that will serve as a datamover, specify the library family set as the family set name for the tape devices that are being used for distributed tape I/O. Save the file.

    In the example, the library is configured on the host, so we use the family set name library1 for the tape devices:

```
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
/dev/scsi/changer/c1t0d5 900       rb        library1     on
 /dev/rmt/60cbn           901       ti        library1     on
 /dev/rmt/61cbn           902       ti        library1     on
 /dev/rmt/62cbn           903       ti        library1     on
 /dev/rmt/63cbn           904       ti        library1     on
:wq
[samsharefs-client1]root@solaris:~#
```

11. If the tape library listed in the /etc/opt/SUNWsamfs/mcf file on the metadata server is *not* configured on the client that will serve as a datamover, use a hyphen (-) as the family set name for the tape devices that are being used for distributed tape I/O.

    In the example, the library is not configured on the host:

```
[samsharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
/dev/rmt/60cbn           901       ti        -           on
/dev/rmt/61cbn           902       ti        -           on
/dev/rmt/62cbn           903       ti        -           on
/dev/rmt/63cbn           904       ti        -           on
:wq
[samsharefs-client2]root@solaris:~#
```

12. If you need to enable or disable distributed tape I/O for particular archive set copies, open the server's `/etc/opt/SUNWsamfs/archiver.cmd` file in a text editor and add the `-distio` parameter to the copy directive. Set `-distio on` to enable or `off` to disable distributed I/O. Save the file, and close the editor.

    In the example, we use the vi editor to turn distributed I/O `off` for copy `1` and `on` for copy `2`:

    ```
    [samsharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
    # archiver.cmd
    # Generated by config api Mon Nov 22 14:31:39 2013
    ...
    #
    # Copy Parameters Directives
    params
    allsets -sort path -offline_copy stageahead
    allsets.1 -startage 10m -startsize 500M -startcount 500000 -distio off
    allsets.2 -startage 24h -startsize 20G  -startcount 500000 -distio on
    :wq
    [samsharefs-mds]root@solaris:~#
    ```

13. On each host, check the `mcf` file for errors by running the `sam-fsd` command, and correct any errors found.

    The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error. In the example, we check the `mcf` file on the Linux client, `sharefs-clientL`:

    ```
    [sharefs-clientL][root@Linux ~]# sam-fsd
    ```

14. On the server, tell the SAM-QFS software to read the modified configuration files and reconfigure itself accordingly. Use the command `samd config`, and correct any errors found.

    In the example, we run the `samd config` command on the server, `sharefs-mds`:

    ```
    [samsharefs-mds]root@solaris:~#  samd config
    ```

15. Stop here.

Connecting Tape Drives Using Persistent Bindings

When you add a host that serves as either a potential metadata server or a distributed I/O datamover client, you must configure removable media devices using persistent bindings. The Solaris operating system attaches drives to the system device tree in the order in which it discovers the devices at startup. This order may or may not reflect the order in which devices are discovered by other file system hosts or the order in which they are physically installed in the tape library. So you need to bind the devices to the new host in the same way that they are bound to the other hosts and in the same order in which they are installed in the removable media library.

The procedures below outline the required steps (for full information, see the `devfsadm` and `devlinks` man pages and the administration documentation for your version of the Solaris operating system):

- If you have moved, added, or removed drives in a library or replaced or reconfigured the library associated with an archiving SAM-QFS shared file system, Update Persistent Bindings to Reflect Changes to the Hardware Configuration.
- If you are adding a new metadata server or datamover client to an archiving SAM-QFS shared file system, Persistently Bind a New File System Host to Removable Media Devices

Update Persistent Bindings to Reflect Changes to the Hardware Configuration

1. Log in to the active metadata server host as `root`.

```
[sharefs-mds]root@solaris:~#
```

2. Create a new drive-mapping file as described in "Determining the Order in Which Drives are Installed in the Library" on page 5-10.

   In the example, the `device-mappings.txt` file looks like this:

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    -----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn ->  ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

3. Open the `/etc/devlink.tab` file in a text editor.

   In the example, we use the `vi` editor:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
# This is the table used by devlinks
# Each entry should have 2 fields; but may have 3.  Fields are separated
# by single tab ('\t') characters.
...
```

4. Using the `device-mappings.txt` file as a guide, add a line to the `/etc/devlink.tab` file that remaps a starting node in the Solaris tape device tree, `rmt/`*`node-number`*, to the first drive in the library. The line should be in the form `type=ddi_byte:tape; addr=`*`device_address`*`,0;   rmt/node-number\M0`, where *`device_address`* is the physical address of the device and *`node-number`* is the device's position in the Solaris device tree. Choose a node number that is high enough to avoid conflicts with any devices that Solaris configures automatically (Solaris starts from node `0`).

   In the example, we note that the device address for the first device in the library, `1`, is `w500104f0008120fe` and see that the device is currently attached to the host at `rmt/1`:

```
[sharefs-mds] vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
```

```
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    -----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

So we create a line in `/etc/devlink.tab` that remaps `rmt/60` to the number `1` drive in the library, `w500104f0008120fe`:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
:w
```

5. Continue to add lines to the `/etc/devlink.tab` file for each tape device that is assigned for SAM-QFS archiving, so that the drive order in the device tree on the metadata server matches the installation order on the library. Save the file, and close the editor.

   In the example, we note the order and addresses of the three remaining devices—library drive `2` at `w500104f00093c438`, library drive `3` at `w500104f000c086e1`, and library drive `4` at `w500104f000c086e1`:

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
...
   2    /dev/rmt/0cbn -> ../../devices/pci@8\.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8\.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8\.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8\.../st@w500104f000b6d98d,0:cbn
```

   Then we map the device addresses to next three Solaris device nodes, maintaining the same order as in the library:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
:wq
[sharefs-mds]root@solaris:~#
```

6. Delete all existing links to the tape devices in `/dev/rmt`.

```
[sharefs-mds]root@solaris:~# rm /dev/rmt/*
```

7. Create new, persistent tape-device links from the entries in the `/etc/devlink.tab` file. Use the command `devfsadm -c tape`.

   Each time that the `devfsadm` command runs, it creates new tape device links for devices specified in the `/etc/devlink.tab` file using the configuration specified by the file. The `-c tape` option restricts the command to creating new links for tape-class devices only:

```
[sharefs-mds]root@solaris:~# devfsadm -c tape
```

8. Repeat the operation on each potential metadata server and datamover in the shared file system configuration. In each case, add the same lines to the `/etc/devlink.tab` file, delete the links in `/dev/rmt`, and run `devfsadm -c tape`.

In the example, we use `ssh` to log in to each host in turn, and configure the same four logical devices, `rmt/60\M0`, `rmt/61\M0`, `rmt/62\M0`, and `rmt/63\M0`:

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
Password:
[sharefs-mds_alt]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
:wq
[sharefs-mds_alt]root@solaris:~# rm /dev/rmt/*
[sharefs-mds_alt]root@solaris:~# devfsadm -c tape
[sharefs-mds_alt]root@solaris:~# exit
sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
:wq
[sharefs-client1]root@solaris:~# rm /dev/rmt/*
[sharefs-client1]root@solaris:~# devfsadm -c tape
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

9. Return to "Configuring Datamover Clients for Distributed Tape I/O" on page 3-53 or "Configuring Additional File System Clients" on page 3-42.

Persistently Bind a New File System Host to Removable Media Devices

1. Log in to the host as `root`.

   ```
   [sharefs-mds]root@solaris:~#
   ```

2. If the physical order of the drives in the media library has changed since the existing file-system hosts were configured, create a new mapping file as described in "Determining the Order in Which Drives are Installed in the Library" on page 5-10.

   In the example, the `device-mappings.txt` file looks like this:

   ```
   [sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
   LIBRARY SOLARIS          SOLARIS
   DEVICE  LOGICAL          PHYSICAL
   NUMBER  DEVICE           DEVICE
   ------- -------------    -----------------------------------------------------
      2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
      1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
      3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
      4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
   ```

3. Open the `/etc/devlink.tab` file in a test editor.

   In the example, we use the `vi` editor:

   ```
   [sharefs-mds]root@solaris:~# vi /etc/devlink.tab
   # Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
   # This is the table used by devlinks
   ```

```
# Each entry should have 2 fields; but may have 3.  Fields are separated
# by single tab ('\t') characters.
...
```

4. Using the `device-mappings.txt` file as a guide, remap a starting node in the Solaris tape device tree, `rmt/`*`node-number`*, to the first drive in the library. Add a line to the `/etc/devlink.tab` file of the form `type=ddi_byte:tape; addr=`*`device_address`*`,0;    rmt/`*`node-number`*`\M0`, where: *`device_address`* is the physical address of the device and *`node-number`* is the device's position in the Solaris device tree. Choose a node number that is high enough to avoid conflicts with any devices that Solaris configures automatically (Solaris starts from node `0`).

   In the example, we note that the device address for the first device in the library, `1`, is `w500104f0008120fe` and see that the device is currently attached to the host at `rmt/1`:

```
[sharefs-mds] vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    -----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

   So we create a line in `/etc/devlink.tab` that remaps `rmt/60` to the number `1` drive in the library, `w500104f0008120fe`:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
:w
```

5. Continue to add lines to the `/etc/devlink.tab` file for each tape device that is assigned for SAM-QFS archiving, so that the drive order in the device tree on the metadata server matches the installation order on the library. Save the file.

   In the example, we note the order and addresses of the three remaining devices—library drive `2` at `w500104f00093c438`, library drive `3` at `w500104f000c086e1`, and library drive `4` at `w500104f000c086e1`:

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
...
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

   Then we map the device addresses to the next three Solaris device nodes, maintaining the same order as in the library:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
:wq
[sharefs-mds]root@solaris:~#
```

6. Delete all existing links to the tape devices in `/dev/rmt`.

   ```
   [sharefs-mds]root@solaris:~# rm /dev/rmt/*
   ```

7. Create new, persistent tape-device links from the entries in the `/etc/devlink.tab` file. Use the command `devfsadm -c tape`.

   Each time that the `devfsadm` command runs, it creates new tape device links for devices specified in the `/etc/devlink.tab` file using the configuration specified by the file. The `-c tape` option restricts the command to creating new links for tape-class devices only:

   ```
   [sharefs-mds]root@solaris:~# devfsadm -c tape
   ```

8. On each potential metadata server and datamover in the shared file system configuration, add the same lines to the `/etc/devlink.tab` file, delete the links in `/dev/rmt`, and run `devfsadm -c tape`.

   In the example, we use `ssh` to log in to the potential metadata server host `sharefs-mds_alt` and the client host `sharefs-client1`. We then configure the same four logical devices, `rmt/60\M0`, `rmt/61\M0`, `rmt/62\M0`, and `rmt/63\M0`, on each:

   ```
   [sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
   Password:
   [sharefs-mds_alt]root@solaris:~# vi /etc/devlink.tab
   ...
   type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
   type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
   type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
   type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
   :wq
   [sharefs-mds_alt]root@solaris:~# rm /dev/rmt/*
   [sharefs-mds_alt]root@solaris:~# devfsadm -c tape
   [sharefs-mds_alt]root@solaris:~# exit
   [sharefs-mds]root@solaris:~# ssh root@sharefs-client1
   Password:
   [sharefs-client1]root@solaris:~# vi /etc/devlink.tab
   ...
   type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60\M0
   type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61\M0
   type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62\M0
   type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63\M0
   :wq
   [sharefs-client1]root@solaris:~# rm /dev/rmt/*
   [sharefs-client1]root@solaris:~# devfsadm -c tape
   [sharefs-client1]root@solaris:~# exit
   [sharefs-mds]root@solaris:~#
   ```

9. Return to "Configuring Datamover Clients for Distributed Tape I/O" on page 3-53 or "Configuring Additional File System Clients" on page 3-42.

Switching from the Active Metadata Server to a Potential Metadata Server

The procedures in this section move the metadata service for the file system from the current host (the active metadata server) to a standby host (the potential metadata server). Which procedure you use depends on the health of the server host that you are replacing:

- Activate a Potential Metadata Server to Replace a Faulty Active Metadata Server

- Activate a Potential Metadata Server to Replace a Healthy Active Metadata Server

Activate a Potential Metadata Server to Replace a Faulty Active Metadata Server

This procedure lets you move the metadata service off of an active metadata server host that has stopped functioning. It activates a potential metadata server, even if a file system is still mounted. Proceed as follows:

> **Caution:** Never activate a potential metadata server until you have stopped, disabled, or disconnected the faulty metadata server!
>
> To activate a potential server when a file system is mounted and the active metadata server is down, you have to invoke the `samsharefs` command with the `-R` option, which acts on raw devices rather than on file-system interfaces. So, if you activate a potential metadata server while the faulty server is still connected to the devices, the faulty server can corrupt the file system.

1. If the active metadata server is faulty, make sure that it cannot access the metadata devices before you do anything else. Power the affected host off, halt the host, or disconnect the failed host from the metadata devices.

2. Wait at least until the maximum lease time has run out, so that all client read, write, and append leases can expire.

3. Log in to a potential metadata server as `root`.

   In the example, we log in to the potential metadata server `sharefs-mds_alt`:

   ```
   [sharefs-mds_alt]root@solaris:~#
   ```

4. Activate the potential metadata server. From the potential metadata server, issue the command `samsharefs -R -s` *`server file-system`*, where *`server`* is the host name of the potential metadata server and *`file-system`* is the name of the SAM-QFS shared file system.

   In the example, the potential metadata server is `sharefs-mds_alt` and the file system name is `sharefs`:

   ```
   [sharefs-mds_alt]root@solaris:~# samsharefs -R -s sharefs-mds_alt sharefs
   ```

5. If you need to check the integrity of a file system and repair possible problems, unmount the file system now using the procedure "Unmount a Shared File System" on page 3-41.

6. If you have unmounted the file system, perform the file system check. Use the command `samfsck -F` *`file-system`*, where `-F` specifies repair of errors and where *`file-system`* is the name of the file system.

   In the example, we check and repair the file system name is `sharefs`:

   ```
   [sharefs-mds_alt]root@solaris:~# samfsck -F sharefs
   ```

7. Stop here.

Activate a Potential Metadata Server to Replace a Healthy Active Metadata Server

You can move the metadata service off of a healthy, active metadata server host and on to a newly activated potential metadata server when required. For example, you might transfer metadata services to an alternate host to keep file systems available while you upgrade or replace the original server host or some of its components. Proceed as follows:

1. Log in to both the active and potential metadata servers as `root`.

   In the example, we log in to the active metadata server, `sharefs-mds`. Then, in a second terminal window, we use secure shell (`ssh`) to log in to the potential metadata server `sharefs-mds_alt`:

   ```
   [sharefs-mds]root@solaris:~#

   [sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
   Password:
   [sharefs-mds-alt]root@solaris:~#
   ```

2. If the active metadata server mounts a SAM-QFS archiving file system, finish active archiving and staging jobs and stop any new activity before proceeding further. See "Idle Archiving and Staging Processes" on page 3-22.

3. If the active metadata server mounts a SAM-QFS archiving file system, idle removable media drives and stop the library-control daemon. See"Stop Archiving and Staging Processes" on page 3-23.

4. If you use a `crontab` entry to run the recycler process, remove the entry and make sure that the recycler is not currently running.

5. Activate the potential metadata server. From the potential metadata server, issue the command `samsharefs -s` *`server file-system`*, where *`server`* is the host name of the potential metadata server and *`file-system`* is the name of the SAM-QFS shared file system.

   In the example, the potential metadata server is `sharefs-mds_alt` and the file system name is `sharefs`:

   ```
   [sharefs-mds_alt]root@solaris:~# samsharefs -s sharefs-mds_alt sharefs
   ```

6. Load the configuration files and start SAM-QFS processes on the potential metadata server. Use the command `samd config`.

   For archiving shared file systems, the `samd config` command restarts archiving processes and the library control daemon. But shared file system clients that are waiting for files to be staged from tape to the primary disk cache must reissue the stage requests.

7. If you still need to use a `crontab` entry to run the recycler process, restore the entry.

8. Stop here.

Converting an Unshared File System to a Shared File System

To convert an unshared file system to a shared file system, carry out the following tasks:

- Create a Hosts File on the Active and Potential Metadata Servers
- Share the Unshared File System and Configure the Clients

Create a Hosts File on the Active and Potential Metadata Servers

On each metadata server, you must create a hosts file that lists network address information for the servers and clients of a shared file system. The hosts file is stored alongside the `mcf` file in the `/etc/opt/SUNWsamfs/` directory. During the initial creation of a shared file system, the `sammkfs -S` command configures sharing using the settings stored in this file. So create it now, using the procedure below.

1. Gather the network host names and IP addresses for the hosts that will share the file system as clients.

   In the examples below, we will share the `samqfs1` file system with the clients `samqfs1-mds_alt` (a potential metadata server), `samqfs1-client1`, and `samqfs1-client2`.

2. Log in to the metadata server as `root`.

   In the example, we log in to the host `samqfs1-mds`:

   ```
   [samqfs1-mds]root@solaris:~#
   ```

3. Using a text editor, create the file `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`* on the metadata server, replacing *`family-set-name`* with the name of the family-set name of the file-system that you intend to share.

   In the example, we create the file `hosts.samqfs1` using the `vi` text editor. We add some optional headings, starting each line with a hash sign (`#`), indicating a comment:

   ```
   [samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
   # /etc/opt/SUNWsamfs/hosts.samqfs1
   #                                             Server   On/  Additional
   #Host Name           Network Interface       Ordinal  Off  Parameters
   #------------------  ----------------------  -------  ---  ----------
   ```

4. Enter the host name of the metadata server in the first column and the corresponding IP address or domain name the second. Separate the columns with whitespace characters.

   In the example, we enter the host name and IP address of the metadata server, `samqfs1-mds` and `10.79.213.117`, respectively:

   ```
   [samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
   # /etc/opt/SUNWsamfs/hosts.samqfs1
   #                                             Server   On/  Additional
   #Host Name           Network Interface       Ordinal  Off  Parameters
   #------------------  ----------------------  -------  ---  ----------
   samqfs1-mds          10.79.213.117
   ```

5. Add a third column, separated from the network address by whitespace characters. In this column, enter the ordinal number of the server (`1` for the active metadata server, `2` for the first potential metadata server, and so on).

   In this example, there is only one metadata server, so we enter `1`:

   ```
   [samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
   # /etc/opt/SUNWsamfs/hosts.samqfs1
   #                                             Server   On/  Additional
   #Host Name           Network Interface       Ordinal  Off  Parameters
   #------------------  ----------------------  -------  ---  ----------
   samqfs1-mds          10.79.213.117           1
   ```

6. Add a fourth column, separated from the server ordinal number by whitespace characters. In this column, enter `0` (zero).

   A `0`, `-` (hyphen), or blank value in the fourth column indicates that the host is `on`—configured with access to the shared file system. A `1` (numeral one) indicates that the host is `off`—configured but without access to the file system (for information on using these values when administering shared file systems, see the samsamqfs1 man page).

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
# /etc/opt/SUNWsamfs/hosts.samqfs1
#                                            Server   On/  Additional
#Host Name            Network Interface      Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
samqfs1-mds          10.79.213.117            1        0
```

7. Add a fifth column, separated from the on/off status column by whitespace characters. In this column, enter the keyword `server` to indicate the currently active metadata server:

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
# /etc/opt/SUNWsamfs/hosts.samqfs1
#                                            Server   On/  Additional
#Host Name            Network Interface      Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
samqfs1-mds           10.79.213.117           1        0    server
```

8. If you plan to include one or more hosts as a potential metadata servers, create an entry for each. Increment the server ordinal each time. But do not include the `server` keyword (there can be only one active metadata server per file system).

   In the example, the host `samqfs1-mds_alt` is a potential metadata server with the server ordinal `2`. Until and unless we activate it as a metadata server, it will be a client:

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
# /etc/opt/SUNWsamfs/hosts.samqfs1
#                                            Server   On/  Additional
#Host Name            Network Interface      Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
samqfs1-mds           10.79.213.117           1        0    server
samqfs1-mds_alt       10.79.213.217           2        0
```

9. Add a line for each client host, each with a server ordinal value of `0`.

   A server ordinal of `0` identifies the host as a client. In the example, we add two clients, `samqfs1-client1` and `samqfs1-client2`.

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
# /etc/opt/SUNWsamfs/hosts.samqfs1
#                                            Server   On/  Additional
#Host Name            Network Interface      Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
samqfs1-mds           10.79.213.17            1        0    server
samqfs1-mds_alt       10.79.213.7             2        0
samqfs1-client1       10.79.213.33            0        0
samqfs1-client2       10.79.213.47            0        0
```

10. Save the `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`* file, and quit the editor.

    In the example, we save the changes to `/etc/opt/SUNWsamfs/hosts.samqfs1` and exit the `vi` editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.samqfs1
# /etc/opt/SUNWsamfs/hosts.samqfs1
#                                            Server   On/  Additional
#Host Name            Network Interface      Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
```

```
samqfs1-mds          10.79.213.117          1        0    server
samqfs1-mds          10.79.213.117          1        0    server
samqfs1-mds_alt      10.79.213.217          2        0
samqfs1-client1      10.79.213.133          0        0
samqfs1-client2      10.79.213.147          0        0
:wq
[samqfs1-mds]root@solaris:~#
```

11. Place a copy of the new `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`* file on any potential metadata servers that are included in the shared file-system configuration.

    In the examples, we place a copy on the host `samqfs1-mds_alt`:

    ```
    [samqfs1-mds]root@solaris:~# sftp root@samqfs1-mds_alt
    Password:
    sftp> cd /etc/opt/SUNWsamfs/
    sftp> put /etc/opt/SUNWsamfs/hosts.samqfs1
    sftp> bye
    [samqfs1-mds]root@solaris:~#
    ```

12. Now Share the Unshared File System and Configure the Clients.

Share the Unshared File System and Configure the Clients

1. Log in to the metadata server as `root`.

   In the example, we log in to the host `samqfs1-mds`:

   ```
   [samqfs1-mds]root@solaris:~#
   ```

2. If you do not have current backup copies of the system files and configuration files, create backups now. See "Backing Up the SAM-QFS Configuration" on page 6-4.
3. If you do not have a current file-system recovery point file and a recent copy of the archive log, create them now. See "Backing Up File Systems" on page 6-1.

   If you set up an automated backup process for the file system during initial configuration, you may not need additional backups.
4. If you are converting an archiving file system, finish active archiving and staging jobs and stop any new activity before proceeding further. See "Idle Archiving and Staging Processes" on page 3-22 and "Stop Archiving and Staging Processes" on page 3-23.
5. Unmount the file system. Use the command `umount` *`family-set-name`*, where *`family-set-name`* is the family-set name of the file-system that you intend to share.

   For more information on mounting and unmounting SAM-QFS file systems, see the `mount_samfs` man page. In the example, we unmount the `samqfs1` file system:

   ```
   [samqfs1-mds]root@solaris:~# umount samqfs1
   [samqfs1-mds]root@solaris:~#
   ```

6. Convert the file system to a SAM-QFS shared file system. Use the command `samfsck -S -F` *`file-system-name`*, where *`file-system-name`* is the family-set name of the file system.

   In the example, we convert the file system named `samqfs1`:

   ```
   [samqfs1-mds]root@solaris:~# samfsck -S -F samqfs1
   ```

7. Open the /etc/opt/SUNWsamfs/mcf file in a text editor, and locate the line for the file system.

   In the example, we use the vi editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
samqfs1              200        ma         samqfs1  on
 /dev/dsk/c0t0d0s0    201        mm         samqfs1  on
 /dev/dsk/c0t3d0s0    202        md         samqfs1  on
 /dev/dsk/c0t3d0s1    203        md         samqfs1  on
```

8. In the mcf file, add the shared parameter to the additional parameters field in the last column of the file system entry. Then save the file and close the editor.

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
samqfs1              200        ma         samqfs1  on       shared
 /dev/dsk/c0t0d0s0    201        mm         samqfs1  on
 /dev/dsk/c0t3d0s0    202        md         samqfs1  on
 /dev/dsk/c0t3d0s1    203        md         samqfs1  on
:wq
[samqfs1-mds]root@solaris:~#
```

9. Open the /etc/vfstab file in a text editor, and locate the line for the file system.

   In the example, we use the vi editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
samqfs1    -        /samqfs1  samfs   -     yes
```

10. In the /etc/vfstab file, and add the shared mount option to mount options field in the last column of the file system entry. Then save the file and close the editor.

```
[samqfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
samqfs1    -        /samqfs1  samfs   -     yes      shared
:wq
[samqfs1-mds]root@solaris:~#
```

11. Initialize the shared file system and host configuration. Use the command samsharefs -u -R *family-set-name*, where *family-set-name* is the family-set name of the file system.

```
[samqfs1-mds]root@solaris:~# samsharefs -u -R samqfs1
```

12. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly:

```
[samqfs1-mds]root@solaris:~# samd config
```

13. Mount the shared file system on the metadata server.

```
[samqfs1-mds]root@solaris:~# mount /samqfs1
```

14. If your hosts are configured with multiple network interfaces, see "Use Local Hosts Files to Route Network Communications" on page 3-68.

15. Add any required clients to the newly shared file system, using the procedures outlined in "Configuring Additional File System Clients" on page 3-42.

Use Local Hosts Files to Route Network Communications

Individual hosts do not require local hosts files. The file system's global file on the metadata server identifies the active metadata server and the network interfaces of active and potential metadata servers for all file system hosts (see "Create a Hosts File on the Active and Potential Metadata Servers" on page 3-63). But local hosts files can be useful when you need to selectively route network traffic between file-system hosts that have multiple network interfaces.

Each file-system host identifies the network interfaces for the other hosts by first checking the `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`* file on the metadata server, where *`family-set-name`* is the name of the file system family specified in the `/etc/opt/SUNWsamfs/mcf` file. Then it checks for its own, specific `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`*`.local` file. If there is no local hosts file, the host uses the interface addresses specified in the global hosts file in the order specified in the global file. But if there is a local hosts file, the host compares it with the global file and uses only those interfaces that are listed in both files in the order specified in the local file. By using different addresses in each file, you can thus control the interfaces used by different hosts.

To configure local hosts files, use the procedure outlined below:

1. On the metadata server host and on each potential metadata server host, create a copy of the global hosts file, `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`*, as described in "Create a Hosts File on the Active and Potential Metadata Servers" on page 3-63.

   For the examples in this section, the shared file system, `sharefs2`, includes an active metadata server, `sharefs2-mds`, and a potential metadata server, `sharefs2-mds_alt`, each with two network interfaces. There are also two clients, `sharefs2-client1` and`sharefs2-client2`.

   We want the active and potential metadata servers to communicate with each other via private network addresses and with the clients via host names that Domain Name Service (DNS) can resolve to addresses on the public, local area network (LAN). So `/etc/opt/SUNWsamfs/hosts.sharefs2`, the file system's global host file, specifies a private network address in the `Network Interface` field of the entries for the active and potential servers and a host name for the interface address of each client. The file looks like this:

```
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129            1        0    server
```

```
sharefs2-mds_alt      172.16.0.130            2        0
sharefs2-client1      sharefs2-client1        0        0
sharefs2-client2      sharefs2-client2        0        0
```

2. Create a local hosts file on each of the active and potential metadata servers, using the path and file name `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`*`.local`, where *`family-set-name`* is the name specified for the shared file system in the `/etc/opt/SUNWsamfs/mcf` file. *Only include interfaces for the networks that you want the active and potential servers to use.*

   In the example, we want the active and potential metadata servers to communicate with each other over the private network, so the local hosts file on each server, `hosts.sharefs2.local`, lists only private addresses for active and potential servers:

```
[sharefs2-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129            1        0    server
sharefs2-mds_alt      172.16.0.130            2        0
:wq
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-mds_alt
Password:
[sharefs2-mds_alt]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129            1        0    server
sharefs2-mds_alt      172.16.0.130            2        0
:wq
[sharefs2-mds_alt]root@solaris:~# exit
[sharefs2-mds]root@solaris:~#
```

3. Create a local hosts file on each of the clients, using the path and file name `/etc/opt/SUNWsamfs/hosts.`*`family-set-name`*`.local`, where *`family-set-name`* is the name specified for the shared file system in the `/etc/opt/SUNWsamfs/mcf` file. *Only include interfaces for the networks that you want the clients to use.*

   In our example, we want the clients to communicate with the server only via the public network. So the file includes only the host names of the active and potential metadata servers:

```
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-client1
Password:
[sharefs2-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                             Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ----------------------  -------  ---  ----------
sharefs2-mds          sharefs2-mds            1        0    server
sharefs2-mds_alt      sharefs2-mds_alt        2        0
:wq
[sharefs2-client1]root@solaris:~# exit
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-client2
Password:
[sharefs2-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                             Server   On/  Additional
```

```
#Host Name           Network Interface       Ordinal  Off   Parameters
#------------------  ----------------------  -------  ---   ----------
sharefs2-mds         sharefs2-mds            1        0     server
sharefs2-mds_alt     sharefs2-mds_alt        2        0
:wq
[sharefs2-client2]root@solaris:~# exit
[sharefs2-mds]root@solaris:~#
```

4. If you started this procedure while finishing the configuration of the server, add clients. Go to "Configuring Additional File System Clients" on page 3-42.

Converting a Shared File System to an Unshared File System

When you need to unshare a file system, proceed as follows:

Convert a Shared Metadata Server to an Unshared System

1. Log in to the metadata server as `root`.

   In the example, we log in to the host `samqfs1-mds`:

   ```
   [samqfs1-mds]root@solaris:~#
   ```

2. Remove the clients from the metadata server configuration using the procedure "Remove the Host from the File System Hosts File" on page 3-51.

3. If you do not have current backup copies of the system files and configuration files, create backups now. See "Backing Up the SAM-QFS Configuration" on page 6-4.

4. If you do not have a current file-system recovery point file and a recent copy of the archive log, create them now. See "Backing Up File Systems" on page 6-1.

   If you set up an automated backup process for the file system during initial configuration, you may not need additional backups.

5. If you are converting an archiving file system, finish active archiving and staging jobs and stop any new activity before proceeding further. See "Idle Archiving and Staging Processes" on page 3-22 and "Stop Archiving and Staging Processes" on page 3-23.

6. Unmount the file system. Use the command `umount` *`family-set-name`*, where *`family-set-name`* is the name specified for the shared file system in the `/etc/opt/SUNWsamfs/mcf` file.

   For more information on mounting and unmounting SAM-QFS file systems, see the `mount_samfs` man page. In the example, we unmount the `samqfs1` file system:

   ```
   [samqfs1-mds]root@solaris:~# umount samqfs1
   ```

7. Convert the SAM-QFS shared file system to an unshared file system. Use the command `samfsck -F -U` *`file-system-name`*, where *`file-system-name`* is the name specified for the shared file system in the `/etc/opt/SUNWsamfs/mcf` file.

   In the example, we convert the file system named `samqfs1`:

   ```
   [samqfs1-mds]root@solaris:~# samfsck -F -U samqfs1
   ```

8. Open the `/etc/opt/SUNWsamfs/mcf` file in a text editor, and locate the line for the file system.

   In the example, we use the `vi` editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
samqfs1              200        ma         samqfs1  on       shared
 /dev/dsk/c0t0d0s0    201        mm         samqfs1  on
 /dev/dsk/c0t3d0s0    202        md         samqfs1  on
 /dev/dsk/c0t3d0s1    203        md         samqfs1  on
```

9. In the mcf file, delete the shared parameter from the additional parameters field in the last column of the file system entry. Then save the file and close the editor.

```
[samqfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
samqfs1              200        ma         samqfs1  on
 /dev/dsk/c0t0d0s0    201        mm         samqfs1  on
 /dev/dsk/c0t3d0s0    202        md         samqfs1  on
 /dev/dsk/c0t3d0s1    203        md         samqfs1  on
:wq
[samqfs1-mds]root@solaris:~#
```

10. Open the /etc/vfstab file in a text editor, and locate the line for the file system.

    In the example, we use the vi editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
samqfs1    -        /samqfs1  samfs   -     yes      shared
```

11. In the /etc/vfstab file, delete the shared mount option from the mount options field in the last column of the file system entry. Then save the file and close the editor.

    In the example, we use the vi editor:

```
[samqfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
samqfs1    -        /samqfs1  samfs   -     yes
:wq
[samqfs1-mds]root@solaris:~#
```

12. Delete the file /etc/opt/SUNWsamfs/hosts.*file-system-name*.

13. Tell the SAM-QFS software to re-read the mcf file and reconfigure itself accordingly:

```
[samqfs1-mds]root@solaris:~# samd config
```

14. Mount the file system.

```
[samqfs1]root@solaris:~# mount /samqfs1
```

15. Stop here.

4

Managing Files and Directories

This chapter addresses the following topics:

- Setting SAM-QFS File Attributes
- Accommodating Large Files
- Working with Linear Tape File System (LTFS) Volumes
- Managing WORM Files
- Managing Directories and Files in SMB/CIFS Shares
- Access Control Lists

Setting SAM-QFS File Attributes

The ability to interact with users via a familiar interface—the standard UNIX file system—is a key advantage of SAM-QFS. Most users do not even need to be aware of the differences. However, SAM-QFS file systems can provide advanced users with significantly greater capabilities when necessary. SAM-QFS *file attributes* let users optimize the behavior of the file system for working with individual files and directories. Users who understand their workloads and the characteristics of their data can significantly improve performance file-by-file. Users can, for example, specify direct or buffered I/O based on the characteristics of the data in a given file or directory. They can preallocate file-system space so that large files can be written more sequentially and can specify the stripe width used when writing particular files or directories.

The `setfa` command sets these file attributes on both new and existing files and directories. The command creates specified files or directories that do not exist. When applied to a directory, it sets the specified attributes on all files and subdirectories in the directory. Subsequently created files and directories inherit these attributes.

The basic tasks are outlined below (for additional information, see the `setfa` man page).

- Restore Default File Attribute Values
- Preallocate File-System Space
- Specify Round-Robin or Striped Allocation for a File
- Allocate File Storage on a Specified Stripe-Group Device.

Restore Default File Attribute Values

1. Log in to the file system host.

```
user@solaris:~#
```

2. To reset the default attribute values on a file, use the command `setfa -d` *`file`*, where *`file`* is the path and name of the file.

   In the example, we reset defaults on the file `/samfs1/data/2014/03/series3.15`:

   ```
   user@solaris:~# setfa -d /samfs1/data/2014/03/series3.15
   ```

3. To recursively reset the default attribute values on a directory and all of its contents, use the command `setfa -r` *`directory`*, where *`directory`* is the path and name of the directory.

   In the example, we reset the defaults on the subdirectory `/samfs1/data/2014/02`:

   ```
   user@solaris:~# setfa -r /samfs1/data/2014/02/
   ```

4. Stop here.

Preallocate File-System Space

Preallocating space for a file insures that there is enough room to write out the entire file sequentially when the file is written. Writing and reading large files in sequential blocks improves efficiency and overall performance by reducing the overhead associated with seeking and with buffering smaller, more scattered blocks of data. Preallocation is thus best for writing a predictable number of large blocks of data. Preallocated but unused space remains part of the file when the file closes and cannot be freed for other use until the entire file is deleted.

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. If you need to preallocate space for writing to an existing file that already contains data, use the command `setfa -L` *`number-bytes file`*, where *`number-bytes`* is an integer or an integer plus `k` for kilobytes, `m` for megabytes, or `g` for gigabytes, and where *`file`* is the name of the file.

   The command `setfa -L` uses standard allocation.It supports striping. Pre-allocated files can grow beyond their pre-allocated size. In the example, we preallocate 121 megabytes for the existing file `tests/series119b`:

   ```
   user@solaris:~# setfa -L 121m tests/series119b
   ```

3. If you need to preallocate space for writing a new file that has no storage blocks assigned, use the command `setfa -l` *`number-bytes file`*, where:

   - `l` is the lower-case letter "L".
   - *`number-bytes`* is an integer or an integer plus `k` for kilobytes, `m` for megabytes, or `g` for gigabytes.
   - *`file`* is the name of the file.

   The command `setfa -l` preallocates the specified number of bytes. The resulting files are fixed at their preallocated size and can neither grow beyond nor shrink below their pre-allocated size. In the example, we create the file `data/2014/a3168445` and preallocate two gigabytes of space for its content:

   ```
   user@solaris:~# setfa -l 2g data/2014/a3168445
   ```

4. Stop here.

Specify Round-Robin or Striped Allocation for a File

By default, SAM-QFS file systems use the allocation method specified for the file system at mount time. But users can specify a preferred allocation method—round-robin or striping with a specified stripe-width—for specified directories or files.

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. To specify round-robin allocation, specify a stripe width of `0` (zero). Use the command `setfa -s 0` *`directory-or-file`*, where *`directory-or-file`* is the name of the directory or file that will be written using the specified allocation method.

   A stripe width of `0` (zero) specifies unstriped, round-robin allocation. The file system starts writing a file on the next available device. It writes successive disk allocation units (DAUs) to the file on the same device until the file is complete or the device runs out of space. If the device runs out of space, the file system moves to the next available device and continues to write disk allocation units. The process repeats until the file is complete. In the example, we specify round-robin allocation for all files written to the `data/field-reports` directory:

   ```
   user@solaris:~# setfa -s 0 data/field-reports
   ```

3. To specify striped allocation, specify a stripe width. Use the command `setfa -s` *`stripe-width directory-or-file`*, where *`stripe-width`* is an integer in the range [`1-255`] and *`directory-or-file`* is the name of the directory or file that will be written using the specified allocation method.

   A stripe width in the range [`1-255`] specifies striped allocation. The file system writes the number of disk allocation units (DAUs) specified in the stripe width to multiple devices in parallel until the file is complete. In the example, we specify striped allocation with a stripe width of `1` for all files written to the directory, so the file allocation for all files written to the `data/field-reports` directory `data/2014/`, so the file system will write one disk allocation unit to each available device until the file is complete:

   ```
   user@solaris:~# setfa -s 1 data/2014/
   ```

4. Stop here.

Allocate File Storage on a Specified Stripe-Group Device

A user can specify the *stripe group* device where round-robin or striped allocation should begin. A SAM-QFS stripe group is a logical volume that stripes data across multiple physical volumes. When round-robin file allocation is in effect, the entire file is written on the designated stripe group. When striped allocation is in effect, the first allocation is made on designated stripe group.

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. To write an entire file to a specific stripe group, use round-robin allocation. Use the command `setfa -s 0 -g`*`stripe-group-number`*, where *`stripe-group-number`* is an integer in the range [`0-127`] that identifies the specified stripe group.

   In the example, we specify that file `reports/site51` will be written using round-robin allocation starting on stripe group `0`:

```
user@solaris:~# setfa -s 0 -g0 reports/site51
```

3. To stripe a file across a number of stripe groups starting from a specified stripe group, use striped allocation. Use the command `setfa -s` *`stripe-width`* `-g`*`stripe-group-number`*, where *`stripe-width`* is an integer in the range [`1-255`] that specifies a number of disk allocation units and *`stripe-group-number`* is the integer in the range [`0-127`] that identifies the specified stripe group.

   In the example, we specify that file `assessments/site52` will be striped across stripe groups, one disk allocation unit per group, starting from stripe group `0`:

   ```
   user@solaris:~# setfa -s 0 -g0 assessments/site52
   ```

4. Stop here.

Accommodating Large Files

SAM-QFS file systems are particularly well suited to working with unusually large files. This section covers the following topics:

- Managing Disk Cache With Very Large Files
- Segmenting Files
- Using Removable Media Files for Large Data Sets

Managing Disk Cache With Very Large Files

When manipulating very large files, pay careful attention to the size of available disk cache. If you try to write a file that is larger than your disk cache, non-archiving file systems return an `ENOSPC` error, while an archiving file system simply waits for space that may never become available, causing the application to block.

SAM-QFS provides two possible alternatives to increasing the size of the disk cache:

- Segmenting Files so that users stage only part of a large file to disk at any given time
- Using Removable Media Files for Large Data Sets so that users never stage data to disk.

Segmenting Files

When you set the SAM-QFS segmentation attribute on a file, the file system breaks the file down into segments of a specified size and manages access requests so that, at any given time, only the currently required segment resides on disk. The remainder of the file resides on removable media.

Segmentation of large files has a number of advantages:

- Users can create and access files that are larger than the available disk cache.

  Since only segments reside in cache at any given time, you only need to choose a segment size that fits in the disk cache. The complete file can grow to any size that the media can accommodate.
- Users can access large files that have been released from the disk cache more quickly. Staging a portion of a large file to disk is much faster than waiting for the entire file to stage.
- The speed and efficiency of archiving can improve when files are segmented, because only changed portions of each file are re-archived.

- Files can be striped across removable media volumes mounted on multiple drives. Archiving and staging operations can then proceed in parallel, further improving performance.

There are two limitations:

- You cannot segment files in a shared file system.
- You cannot segment binary executable files, because the Solaris memory-mapping function, `mmap()`, cannot map the bytes in a segmented file to the address space of a process.

To create segmented files, proceed as follows:

Segment a File

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. Select or, if necessary, create the file(s) that you need to segment.
3. To segment a single file, use the command `segment [-s stage_ahead] -l` *`segment_size file-path-name`*, where:

   - `stage_ahead` (optional) is an integer specifying the number of consecutive extra segments to read when a given segment is accessed. Well-chosen values can improve utilization of the system page cache and thus improve I/O performance. The default is `0` (disabled).
   - `segment_size` is an integer and a unit that together specify the size of each segment. Supported units are `k` (kilobytes), `m` (megabytes, and `g` (gigabytes). The minimum size is one megabyte (`1m` or `1024k`).
   - *`file-path-name`* is the path and file name of the file.

   For full details, see the `segment` man page. In the example, we segment the file `201401.dat` using a 1.5-megabyte (`1536k`) segment size:

   ```
   user@solaris:~# segment -l 1536k 201401.dat
   ```

4. To recursively segment the files in a directory and its subdirectories, use the command `segment [-s` *`stage_ahead`*`] -l` *`segment_size`* `-r` *`directory-path-name`*, where *`directory-path-name`* is the path and name of the starting directory.

   In the example, we segment all files in the `/samqfs1/data` directory and its subdirectories using a 1-megabyte (`1m`) segment size:

   ```
   user@solaris:~# segment -l 1m -r /samqfs1/data
   ```

5. Stop here.

Stripe a Segmented File Across Multiple Volumes

You configure segmented files for striped I/O by assigning them to an archive set that specifies multiple drives. Proceed as follows:

1. Log in to the host as `root`.

   ```
   root@solaris:~#
   ```

2. Open the file `/etc/opt/SUNWsamfs/archiver.cmd` in a text editor.

   In the example, we use the `vi` editor to open the file:

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# Configuration file for SAM-QFS archiving file systems ...
```

3. To stripe segmented files across drives, specify the use of at least two drives for each copy of each archive set that contains segmented files. In the `archiver.cmd` file, locate the `params` section, and make sure that the parameters for each copy include the `-drives` *`number`* parameter, where *`number`* is two (`2`) or more. Make any required changes, save the file, and close the editor.

   In the example, the `archiver.cmd` file specifies two drives for all three copies of all configured archive sets:

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# Configuration file for SAM-QFS archiving file systems ...
...
#-----------------------------------------------------------------------
# Copy Parameters
params
allsets -sort path -offline_copy stageahead -reserve set
allsets.1 -startage 10m -drives 2
allsets.2 -startage 24h -drives 2
allsets.3 -startage 48h -drives 2
endparams
...
:wq
root@solaris:~#
```

4. Check the `archiver.cmd` file for errors. Use the command `archiver -lv`.

   The `archiver -lv` command prints the `archiver.cmd` file to screen and generates a configuration report if no errors are found. Otherwise, it notes any errors and stops.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
...
 Total space available:  300T
root@solaris:~#
```

5. Tell the SAM-QFS software to re-read the `archiver.cmd` file and reconfigure itself accordingly. Use the `/opt/SUNWsamfs/sbin/samd config` command.

```
root@solaris:~# samd config
```

6. Stop here.

Using Removable Media Files for Large Data Sets

SAM-QFS *removable media files* reside entirely on removable media and thus never occupy space in the file-system disk cache. The file system reads removable media files from removable media directly into memory. So the storage medium does not limit the size of the file at all. Removable files that exceed the capacity of a single media cartridge can become multiple-cartridge, *volume overflow files*. The file system reads and writes data to the media sequentially.

In most respects, removable media files look like typical UNIX files. They have permissions, a user name, a group name, and a file size. When a user or application requests a removable media file, the system automatically mounts the corresponding volume(s) and the user accesses the data from memory, much as if the data were on disk. But removable media files differ from other SAM-QFS files in two major ways:

they are never archived by the Storage Archive Manager software, and they are not supported over NFS.

The Storage Archive Manager software does not manage removable media files. The files are never archived or released, and the media that contains them is never recycled. This makes removable media files useful when you need to use removable media for purposes other than archiving. These files are ideal for creating removable disaster-recovery volumes that backup your SAM-QFS configuration and metadata dump files. You can also read data from foreign volumes (volumes created by other applications) by loading the volume read-only and reading the files into memory as removable media files.

Since removable media files cannot be released and the associated volume(s) cannot be recycled. you should generally segregate removable media files on dedicated volumes, rather than mixing them in with archive copies.

Create a Removable Media or Volume Overflow File

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. Select the SAM-QFS file system, path, and file name for the removable media file.

   Once the removable media file is created, the file system will address requests for this path and file name using data from removable media.

3. Create the removable media file. Use the command `request -m` *`media-type`* `-v` *`volume-specifier data-file`*, where:*`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types", *`data-file`* is the path and name that you selected for the removable media file, and *`volume-specifier`* is one of the following:

   - a volume serial number or a slash-delimited list of volume serial numbers

     In the first example, we create `file1` on LTO (`li`) volume `VOL080`:

     ```
     user@solaris:~# request -m li -v VOL080 /samqfs1/data/file1
     ```

     In the second example, we create `file2` on LTO (`li`) volumes `VOL080`, `VOL082`, and `VOL098`:

     ```
     user@solaris:~# request -m li -v VOL081/VOL082/VOL098 /samqfs1/data/file2
     ```

   - `-l` *`volume-list-file`*, where *`volume-list-file`* is the path and name of a file that, on each line, lists a single volume serial number and, optionally, a space and a decimal or hexadecimal number specifying a starting position on the specified volume (prefix hexadecimals with `0x`).

     In the example, using the `vi` editor, we create `file3` on the LTO (`li`) volumes listed in the file `vsnsfile3`:

     ```
     user@solaris:~# vi vsnsfile3
     VOL180
     VOL181
     VOL182
     :wq
     user@solaris:~# request -m li -v -l vsnsfile3 /samqfs1/data/file3
     ```

4. Stop here.

Read a Foreign Tape Volume as a Removable Media File

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. Make sure that the foreign tape is barcoded, write protected, opened read-only, and positioned to `0`.
3. Select the SAM-QFS file system, path, and file name for the removable media file.

   Once the removable media file is created, the file system will address requests for this path and file name using data from the foreign tape.

4. Create the removable media file using the `-N` (foreign media) option. Use the command `request -m` *`media-type`* `-N -v` *`volume-serial-number`* *`data-file`*, where:

   - *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
   - *`volume-serial-number`* is the volume serial number of the foreign tape.
   - *`data-file`* is the path and name for the removable media file.

   In the example, we create a removable media file for the foreign LTO (`li`) volume `FOR991`:

   ```
   user@solaris:~# request -m li -N -v FOR991 /samqfs1/foreignfile
   ```

5. Stop here.

Working with Linear Tape File System (LTFS) Volumes

Linear Tape File System is a self-describing tape format that organizes the data on sequential-access tape media into a file system, so that files can be accessed much as if they resided on random-access disk. Starting with Release 5.4, SAM-QFS provides extensive support for LTFS. The software lets you use LTFS files in SAM-QFS file systems and supplies tools for creating, accessing, and managing LTFS media.

This section addresses the following topics:

- Importing LTFS Media Into the Library
- Attaching LTFS Directories and Files to a SAM-QFS File System
- Accessing LTFS Media Using SAM-QFS Software
- Managing LTFS Media Using SAM-QFS Software

Importing LTFS Media Into the Library

The SAM-QFS software automatically recognizes LTFS media. So you can import LTFS volumes with the `samimport` command, just as you would any other media. See "Importing and Exporting Removable Media" on page 5-2 and the `samimport` man page for additional information.

Attaching LTFS Directories and Files to a SAM-QFS File System

The SAM-QFS software can attach Linear Tape File System (LTFS) directories and files to a SAM-QFS file system, so that they can be accessed and managed as if they were themselves SAM-QFS files. The software copies the LTFS meta-data from the LTFS volume to an empty directory in a SAM-QFS file system. Using this metadata, SAM-QFS manages the LTFS media and files as it would an archived SAM-QFS file.

LTFS files are staged from the LTFS media to the SAM-QFS disk cache for use, either when users access them or all at once, as soon as the LTFS metadata is in place. The SAM-QFS file system's archiving and space-management policies apply as they would for any SAM-QFS file.

This section describes the following tasks:

- Making LTFS Files Accessible On Demand
- Making LTFS Files Immediately Accessible in the Disk Cache

Making LTFS Files Accessible On Demand

When you *attach* LTFS files to a SAM-QFS file system, the SAM-QFS software copies file-system metadata from the LTFS volume to a specified directory in the SAM-QFS file system. Files will then be staged to the disk cache when users access them. To attach LTFS files, proceed as follows:

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. In the SAM-QFS file system that will host the LTFS files, create the directory that will hold the LTFS metadata.

   In the example, we create the directory `ltfs1/` under the file-system mount point `/samqfs1`:

   ```
   user@solaris:~# mkdir /samqfs1/ltfs1
   user@solaris:~#
   ```

3. Attach the LTFS files to the SAM-QFS file system. Use the command `samltfs attach` *`LTFS-media-type`*`.`*`LTFS-volume-serial-number SAMQFS-directory`*, where:

   - *`LTFS-media-type`* is the two-character media type code for the type of media that holds the LTFS data (see Appendix A, "Glossary of Equipment Types").
   - *`LTFS-volume-serial-number`* is the six-character, alphanumeric volume serial number of the LTFS volume.
   - The specified media type and volume serial number identify a volume that the catalog lists as an LTFS volume.

     In the SAM-QFS catalog, LTFS media are unlabeled and marked `non-SAM` and `tfs`.
   - *`SAMQFS-directory`* is the path and name of the directory that will hold LTFS metadata.

   In the example, we attach LTO (`li`) volume `TFS233`:

   ```
   user@solaris:~# samltfs attach li.TFS233 /samqfs1/ltfs1
   user@solaris:~#
   ```

4. Stop here.

Making LTFS Files Immediately Accessible in the Disk Cache

When you *ingest* LTFS files into a SAM-QFS file system, the SAM-QFS software copies file-system metadata from the LTFS volume to a specified directory in the SAM-QFS file system and immediately stages all files to the disk cache. To ingest LTFS files, proceed as follows:

1. Log in to the file system host.

```
user@solaris:~#
```

2. In the SAM-QFS file system that will host the LTFS files, create the directory that will hold the LTFS metadata.

   In the example, we create the directory `ltfs2/` under the file-system mount point `/samqfs1`:

   ```
   user@solaris:~# mkdir /samqfs1/ltfs2
   user@solaris:~#
   ```

3. Attach the LTFS files to the SAM-QFS file system. Use the command `samltfs ingest` *`LTFS-media-type`*`.`*`LTFS-volume-serial-number SAMQFS-directory`*, where:

   - *`LTFS-media-type`* is the two-character media type code for the type of media that holds the LTFS data (see Appendix A, "Glossary of Equipment Types").
   - *`LTFS-volume-serial-number`* is the six-character, alphanumeric volume serial number of the LTFS volume.
   - The specified media type and volume serial number identify a volume that the catalog lists as an LTFS volume.

     In the SAM-QFS catalog, LTFS media are unlabeled and marked `non-SAM` and `tfs`.
   - *`SAMQFS-directory`* is the path and name of the directory that hold LTFS metadata.

   In the example, we ingest LTO (`li`) volume `TFS234`:

   ```
   user@solaris:~# samltfs ingest li.TFS234 /samqfs1/ltfs2
   user@solaris:~#
   ```

4. Stop here.

Accessing LTFS Media Using SAM-QFS Software

SAM-QFS software can also load and unload LTFS media and mount or dismount LTFS file systems on the host using the LTFS mount point specified in the SAM-QFS `defaults.conf` file.

- Loading an LTFS Volume Into a Tape Drive and Mounting the LTFS File System
- Unmounting an LTFS File System and Unloading the Volume from the Tape Drive
- Displaying LTFS Configuration and Status Information.

Loading an LTFS Volume Into a Tape Drive and Mounting the LTFS File System

1. Log in to the file system host.

   ```
   user@solaris:~#
   ```

2. Load the LTFS volume into a tape drive and mount the file system on the mount point specified in the `defaults.conf` file. Use the command `samltfs load` *`LTFS-media-type`*`.`*`LTFS-volume-serial-number`*, where:

   - *`LTFS-media-type`* is the two-character media type code for the type of media that holds the LTFS data (see Appendix A, "Glossary of Equipment Types").
   - *`LTFS-volume-serial-number`* is the six-character, alphanumeric volume serial number of the LTFS volume.

- The specified media type and volume serial number identify a volume that the catalog lists as an LTFS volume.

  In the SAM-QFS catalog, LTFS media are unlabeled and marked `non-SAM` and `tfs`.

In the example, we load LTO (`li`) volume `TFS434` and mount it on the directory specified in the `defaults.conf` file, `/mnt/ltfs`:

```
user@solaris:~# samltfs load li.TFS234
```

3. Stop here.

Unmounting an LTFS File System and Unloading the Volume from the Tape Drive

1. Log in to the file system host.

```
user@solaris:~#
```

2. Unmount the LTFS file system and unload the corresponding volume from the tape drive. Use the command `samltfs unload` *`LTFS-media-type.LTFS-volume-serial-number`*, where:

- *`LTFS-media-type`* is the two-character media type code for the type of media that holds the LTFS data (see Appendix A, "Glossary of Equipment Types").
- *`LTFS-volume-serial-number`* is the six-character, alphanumeric volume serial number of the LTFS volume.
- The specified media type and volume serial number identify an LTFS volume that the catalog lists as an LTFS volume.

  In the SAM-QFS catalog, LTFS media are unlabeled and marked `non-SAM` and `tfs`.

In the example, we unmount the LTFS file system and unload LTO (`li`) volume `TFS435`:

```
user@solaris:~# samltfs load li.TFS435
```

3. Stop here.

Managing LTFS Media Using SAM-QFS Software

The SAM-QFS software provides the basic tools needed for creating, erasing, and validating LTFS media:

- Format a Volume as an LTFS File System
- Erase LTFS Data and Remove LTFS Formatting and Partitions from a Volume
- Check the Integrity of an LTFS File System

Format a Volume as an LTFS File System

1. Log in to the file system host.

```
user@solaris:~#
```

2. Partition and format a removable media volume for the LTFS file system. Use the command `samltfs mkltfs` *`media-type.volume-serial-number`*, where:

- *`media-type`* is the two-character media type code for an LTFS-compatible type of media (see Appendix A, "Glossary of Equipment Types").

- *volume-serial-number* is the six-character alphanumeric volume serial number of the volume.

In the example, we partition LTO (`li`) volume `VOL234` and format it as an LTFS volume:

```
user@solaris:~# samltfs mkltfs li.VOL234
```

3. Stop here.

Erase LTFS Data and Remove LTFS Formatting and Partitions from a Volume

1. Log in to the file system host.

```
user@solaris:~#
```

2. Erase the LTFS volume and restore it to general use. Use the command `samltfs unltfs` *media-type.volume-serial-number*, where:

   - *media-type* is the two-character media type code for an LTFS-compatible type of media (see Appendix A, "Glossary of Equipment Types").
   - *volume-serial-number* is the six-character alphanumeric volume serial number of the volume.

   In the example, we erase the LTFS file system data and metadata and remove the partitions on LTO (`li`) volume `VOL234`:

```
user@solaris:~# samltfs unltfs li.VOL234
```

3. Stop here.

Check the Integrity of an LTFS File System

1. Log in to the file system host.

```
user@solaris:~#
```

2. Check the integrity of the LTFS file system. Use the command `samltfs ltfsck` *LTFS-media-type.LTFS-volume-serial-number*, where:

   - *LTFS-media-type* is the two-character media type code for the type of media that holds the LTFS data (see Appendix A, "Glossary of Equipment Types").
   - *LTFS-volume-serial-number* is the six-character, alphanumeric volume serial number of the LTFS volume.
   - The specified media type and volume serial number identify an LTFS volume that the catalog lists as an LTFS volume.

     In the SAM-QFS catalog, LTFS media are unlabeled and marked `non-SAM` and `tfs`.

   In the example, we check the LTFS file system on LTO (`li`) volume `VOL234`:

```
user@solaris:~# samltfs ltfsck li.VOL234
```

3. Stop here.

Displaying LTFS Configuration and Status Information

To display the configuration and status of LTFS, use the command `samltfs status`.

```
user@solaris:~# samltfs status
```

Managing WORM Files

Once a file system has been configured with write-once read-many (WORM) support as described in the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide* (*Customer Documentation Library*, `docs.oracle.com`), you can WORM-enable directories and files. This section starts with a brief overview of the SAM-QFS implementation of WORM, "Understanding WORM File Systems". It then provides instructions for carrying out the following tasks:

- Activate WORM Retention for a File
- Find and List WORM Files

Understanding WORM File Systems

WORM files are used in many applications for legal and archival reasons. WORM-enabled SAM-QFS file systems support default and customizable file-retention periods, data and path immutability, and subdirectory inheritance of the WORM setting. Depending on how your file systems are configured, you use one of two SAM-QFS WORM modes:

- standard compliance mode (the default)

  The standard WORM mode starts the WORM retention period when a user sets UNIX `setuid` permission on a directory or non-executable file (`chmod 4000` *`directory|file`*). Since setting `setuid` (*set user ID upon execution*) permission on an executable file presents security risks, files that also have UNIX execute permission cannot be retained using this mode.

- emulation mode

  The WORM emulation mode starts the WORM retention period when a user makes a writable file or directory read-only (`chmod 555` *`directory|file`*), so executable files can be retained.

Both standard and emulation modes have a strict WORM implementation and a less restrictive, *lite* implementation that relaxes some restrictions for `root` users. Both strict and lite implementations do not allow changes to data or paths once retention has been triggered on a file or directory. The strict implementations do not let anyone shorten the specified retention period (by default, 43,200 minutes=30 days) or delete files or directories prior to the end of the retention period. They also do not let anyone use `sammkfs` to delete volumes that hold currently retained files and directories. The strict implementations are thus well-suited to meeting legal and regulatory compliance requirements. The lite implementations let `root` users shorten retention periods, delete files and directories, and delete volumes using the `sammkfs` command.

You create WORM directories and files using a *trigger* action. File systems mounted with the `worm_capable` or `worm_lite` mount options use the standard compliance-mode trigger, while those mounted with `worm_emul` or `emul_lite` options use the emulation-mode trigger.

Trigger actions have different effects on directories and files. When applied to a directory, the trigger action enables WORM support for the directory but does not otherwise affect the user's ability to create and edit files within the directory. You can also delete WORM-enabled directories, as long as they do not contain WORM files. When applied to a file within a WORM-enabled directory, the trigger sets a retention period for the file, and the file system no longer allows modifications to the file data or the path to the data. When the retention period expires, you can only delete the file or extend the retention time. Consequently, you must use care when applying the WORM trigger, because changes are irrevocable at the file system level.

You can create both hard and soft links to WORM files. You can only create hard links with files that reside in a WORM-capable directory. After a hard link is created, it has the same WORM characteristics as the original file. Soft links can also be established, but a soft link cannot use the WORM features. Soft links to WORM files can be created in any directory in a SAM-QFS file system.

Activate WORM Retention for a File

1. Log in to the file-system server.

   ```
   user@solaris:~#
   ```

2. See if the directory that holds the file has been WORM-enabled. Use the command `sls -Dd` *`directory`*, where *`directory`* is the path and name of the directory. Look for the attribute `worm-capable` in the output of the command.

   For full information on the command, see the `sls` man page. Usually, directories will be WORM-enabled, because, when one user WORM-enables a directory, all current and future child directories inherit the WORM capability. In the first example, we find that our target directory, `/samqfs1/records`, is already worm-enabled:

   ```
   user@solaris:~# sls -Dd /samqfs1/records/2013/
   /samma1/records/2013:
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:      0  inode:     1048.1
     project: user.root(1)
     access:      Mar  3 12:15  modification: Mar  3 12:15
     changed:     Mar  3 12:15  attributes:   Mar  3 12:15
     creation:    Mar  3 12:15  residence:    Mar  3 12:15
     worm-capable        retention-period: 0y, 30d, 0h, 0m
   ```

   But in the second example, we find that our target directory, `/samqfs1/documents`, is *not* worm-enabled:

   ```
   user@solaris:~# sls -Dd /samqfs1/documents
   /samqfs1/documents
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:      0  inode:     1049.1
     project: user.root(1)
     access:      Mar  3 12:28  modification: Mar  3 12:28
     changed:     Mar  3 12:28  attributes:   Mar  3 12:28
     creation:    Mar  3 12:28  residence:    Mar  3 12:28
   ```

3. If the directory is not WORM-enabled and if the file system was mounted with the `worm_capable` or `worm_lite` mount option, use the standard compliance-mode trigger to WORM-enable the directory. Use the Solaris command `chmod 4000` *`directory-name`*, where *`directory-name`* is the path and name of the directory that will hold the WORM files.

   The command `chmod 4000` sets the `setuid` (`set user ID upon execution`) attribute on the file, which is the trigger action for standard compliance mode. In the example, we WORM-enable the directory `/samqfs1/documents` and check the result with `sls -Dd`:

   ```
   user@solaris:~# chmod 4000 /samqfs1/documents
   user@solaris:~# sls -Dd /samqfs1/documents
   /samqfs1/documents
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:      0  inode:     1049.1
   ```

```
  project: user.root(1)
  access:      Mar  3 12:28  modification: Mar  3 12:28
  changed:     Mar  3 12:28  attributes:   Mar  3 12:28
  creation:    Mar  3 12:28  residence:    Mar  3 12:28
  worm-capable        retention-period: 0y, 30d, 0h, 0m
```

4. If the directory is not WORM-enabled and if the file system was mounted with the `worm_emul` or `emul_lite` mount option, use the emulation-mode trigger to WORM-enable the directory. Use the Solaris command `chmod 555` *`directory-name`*, where *`directory-name`* is the path and name of the directory that will hold the WORM files.

   The command `chmod 555` removes write permissions for the directory, which is the trigger action for emulation mode. In the example, we WORM-enable the directory `/samqfs1/documents` and check the result using the command `sls -Dd`:

```
user@solaris:~# chmod 555 /samqfs1/documents
user@solaris:~# sls -Dd /samqfs1/documents
/samqfs1/documents
  mode: drwxr-xr-x    links:   2  owner: root        group: root
  length:      4096  admin id:      0  inode:     1049.1
  project: user.root(1)
  access:      Mar  3 12:28  modification: Mar  3 12:28
  changed:     Mar  3 12:28  attributes:   Mar  3 12:28
  creation:    Mar  3 12:28  residence:    Mar  3 12:28
  worm-capable        retention-period: 0y, 30d, 0h, 0m
```

5. If you need to retain the file for some period other than the default for the file system, specify the required retention time by changing the access time for the file. Use the Solaris command `touch -a -t`*`expiration-date`*, where *`expiration-date`* is a string of numerals consisting of a four-digit year, a two-digit month, a two-digit day of the month, a two-digit hour of the day, a two digit minute within the hour, and, optionally, a two-digit second within the minute.

   Note that Oracle Solaris UNIX utilities such as `touch` cannot extend a retention period beyond 10:14 PM on 01/18/2038. These utilities use signed 32–bit numbers to represent time in seconds starting from 01/01/1970. So use a default retention period if you need to retain files beyond this cut-off date.

   In the example, we set the retention period to expire on March 4, 2018 at 11:59 AM for the file `/samqfs1/plans/master.odt`:

```
user@solaris:~# touch -a -t201803041159  /samqfs1/plans/master.odt
```

6. If the file system was mounted with the `worm_capable` or `worm_lite` mount option, use the standard compliance trigger to activate WORM retention for a file. Use the Solaris command `chmod 4000` *`directory-name`*, where *`directory-name`* is the path and name of the directory that will hold the WORM files.

   The trigger action for standard compliance mode, `chmod 4000`, sets the `setuid` (*set user ID upon execution*) attribute on the specified file. Setting this attribute on an executable file is insecure. So, if the file system was mounted with the `worm_capable` or `worm_lite` mount option, you cannot set WORM protections on UNIX-executable files.

   In the example, we activate WORM retention for the file `master.odt`. We check the result with `sls -D`, and note that `retention` is now `active`:

```
user@solaris:~# chmod 4000 /samqfs1/plans/master.odt
user@solaris:~# sls -Dd /samqfs1/plans/master.odt
/samqfs1/plans/master.odt:
  mode: -r-xr-xr-x    links:   1  owner: root        group: root
  length:       104  admin id:      0  inode:     1051.1
  project: user.root(1)
  access:      Mar  4 2018  modification: Mar  3 13:14
  changed:     Mar  3 13:16  retention-end: Apr  2 14:16 2014
  creation:    Mar  3 13:16  residence:    Mar  3 13:16
  retention:   active        retention-period: 4y, 0d, 0h, 0m
```

7. If the file system was mounted with the `worm_emul` or `emul_lite` mount option, use the emulation-mode trigger to activate WORM retention for a file. Use the Solaris command `chmod 555` *`directory-name`*, where *`directory-name`* is the path and name of the directory that will hold the WORM files.

   The command `chmod 555` removes write permissions for the directory, which is the trigger action for emulation mode. In the example, we activate WORM retention for the file `master-plan.odt`. We check the result with `sls -D`, and note that `retention` is now `active`:

```
user@solaris:~# chmod 555 /samqfs1/plans/master.odt
user@solaris:~# sls -Dd /samqfs1/plans/master.odt
/samqfs1/plans/master.odt:
  mode: -r-xr-xr-x    links:   1  owner: root        group: root
  length:       104  admin id:      0  inode:     1051.1
  project: user.root(1)
  access:      Mar  4 2018  modification: Mar  3 13:14
  changed:     Mar  3 13:16  retention-end: Apr  2 14:16 2014
  creation:    Mar  3 13:16  residence:    Mar  3 13:16
  retention:   active        retention-period: 4y, 0d, 0h, 0m
```

8. Stop here.

Find and List WORM Files

To find and list WORM files that meet specified search criteria, use the `sfind` command. Proceed as follows:

1. Log in to the file-system server.

```
user@solaris:~#
```

2. To list files that are WORM protected and being actively retained, use the command `sfind` *`starting-directory`* -ractive, where *`starting-directory`* is the path and name for the directory where you want the listing process to start.

```
user@solaris:~# sfind /samqfs1/ -ractive
/samqfs1/documents/2013/master-plan.odt
/samqfs1/documents/2013/schedule.ods
/samma1/records/2013/progress/report01.odt
/samma1/records/2013/progress/report02.odt
/samma1/records/2013/progress/report03.odt ...
user@solaris:~#
```

3. To list WORM protected files for which the retention period has expired, use the command `sfind` *`starting-directory`* `-rover`, where *`starting-directory`* is the path and name for the directory where you want the listing process to start.

```
user@solaris:~# sfind /samqfs1/  -rover
/samma1/documents/2007/master-plan.odt
/samma1/documents/2007/schedule.ods
user@solaris:~#
```

4. To list WORM protected files for which the retention period will expire after a specified date and time, use the command `sfind` *`starting-directory`* `-rafter` *`expiration-date`*, where *`starting-directory`* is the path and name for the directory where you want the listing process to start and *`expiration-date`* is a string of numerals consisting of a four-digit year, a two-digit month, a two-digit day of the month, a two-digit hour of the day, a two digit minute within the hour, and, optionally, a two-digit second within the minute.

   In the example, we list any files for which the retention period expires after January 1, 2015 at one minute after midnight:

   ```
   user@solaris:~# sfind /samqfs1/ -rafter 201501010001
   /samqfs1/documents/2013/master-plan.odt
   user@solaris:~#
   ```

5. To list WORM protected files that must remain in the file system for at least a specified amount of time, use the command `sfind` *`starting-directory`* `-rremain` *`time-remaining`*, where *`time-remaining`* is a string of non-negative integers paired with the following labels representing units of time:

   - `y` for years
   - `d` for days
   - `h` for hours
   - `m` for minutes.

   In the example, we find all files under the directory `/samqfs1/` that will be retained for at least three more years:

   ```
   user@solaris:~# sfind /samqfs1/ -rremain 3y
   /samqfs1/documents/2013/master-plan.odt
   user@solaris:~#
   ```

6. To list WORM protected files that must remain in the file system for more than a specified amount of time, use the command `sfind` *`starting-directory`* `-rlonger` *`time`*, where *`time`* is a string of non-negative integers paired with the following labels representing units of time:

   - `y` for years
   - `d` for days
   - `h` for hours
   - `m` for minutes.

   In the example, we find all files under the directory `/samqfs1/` that will be retained for more than three years and ninety days:

   ```
   user@solaris:~# sfind /samqfs1/ -rremain 3y90d
   /samqfs1/documents/2013/master-plan.odt
   user@solaris:~#
   ```

7. To list WORM protected files that must remain in the file system permanently, use the command `sfind` *`starting-directory`* `-rpermanent`.

   In the example, we find that no files under the directory `/samqfs1/` are being retained permanently:

   ```
   user@solaris:~# sfind /samqfs1/ -rpermanent
   user@solaris:~#
   ```

8. Stop here.

Managing Directories and Files in SMB/CIFS Shares

This section addresses the following tasks:

- Managing Extended System Attributes in SMB/CIFS Shares
- Access Control Lists

Managing Extended System Attributes in SMB/CIFS Shares

Extended file attributes support SMB/CIFS file sharing by associating SAM-QFS files with non-UNIX metadata that can be interpreted by Microsoft Windows file systems. This section starts with a brief overview of the extended attributes supported by SAM-QFS. It then provides basic instructions for the following tasks:

- Display Extended System Attributes
- Modify Extended System Attributes

SAM-QFS Supported Extended System Attributes

Extended system attributes are Boolean (true or false) values expressed by an attribute `name` with the value `true` or the negation of the name, `noname`, with the value `false`. SAM-QFS provides the following extended system attributes in support of SMB/CIFS file sharing:

- `appendonly` means that users can only append data to the file. `noappendonly` means that this restriction is not in effect.
- `archive` means that the file has changed since it was last copied/backed up. `noarchive` means that the file has not changed since it was last copied/backed up. SAM-QFS does not currently use this attribute.
- `hidden` means that the file is not displayed in file listings by default. `nohidden` means that the file is displayed by default.
- `immutable` means that the directory or file and its contents cannot be changed or deleted. `noimmutable` means that the directory or file can be changed or deleted.
- `nodump` means that the file cannot be backed up. `nonodump` means that the file can be backed up. Oracle Solaris does not use this attribute.
- `nounlink` means that the file or the directory and its contents cannot be deleted or renamed. `nonounlink` means that the file or the directory and its contents can be deleted or renamed.
- `offline` means that the file has been released from a SAM-QFS file system. Microsoft Windows systems will not preview the file. `nooffline` means that the file is online and has not been released from a SAM-QFS file system.
- `readonly` means that the file cannot be deleted or modified. `noreadonly` means that the file can be deleted or modified. The attribute is ignored when applied to directories.
- `sparse` means that the stored file contains only non-zero data, with zeroes reduced to ranges that are restored by the file system when the file is accessed or copied to a file system that does not support sparse files. `nosparse` means that the file is not sparse.

- `system` means that the file is critical to the Microsoft Windows operating system, must not be altered, deleted, and should not be displayed in listings by default. `nosystem` means that the file is not a system file.

Display Extended System Attributes

To view the extended system attributes of a SAM-QFS file, use the Solaris command `ls -/v` *`file`*, where *`file`* is the path and name of the file.

In the example, we list extended attributes for the file `/samqfs1/documents/master-plan.odt`:

```
user@solaris:~# ls -/v /samqfs1/documents/master-plan.odt
-rw-r--r--   1 root root  40560 Mar 4 15:52 /samqfs1/documents/master-plan.odt
{archive,nohidden,noreadonly,nosystem,noappendonly,nonodump,noimmutable,nonounlink
, nooffline,nosparse}
user@solaris:~#
```

Modify Extended System Attributes

To change a system attribute value for a file to a specified value, use the Solaris command `chmod S+v{`*`attributes`*`)`, where *`attributes`* is a comma-delimited list of SAM-QFS Supported Extended System Attributes.

See the `chmod` man page for a comprehensive explanation of syntax and available options. In the example, we change the archive attribute from `noarchive` (false) to `archive` (true):

```
root@solaris:~# ls -/v /samqfs1/documents/master-plan.odt
-r-xr-xr-x 1 root root 40561 Mar 4 15:52 /samqfs1/documents/master-plan.odt
{noarchive,nohidden,readonly,nosystem,noappendonly,nonodump,noimmutable,
nonounlink,offline,nosparse}
root@solaris:~# chmod S+v{archive} /samqfs1/documents/master-plan.odt
root@solaris:~# ls -/v /samqfs1/documents/master-plan.odt
-r-xr-xr-x 1 root root 40561 Mar 4 15:52 /samqfs1/documents/master-plan.odt
{archive,nohidden,readonly,nosystem,noappendonly,nonodump,noimmutable,
nonounlink,offline,nosparse}
```

Access Control Lists

An Access Control List (ACL) is a table that defines access permissions for a file or directory. Each record or Access Control Entry (ACE) in the table defines the access writes of a particular user, group, or class of users or groups. By default, new file systems that you create with SAM-QFS Release 5.4 use the Access Control List (ACL) implementation introduced in Network File System (NFS) version 4 and Solaris 11.

A comprehensive account of Solaris ACL administration, syntax, and usage is outside the scope of this document. For full information, see the chapter "Using ACLs and Attributes to Protect Oracle Solaris ZFS Files" in the volume *Oracle Solaris 11.1 Administration: ZFS File Systems*, available in the *Oracle Solaris 11.1 Information Library* at `docs.oracle.com` and the Solaris `ls` and `chmod` man pages.

5

Managing Libraries, Media, and Drives

This chapter covers the following topics:

- Managing Automated Media Libraries
- Managing Drives
- Managing Removable Media

Managing Automated Media Libraries

- Taking the Library On and Off Line
- Importing and Exporting Removable Media
- Maintaining Library Catalogs
- Determining the Order in Which Drives are Installed in the Library

Taking the Library On and Off Line

- Take the Library Offline
- Bring the Library Online

Take the Library Offline

If you need to stop SAM-QFS operations on only one library or if you need to power down a library, start by taking the library offline as described below:

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. Finish up active archiving and staging jobs and keep any new jobs from starting. See "Idle Archiving and Staging Processes" on page 3-22.

3. Stop drive and library activity. See "Stop Archiving and Staging Processes" on page 3-23.

4. Take the library offline. Use the command `samcmd off` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number assigned to the library in the `/etc/opt/SUNWsamfs/mcf` file.

   Placing a library in the `off` state stops I/O operations and removes the library from the control of the SAM-QFS software. Drives remain in the automated library remain in the `on` state. In the example, we take library `800` offline and check the result using `samcmd c`:

   ```
   root@solaris:~# samcmd off 800
   ```

```
root@solaris:~# samcmd c
Device configuration samcmd     5.4 14:34:04 Mar  7 2014
samcmd on samqfs1host
Device configuration:
ty   eq state   device_name                       fs   family_set
sn  800 off     /dev/scsi/changer/c1t2d0          800  lib800
li  801 on      /dev/rmt/0cbn                     800  lib800
li  802 on      /dev/rmt/1cbn                     800  lib800
li  803 on      /dev/rmt/2cbn                     800  lib800
li  804 on      /dev/rmt/3cbn                     800  lib800
hy  900 on      historian                         900
root@solaris:~#
```

5. When you are ready, Bring the Library Online.

Bring the Library Online

When a library enters the on state, t

1. Log in to the file system host as `root`.

```
root@solaris:~#
```

2. Bring the library online. Use the command `samcmd on` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number assigned to the library in the `/etc/opt/SUNWsamfs/mcf` file.

   The library comes online. SAM-QFS software queries the device state and updates the catalog as needed. In the example, we bring library **`800`** online and check the result using **`samcmd c`**:

```
root@solaris:~# samcmd on 800
root@solaris:~# samcmd c
Device configuration samcmd     5.4 15:04:14 Mar  7 2014
samcmd on samqfs1host
Device configuration:
ty   eq state   device_name                       fs   family_set
sn  800 on      /dev/scsi/changer/c1t2d0          800  lib800
li  801 on      /dev/rmt/0cbn                     800  lib800
li  802 on      /dev/rmt/1cbn                     800  lib800
li  803 on      /dev/rmt/2cbn                     800  lib800
li  804 on      /dev/rmt/3cbn                     800  lib800
hy  900 on      historian                         900
root@solaris:~#
```

3. Stop here.

Importing and Exporting Removable Media

Many automated libraries include a loading bay that lets you add or remove media cartridges without physically entering the library. Depending on the vendor, it may be called the mailbox, mailslot, media access port (MAP), or cartridge access port (CAP). This section describes the SAM-QFS commands that support this type of library:

- Import Removable Media Cartridges
- Export Removable Media Cartridges

If your library does *not* include a mailbox, consult the library vendor's documentation and your local site policies for instructions on adding and removing library media. When the library reinitializes following the change and audits its contents, the SAM-QFS software will update its library and historian catalogs automatically.

Import Removable Media Cartridges

If the library mailbox contains media cartridges when the SAM-QFS software starts, the software automatically loads them into the library. Once the software is running, you can import media from the mailbox at any time using the following procedure:

1. Place media cartridge(s) in the mailbox according to the library vendor's instructions.

2. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

3. Import the cartridge(s) into the automated library. Use the command `samimport` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number specified for the library in the `/etc/opt/SUNWsamfs/mcf` file.

   The SAM-QFS software assigns the media to storage slots and catalogs their locations. In the example, we import media into library `800`.

   ```
   root@solaris:~# samimport 800
   ```

4. Stop here.

Export Removable Media Cartridges

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. If required, add an informational note to the catalog record for a cartridge before exporting it. Use the command `chmed -I "`*`note`*`"` *`identifier`*, where *`note`* is a string of up to 128 characters and *`identifier`* is either of the following:

   - *`mediatype`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the six-character, alphanumeric string that uniquely identifies the volume within the library.
   - *`library-equipment-number`*`:`*`slot`*, where *`library-equipment-number`* is the equipment ordinal number specified for the automated tape library in the `/etc/opt/SUNWsamfs/mcf` file and *`slot`* is the slot address where the cartridge resides within the library.

   The note will be retained in the historian catalog after the volume has been exported. In the example, we add a note to catalog entry for LTO (`li`) cartridge `VOL054`:

   ```
   root@solaris:~# chmed -I "To vault 20140411" li.VOL054
   ```

3. To move a cartridge from a specified storage slot to the mailbox, use the command `samexport` *`library-equipment-number`*`:`*`slot`*, where *`library-equipment-number`* is the equipment ordinal number specified for the automated tape library in the `/etc/opt/SUNWsamfs/mcf` file and *`slot`* is the slot address where the cartridge resides within the library.

   In the example, we export the magnetic tape cartridge located in slot `11` of library `800`:

   ```
   root@solaris:~# samexport 800:11
   ```

4. To move a specified cartridge to the mailslot, use the command `samexport` *`mediatype`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character

media type codes listed Appendix A, "Glossary of Equipment Types" and *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the library.

The SAM-QFS software adds the cartridge to the catalog maintained by the SAM-QFS **historian**. In the example, we export the LTO (li) tape cartridge VOL109:

```
root@solaris:~# samexport li.VOL109
```

5. Remove media cartridge(s) from the mailbox according to the library vendor's instructions.

6. Stop here.

Maintaining Library Catalogs

SAM-QFS library catalogs are the software's internal representation of the automated library and its contents. If the automated library is direct-attached, the SAM-QFS software has full control over the library and its contents. The SAM-QFS library catalog entries are, accordingly, a one-to-one representation of the slots in the physical library. If the automated library is network-attached, SAM-QFS accesses only the parts of the library that the library software makes available in the form of a virtual library or library partition. So the SAM-QFS library catalog entries reflect only the contents of a portion of the library.

This section explains the following tasks:

- View the Library Catalog
- Audit the Contents of a Library Slot
- Audit the Entire Direct-Attached Automated Library
- Clear a Media Error from the Catalog

View the Library Catalog

1. To view the most commonly used library catalog information, use the command samcmd v *library-equipment-number*, where *library-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the automated tape library.

   In the example, we display the catalog for library 800

```
root@solaris:~# samcmd v 800
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                         count 32
Robot VSN catalog by slot       : eq 800
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li VOL001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li VOL002
   2     2014/03/14 11:26    3   0%  -il-o-b-----  li VOL003
   3     2014/03/14 10:33    3   0%  -il-o-b-----  li VOL004
   4     2014/03/14 11:34    5   0%  -il-o-b-----  li VOL005
   5     2014/03/14 11:32    2   0%  -ilEo-b----f  li VOL006 MEDIA ERROR
   6     2014/03/13 18:07    2   0%  -il-o-b-----  li VOL007
   7     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL008
   8     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL009
...
  18     2014/03/13 18:16    1   0%  -il-o-b-----  li VOL019
  19     none               50   0%  -il-oCb-----  li CLN020
```

2. To determine the status of a volume using the `samcmd v` display, examine the entry in the `flags` column and consult the list below:
   - `A` means that the slot needs an audit.
   - `C` means that the slot contains a cleaning cartridge.
   - `E` means that the volume is bad or the cleaning media has expired.
   - `L` means that the volume is a Linear Tape File System (LTFS) volume.
   - `N` means that the volume is foreign media (not in SAM-QFS format).
   - `R` means that the volume is read-only (a software flag).
   - `U` means that the volume is unavailable.
   - `W` means that the volume is physically write-protected.
   - `X` means that the slot is an export slot.
   - `b` means that the volume has a bar code.
   - `c` means that the volume is scheduled for recycling.
   - `f` means that the archiver found the volume full or corrupted.
   - `d` means that the volume has a duplicate volume serial number (VSN).
   - `l` means that the volume is labeled.
   - `o` means that the slot is occupied.
   - `p` means that the volume is a high priority volume.
   - `-` means that the corresponding flag is not set.
3. To identify the type of media used for a volume using the `samcmd v` display, consult the `ty` column and look up the code displayed in Appendix A, "Glossary of Equipment Types" or in the `mcf` man page.
4. To list all information in the catalog, use the command `dump_cat` *`catalog-path-name`*, where *`catalog-path-name`* is the path and file name of the catalog file, as specified in the `/etc/opt/SUNWsamfs/mcf` file.

   In the example, we dump the catalog file `catalog/800_cat`.

   ```
   root@solaris:~# dump_cat catalog/800_cat
   # audit_time Wed Dec 31 17:00:00 1969
   # version 530  count 32 mediatype
   #Index VSN    Barcode  Type PTOC   Access Capacity ...  LVTime LVPos
   #
    0     S00001 S00001L4 li   0x747    875   512000 ...       0   0x3
    1     S00002 S00002L4 li   0x5db    866   512000 ...       0   0x3
   13     S00014 S00014L4 li       0      4   512000 ...       0   0
   17     S00018 S00018L4 li       0      1   512000 ...       0   0
   18     S00003 S00003L4 li       0      3   512000 ...       0   0
   ```
5. Stop here.

Audit the Contents of a Library Slot

To update the library catalog with the reported space remaining on a removable media volume, audit the library slot. Use the command `auditslot`.

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. To audit a specified tape volume, skip to EOD (*end of data*), and update the space available, use the command `auditslot -e` *`library-equipment-number`*`:`*`slot`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library and *`slot`* is the location of the cartridge within the library.

   The `auditslot` command loads the cartridge that contains the volume, reads the label, and updates the library catalog entry for the slot. Note that you cannot interrupt skipping to EOD once you start it, and, under certain conditions, it can take hours to complete. In the example we audit slot `11` in tape library `800`:

   ```
   root@solaris:~# auditslot -e 800:11
   root@solaris:~#
   ```

3. To audit a specified tape or optical volume, use the command `auditslot` *`library-equipment-number`*`:`*`slot`*`[:`*`side`*`]`, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library, *`slot`* is the location of the cartridge within the library, and *`side`* (optional) is the specified side of a two-sided optical disk.

   In the example we audit side `1` of the volume in slot `21` of optical library `700`:

   ```
   root@solaris:~# auditslot 800:21:1
   root@solaris:~#
   ```

4. Stop here.

Audit the Entire Direct-Attached Automated Library

A full audit loads each cartridge into a drive, reads the label, and updates the library catalog. Audit a library in the following situations:

- after moving cartridges in the automated library without using SAM-QFS commands
- when the library catalog may be unreliable (following a power outage, for example)
- when you have added, removed, or moved cartridges in an automated library that has no mailbox.

To perform a full audit, use the command `samcmd audit` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library.

Note that a full audit can take a long time, depending on the number of slots that contain media.

In the example we audit tape library `800`:

```
root@solaris:~# audit 800
root@solaris:~#
```

Clear a Media Error from the Catalog

When SAM-QFS has problems using a removable media cartridge, it sets an error flag on the corresponding catalog entry. The media may be worn, damaged, or, the case of cleaning media, expired, in which case the media should not be reused. But problems accessing media can also result from faults in the drive, in which case the media can be reused without difficulty. In the latter case, you need to clear the error flag for the cartridge.

Be sure that you know the nature of the problem before clearing error flags. Error flags are critical to SAM-QFS operations and to the security of your data. You do not want to clear this flag if a cartridge is actually faulty.

Once you are sure, you can clear the error and try to use the cartridge. Proceed as follows:

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. Check the status of removable media volumes. Use the command `samcmd r`.

   In the example, the `samcmd r` command shows that drive `801` has set the error flag on LTO (`li`) volume `VOL004`.

   ```
   root@solaris:~# samcmd r
   Removable media status: all          samcmd 5.4       17:40:11 Mar 13 2014
   ty   eq  status      act  use  state   vsn
   li   801 -E-------r     0   0%  notrdy  VOL004  MEDIA ERROR
           MEDIA ERROR
   li   802  ---------p    0   0%  notrdy
           empty
   li   803  ---------p    0   0%  notrdy
           empty
   li   804  ---------p    0   0%  notrdy
           empty
   root@solaris:~#
   ```

3. If the drive that set the error flag is suspect, unload the cartridge and clear the error flag. Use the command `samcmd unload` *`drive-number`*, where *`drive-number`* is the equipment-ordinal number specified for the drive in the `/etc/opt/SUNWsamfs/mcf` file.

   In the example, we unload drive `801`:

   ```
   root@solaris:~# samcmd unload 801
   ```

4. To clear the media error flag for a specified volume serial number and media type, use the command `chmed -E` *`media-type`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

   In the example, we clear the error flag on LTO (`li`) volume `VOL004`:

   ```
   root@solaris:~# chmed -E li.VOL004
      3:0 li VOL004     Ail---b-----    2.3T    2.3T    0          0 800 4   0 //
   root@solaris:~#
   ```

5. To clear the media error flag for a cartridge that resides in a specified library slot, use the command `chmed -E` *`library-equipment-number`*`:`*`slot`*`[:`*`disk-side`*`]`, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library, *`slot`* is the slot address where the target volume resides within the library, and the optional *`disk-side`* value, either `1` or `2`, specifies one of the sides of a two-sided magneto-optical disk.

   In the example, we clear the error flag on the cartridge in slot `31` of library `800`:

```
root@solaris:~# chmed -E 800:31
```

6. Update the library catalog to reflect the change. Use the command `auditslot -e` *`library-equipment-number`*`:`*`slot`*`[:`*`disk-side`*`]`.

   In the example, we update the catalog by auditing slot `31` of library `800`:

   ```
   root@solaris:~# auditslot -e 800:31
   root@solaris:~#
   ```

7. Mount the cartridge in a different drive, and see if the error recurs. Use the command `samcmd load` *`media-type`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

   ```
   root@solaris:~# samcmd load li.VOL004
   root@solaris:~#
   ```

8. Re-check the status of removable media volumes. Use the command `samcmd r`.

   ```
   root@solaris:~# samcmd r
   Removable media status: all          samcmd 5.4        17:42:10 Mar 13 2014
   ty   eq  status      act  use  state  vsn
   li  801  ---------p    0   0%  notrdy
           empty
   li  802  --l------r    0   0%  ready   VOL004
           idle
   li  803  ---------p    0   0%  notrdy
           empty
   li  804  ---------p    0   0%  notrdy
           empty
   root@solaris:~#
   ```

9. If the error does not recur on the new drive, the cartridge is probably OK.

10. If the error recurs, consider retiring the removable media volume.

11. Stop here.

Managing the Historian Catalog

The SAM-QFS historian is a pseudo-library that has a catalog but no equipment. The historian catalogs volumes that are no longer under direct SAM-QFS control. It thus maintains a record of any volumes that have been exported from a library and sent for offsite storage and volumes that are hand-loaded into standalone drives. SAM-QFS automatically updates the historian catalog when you export volumes from the library. But you can also use the historian for manual record keeping by adding and/or removing records and attaching notes. In general, you interact with the historian much as you would with a physical media library:

This section outlines the following tasks:

- View the Historian Catalog
- Add an Entry to the Historian Catalog
- Remove an Entry from the Historian Catalog.

View the Historian Catalog You view the historian catalog exactly as you would that of a physical library. Use the command `samcmd v` *`historian-equipment-number`*, where *historian-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the historian.

In the example, we display the catalog for a historian that has the equipment ordinal number 900:

```
root@solaris:~# samcmd v 900
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                         count 32
Robot VSN catalog by slot       : eq 900
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li EXT001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li EXT002
```

Add an Entry to the Historian Catalog

1. To add an entry to the historian catalog for a specified volume serial number, use the command samimport -v *volume-serial-number* -m *mediatype historian-equipment-number*, where:

   - *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the catalog.
   - *mediatype* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
   - *historian-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the historian.

   In the example, we add a record for the LTO (li) volume EXT003 to the catalog for historian 900:

   ```
   root@solaris:~# samimport -v EXT003 -m li 900
   ]
   ```

2. To add an entry to the historian catalog for a specified barcode, use the command samimport -b *barcode* -m *mediatype historian-equipment-number*, where *barcode* is the barcode affixed to the corresponding physical cartridge.

   In the example, we add a record for the LTO (li) volume with barcode EXT003L4 to the catalog for historian 900:

   ```
   root@solaris:~# samimport -b EXT003L4 -m li 900
   ```

3. Stop here.

Remove an Entry from the Historian Catalog To remove an entry from the historian catalog, use the command samexport *historian-equipment-number*:*slot*, where *historian-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the historian and *slot* is the historian slot address for the record.

In the example, we remove the record for volume EXT002 in slot 1 of the catalog for historian 900:

```
root@solaris:~# samcmd v 900
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                         count 32
Robot VSN catalog by slot       : eq 900
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li EXT001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li EXT002
   2     2014/03/13 17:57  866   0%  -il-o-b-----  li EXT003
```

```
root@solaris:~# samexport 900:1
```

Update Historian Information You can update the information field in a historian catalog entry to note changes to the disposition or status of an exported volume. Use the command `chmed -I "note" identifier`, where *note* is a string of up to 128 characters and *identifier* is either of the following:

- *mediatype.volume-serial-number*, where *mediatype* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the library. Or use the command
- *library-equipment-number*:*slot*, where *library-equipment-number* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library and *slot* is the slot address where the cartridge resides within the library.

In the example, we note that LTO (`li`) cartridge `VOL06E` has been recalled from the vault, successfully validated, and returned to the vault:

```
root@solaris:~# chmed -I "validated and revaulted 20140310" li.VOL06A
```

Determining the Order in Which Drives are Installed in the Library

If your automated library contains more than one drive, the order of the drives in the `mcf` file must be the same as the order in which the drives are seen by the library controller. This order can be different from the order in which devices are seen on the host and reported in the host `/var/adm/messages` file. So whenever you configure a SAM-QFS metadata server and datamover host, change libraries, or change the configuration of a library, you should check the drive order by carrying out the tasks listed below:

- Gather Drive Information for the Library and the Solaris Host
- Either Map the Drives in a Direct-Attached Library to Solaris Device Names or Map the Drives in an ACSLS-Attached Library to Solaris Device Names, depending on the equipment you are using.

Gather Drive Information for the Library and the Solaris Host

1. Consult the library documentation. Note how drives and targets are identified. If there is a local operator panel, see how it can be used to determine drive order.
2. If the library has a local operator panel mounted on the library, use it to determine the order in which drives attach to the controller. Determine the SCSI target identifier or World Wide Name of each drive.
3. Log in to the Solaris host as `root`.

   ```
   root@solaris:~#
   ```

4. List the Solaris logical device names in `/dev/rmt/`, redirecting the output to a text file.

   In the example, we redirect the listings for `/dev/rmt/` to the file `device-mappings.txt` in the `root` user's home directory:

   ```
   root@solaris:~# ls -l /dev/rmt/ > /root/device-mappings.txt
   ```

5. Now, Map the Drives in a Direct-Attached Library to Solaris Device Names or Map the Drives in an ACSLS-Attached Library to Solaris Device Names.

Map the Drives in a Direct-Attached Library to Solaris Device Names

For each Solaris logical drive name listed in `/dev/rmt/` and each drive that the library assigns to the SAM-QFS server host, carry out the following procedure:

1. If you are not already logged in to the SAM-QFS Solaris host, log in as `root`.

   ```
   root@solaris:~#
   ```

2. If SAM-QFS is already installed, stop all running archiving processes so that drives are not in use. See "Idle Archiving and Staging Processes" and "Stop Archiving and Staging Processes" on page 3-23.

3. In a text editor, open the device mappings file that you created in the procedure "Gather Drive Information for the Library and the Solaris Host" on page 5-10, organize it into a simple table, and save the changes.

   You will need to refer to this information in subsequent steps. In the example, we have used the `vi` editor to delete the permissions, ownership, and date attributes from the `/dev/rmt/` list, while adding headers and space for library device information:

   ```
   root@solaris:~# vi /root/device-mappings.txt
   LIBRARY SOLARIS          SOLARIS
   DEVICE  LOGICAL          PHYSICAL
   NUMBER  DEVICE           DEVICE
   ------- -------------    -----------------------------------------------------
           /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
           /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
           /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
           /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
   :w
   ```

4. On the library, make sure that all drives are empty.

5. Load a tape into the first drive in the library that you have not yet mapped to a Solaris logical device name.

   For the purposes of the examples below, we load an LTO4 tape into an HP Ultrium LTO4 tape drive.

6. If you are mapping the drives in a tape library, identify the Solaris `/dev/rmt/` entry that corresponds to the drive that mounts the tape. Until you identify the drive, run the command `mt -f /dev/rmt/`*`number`* `status` where *`number`* identifies the drive in `/dev/rmt/`.

   In the example, the drive at `/dev/rmt/0` is empty, but the drive at `/dev/rmt/1` holds the tape. So the drive that the library identifies as drive 1 corresponds to Solaris `/dev/rmt/1`:

   ```
   root@solaris:~# mt -f /dev/rmt/0 status
   /dev/rmt/0: no tape loaded or drive offline
   root@solaris:~# mt -f /dev/rmt/1 status
   HP Ultrium LTO 4 tape drive:
      sense key(0x0)= No Additional Sense   residual= 0   retries= 0
      file no= 0   block no= 3
   ```

7. In the device-mappings file that you created in the previous procedure, locate the entry for the Solaris device that holds the tape, and enter the library's device identifier in the space provided. Then save the file.

   In the example, enter `1` in the `LIBRARY DEVICE NUMBER` field of the row for `/dev/rmt/1`:

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    -----------------------------------------------------
        /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
        /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
        /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
:w
```

8. Unload the tape.

9. Repeat this procedure until the device-mappings file holds entries that map all devices that the library assigns to the SAM-QFS host to Solaris logical device names. Then save the file and close the editor.

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    -----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
:wq
root@solaris:~#
```

10. Stop here. Keep the mappings file for later use.

Map the Drives in an ACSLS-Attached Library to Solaris Device Names

1. If you are not already logged in to the SAM-QFS Solaris host, log in as `root`.

```
root@solaris:~#
```

2. If SAM-QFS is already installed, stop all running archiving processes, so that drives are not in use. See "Idle Archiving and Staging Processes" and "Stop Archiving and Staging Processes".

3. In a text editor, open the device mappings file that you created in the procedure "Gather Drive Information for the Library and the Solaris Host" on page 5-10, and organize it into a simple table.

   You will need to refer to this information in subsequent steps. In the example, we are using the `vi` editor to delete the permissions, ownership, and date attributes from the `/dev/rmt/` list, while adding headers and space for library device information:

```
root@solaris:~# vi /root/device-mappings.txt
SOLARIS LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
----------------------  --------------------  --------------------------------
/dev/rmt/0
/dev/rmt/1
```

```
/dev/rmt/2
/dev/rmt/3
```

4. For each logical device name listed in /dev/rmt/, display the serial number using the command luxadm display /dev/rmt/*number*, where *number* identifies the drive in /dev/rmt/.

   In the example, we obtain HU92K00200, the serial number of device /dev/rmt/0:

```
root@solaris:~# luxadm display /dev/rmt/0
DEVICE PROPERTIES for tape: /dev/rmt/0
Vendor: HP
Product ID: Ultrium 4-SCSI
Revision: G25W
Serial Num: HU92K00200
...
Path status: Ready
root@solaris:~#
```

5. Then, using a text editor, enter the serial number of each device in the corresponding row of your device-mappings.txt file.

   In the example, we record the serial number for device /dev/rmt/0 in the device-mappings.txt file using the vi editor:

```
root@solaris:~# vi /root/device-mappings.txt
SOLARIS LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
----------------------  --------------------  --------------------------------
/dev/rmt/0              HU92K00200
/dev/rmt/1
/dev/rmt/2
/dev/rmt/3
```

6. For each device serial number mapped to /dev/rmt/, obtain the corresponding ACSLS drive address. Use the ACSLS command display drive * -f serial_num.

   In the example, we obtain the ACSLS addresses of devices HU92K00200 (/dev/rmt/0), HU92K00208 (/dev/rmt/1), HU92K00339 (/dev/rmt/2), HU92K00289 (/dev/rmt/3):

```
 ACSSA> display drive * -f serial_num
 2014-03-29 10:49:12 Display Drive
 Acs Lsm Panel Drive Serial_num
 0   2   10    16    331002031352
 0   2   10    17    HU92K00200
 0   2   10    18    HU92K00208
 0   3   10    10    HU92K00339
 0   3   10    11    HU92K00189
 0   3   10    12    HU92K00289
root@solaris:~#
```

7. Using a text editor, enter the ACSLS address for each serial number in the corresponding row of the device-mappings.txt file. Save the file, and close the editor.

   In the example, we record the information in the device-mappings.txt file using the vi editor:

```
root@solaris:~# vi /root/device-mappings.txt
SOLARIS LOGICAL DEVICE DEVICE SERIAL NUMBER ACSLS DEVICE ADDRESS
---------------------- -------------------- ---------------------------------
```

```
/dev/rmt/0          HU92K00200          (acs=0, lsm=2, panel=10, drive=17)
/dev/rmt/1          HU92K00208          (acs=0, lsm=2, panel=10, drive=18)
/dev/rmt/2          HU92K00339          (acs=0, lsm=2, panel=10, drive=10)
/dev/rmt/3          HU92K00289          (acs=0, lsm=2, panel=10, drive=12)
:wq
root@solaris:~#
```

8. Stop here. Keep the mappings file for later use.

Managing Drives

You can handle a variety of drive management tasks from the SAM-QFS interfaces, including the following:

- Loading and Unloading Drives
- Cleaning Tape Drives
- Using Drives With Encryption Capability
- Handling Drive Problems

Loading and Unloading Drives

When removable media are stored in automated libraries, file-system archiving and staging processes automatically load cartridges into drives as required. But you can also load cartridges on demand when managing removable media files, backing up the SAM-QFS configuration, or recovering a file system. This section covers the following topics:

- Loading and Unloading Drives Installed in an Automated Library
- Manually Loading and Unloading Standalone Drives
- Notifying Operators When Volumes Must Be Loaded Manually

Loading and Unloading Drives Installed in an Automated Library

- Load a Drive from a Specified Library Location
- Load a Drive with a Specified Media Type and Volume Serial Number
- Unload a Specified Drive in the Library

Load a Drive from a Specified Library Location Use the command `samcmd load` *`library-equipment-number`*`:`*`slot`*`[:`*`disk-side`*`]`, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library, *`slot`* is the slot address where the target volume resides within the library, and the optional *`disk-side`* value, either `1` or `2`, specifies one of the sides of a two-sided magneto-optical disk.

The cartridge is loaded in the next available drive in the library. In the example, we load the magnetic tape cartridge located in slot `11` of library `800`:

```
root@solaris:~# samcmd load 800:11
```

Load a Drive with a Specified Media Type and Volume Serial Number Use the command `samcmd load` *`mediatype`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

The cartridge is loaded in the next available drive in the library. In the example, we load the LTO (`li`) tape cartridge `VOL109`:

```
root@solaris:~# samcmd load li.VOL109
```

Unload a Specified Drive in the Library Use the command `samcmd unload` *`drive-equipment-number`*, where *`drive-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive.

The cartridge is unloaded, even if the drive is `unavail`. In the example, we unload drive `801`:

```
root@solaris:~# samcmd unload 801]
```

Manually Loading and Unloading Standalone Drives

The SAM-QFS software treats standalone removable media drives as if they were small, single-slot libraries with their own catalogs.

Load a Cartridge Into a Standalone Drive To load a standalone drive, place the cartridge in the drive according to the manufacturer's instructions. The SAM-QFS system recognizes that the cartridge is loaded, reads the label, and updates the catalog for the drive.

Unload a Cartridge from a Standalone Drive 1.Idle the drive. Use the command `samcmd idle` *`drive-equipment-number`*, where *`drive-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive.

When a drive is idled, the SAM-QFS software finishes any current archiving processes that use the drive but does not start any new ones. When current activity completes, the software sets the drive `off`.

```
root@solaris:~# samcmd idle 801
```

2. Wait until SAM-QFS finishes using the drive and sets it `off`.

   You can check on the status of the drive using the command `samcmd r`.

3. Remove the cartridge according to the vendor's instructions.
4. Stop here.

Notifying Operators When Volumes Must Be Loaded Manually

If you are using a standalone drive or if you store required cartridges in a vault or some other location outside the library, the SAM-QFS software can send email to a specified address when it needs an operator to load a non-resident cartridge. To enable this feature, follow the procedure below:

Enable Load Notification

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. Copy the file, `load_notify.sh`, from `/opt/SUNWsamfs/examples/` to `/etc/opt/SUNWsamfs/scripts/`.

   ```
   root@solaris:~# cp /opt/SUNWsamfs/examples/load_notify.sh \
   /etc/opt/SUNWsamfs/scripts/
   root@solaris:~#
   ```

3. Open the file /etc/opt/SUNWsamfs/defaults.conf in a text editor, and search for the directive exported_media. Uncomment the line or add it, if necessary, and set its value to exported_media=available.

   In the example, we use the vi editor:

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults.
   ...
   exported_media=available
   ```

4. In the file /etc/opt/SUNWsamfs/defaults.conf, search for the directive attended. Uncomment the line or add it, if necessary, and set its value to attended=yes. Save the file, and close the editor.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults. ...
   # These are the defaults. ...
   exported_media=available
   attended=yes
   :wq
   root@solaris:~#
   ```

5. Open the file /etc/opt/SUNWsamfs/scripts/load_notify.sh in a text editor, and locate the default recipient of the notification email, root.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/scripts/load_notify.sh
   #       Notify operator to load volume.
   ...
   # Change the email address on the following line to send email to
   # the appropriate recipient.
   /bin/ppriv -s I=basic -e /usr/bin/mailx -s "SAM-FS needs VSN $5" root <<EOF
   ...
   ```

6. In the file /etc/opt/SUNWsamfs/scripts/load_notify.sh, change the recipient of the notification email from the default, root, to the operator responsible for the non-resident volumes. Save the file and close the editor.

   In the example, we change the recipient to tapetech:

   ```
   #       Notify operator to load volume.
   ...
   /bin/ppriv -s I=basic -e /usr/bin/mailx -s "SAM-FS needs VSN $5" tapetech <<EOF
   ...
   :wq
   root@solaris:~#
   ```

7. Reinitialize the SAM-QFS software. Use the sam-fsd command.

   The sam-fsd is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

   ```
   root@solaris:~# sam-fsd
   ```

8. Tell the SAM-QFS software to re-read the mcf file and reconfigure file systems and hardware accordingly. Use the command samd config:

   ```
   root@solaris:~# samd config
   ```

9. Stop here.

Cleaning Tape Drives

Modern tape drives like the SAM-QFS SAM-QFS T10000D and Linear Tape Open (LTO) drives self-monitor and request cleaning when and as needed. The SAM-QFS software honors these requests and automatically loads a cleaning cartridge when required. So in most cases, you need only insure that your library contains adequate cleaning cartridges and that SAM-QFS is able to locate them (see Provide Sufficient Cleaning Cartridges below).

When drive-requested cleaning is not feasible, you can initiate cleaning manually. But be advised: most manufacturers emphatically discourage routine cleaning in the absence of a request from the drive. Cleaning cartridges are abrasive, and overuse can damage drives and media. So exercise caution, and follow the manufacturer's recommendations.

- Provide Sufficient Cleaning Cartridges
- Enable Automatic Tape-Drive Cleaning (Recommended)
- Clean a Tape Drive Manually.

Provide Sufficient Cleaning Cartridges

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. If you plan to configure automatic cleaning (recommended) and if your library has more than two drives, make sure that you provide at least two cleaning cartridges for each file-system catalog that lists tapes in the library.

   If a cleaning cartridge is unavailable when a drive requires cleaning, the SAM-QFS software sets the drive state to `down` until cleaning can be completed.

3. Place the cartridge(s) in the library mail slot (cartridge access port).

4. Import the cleaning cartridge(s) into the automated library. Use the command `samimport` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library.

   In the example, we have placed cleaning cartridges in the mailslot (cartridge access port) of library `800` and imported them into the library.

   ```
   root@solaris:~# samimport 800
   ```

5. If the cleaning cartridge label reads `CLEAN` or starts with the letters `CLN`, stop here.

   The SAM-QFS software recognizes the cleaning cartridge and moves it from the mailbox to a storage slot. SAM-QFS updates the library catalog, sets the cleaning media flag, and sets the access count to the maximum number of cleanings recommended for the media type (each time the cartridge is used to clean a drive, this count decrements).

6. If the cartridge is not labeled, flag it as cleaning media. Use the command `chmed +C` *`library-equipment-number`*`:`*`slot`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library and *`slot`* is the location of the cleaning cartridge within the library.

   In the example, we set the `C` (cleaning-media) flag on the cartridge in slot `31` of library `800`.

   ```
   root@solaris:~# chmed +C 800:31
   ```

7. Set the access count to the maximum number of cleanings recommended for the media type. Use the command `chmed -count` *`cleanings library-equipment-number`*`:`*`slot`*, where:

   - *`cleanings`* is the maximum number of cleanings.
   - *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library.
   - *`slot`* is the location of the cleaning cartridge within the library.

   Each time the cartridge is used to clean a drive, the cleaning count decrements. In the example, we set the count to a maximum of `50` cleanings, the maximum recommended for the LTO (type `li`) cleaning cartridges used in library `800`:

   ```
   root@solaris:~# chmed -count 50 800:31
   ```

8. Next, Enable Automatic Tape-Drive Cleaning (Recommended) or stop here.

Enable Automatic Tape-Drive Cleaning (Recommended)

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. If your library has more than two drives, make sure that you provide at least two cleaning cartridges for each file-system catalog that lists tapes in the library.

   Having enough cleaning media minimizes drive downtime. If a cleaning cartridge is unavailable when a drive requires cleaning, the SAM-QFS software sets the drive state to `down` until cleaning can be completed.

3. If your library includes an Auto Clean feature that you wish to use, configure the feature according to the library manufacturer's recommendations. Stop here.

   Now, when drives request cleaning, the library will automatically supply the required cleaning media.

4. If your library includes an Auto Clean feature that you *do not* wish to use, disable the feature according to the manufacturer's recommendations.

5. Open the file `/etc/opt/SUNWsamfs/defaults.conf` in a text editor, and enable SAM-QFS automatic cleaning. Add the line `tapeclean = all autoclean on logsense on`. Then save the file and close the editor.

   In the example, we use the `vi` editor:

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults. ...
   #sef = all on once
   ...
   tapeclean = all autoclean on logsense on
   :wq
   root@solaris:~#
   ```

6. Reinitialize the SAM-QFS software. Use the `sam-fsd` command.

   The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

   ```
   root@solaris:~# sam-fsd
   ```

7. Tell the SAM-QFS software to re-read the `mcf` file and reconfigure file systems and hardware accordingly. Use the command `samd config`:

```
root@solaris:~# samd config
```

8. Stop here.

Clean a Tape Drive Manually

1. Check the drive manufacturer's guidelines for manual cleaning before proceeding.

   Exercise caution. Over-frequent cleaning is a common cause of drive damage, and many manufacturers now strongly discourage routine or scheduled cleanings. So make sure that you understand when your drives need to be cleaned.

2. Monitor the device logs for indications that drives need cleaning. There is one log in the directory `/var/opt/SUNWsamfs/devlog/` for each *`drive-equipment-number`*, where *`drive-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive. Also check the file `/var/adm/messages` for device errors.

3. Clean the tape drive. Use the command `cleandrive` *`drive-equipment-number`*.

   In the example, we clean drive `802`:

   ```
   root@solaris:~# cleandrive 802
   ```

4. Stop here.

Using Drives With Encryption Capability

If you are archiving files to drives that have encryption capability, consider the following recommendations when planning archiving operations:

- Do not mix non-encrypted and encryption-capable drives in a library.
- After a drive has encryption enabled, it cannot be disabled.
- Do not mix encrypted and non-encrypted files on a tape.
- An encrypted drive cannot append to a tape that contains non-encrypted data.
- An encryption-enabled drive can read non-encrypted data.

For further information, consult the documentation for your drives and encryption key-management system.

Handling Drive Problems

In general, you handle drive problems according to the drive vendor's recommendations. This section explains the procedure you should follow to prepare the SAM-QFS software and file systems for drive maintenance, troubleshooting, and repair.

Taking the Drive Offline for Maintenance or Repair

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. Quiesce archiving and staging processes, as described in "Idle Archiving and Staging Processes" on page 3-22.

3. Stop archiving and staging processes, and take the drives offline. Use the procedure "Stop Archiving and Staging Processes" on page 3-23.

4. Carry out the vendor-specified maintenance, diagnostic. or repair procedures.

For example, before attempting to remove a stuck cartridge, be sure to check the vendor's recommendations. Improperly removing a stuck cartridge can damage the cartridge and the drive.

5. When the drive is again operational, bring the library and drives online and restart archiving and staging processes. Use the procedure "Restart Archiving and Staging Processes" on page 3-24.
6. Stop here.

Return Media to the Library Following a Drive Problem

If drive problems occur with media mounted in the drive, you may need to remove the media manually as part of the repair process. This can leave the catalog inconsistent. So follow the appropriate procedure below:

Returning Media to a Library that Has Not Performed an Automatic Audit

1. If the library does *not* perform an automatic audit when the library and drive are brought back online following repairs, return the cartridge to its storage slot by hand.

   In this case, the catalog has not been updated and continues to list the cartridge among the library contents. So you correct the discrepancy by putting the cartridge back in its slot.

2. Update the catalog to show that the slot is again occupied. Use the command `chmed` *`library-equipment-number`*`:`*`slot`*, where *`slot`* is the address of the slot within the library.

   In the example, we update the status of slot `42` in library `800`:

   ```
   root@solaris:~# chmed +o 800:42
   root@solaris:~#
   ```

3. Stop here.

Returning Media to a Library After an Automatic Audit

1. If the library performs an automatic audit when the library and drive are brought back online following repairs, place the cartridge in the library mail slot (cartridge access port).
2. Import the cartridge into the library. Use the command `samimport` *`library-equipment-number`*.

   In this case, the audit has reconciled the catalog, which no longer lists the cartridge in the library. So importing the cartridge adds it to both the library and the catalog. In the example, we have placed the cartridge in the mailslot of library `800` and imported it into the library.

   ```
   root@solaris:~# samimport 800
   ```

3. Stop here.

Managing Removable Media

This section covers the following topics:

- Labeling Removable Media
- Displaying Data Integrity Validation (DIV) Settings and Status

- Verifying the Integrity of Tape Media

Labeling Removable Media

Caution: Labeling or relabeling a cartridge renders any data on the cartridge permanently inaccessible. Relabel a cartridge only if you are certain that you do not need the data that is stored on it.

The labeling process writes identifying information on the recording media and initializes it for use (see ANSI X3.27-1987, *File Structure and Labeling of Magnetic Tapes for Information Interchange*, for full information).

When you need to label media, select the appropriate procedure below:

- Generate Labels from Barcodes
- Label a New Tape or Relabel an Existing Tape
- Label a New Optical Disk or Relabel an Existing Optical Disk.

Generate Labels from Barcodes

To automatically label write-enabled, unlabeled cartridges with a volume serial number (VSN) derived from the barcodes on the cartridges, proceed as follows.

1. Make sure that all barcodes are readable.
2. Log in to the file system host as `root`.

```
root@solaris:~#
```

3. Open the `/etc/opt/SUNWsamfs/defaults.conf` file in a text editor. Locate the `labels` directive, and note its value.

   In the example, we use the `vi` editor. The `labels` directive is set to `barcodes` (the default).

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
#labels = barcodes
:w
root@solaris:~#
```

4. If you need to generate the volume serial number (VSN) from the first six characters in the barcode, use the default setting, `labels = barcodes`.

   If the `labels` directive is commented out, set to the value `barcodes`, or missing, you can close the file without making changes. Otherwise, set the `labels` parameter to `barcodes`, save the file, and close the editor.

   In the example, the directive is commented out, so the default setting has not been changed. We thus close the file and the editor, leaving the default setting unchanged:

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
#labels = barcodes
:q
root@solaris:~#
```

5. If you need to generate the volume serial number (VSN) from the last six characters in the cartridge's barcode, set the value of the `labels` parameter to `barcodes_low`. Save the file and close the editor.

   In the example, we replace the line `#labels = barcodes` with `labels = barcodes_low`:

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults.
   ...
   labels = barcodes_low
   :wq
   root@solaris:~#
   ```

6. If you edited the `defaults.conf` file, run the `sam-fsd` command.

   The `sam-fsd` is an initialization command that reads SAM-QFS configuration files. It will stop if it encounters an error:

   ```
   root@solaris:~# sam-fsd
   ```

7. If you edited the `defaults.conf` file, tell the SAM-QFS software to re-read the `mcf` file and reconfigure itself accordingly:

   ```
   [metadata-server]root@solaris:~# samd config
   ```

8. Stop here.

Label a New Tape or Relabel an Existing Tape

1. Log in to the file system host as `root`.

   ```
   root@solaris:~#
   ```

2. To label a new tape that is loaded into a drive, use the command `tplabel -new` *`volume-serial-number drive-equipment-number`*, where *`volume-serial-number`* is the required volume serial number and *`drive-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive.

   In the example, we assign the volume serial number `VOL600` to the new tape cartridge in drive `803`:

   ```
   root@solaris:~# tplabel -new -vsn VOL600 803
   root@solaris:~#
   ```

3. To label a new tape that resides in an automated media library, use the command `tplabel -new` *`volume-serial-number library-equipment-number:slot`*, where *`volume-serial-number`* is the required volume serial number, *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive, and *`slot`* is the location of the cartridge within the library.

   In the example, we assign the volume serial number `VOL601` to the new tape cartridge in slot `19` of library `800`:

   ```
   root@solaris:~# tplabel -new -vsn VOL601 800:19
   root@solaris:~#
   ```

4. To relabel an existing tape that is loaded into a drive, use the command `tplabel -old` *`volume-serial-number drive-equipment-number`*, where

volume-serial-number is the required volume serial number and *drive-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the drive.

You can reuse the existing volume serial number if required. In the example, we assign the volume serial number VOL120 to the tape cartridge in drive 804:

```
root@solaris:~# tplabel -old -vsn VOL120 804
root@solaris:~#
```

5. To relabel an existing tape that resides in an automated media library, use the command tplabel -old *volume-serial-number library-equipment-number:slot*, where *volume-serial-number* is the required volume serial number, *library-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the drive, and *slot* is the location of the cartridge within the library.

   You can reuse the existing volume serial number if required. In the example, we assign the volume serial number VOL121 to the tape cartridge in slot 23 of library 800:

```
root@solaris:~# tplabel -old -vsn VOL601 800:23
root@solaris:~#
```

6. Stop here.

Label a New Optical Disk or Relabel an Existing Optical Disk

1. Log in to the file system host as root.

```
root@solaris:~#
```

2. To label a new optical cartridge that is loaded into a drive, use the command odlabel -new *volume-serial-number drive-equipment-number*[:*side*], where:

   - *volume-serial-number* is the required volume serial number.
   - *drive-equipment-number* is the equipment ordinal number that the */etc/opt/SUNWsamfs/mcf* file assigns to the drive.
   - *side* (optional) is the specified side of a two-sided disk.

   In the example, we assign the volume serial number OD1700 to the new, single-sided optical cartridge in drive 701:

```
root@solaris:~# odlabel -new -vsn OD1700 701
root@solaris:~#
```

3. To label a new optical cartridge that resides in an automated media library, use the command odlabel -new *volume-serial-number library-equipment-number*:*slot*[:*side*], where:

   - *volume-serial-number* is the required volume serial number.
   - *library-equipment-number* is the equipment ordinal number that the /etc/opt/SUNWsamfs/mcf file assigns to the drive.
   - *slot* is the location of the cartridge within the library, and *side* (optional) is the specified side of a two-sided disk.

   In the example, we assign the volume serial number OD1701 to side 2 of the new, two-sided optical cartridge in slot 42 of library 700:

```
root@solaris:~# odlabel -new -vsn OD1701 700:42:2
```

```
root@solaris:~#
```

4. To relabel an existing optical cartridge that is loaded into a drive, use the command `odlabel -old` *`volume-serial-number`* *`drive-equipment-number`*`[:`*`side`*`]`, where:

   - *`volume-serial-number`* is the required volume serial number.
   - *`drive-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive.
   - *`side`* (optional) is the specified side of a two-sided disk.

   You can reuse the existing volume serial number if required. In the example, we assign the volume serial number `OD1120` to the optical cartridge in drive `702`:

   ```
   root@solaris:~# odlabel -old -vsn OD1120 702
   root@solaris:~#
   ```

5. To relabel an existing optical cartridge that resides in an automated media library, use the command `odlabel -old` *`volume-serial-number`* *`library-equipment-number`*`:`*`slot`*`[:`*`side`*`]`, where:

   - *`volume-serial-number`* is the required volume serial number.
   - *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the library.
   - *`side`* (optional) is the specified side of a two-sided disk.

   You can reuse the existing volume serial number if required. In the example, we assign the volume serial number `OD1121` to the optical cartridge in slot `23` of library `700`:

   ```
   root@solaris:~# odlabel -old -vsn OD1121 800:23
   root@solaris:~#
   ```

6. Stop here.

Displaying Data Integrity Validation (DIV) Settings and Status

This section covers the following tasks:

- Display the DIV Setting
- Monitor the Verify After Write Status of Archive Files
- Monitor the Verify After Write Status of Devices.

Display the DIV Setting

To display the Data Integrity Validation (DIV) setting, use the command `samcmd L` and pipe the output to the Solaris `grep` command: `samcmd L | grep div`.

In the example, DIV is `OFF`:

```
root@solaris:~# samcmd L | grep div
div            OFF
root@solaris:~#
```

Monitor the Verify After Write Status of Archive Files

To monitor the verification status of archive files during archiving, use the `samu` interface. Use the command `samu -d a`.

```
root@solaris:~# samu -d a
Archiver status                         samu 5.4         22:22:31 Mar 4 2014
sam-archiverd: Archiving files
sam-arfind:  samqfs1 mounted at /samqfs1
Files waiting to start     12,576  schedule     26,695  archiving     13,120
...
sam-arcopy: qfs.arset1.2.344 ti.TKC960
Verifying archive file at position 1175
```

Monitor the Verify After Write Status of Devices

To monitor the verification status of devices during archiving, use the `samu` interface. Use the command `samu -d s`:

```
root@solaris:~# samu -d s
Device status                        samu 5.4           22:27:53 Mar 4 2014
ty     eq state  device_name                     fs  status
sn    800 on     /dev/scsi/changer/c1t2d0        800 n--------r
ti    801 on     /dev/rmt/0cbn                   800 ---------p
...
hy    805 on     historian                       805 ----------
ti     91 on     /dev/rmt/4cbn                    90 -l----oVr
Verify averaging 240.9 MB/s
```

Verifying the Integrity of Tape Media

When you need to verify the data integrity of particular tape volumes on demand, use the SAM-QFS `tpverify` command (for instructions on configuring automated Periodic Media Verification, see the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide*). The `tpverify` command supports SAM-QFS T10000C/D, LTO, and other commonly used media. T10000C/D media are verified using SAM-QFS Data Integrity Validation. Other formats are checked using the widely supported SCSI `verify(6)` command.

The following sections outline some of the ways in which `tpverify` can be used. See the `tpverify` man page for full details on syntax and options:

- Verify the Data on a Tape Specified by Library Location
- Verify the Data on a Tape Specified by Media Type and Volume Serial Number
- Verify the Data on a Tape Using a Specified Drive
- Restart Data Verification from the Start of the Tape
- Verify ECC for All Blocks on a T10000C/D Tape
- Verify ECC and DIV Checksums for All Blocks on a T10000C/D Tape
- Rebuild the Media Information Region (MIR) of a T10000C/D Tape
- Cancel Data Verification for a Specified Tape
- Display the DIV Status and Verification Progress for a Tape

Verify the Data on a Tape Specified by Library Location

Use the command `tpverify` *`library-equipment-number`*`:`*`slot`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library and *`slot`* is the slot address where the target volume resides within the library.

The `tpverify` command locates the last tape position that was verified by checking the library media catalog. It then loads the tape into the first available drive and starts validating from the point where it last stopped, using the default method—the `tpverify` Standard method for T10000C/D media or SCSI `verify(6)` for other media. The Standard method is optimized for speed and verifies the edges, beginning, end, and first 1,000 blocks of SAM-QFS media.

In the example, we validate the T10000D tape stored in slot `9` on library `800` using the Standard method:

```
root@solaris:~# tpverify 800:9
```

Verify the Data on a Tape Specified by Media Type and Volume Serial Number

Use the command `tpverify` *`mediatype`*`.`*`volume-serial-number`*, where *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

The `tpverify` command locates the last tape position that was verified by checking the library media catalog. It then loads the tape into the first available drive and starts validating from the point where it last stopped, using the default method—the `tpverify` Standard method for T10000C/D media or SCSI `verify(6)` for other media.

In the example, we validate LTO (**`li`**) volume **`VOL006`** using the SCSI **`verify(6)`** command:

```
root@solaris:~# tpverify li.VOL006
```

Verify the Data on a Tape Using a Specified Drive

Use the command `tpverify` *`library-equipment-number`*`:`*`slot`* *`device-equipment-number`*, where:

- *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *`slot`* is the slot address where the target volume resides within the library.
- *`device-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the drive.

In the example, we validate the T10000D tape stored in slot `17` on library `800` using drive `803`:

```
root@solaris:~# tpverify 800:17 803
```

Restart Data Verification from the Start of the Tape

Use the command `tpverify -a` *`library-equipment-number`*`:`*`slot`* or `tpverify -a` *`mediatype`*`.`*`volume-serial-number`*, where:

- *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *`slot`* is the slot address where the target volume resides within the library.
- *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".

- *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the library.

The `-a` option overrides the default behavior and starts verifying from the beginning of the media, ignoring the position recorded in the media catalog.

In the example, we validate LTO (`li`) volume `VOL016` from the beginning of the tape:

```
root@solaris:~# tpverify -a li.VOL016
```

Verify ECC for All Blocks on a T10000C/D Tape

Use the command `tpverify -C` *library-equipment-number*`:`*slot* or `tpverify -C` *mediatype*`.`*volume-serial-number* where:

- *library-equipment-number* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *slot* is the slot address where the target volume resides within the library.
- *mediatype* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
- *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the library.

The `tpverify` command locates the last tape position that was verified by checking the library media catalog. It then starts validating from the point where it last stopped, using the Complete method specified by the `-C` option. The Complete method is more thorough than the standard method but can also be significantly slower. It checks Error Correction Codes (ECC) on all blocks on the media.

In the example, we validate T10000D (`ti`) volume `VOL516` using the Complete method:

```
root@solaris:~# tpverify -C ti.VOL516
```

Verify ECC and DIV Checksums for All Blocks on a T10000C/D Tape

Use the command `tpverify -P` *library-equipment-number*`:`*slot* or `tpverify -P` *mediatype*`.`*volume-serial-number*, where:

- *library-equipment-number* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *slot* is the slot address where the target volume resides within the library.
- *mediatype* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
- *volume-serial-number* is the alphanumeric string that uniquely identifies the volume within the library.

The `tpverify` command locates the last tape position that was verified by checking the library media catalog. It then starts validating from the point where it last stopped, using the Complete Plus method specified by the `-P` option. The Complete Plus method is very thorough but also even slower than the other methods. It checks Error Correction Codes (ECC) and Data Integrity Validation checksums on all blocks on the media.

In the example, we validate T10000D (`ti`) volume `VOL521` using the Complete Plus method:

```
root@solaris:~# tpverify -P ti.VOL521
```

Rebuild the Media Information Region (MIR) of a T10000C/D Tape

Use the command `tpverify -M` *`library-equipment-number`*`:`*`slot`* or `tpverify -M` *`mediatype`*`.`*`volume-serial-number`*, where:

- *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *`slot`* is the slot address where the target volume resides within the library.
- *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
- *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

The `tpverify` command rebuilds a missing or damaged media information region (MIR) on an SAM-QFS SAM-QFS tape cartridge, even if the tape is marked bad in the media catalog. Rebuilding is automatically specified when MIR damage is detected.

In the example, we validate T10000D (`ti`) volume `VOL523` using the MIR Rebuild method:

```
root@solaris:~# tpverify -M ti.VOL523
```

Cancel Data Verification for a Specified Tape

Use the command `tpverify -c` *`library-equipment-number`*`:`*`slot`* or `tpverify -c` *`mediatype`*`.`*`volume-serial-number`*, where:

- *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.
- *`slot`* is the slot address where the target volume resides within the library.
- *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types".
- *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

The `tpverify -c` command cancels the current verification operation and records the last verified position on the tape in the media catalog. So you can stop a verification job to free a drive or volume for archiving or staging and then resume verification at the same point later.

In the example, we cancel verification of T10000D (`ti`) volume `VOL533`:

```
root@solaris:~# tpverify -c ti.VOL523
```

Display the DIV Status and Verification Progress for a Tape

Use the command `itemize -2` *`library-equipment-number`*, where *`library-equipment-number`* is the equipment ordinal number that the `/etc/opt/SUNWsamfs/mcf` file assigns to the automated tape library.

The `itemize -2` command catalogs the media in the specified library and lists the DIV status and verification progress for each volume.

In the example, we display verification status for volumes in the library with equipment ordinal number `800`. The `lvtime` (time last verified) fields display the time when `tpverify` last completed a full verification of the tape. A `status` field value of `div` indicates that the tape is DIV-capable while a value of `none` indicates that it is not. The `lvpos` (last verified position) fields show where `tpverify` was last canceled and where it will start when run again.

```
root@solaris:~# itemize -2 800
Robot VSN catalog: eq: 800       count: 60
slot    access_time  count  use  ty vsn
        lvtime      status         lvpos
   0    Apr  2 16:34     6   0%  ti VOL519
        Apr  2 09:23   div          0
   1    Apr  2 16:17    28  29%  ti VOL510
        Apr  2 16:17   div          0x9bb9
   2    none             0   0%  ti VOL511
        none          none          0
  ...
root@solaris:~#
```

Managing Periodic Media Validation

Periodic Media Validation is the automated form of the `tpverify` command. This section provides instructions for maintenance tasks that may occasionally be necessary (for instructions on configuring Periodic Media Verification, see the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide*). These tasks include:

- View and Validate the verifyd.cmd Configuration File
- Reload the verifyd.cmd Configuration File
- Display All Defects Listed in the Periodic Media Verification Tape Defects Database
- Display Defects Listed for a Particular Volume
- Clear Defects Listed in the Periodic Media Verification Tape Defects Database.

View and Validate the `verifyd.cmd` Configuration File

To view the `verifyd.cmd` file at any time or to validate the file following editing, use the command `tpverify -x`.

The `tpverify -x` command checks the `/etc/opt/SUNWsamfs/verifyd.cmd` file and either calls out errors or displays the contents of the file.

```
root@solaris:~# tpverify -x
Reading '/etc/opt/SUNWsamfs/verifyd.cmd'.
PMV: off
     Run-time:
     Start Time: 2200
End Time: 0500
PMV Scan: all
PMV Method: Standard
STA Scan: off
Action: none
PMV VSNs: all
PMV Policy:
     Last Verified Age: 6m
root@solaris:~#
```

Reload the verifyd.cmd Configuration File

To reload the `verifyd.cmd` file without stopping the verification process, use the command `tpverify -r`.

```
root@solaris:~# tpverify -r
root@solaris:~#
```

Display All Defects Listed in the Periodic Media Verification Tape Defects Database

To list all defects that have been identified by Periodic Media Verification and stored in the tape defects database, use the command `tpverify -l`.

In the example, there are no defects in the database:

```
root@solaris:~# tpverify -l
No defects found.
root@solaris:~#
```

Display Defects Listed for a Particular Volume

To list all defects that have been identified on a particular volume, use the command `tpverify -l` *`mediatype.volume-serial-number`*, where the optional *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

In the example, there are no defects listed in the database for the LTO (`ti`) volume `VOL514`:

```
root@solaris:~# tpverify -l ti.VOL514
No defects found.
root@solaris:~#
```

Clear Defects Listed in the Periodic Media Verification Tape Defects Database

To delete all defects that have been identified by Periodic Media Verification from the tape defects database, use the command `tpverify -d`.

To delete all defects listed for a particular volume, use the command `tpverify -d` *`mediatype.volume-serial-number`* where the optional *`mediatype`* is one of the two-character media type codes listed Appendix A, "Glossary of Equipment Types" and *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume within the library.

```
root@solaris:~# tpverify -d
root@solaris:~# tpverify -d ti.VOL514
root@solaris:~#
```

6

Backing Up the Configuration and File Systems

During SAM-QFS installation and configuration, you created secure locations for storing recovery point files and copies of the archiver log. You also configured automated processes for creating recovery points, backing up the logs, and protecting the system configuration. These steps provide the core protection for your file systems. But you may also need to take unscheduled protective measures from time to time as well:

- Back up the SAM-QFS configuration and file systems prior to anticipated, potentially disruptive events, such as major changes to physical infrastructure or data center facilities.
- Back up the SAM-QFS configuration and file systems after upgrading or reconfiguring software, operating systems, or host platforms, so that the current configuration is protected.
- Gather required configuration and status information prior to engaging SAM-QFS support services.

This chapter outlines the procedures for collecting, creating, and storing configuration and file-system recovery files on an as-needed basis. It contains three major sections:

- Backing Up File Systems
- Backing Up the SAM-QFS Configuration
- Gathering Configuration and Diagnostic Information with samexplorer

Note that this chapter uses the command line interface for all tasks. If you wish to use the SAM-QFS Manager graphical user interface, refer to the online help for detailed procedures.

Backing Up File Systems

This section starts with a brief review of SAM-QFS file system protection, "Understanding Recovery Points and Archive Logs". Then it provides procedures for carrying out the following tasks:

- Create a Recovery Point on Demand
- Back Up the Archiver Log
- Gathering Configuration and Diagnostic Information with samexplorer
- Manually Backup the SAM-QFS Configuration

Understanding Recovery Points and Archive Logs

To protect a file system, you need to do two things:

- You must protect the files that hold your data.
- You must protect the file system itself, so that you can use, organize, locate, access, and manage your data.

Copying the files from a SAM-QFS file system to backup media protects the data. But if a disk device or RAID group subsequently fails and cannot be recovered, you are left with no easy way to use the backed up data. You have to create a substitute file system, identify each file, determine its proper location within the new file system, ingest it, and recreate lost relationships between the file and the users, applications, and files that depend on it. This kind of recovery is, at best, a daunting and long drawn-out process.

So, for fast, efficient recovery, you have to actively protect both the file-system data and the metadata that make files and archive copies usable. You must make copies of files and you must back up directory paths, inodes, access controls, symbolic links, and pointers to any copies archived on removable media. You protect SAM-QFS file-system metadata by creating *recovery points*. A recovery point is a compressed file that stores point-in-time copies of the metadata for a specified file system. You protect file-system data in either of two ways. Archiving file systems protect data automatically. They copy files to archival media—tape and/or archival disk volumes—as files are created or modified. To fully protect data, you configure the archiver to make at least two backup file copies, with at least one on removable media. You protect stand-alone (non-archiving) file systems by including the data files in each recovery point file.

In the event of a data loss—anything from accidental deletion of a user file to catastrophic loss of a whole file system—you locate the last recovery point at which the file or file system remained intact and restore the file or file system. To restore an archiving file system, do not need to restore the files to their original locations on disk. You only need to recover the file-system metadata from the recovery point. Once the metadata is in place, the SAM-QFS software will then stage archival copies of the data to disk automatically, as users access the files. To restore a standalone file system, you recover both the metadata and the data files from the recovery point. Recovery points that contain file data can be extremely large and you must restore both the data and the metadata to disk. So recovery can take considerable time.

While a recent recovery point greatly simplifies recovery, it is a point-in-time backup copy. So a recovery point is seldom a complete record of the state of the file system at the time when a failure occurs. Inevitably, at least a few files are created and changed after one recovery point is completed and before the next one is created. You can—and should—minimize this problem by scheduling creation of recovery points frequently and at times when the file system is not in use. But, in practice, scheduling has to be a compromise, because the file system exists to be used.

For this reason, you must also save point-in-time copies of the *archiver log file*. As each data file is archived, the log file records the volume serial number of the archival media, the archive file, and the path and file name. With this information, you can recover any files that are missing from the recovery point using Solaris or SAM-QFS `tar` utilities. However, this information is volatile. Like most system logs, the archiver log grows rapidly and must thus be overwritten frequently. If you do not make regular copies to compliment your recovery points, you will not have log information when you need it.

The remainder of this section provides instructions for creating and managing recovery points. It contains the following subsections:

- Create a Recovery Point on Demand
- Back Up the Archiver Log

Create a Recovery Point on Demand

Sometimes you may need to capture the metadata from an archiving file system at a point in time outside the normal schedule. Whenever you anticipate potentially disruptive system or facilities maintenance, for example, you can create before and after recovery points to make sure that file systems are protected.

To initiate creation of an unscheduled, on-demand recovery point on demand, proceed as follows:

1. Log in to the SAM-QFS server host as `root`.

   ```
   root@solaris:~#
   ```

2. Select an independent location where the recovery point will be stored.

   In the example, we create a subdirectory, `unscheduled/`, under the directory that we created for recovery points when we configured file systems (see the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide*). The `/zfs1` file system is remotely located and has no components in common with the SAM-QFS file system:

   ```
   root@solaris:~# mkdir /zfs1/samqfs_recovery/unscheduled
   root@solaris:~#
   ```

3. Change to the file system's root directory.

   In the example, we change to the mount-point directory `/samqfs`:

   ```
   root@solaris:~# cd /samqfs
   root@solaris:~#
   ```

4. If you are backing up an archiving file system where the data is copied to removable media, back up the metadata only. Use the command `samfsdump -f` *`recovery-point`*, where *`recovery-point`* is the path and file name of the finished recovery point file.

   See the `samfsdump` man page for additional details. In the example, we create an unscheduled recovery point for the `samqfs` file system prior to a scheduled, maintenance-related power outage. We create the recovery-point file `20140215pre-outage` in the remote directory `/zfs1/samqfs_recovery/unscheduled/`:

   ```
   root@solaris:~# cd /samqfs
   root@solaris:~# samfsdump -f \
   /zfs1/samqfs_recovery/unscheduled/20140215pre-outage
   root@solaris:~#
   ```

5. If you are backing up a standalone file system where data is not copied to removable media, back up both the metadata and the data. Use the command `samfsdump -U -f` *`recovery-point`*, where *`recovery-point`* is the path and file name of the finished recovery point file.

   Note that recovery-point files that include data as well as metadata can be extremely large. See the `samfsdump` man page for additional details. In the example, we create an unscheduled recovery point for the `samqfs` file system. We create the recovery-point file `20140215pre-outage` in the remote directory `/zfs1/samqfs_recovery/unscheduled/`:

```
root@solaris:~# cd /samqfs
root@solaris:~# samfsdump -f -U \
/zfs1/samqfs_recovery/unscheduled/20140215pre-outage
root@solaris:~#
```

6. If you are backing up an archiving file system, Back Up the Archiver Log.
7. Otherwise, depending on the situation, you may also want to Run samexplorer (see Gathering Configuration and Diagnostic Information with samexplorer) and Manually Backup the SAM-QFS Configuration.

Back Up the Archiver Log

While recovery point files contain almost all of the information needed when restoring a file system, they do not hold the metadata for files created or modified after the recovery point was created. Because archiver logs list all of the files that have been archived and their locations on cartridges, archiver logs can be used to recover any files that were archived after the creation of the recovery point. So, if possible, create an unscheduled copy of the archiver log file whenever you create an unscheduled recovery point. Proceed as follows.

1. Log in to the SAM-QFS server host as `root`.

   ```
   root@solaris:~#
   ```

2. Select an independent location where the backed up archiver log will be stored.

   In the example, we decide to store the log in the same directory as the corresponding unscheduled recovery point that we created above. The `/zfs1` file system is remotely located and has no components in common with the SAM-QFS file system:

   ```
   root@solaris:~# ls /zfs1/samqfs_recovery/unscheduled
   20140215pre-outage
   root@solaris:~#
   ```

3. Copy the current archiver log to the chosen location and give it a unique name. Use the command `cp /var/adm/samqfs.archive.log` *`path`*`/"date +%y%m%d";`, where *`path`* is the path to the chosen location.

   ```
   root@solaris:~# cp /var/adm/samqfs.archive.log \
   /zfs1/samqfs_recovery/unscheduled/20140215pre-outage/"date +%y%m%d".archive.log
   root@solaris:~#
   ```

4. Depending on the situation, you may also want to Run samexplorer (see Gathering Configuration and Diagnostic Information with samexplorer) and Manually Backup the SAM-QFS Configuration.

Backing Up the SAM-QFS Configuration

Whenever you change the SAM-QFS configuration, protect your investment by backing up all modified configuration files and related information. Carry out the following tasks:

- Manually Backup the SAM-QFS Configuration
- Run samexplorer and safely store the output.

Manually Backup the SAM-QFS Configuration

For full redundancy, create a local copy of the configuration files whenever you make significant changes to software, operating systems, or hosts. Proceed as follows:

1. Log in to the file-system host as `root`.

   ```
   root@solaris:~#
   ```

2. In the subdirectory that holds your backup configuration information, create a subdirectory for manual backup copies of your SAM-QFS configuration. Use the command `mkdir` *`mount-point`*`/`*`path`*, where *`mount-point`* is the mount point directory for the selected independent file system and *`path`* is the path and name of the chosen directory.

   In the example, we are configuring recovery points for the archiving file system `/samqfs`. So we have created the directory `/zfs1/sam_config/samconfig`:

   ```
   root@solaris:~# mkdir /zfs1/sam_config/samconfig
   ```

3. In the subdirectory that holds manual backup copies of your SAM-QFS configuration, create a subdirectory for the current SAM-QFS configuration. Use the command `mkdir` *`mount-point`*`/`*`path`*`/`*`subdirectory`*, where *`mount-point`* is the mount point for the selected independent file system and *`path`*`/`*`subdirectory`* is the path and name of the chosen subdirectory.

   In the example, we create a subdirectory in the directory that we created for this purpose during initial configuration, `/zfs1/sam_config/samconfig`. We use the date to name the subdirectory:

   ```
   root@solaris:~# mkdir /zfs1/sam_config/samconfig/20140127
   ```

4. Copy the configuration files to another file system.

   ```
   /etc/opt/SUNWsamfs/
        mcf
        archiver.cmd
        defaults.conf
        diskvols.conf
        hosts.family-set-name
        hosts.family-set-name.local
        preview.cmd
        recycler.cmd
        releaser.cmd
        rft.cmd
        samfs.cmd
        stager.cmd
        inquiry.conf
        samremote                   # SAM-Remote server configuration file
        family-set-name             # SAM-Remote client configuration file
        network-attached-library    # Parameters file
        scripts/*                   # Back up all locally modified files
   /var/opt/SUNWsamfs/
   ```

5. Back up all library catalog data, including that maintained by the historian. For each catalog, use the command `/opt/SUNWsamfs/sbin/dump_cat -V` *`catalog-file`*, where *`catalog-file`* is the path and name of the catalog file. Redirect the output to *`dump-file`*, in a new location.

   In the example, we dump the catalog data for `library1` to the file `library1cat.dump` in a directory on the independent NFS-mounted file system `zfs1`:

```
root@solaris:~# dump_cat -V /var/opt/SUNWsamfs/catalog/library1cat > \
/zfs1/sam_config/20140513/catalogs/library1cat.dump
```

6. Copy system configuration files that were modified during SAM-QFS installation and configuration. These may include:

   ```
   /etc/
        syslog.conf
        system
        vfstab
   /kernel/drv/
        sgen.conf
        samst.conf
        samrd.conf
        sd.conf
        ssd.conf
        st.conf
   /usr/kernel/drv/dst.conf
   ```

7. Copy any custom shell scripts and `crontab` entries that you created as part of the SAM-QFS configuration to the selected subdirectory.

   For example, if you created a `crontab` entry to manage creation of recovery points, you would save a copy now.

8. Copy any custom shell scripts and `crontab` entries that you created as part of the SAM-QFS configuration to the selected subdirectory.

   For example, if you created a `crontab` entry to manage creation of recovery points, you would save a copy now.

9. Record the revision level of the currently installed software, including SAM-QFS SAM-QFS, Solaris, and Solaris Cluster (if applicable), and save a copy of the information in a `readme` file in the chosen subdirectory.

10. In the chosen subdirectory, save copies of downloaded SAM-QFS SAM-QFS, Solaris, and Solaris Cluster packages so that you can restore the software quickly, should it become necessary.

11. Next, Run samexplorer.

Gathering Configuration and Diagnostic Information with samexplorer

The `samexplorer` is a diagnostic tool that captures and reports comprehensive configuration and status information for the SAM-QFS software and file systems. Whenever you make changes to your SAM-QFS configuration, you should run `samexplorer` and store the resulting report with your backup copies of the configuration files. You should also run `samexplorer` when you are troubleshooting a problem and whenever you are requested to do so by SAM-QFS support services personnel. Proceed as follows:

Run samexplorer

1. Log in to the file-system host as `root`.

2. In the directory that holds your backup configuration information, create a subdirectory for `samexplorer` reports. Use the command `mkdir` *`mount-point/path`*, where *`mount-point`* is the mount point directory for the selected independent file system and *`path`* is the path and name of the chosen directory.

In the example, we create the new subdirectory in the directory that we created for this purpose during initial configuration, `/zfs1/sam_config/`. We name the new subdirectory `explorer/`:

```
root@solaris:~# mkdir /zfs1/sam_config/explorer
```

3. Create the `samexplorer` report in the selected directory. Use the command `samexplorer` *`path`*`/`*`hostname`*`.`*`YYYYMMDD`*`.`*`hhmmz`*`.tar.gz`, where *`path`* is the path to the chosen directory, *`hostname`* is the name of the SAM-QFS file system host, and *`YYYYMMDD`*`.`*`hhmmz`* is a date and time stamp.

   The default file name is `/tmp/SAMreport.`*`hostname`*`.`*`YYYYMMDD`*`.`*`hhmmz`*`.tar.gz`. In the example, we create a report for the host `samhost1` dated January 30, 2014, at 4:59 PM Mountain time:

```
root@solaris:~# samexplorer \
/zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz

     Report name:
/zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
     Lines per file:  1000
     Output format:   tar.gz (default) Use -u for unarchived/uncompressed.

     Please wait.............................................
     Please wait.............................................
     Please wait......................................

     The following files should now be ftp'ed to your support provider
     as ftp type binary.

     /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
```

4. Repeat this procedure whenever you significantly reconfigure your file systems.
5. Stop here. The SAM-QFS configuration is backed up.

7

Migrating to New Storage Media

To migrate archive files from old to new media, you need to identify the files that will migrate, stage them to the disk cache, and then write them to the new media, without interfering with normal SAM-QFS file-system operations. This chapter covers the following tasks:

- Planning for Data Migration
- Edit the archiver.cmd file
- Migrating Data to Replacement Media

Planning for Data Migration

Before you start moving data, you need to carry out the following tasks.

- Estimate Available Resources
- Plan Post-Migration Disposition of Old Media

Estimate Available Resources

The details of the actual data-migration process depend largely on two factors: the amount of disk storage available and the number of tape drives available. The migration works by staging files from tapes in the old media format to the disk cache and then rearchiving the migrating files from disk cache to tapes in the new media format.

Ideally, you would stage all the files on any given tape volume to disk at once and then immediately archive them to new media. But, to do this, you would have to dedicate significant resources for the duration of the migration:

- disk space equivalent to the capacity of a full tape
- exclusive use of a drive that reads the old tape format
- exclusive use of a drive that writes the new format.

The above is not a problem if you can quiesce the file system for the duration of the migration. But migrating data in a production setting, without unduly interfering with ongoing file system and archive operations, requires some thought. If disk space or tape drives are in short supply, you need to identify the resources that you can reasonably spare for migration and then adjust the migration process. So proceed as follows:

1. Estimate the amount of disk cache that you can use for migration without impeding normal file-system operations.

2. Estimate the number of tape drives that you can afford to dedicate to migration.

   If only a limited number of tape drives are available, plan on throttling the staging and archiving processes so that the migration process does not impede normal operations.

3. Based on your estimates above, decide on staging and archiving parameters. Determine the maximum number of migrating files that the available disk space will hold at any one time and the maximum rate at which files can be moved out of cache and on to new media.

4. Once you have estimated the resources required, proceed with Plan Post-Migration Disposition of Old Media.

Plan Post-Migration Disposition of Old Media

Following data migration, you may or may not need to retain the old media. So identify your requirements and plan for media disposition now, before you begin the data-migration process.

If you have recovery-point files that were created with `samfsdump` prior to the migration, the file-system metadata in the dump file still refers to the old tape volumes. You could not restore the file system to these recovery-points without access to the data on the original media. So, at a minimum, you should retain the old media until you are sure that you will not need to recover the file system to the state it was in at the time the older recovery-point files were created. Once you have rerun `samfsdump` enough times to create recovery points that reference the new volumes, you can dispose of the old volumes.

Once all current recovery-point files point to new media, you can export the old volumes from the robotic library or update and relabel them for re-use, depending on the media type. For example, you might simply export DLT media for disposal. But you could relabel media for an older Oracle StorageTek T10000C drive and use it with a newer T10000D drive.

Once you have decided how you will dispose of the old media, you are ready to start Configuring the Archiving Process for Migration.

Configuring the Archiving Process for Migration

During the migration process, you modify the archiver configuration file, `archiver.cmd`, to include the new, replacement media type alongside the type that you are replacing.

Edit the `archiver.cmd` file

Modify the `archiver.cmd` file so that the second archive copy is sent to the new media. There is no reason to add an additional archive copy.

1. Open the `/etc/opt/SUNWsamfs/archiver.cmd` file in a text editor.

   The archiving policy specifies two copies, both of which are written to the media type that we want to replace. In the example, we open the file in the `vi` editor. We want to replace the DLT cartridges (type `lt`):

   ```
   root@solaris: vi /etc/opt/SUNWsamfs/archiver.cmd
   # =============================================
   # /etc/opt/SUNWsamfs/archiver.cmd
   # ---------------------------------------------
   ...
   ```

```
# --------------------------------------------
# VSN Directives
vsns
samqfsms.1 lt .*
all.1 lt .*
all.2 lt .*
endvsns
```

2. In the directives for copy `2`, change the specified media type to the identifier for the new media, save the file, and close the text editor.

   In the example, we want to migrate data from the old DLT tapes to new LTO cartridges. So, in copy `2`, we change the old media type, `lt` (DLT), to `li` (LTO):

```
root@solaris: vi /etc/opt/SUNWsamfs/archiver.cmd
# ============================================
# /etc/opt/SUNWsamfs/archiver.cmd
# --------------------------------------------
...
# --------------------------------------------
# VSN Directives
vsns
samqfsms.1 lt .*
all.1 lt .*
all.2 li .*
~
:wq
root@solaris:~#
```

3. Check the `archiver.cmd` file for syntax errors. Run the command `archiver -lv`, and correct errors until no errors are found.

   The `archiver -lv` command will print out the file line-by-line. If it encounters an error, it will stop running at the point where the error occurred.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
1: # ============================================
2: # /etc/opt/SUNWsamfs/archiver.cmd
3: # --------------------------------------------
4: # Global Directives
5: logfile = /var/opt/SUNWsamfs/archiver.log
6: # --------------------------------------------
7: # File System Directives:
8: fs = samqfsms
9: all .
10: 1 5m ...
```

4. Once the modified `archiver.cmd` file is error-free, load it into the current configuration using the command `samd config`:

```
root@solaris:~# samd config
```

5. Next, Migrate Data from One Cartridge to Another.

Migrating Data to Replacement Media

The recommended procedure for migrating data uses `sfind`, the SAM-QFS archival file-system extension of the GNU `find` command. The `sfind` command is used to locate files on a specified tape volume and launch the `stage` and `rearchive` commands against all files found. (If you are unfamiliar with the `sfind`, `stage`, and/or `rearchive` commands, you should review their respective man pages now.)

For each tape cartridge that holds data that must be migrated, proceed as follows:

Migrate Data from One Cartridge to Another

1. Log in to the file-system host as `root`.

   ```
   root@solaris:~#
   ```

2. Move to the mount-point directory for the file system that holds the files that you are migrating.

   In the example, we are migrating archived copies of files stored in the `samqfs1` file system mounted at `/samqfs1`:

   ```
   root@solaris:~# cd /samqfs1
   ```

3. Select a tape volume.

   When migrating data from media type to media type, work with one volume at a time. The examples below, we work with volume serial number `VOL008`.

4. First, search the selected volume for damaged files that cannot be successfully staged. Use the SAM-QFS command `sfind . -vsn` *`volume-serial-number`* `-damaged`, where *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume in the library.

   In the example, we start the search from the current working directory (`.`). The `-vsn` parameter limits the search to files that are found on our current tape, `VOL008`. The `-damaged` flag limits the search to files that cannot be successfully staged:

   ```
   root@solaris:~# sfind . -vsn VOL008 -damaged
   ```

5. If the `sfind` search for damaged files returns any results, try to fix the file. Use the command `undamage -m` *`media-type`* `-vsn` *`volume-serial-number file`*, where *`media-type`* is one of the two-character media type codes listed in Appendix A, "Glossary of Equipment Types", *`volume-serial-number`* is the alphanumeric string that uniquely identifies the volume, and *`file`* is the path and name of the damaged file.

   Sometimes a transitory I/O error causes a copy to be marked damaged. The SAM-QFS `undamage` command clears this condition. In the example, the archive file copy `/samqfs1/data0008/20131025DAT` is reported damaged. So we undamage it and retry the search for damaged files:

   ```
   root@solaris:~# sfind . -vsn VOL008 -damaged
   /samqfs1/data0008/20131025DAT
   root@solaris:~# undamage -m lt -vsn VOL008 /samqfs1/data0008/20131025DAT
   root@solaris:~# sfind . -vsn VOL008 -damaged
   root@solaris:~#
   ```

6. If the `sfind` command again lists the file as damaged, the copy is unusable. See if the archive contains another, undamaged copy of the file. To list the available copies, use the command `sls -d` *`file`*, where *`file`* is the path and name of the file. To check the status of any copies found, use the command `sfind` *`file`* `-vsn` *`volume-serial-number`*

   In the example, we list all copies of the file `/samqfs1/data0008/20131025DAT` and find:

```
root@solaris:~# sls -d /samqfs1/data0008/20131025DAT
20131025DAT:
mode: -rw-r--r--  links:   1  owner: root      group: other
            length:    319279  admin id:      7  inode: 1407.5
            project: system(0)
            offline;  archdone;  stage -n;
            copy 1: ---- May 21 07:12     1e4b1.1    lt VOL008
            copy 2: ---- May 21 10:29     109c6.1    lt VOL022
...
root@solaris:~# sfind file -vsn volume-serial-number
/samqfs1/data0008/20131025DAT
root@solaris:~#
```

7. If a copy is unusable and another undamaged copy of the file exists, archive the file so that two good copies exist in the archive before you proceed further. Then unarchive the damaged copy.

   In the example, copy 1 of file /samqfs1/data0008/20131025DAT on volume VOL008 is unusable, so we issue the archive command with the -c option to create a valid copy 1 before unarchiving the damaged copy on volume VOL008:

   ```
   root@solaris:~# archive -c 1 /samqfs1/data0008/20131025DAT
   root@solaris:~# unarchive -m lt -vsn VOL008 /samqfs1/data0008/20131025DAT
   ```

8. If no usable copies exist, see if the file is resident in cache. If it is, archive the file so that two good copies exist in the archive before you proceed. Once two good copies exist, unarchive damaged copy.

   In the example, both copy 1 on volume VOL008 and copy 2 on volume VOL022 are unusable, so we issue the archive command to create two valid copies before unarchiving the damaged copy on volume VOL008:

   ```
   root@solaris:~# archive /samqfs1/data0008/20131025DAT
   root@solaris:~# unarchive -m lt -vsn VOL008 /samqfs1/data0008/20131025DAT
   ```

9. If no usable copies exist and if the file is not resident in the disk cache, the data has probably been lost. If the data is critical, consult a specialist data recovery firm for assistance. Otherwise, unarchive the damaged copy.

   In the example, both copy 1 on volume VOL008 and copy 2 on volume VOL022 are unusable and the file is no longer in the disk cache. The data is not critical, so we unarchive the damaged copy on volume VOL008:

   ```
   root@solaris:~# archive /samqfs1/data0008/20131025DAT
   root@solaris:~# unarchive -m lt -vsn VOL008 /samqfs1/data0008/20131025DAT
   ```

10. When the sfind search for damaged files returns no results, stage the all files from the current tape to the disk cache. Use the command sfind . -vsn *volser* -offline -exec stage {}\;, where *volser* is the volume serial number (VSN) of the tape volume.

    The -exec parameter runs the command line stage {}\; against each result of the sfind search, replacing the paired braces {} with each file name found.

    In the example, the -vsn parameter again limits the search to files that are found on our current tape, VOL008. The -offline parameter further restricts the sfind output to files that are on tape and not already resident in cache. This precaution keeps files from being damaged during staging.

    ```
    root@solaris:~# sfind . -vsn VOL008 -offline -exec stage {} \;
    ```

11. Once files have been staged from tape, selectively rearchive them. Use the command `sfind . -vsn` *`volser`* `-online -exec rearch -r -m` *`media-type`* `{}\;` where *`media-type`* is the type of media from which you are migrating.

    The `-exec` parameter runs the command line `rearch -r -m` *`media-type`* `{}\;` against each result of the `sfind` search, replacing the paired braces `{}` with each file name found. The `-vsn` parameter again limits the search to files that are found on our current tape. The `-online` parameter further restricts the `sfind` output to files that are currently resident in cache rather than those still on tape. This precaution keeps files from being damaged during archiving. The `-r` parameter tells `rearchive` to recurse through directories and subdirectories.

    In the example, the `-vsn` parameter value is `VOL008`, and the value of the `-m` parameter specifies `lt`, for DLT media:

    ```
    root@solaris:~# sfind . -vsn VOL008 -online -exec rearch -r -m lt {} \;
    ```

12. Repeat the preceding step until the `sfind` search finds no more files.

13. When all files have been rearchived, dispose of the tape as planned (see "Plan Post-Migration Disposition of Old Media" on page 7-2).

14. Repeat this procedure until data has been migrated from all old media to new media.

A

Glossary of Equipment Types

The value of the `Equipment Type` field of the Master Configuration File (`mcf`) identifies devices and device configurations within the SAM-QFS software using two-character codes. This glossary lists the codes for quick reference when working with the samples or when interpreting an existing `mcf` (for full details see the `mcf(4)` man page).

For convenience, the codes are divided into three sections and then listed alphabetically:

- Recommended Equipment and Media Types
- Other Equipment and Media Types

Recommended Equipment and Media Types

This section describes all of the equipment codes that you normally need: the generic equipment codes (`rb`, `tp`, and `od`) and codes for identifying network-attached library interfaces and the SAM-QFS historian.

The generic equipment codes `rb`, `tp`, and `od` are the preferred equipment type codes for all SCSI-attached libraries, tape drives, and optical disk devices. When you specify a generic equipment type, SAM-QFS can automatically set the correct type based on SCSI vendor codes.

g*XXX*

Where *XXX* is an integer in the range `[0-127]`, a striped group of disk devices that is part of an `ma` disk-cache family set.

hy

The SAM-QFS historian, an optional, virtual library that maintains a media catalog, but has no associated hardware. Used for tracking exported media.

ma

A high-performance QFS file system that maintains file-system metadata on one or more dedicated `mm` disk devices. File data resides on separate `md`, `mr`, or `g`*XXX* data devices.

md

A disk device that stores file data for an `ma` file system or data and metadata for an `ms` file system. `md` devices store file data in small, 4-kilobyte Disk Allocation Units (DAUs) and large, 16-, 32-, or 64-kilobyte DAUs. The default DAU is 64-kilobytes.

mm

A disk device that stores file-system metadata for a high-performance `ma` file system.

mr
A disk device that stores file data for an ma file system. mr devices store file data in large, fully adjustable Disk Allocation Units (DAUs) that are multiples of 8 kilobytes in the range 8-65528 kilobytes. The default DAU is 64 kilobytes.

ms
A SAM-QFS file system that maintains file-system metadata on the same devices that store file data.

od
Any SCSI-attached optical disk. SAM-QFS sets the appropriate equipment type automatically using the SCSI vendor code.

rb
Any SCSI-attached tape library. SAM-QFS sets the appropriate equipment type automatically using the SCSI vendor code.

rd
The SAM-Remote pseudo-device. In the Master Configuration File (`mcf`), the corresponding `Equipment Identifier` field has to contain the path to the pseudo-device (such as `/dev/samrd/rd2`). The corresponding `Family Set` field has to contain the hostname of the SAM-Remote server.

sc
A SAM-Remote client system. In the Master Configuration File (`mcf`), the corresponding `Equipment Identifier` field has to contain the path the SAM-Remote client-configuration file for the client. The corresponding `Family Set` field has to contain the family set name of the server. The `Additional Parameters` field must contain the full path to the client's library catalog file.

sk
An Oracle StorageTek ACSLS interface to a network-attached library. In the Master Configuration File (`mcf`), the corresponding `Equipment Identifier` field has to contain the path to the parameters file for the ACSLS interface. For more information, see the `stk(7)` man page.

ss
A SAM-Remote server. In the Master Configuration File (`mcf`), the corresponding `Equipment Identifier` field has to contain the path to the SAM-Remote server-configuration file. The corresponding `Family Set` field has to contain the family set name of the server, which must match the name used in the `Family Set` field of the `mcf` on the client.

tp
Any SCSI-attached tape drive. SAM-QFS sets the appropriate equipment type automatically using the SCSI vendor code.

Other Equipment and Media Types

The equipment types listed in this section are also supported.

Note that, in most cases, Oracle recommends identifying SCSI-attached libraries, tape drives, and optical disk devices using the generic equipment types `rb`, `tp`, and `od`. The generic equipment types tell SAM-QFS to identify the hardware dynamically, using SCSI vendor IDs. The type codes below are essential when migrating from one media type to another and may sometimes be useful for management purposes. But using them in a Master Configuration File (`mcf`), for example, hard-codes a static equipment configuration that may not match the actual hardware.

ac
A Sun 1800, 3500, or L11000 tape library.

at
A Sony AIT-4 or AIT-5 tape drive.

cy
A Cygnet optical disk library.

d3
A StorageTek D3 tape drive.

dm
A Sony DMF library.

ds
A DocuStore or Plasmon optical disk library.

dt
A DAT 4-mm tape drive.

e8
An Exabyte X80 library.

fd
A Fujitsu M8100 128-track tape drive.

h4
An HP SL48 or SL24 library.

hc
An Hewlett Packard L9-/L20-/L60-series library.

i7
An IBM 3570 tape drive.

ic
An IBM 3570 media changer.

il
An IBM 3584 tape library.

li
An LTO-3 or later tape drive.

lt
A Digital Linear Tape (DLT), Super DLT, or DLT-S4 tape drive.

me
A Metrum library.

mf
An IBM Multi Function optical drive.

mo
A 5.25-in erasable optical drive.

o2
A 12-in WORM drive.

ov
An Overland Data Inc. Neo Series tape library.

pd
A Plasmon D-Series DVD-RAM library.

q8
A Qualstar 42xx, 62xx, or 82xx library.

s3
A StorageTek SL3000 library.

s9
An Oracle StorageTek 97xx series library.

se
A StorageTek 9490 tape drive.

sf
A StorageTek T9940 tape drive.

sg
A StorageTek 9840C or later tape drive.

sl
A Spectra Logic or Qualstar tape library.

st
A StorageTek 3480 tape drive.

ti
A StorageTek T10000 (Titanium) tape drive.

vt
A Metrum VHS (RSP-2150) tape drive.

wo
A 5.25-in optical WORM drive.

xt
An Exabyte (850x) 8-mm tape drive.

B

Media Status Flags

Media flags have the following meanings:

- `A` means that the slot needs an audit.
- `C` means that the slot contains cleaning cartridge.
- `E` means that the volume is bad or the cleaning media has expired.
- `L` means that the volume is a Linear Tape File System (LTFS) volume.
- `N` means that the volume is not in SAM-QFS format.
- `R` means that the volume is read-only (a software flag).
- `U` means that the volume is unavailable.
- `W` means that the volume is physically write-protected.
- `X` means that the slot is an export slot.
- `b` means that the volume has a bar code.
- `c` means that the volume is scheduled for recycling
- `f` means that the archiver found the volume full or corrupt.
- `d` means that the volume has a duplicate volume serial number (VSN)
- `l` means that the volume is labeled.
- `o` means that the slot is occupied.
- `p` means that the volume is a high priority volume.
- `-`, when used in displays, means that the corresponding flag is not set.

For example, the `samcmd v` lists catalog information, including media flags for each cataloged volume:

```
root@solaris:~# samcmd v 800
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                         count 32
Robot VSN catalog by slot       : eq 800
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li VOL001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li VOL002
   2     2014/03/14 11:26    3   0%  -il-o-b-----  li VOL003
   3     2014/03/14 10:33    3   0%  -il-o-b-----  li VOL004
   4     2014/03/14 11:34    5   0%  -il-o-b-----  li VOL005
   5     2014/03/14 11:32    2   0%  -ilEo-b----f  li VOL006 MEDIA ERROR
   6     2014/03/13 18:07    2   0%  -il-o-b-----  li VOL007
   7     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL008
  10     2014/03/13 18:10    2   0%  -il-o-b-----  li VOL011
```

C

Mount Options in a Shared File System

The SAM-QFS shared file system can be mounted with several mount options. This chapter describes some of these options within the context of their roles.

Shared File System Mount Options

You can specify most mount options by using the mount command, by entering them in the /etc/vfstab file, or by entering them in the samfs.cmd file. For example, the following /etc/vfstab file includes mount options for a shared file system:

```
sharefs - /sfs samfs - no shared,mh_write
```

You can change some mount options dynamically by using the samu operator utility. For more information about these options, see the *StorageTek Storage Archive Manager and StorageTek QFS samu Command Reference*.

For more information about any of these mount options, see the mount_samfs man page.

bg: Mounting in the Background

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.

retry: Reattempting a File System Mount

The retry mount option specifies the number of times that the system should attempt to mount a file system. The default is 10000.

shared: Declaring a SAM-QFS Shared File System

The shared mount option declares a file system to be a SAM-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 SAM-QFS shared file system. The presence of this option in a samfs.cmd file or on the mount command does not cause an error condition, but it does not mount the file system as a shared file system.

minallocsz and maxallocsz: Tuning Allocation Sizes

The minallocsz and maxallocsz options to the mount command specify an amount of space, in kilobytes. These options set the minimum block allocation size. If a file is growing, the metadata server allocates blocks when an append lease is granted. Use -o minallocsz=*n* to specify the initial size of this allocation. 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`* setting.

You can specify these `mount` options on the `mount` command line, in the `/etc/vfstab` file, or in the `samfs.cmd` file.

`rdlease`, `wrlease`, and `aplease`: Using Leases in a SAM-QFS Shared File System

By default, when hosts share files, the SAM-QFS metadata server maintains file-system consistency by issuing I/O *leases* to itself and its clients. A lease grants a shared host permission to perform an operation on a file for a specified period. A *read lease* lets a host read file data. A *write lease* lets a host overwrite existing file data. An *append lease* lets a host write additional data at the end of a file. The metadata server can renew leases as necessary.

Reads and writes to a SAM-QFS shared file system should thus provide near-POSIX behavior for data. For metadata, however, access time changes might not be seen immediately on other hosts. Changes to a file are pushed to disk at the end of a write lease. When a read lease is acquired, the system invalidates any stale cache pages so that the newly written data can be seen.

The following mount options set the duration of the leases:

- `-o rdlease=` *`number-seconds`* specifies the maximum amount of time, in seconds, for the read lease.
- `-o wrlease=` *`number-seconds`* specifies the maximum amount of time, in seconds, for the write lease.
- `-o aplease=` *`number-seconds`* specifies the maximum amount of time, in seconds, for the append lease.

In all three cases, *`number-seconds`* is an integer in the range [`15-600`]. The default time for each lease is `30` seconds. A file cannot be truncated if a lease is in effect. For more information about setting these leases, see the `mount_samfs` man page.

If you change the metadata server because the current metadata server is down, you must add the lease time to the changeover time because all leases must expire before an alternate metadata server can assume control.

Setting a short lease time causes more traffic between the client hosts and the metadata server because the lease must be renewed after it has expired.

`mh_write`: Enabling Multiple Host Reads and Writes

The `mh_write` option controls write access to the same file from multiple hosts. If `mh_write` is specified as a mount option on the metadata server host, the SAM-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.

By default, `mh_write` is disabled, and only one host has write access to a file for the duration of the `wrlease` mount option. If the SAM-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.

When `mh_write` is enabled on the metadata server, SAM-QFS supports the following:

- Multiple reader hosts and paged I/O
- Multiple reader and/or writer hosts and direct I/O only if there are writers

- One append host (other hosts read or write) and direct I/O only if there are writers.

Mounting a file system with the mh_write option does not change locking behavior. File locks behave the same regardless of whether mh_write is in effect. But, in other respects, behavior might be less consistent. When there are simultaneous readers and writers, the SAM-QFS shared file system uses direct I/O for all host access to a file. Therefore, page-aligned I/O should be visible immediately to other hosts. However, non-page-aligned I/O can result in stale data being visible, or even written to the file, because the normal lease mechanism preventing such occurrences has been disabled.

For this reason, you should specify the mh_write option only when multiple hosts need to write the same file simultaneously and when hosted applications perform page-aligned I/O and coordinate conflicting writes. In other cases, data inconsistency could occur because, even using flock() with mh_write to coordinate between hosts does not guarantee consistency. For more information, see the mount_samfs man page.

min_pool: Setting the Minimum Number of Concurrent Threads

The min_pool mount option sets the minimum number of concurrent threads for the SAM-QFS shared file system. The default setting is min_pool=64 on Oracle Solaris systems. This setting means that at least 64 active threads will be in the thread pool on Oracle Solaris. You can adjust the min_pool setting to any value in the range [8-2048], depending on the SAM-QFS shared file system's activity.

The min_pool mount option must be set in the samfs.cmd file. It will be ignored if set in the /etc/vfstab file or at the command line.

meta_timeo: Retaining Cached Attributes

The meta_timeo mount option determines how long the system waits between checks on the metadata information. By default, the system refreshes metadata information every three seconds. For example, an ls command entered in a SAM-QFS shared file system with several newly created files might not return information about all the files until three seconds have passed. The syntax for the option is meta_timeo=*seconds* where *seconds* is an integer in the range [0-60].

stripe: Specifying Striped Allocation

By default, data files in the SAM-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. The maximum performance for one file will be the speed of a slice or striped group. For more information about file allocation methods, see the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide* (*SAM-QFS Customer Documentation Library*, docs.oracle.com.).

sync_meta: Specifying the Frequency With Which Metadata Is Written

You can set the sync_meta option to sync_meta=1 or sync_meta=0.

The default setting is sync_meta=1, which means that a SAM-QFS shared file system writes file metadata to disk every time the metadata changes. This setting slows data performance but ensures data consistency. This setting must be in effect if you want to change the metadata server.

If you set `sync_meta=0`, the SAM-QFS shared file system writes the metadata to a buffer before writing it to disk. This delayed write delivers higher performance but decreases data consistency after an unscheduled machine interruption.

`worm_capable` and `def_retention`: Enabling WORM Functionality

The `worm_capable` mount option lets the file system support WORM files. The `def_retention` mount option sets the default retention time using the format `def_retention=`*`M`*`y`*`N`*`d`*`O`*`h`*`P`*`m`.

In this format, *`M`*, *`N`*, *`O`*, and *`P`* are non-negative integers and `y`, `d`, `h`, and `m` stand for years, days, hours, and minutes, respectively. Any combination of these units can be used. For example, `1y5d4h3m` indicates 1 year, 5 days, 4 hours, and 3 minutes; `30d8h` indicates 30 days and 8 hours; and `300m` indicates 300 minutes. This format is backward compatible with the formula in previous software versions, in which the retention period was specified in minutes.

Seethe *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide* (*SAM-QFS Customer Documentation Library*, `docs.oracle.com`.) for more information.

D

Configuration Directives for Archiving

This appendix lists directives used in the SAM-QFS command (`.cmd`) files that configure archiving file systems and related software operations (for comprehensive descriptions, see the SAM-QFS man pages). Directives are divided into the following sections:

- Archiving Directives
- Staging Directives
- Preview Request Directives

You can configure SAM-QFS command files either from the command line, as described here, or by using the SAM-QFS Manager software. For information on SAM-QFS Manager, see the online help.

Archiving Directives

This section provides usage information for the *archiving directives* that make up the `archiver.cmd` file. Archiving directives define the archive sets that control how files are archived, the media used, and the overall behavior of the archiving software.

There are four basic types of archiving directives:

- Global Archiving Directives
- File System Directives
- Copy Parameters
- Volume Serial Number (VSN) Association Directives

Both global and file system directives control how files are archived. But the archiver evaluates the file system-specific directives before evaluating the global directives. So file system directives override global directives when there is a conflict. Similarly, within the file system directives, the first directive listed overrides any subsequent, conflicting directives.

Global Archiving Directives

Global directives control the overall archiver operation and enable you to optimize operations for all configured file systems. Global directives are specified by entering equal sign (=) in the second field or the absence of additional fields. Global directives start the `archiver.cmd` file and end at the first of the File System Directives.

archivemeta: Controlling Whether Metadata Is Archived

The archivemeta directive controls whether file system metadata is archived. If files are often moved around and there are frequent changes to the directory structures in a file system, archive the file system metadata. But if the directory structures are reasonably stable, you can disable metadata archiving and reduce the actions performed by removable media drives. By default, metadata is not archived.

This directive has the following format:

```
archivemeta=state
```

For *state*, specify either on or off. The default is off.

The archiving process for metadata depends on whether you are using a Version 1 or a Version 2 superblock, as follows:

- For Version 1 file systems, the archiver archives directories, removable media files, segment index inodes, and symbolic links as metadata.
- For Version 2 file systems, the archiver archives directories and segment index inodes as metadata. Removable media files and symbolic links are stored in inodes rather than in data blocks. They are not archived. Symbolic links are archived as data.

archmax: Controlling the Size of Archive Files

The archmax directive specifies the maximum size of an archive (.tar) file. After the *target-size* value is met, no more user files are added to the archive file. Large user files are written in a single archive file.

To change the defaults, use the following directive:

```
archmax=media target-size
```

where *media* is one of the media types defined in Appendix A and in the mcf man page and where *target-size* is the maximum size of the archive file. This value is media-dependent. By default, archive files written to optical discs are no larger than 5 megabytes. The default maximum archive file size for tapes is 512 megabytes.

Setting large or small sizes for archive files has advantages and disadvantages. For example, if you are archiving to tape and archmax is set to a large size, the tape drive stops and starts less often. However, when writing large archive files, a premature end-of-tape causes a large amount of tape to be wasted. As a best practice, do not set the archmax directive to be more than 5 percent of the media capacity.

The archmax directive can also be set for an individual archive set.

bufsize: Setting the Archiver Buffer Size

By default, a file being archived is copied to archive media using a memory buffer. You can use the bufsize directive to specify a non-default buffer size and, optionally, to lock the buffer. These actions can improve performance. You can experiment with different *buffer-size* values. This directive has the following format:

```
bufsize=media buffer-size [lock]
```

where:

- *media* is one of the media types defined in Appendix A and in the mcf man page
- *buffer-size* is a number in the range [2-1024]. The default is 4. This value is multiplied by the dev_blksize value for the media type, and the resulting buffer

size is used. The dev_blksize value is specified in the defaults.conf file. See the defaults.conf man page for details.

- lock indicates whether the archiver can use locked buffers when making archive copies.

  If lock is specified, the archiver sets file locks on the archive buffer in memory for the duration of the sam-arcopy operation. This action avoids the overhead associated with locking and unlocking the buffer for each I/O request and results in a reduction in system CPU time.

  The lock argument must be specified only on large systems with large amounts of memory. Insufficient memory can cause an out-of-memory condition. The lock argument is effective only if direct I/O is enabled for the file being archived. By default, lock is not specified and the file system sets the locks on all direct I/O buffers, including those for archiving.

You can specify a buffer size and a lock for each archive set by using the archive set copy Parameters, -bufsize and -lock. For more information, see "Archive Copy Directives" on page D-8.

drives: Controlling the Number of Drives Used for Archiving

By default, the archiver uses all of the drives in an automated library for archiving. To limit the number of drives used, use the drives directive. This directive has the following format:

```
drives=auto-lib count
```

where *auto-lib* is the family set name of the automated library as defined in the mcf file and *count* is the number of drives to be used for archiving activities.

You can also use the archive set copy parameters -drivemax, -drivemin, and -drives for this purpose.

examine: Controlling Archive Scans

The examine directive sets the *method* that the archiver uses to identify files that are ready for archiving:

```
examine=method
```

where *method* is one of the following directives:

- noscan, the default, specifies *continuous archiving*. After the initial scan, directories are scanned only when the content changes and archiving is required. Directory and inode information is not scanned. This archiving method provides better performance than scan archiving, particularly for file systems with more than 1,000,000 files.
- scan specifies *scan archiving*. The initial file system scan is a directory scan. Subsequent scans are inode scans.
- scandirs specifies scan archiving on directories only. If the archiver finds a directory with the no_archive attribute set, the directory is not scanned. If you have files that do not change, place them in this type of directory to reduce the amount of time spent on archiving scans.
- scaninodes specifies scan archiving on inodes only.

interval: Specifying an Archive Interval

The archiver runs periodically to examine the status of all mounted archived-enabled file systems. The timing is controlled by the archive interval, which is the time between scan operations on each file system. To change the archive interval, use the interval directive.

The interval directive initiates full scans only when continuous archiving is not set and no startage, startsize, or startcount Parameters have been specified. If continuous archiving is set (examine=noscan), the interval directive acts as the default startage value. This directive has the following format:

```
interval=time
```

For *time*, specify the amount of time you want between scan operations on a file system. By default, *time* is interpreted in seconds and has a value of 600, which is 10 minutes. You can specify a different unit of time, such as minutes or hours.

If the archiver receives the samu utility's arrun command, it begins scanning all file systems immediately. If the examine=scan directive is also specified in the archiver.cmd file, a scan is performed after arrun or arscan is issued.

If the hwm_archive mount option is set for the file system, the archive interval can be shortened automatically. This mount option specifies that the archiver commences its scan when the file system is filling up and the high-water mark is crossed. The high=*percent* mount option sets the high-water mark for the file system.

For more information about specifying the archive interval, see the archiver.cmd and mount_samfs man pages.

logfile: Specifying an Archiver Log File

The archiver can produce a log file that contains information about each file that is archived, re-archived, or unarchived. The log file is a continuous record of archival action. By default, archiver log files are not enabled. To specify a log file, use the logfile directive. This directive has the following format:

```
logfile=pathname
```

For *pathname*, specify the absolute path and name of the log file. The logfile directive can also be set for an individual file system.

Archiver log files are essential for recovering damaged or lost file systems and can be valuable for monitoring and analysis. So you should enable archiver logs and back them up. See the *StorageTek Storage Archive Manager and StorageTek QFS Software Installation and Configuration Guide* (*SAM-QFS Customer Documentation Library*, docs.oracle.com.) for more information.

notify: Renaming the Event Notification Script

The notify directive sets the name of the archiver's event notification script file. This directive has the following format:

```
notify=filename
```

For *filename*, specify the name of the file containing the archiver event notification script or the full path to this file. The default file name is /etc/opt/SUNWsamfs/scripts/archiver.sh.

The archiver executes this script to process various events in a site-specific manner. The script is called with one of the following keywords for the first argument: emerg, alert, crit, err, warning, notice, info, and debug.

Additional arguments are described in the default script. For more information, see the archiver.sh man page.

ovflmin: Controlling Volume Overflow

When volume overflow is enabled, the archiver can create archived files that span multiple volumes. When a file size exceeds the specified minimum size, the archiver writes the remaining portion of this file to another volume of the same type. The portion of the file written to each volume is called a *section*. The sls command lists the archive copy, showing each section of the file on each volume.

The archiver controls volume overflow through the ovflmin directive. By default, volume overflow is disabled. To enable volume overflow, use the ovflmin directive in the archiver.cmd file. This directive has the following format:

```
ovflmin = media minimum-file-size
```

where *media* is one of the media types defined in Appendix A and in the mcf man page and *minimum-file-size* is the size of the smallest file that you want to trigger the volume overflow. The ovflmin directive can also be set for an individual archive set.

Use volume overflow with caution after assessing its effects. Disaster recovery and recycling are much more difficult with files that span volumes.

Volume overflow files do not generate checksums. For more information on using checksums, see the ssum man page.

scanlist_squash: Controlling Scanlist Consolidation

The scanlist_squash parameter controls scanlist consolidation. The default setting is off. This parameter can be either global or file system-specific.

When this option is enabled, the scan list entries for files in two or more subdirectories with the same parent directory that need to be scanned by sam-arfind at a much later time are consolidated. These directories are combined upwards to the common parent, which results in a deep recursive scan of many subdirectories. This consolidation can cause a severe performance penalty if archiving on a file system that has a large number of changes to many subdirectories.

setarchdone: Controlling the Setting of the archdone Flag

The setarchdone Parameter is a global directive that controls the setting of the archdone flag when the file is examined by sam-arfind. This directive has the following format:

```
setarchdone=state
```

where *state* is either on or off.

When all archive copies for a file have been made, the archdone flag is set for that file to indicate that no further archive action is required. During directory scans, the archdone flag is also set for files that will never be archived. Because evaluating whether a file will ever be archived can affect performance, the setarchdone directive gives you control over this activity. This directive controls the setting of the archdone flag only on files that will never be archived. It does not affect the setting of the archdone flag after archive copies are made.

The default setting for the directive is off if the examine directive is set to scandirs or noscan.

wait: Delaying Archiver Startup

The wait directive causes the archiver to wait for a start signal from samu or SAM-QFS File System Manager. By default, the archiver begins archiving when started by sam-fsd. This directive has the following format:

```
wait
```

The wait directive can also be set for an individual file system.

File System Directives

File system directives define archiving behavior for specific file systems:

- fs: Specifying a File System
- copy-number [archive-age]: Specifying Multiple Copies of File System Metadata
- interval, logfile, scanlist as File System Directives

fs: Specifying a File System

Each fs=*file-system-name* directive introduces a sequence of archiving directives that apply only to the named file system, *file-system-name*. This directive has the following format:

```
fs=file-system-name
```

where *file-system-name* is the file system name defined in the mcf file.

General directives and archive set association directives that occur after an fs= directive apply only to the specified file system.

copy-number [*archive-age*]: Specifying Multiple Copies of File System Metadata

File system metadata includes path names in the file system. If you have frequent changes to directories, the new path names cause the creation of new archive copies and result in frequent loading of the volumes specified for metadata. If more than one copy of the metadata is required, place copy definitions in the archiver.cmd file immediately after the fs= directive.

```
copy-number [archive-age]
```

where time is expressed as one or more combinations of an integer and a unit of time. Units include s (seconds), m (minutes), h (hours), d (days), w (weeks), and y (years). By default, SAM-QFS makes only a single copy of the metadata.

In the example, copy 1 of the metadata for the fs=samma1 file system is made after 4 hours (4h) and copy 2 is made after twelve hours (12h):

```
# General Directives
archivemeta = off
examine = noscan
# Archive Set Assignments
fs = samma1
1 4h
2 12h
```

`interval`, `logfile`, `scanlist` as File System Directives

Several directives can be specified both as global directives for all file systems and as directives specific to only one file system. These directives are described in the following sections:

- `interval`: Specifying an Archive Interval
- `logfile`: Specifying an Archiver Log File
- `scanlist_squash`: Controlling Scanlist Consolidation
- `wait`: Delaying Archiver Startup

`archive-set-name`: the Archive Set Assignment Directive

The archive set assignment directive specifies which files will be archived together. You can specify files very narrowly, using the wide range of selection criteria described below. But avoid doing so unless absolutely necessary. In general, you should configure the smallest of number of the most inclusive archive sets possible. Archive sets have exclusive use of a set of archival media. So large numbers of archive sets each defined by excessively restrictive assignment criteria cause poor media utilization, high system overhead, and reduced performance. In extreme cases, jobs may fail due to lack of usable media, even though ample capacity remains in the library.

Each archive set assignment directive has the following format:

```
archive-set-name path [-access interval [-nftv]] [-after date-time] [-minsize
size] [-maxsize size] [-user username] [-group groupname] [-name regex]
```

where:

- *`archive-set-name`* is the administrator-defined name for the archive set.

  Names can contain up to 29 characters in any combination of upper and/or lower case letters [`A-Za-z`], numerals [`0-9`], and underscore characters (_), as long as the first character is a letter. You cannot include any other characters, including spaces, and you cannot use the names of the SAM-QFS special archive sets `no_archive` and `allsets` for your own archive sets.

- *`path`* specifies the path relative to the mount point of the subdirectory where archiving starts within the file system. All files in the starting directory and its subdirectories are archived. To include all of the files in a file system, use the dot (`.`) character. A leading slash (`/`) is not allowed in the path.

- `-access` rearchives files that have not been accessed for the amount of time specified by *`interval`*, where *`interval`* is an integer followed by one of following units: `s` (seconds), `m` (minutes), `h` (hours), `d` (days), `w` (weeks), and `y` (years).

  This parameter lets you schedule rearchiving of less used files from higher to lower cost media. The software validates the access and modification times for files to ensure that they are greater than or equal to the file creation time and less than or equal to the time at which the file is examined. The `-nftv` (no file time validation) parameter disables this validation.

- `-after` archives only files that have been created or modified after *`date-time`*, where *`date-time`* is an expression of the form *`YYYY-MM-DD`* `[`*`hh`*`:`*`mm`*`:`*`ss`*`] [Z]` and where `YYYY`, *`MM`*, *`DD`*, *`hh`*, *`mm`*, and *`ss`* are integers representing the year, month, day, hour, minutes, and seconds, respectively. The optional `Z` parameter sets the time zone to Coordinated Universal Time (UTC). The defaults are `00:00:00` and local time.

- -minsize and -maxsize archive only those files that are over or under the specified *size*, where *size* is an integer followed by one of the following units: b (bytes), k (kilobytes), M (megabytes), G (gigabytes), T (terabytes), P (petabytes), E (exabytes).
- -user *username* and -group *groupname* archive only files that belong to the specified user and/or group.
- -name archives all files that have path and file names matching the pattern defined by the regular expression *regex*.

Archive Copy Directives

By default, the archiver writes a single archive copy for files in the archive set when the archive age of the file is four minutes. To change the default behavior, use archive copy directives. Archive copy directives must appear immediately after the archive set assignment directive to which they pertain.

The archive copy directives begin with a *copy-number* value of 1, 2, 3, or 4. The digit is followed by one or more arguments that specify archive characteristics for that copy. Each archive copy directive has the following format:

```
copy-number [archive-age] [-release [attribute] [-norelease][-stage[attribute]
[unarchive-age]
```

where:

- The optional *archive-age* parameter is the time that a new or modified file must spend in the disk cache before it becomes eligible for archiving. Specify *archive-age* as one or more combinations of an integer and a unit of time, where units include **s** (seconds), **m** (minutes), h (hours), d (days), w (weeks), and y (years). The default is 4m (4 minutes).
- The optional -release parameter clears the SAM-QFS releaser software to free the disk space used by files as soon as an archive copy has been made. The optional release *attribute* is -a, -n, or -d. The -a (**associative staging**) attribute requires that the software stage all files that have been released from the archive set when any one of them is accessed. The -n attribute requires that the software read directly from the archive media and never stage files.The -d attribute resets the default staging behavior.
- The optional -norelease parameter does not clear the SAM-QFS releaser software to free the disk space used by files until all copies marked with -norelease have been made.
- -release -norelease, used together, require that the SAM-QFS software free the disk space used by files immediately after all copies that are flagged -release -norelease are made. SAM-QFS does not wait for the releaser process to run.
- The optional -stage parameter *attribute* is -a, -c *copy-number*, -f, -I, -i *input_file*, -w, -n, -p, -V, -x, -r, -d, where:

  -a (associative staging) requires staging of all files from the archive set when any one of them is accessed.

  -c *copy-number* requires that the software stage from the specified copy number.

  -n requires that the software read directly from the archive media and never stage files.

  -w requires that the software wait for each file to be successfully staged before proceeding (not valid with -d or -n).

  -d resets the default staging behavior.

- *unarchive-age* parameter specifies the amount of time that an archival copy of a file spends in the archive before it is unarchived to free space on the media for reuse. Time is expressed as one or more combinations of an integer and a unit of time, where units include `s` (seconds), `m` (minutes), `h` (hours), `d` (days), `w` (weeks), and `y` (years).

The example below contains two copy directives for archive set `allsamma1`. The first directive does not release copy `1` until it reaches an archive age of five minutes (`5m`). The second directive does not release copy `2` until it reaches an archive age of one hour (`1h`) and unarchives copy `2` once it reaches the unarchive age of seven years and six months (`7y6m`):

```
# Archive Set Assignments
fs = samma1
logfile = /var/adm/samma1.archive.log
allsamma1 .
    1 -norelease 5m
    2 -norelease 1h 7y6m
```

Copy Parameters

Copy parameters define how the copies specified by an archive set are created.The archive set copy parameters section of the `archiver.cmd` file begins with the `params` directive and ends with the `endparams` directive:

```
params
allsets -sort path -offline_copy stageahead
allsets.1 -startage 10m -startsize 10M -drives 10 -archmax 1G
allsets.2 -startage 1h -startsize 1G -drives 2 -archmax 10G -reserve set
endparams
```

Each copy parameter takes the following form:

```
archive-set-name[.copy-number][R] [-startage time] [-startcount count] [-startsize
size] [-archmax maximum-size] [-bufsize=buffer-size] [-drivemax maximum-size]
[-drivemin minimum-size] [-drives number] [-fillvsns] [-lock] [-offline_copy
method] [-sort criterion] [-rsort criterion] [-recycle_dataquantity size]
[-recycle_hwm percent] [-recycle_ignore] [-recycle_mailaddr mail-address]
[-recycle_mingainpercentage] [-recycle_vsncountcount ] [-recycle_minobs
percentage] [-unarchagetime_ref] [-tapenonstop] [-reserve keyword ] [-priority
multiplier ranking]
```

where:

- *archive-set-name* is the name of an archive set defined by an archive set assignment directive in the File System Directives or the special directive `allsets`, which applies the specified copy parameters to all defined archive sets. Set parameters for `allsets` first, before specifying parameters for individual archive sets. Otherwise, the parameters for individual archive sets will override the `allsets` specification, defeating its purpose.
- *.copy-number* limits the application of the specified copy parameters to the archive copy specified by *copy-number*, where *copy-number* is an integer in the range [`1-4`], and the optional `R` limits application of the parameters to rearchived copies.
- `-startage` *time* specifies interval between the moment when the first file is added to an archive request and the moment when archiving actually begins. Specify *time* as one or more combinations of an integer and a unit of time, where units

include s (seconds), m (minutes), h (hours), d (days), w (weeks), and y (years). The default is 2h (two hours).

- -startcount *count* specifies the minimum number of files in an archive request. Archiving begins when the number of files awaiting archiving reaches this threshold. By default, *count* is not set.
- -startsize *size* specifies the minimum size, in bytes, of an archive request. Archiving begins when the total size of the files awaiting archiving reaches this threshold. By default, *size* is not set.
- -archmax limits the size of an archive file to no more than *maximum-size*, where *maximum-size* is media-dependent. The default maximum archive file size for magnetic tape is 512 megabytes. Archive files written to optical discs are no larger than 5 megabytes.

  See "archmax: Controlling the Size of Archive Files" on page D-2 for a description of the global archiving directive of the same name.
- -bufsize= *buffer-size* sets the size of the buffer that holds the holds the archive file as it is being written out to archival media to *buffer-size*dev_blksize*, where *buffer-size* is an integer in the range [2-32] and *dev_blksize* is the block size specified for the media type in the defaults.conf file. The default is 4.
- -drivemax limits the amount of data archived using one drive to no more than *maximum-size* megabytes, where *maximum-size* is an integer. By default, *maximum-size* is not specified.

  When multiple drives are specified using the -drives parameter, limiting the amount of data written to any one drive can improve drives can help to balance workloads and improve overall drive utilization.
- -drivemin *minimum-size* limits the amount of data archived using one drive to at least *minimum-size* megabytes, where *minimum-size* is an integer. The default is the value of -archmax (if specified) or the value listed for the media type in the defaults.conf file.

  Setting a lower limit on the amount of data written to a drive can improves drive utilization and efficiency. Set *minimum-size* large enough that the transfer time significantly exceeds the time required to load, position, and unload media. When -drivemin is specified, multiple drives are only used when data transfers are large enough.
- -drives *number* limits the number of drives used for archiving to at most *number*, where *number* is an integer. The default is **1**.

  Setting a higher maximum number of drives can improve performance when archive sets contain large files or large numbers of files. If the available drives operate at different speeds, specifying multiple drives can also balance these variations and increase archiving efficiency.
- -fillvsns forces the archiving process to use smaller archive files that fill archival media volumes more completely.

  By default, the archiver selects a volume with enough space to hold the all files in an archive copy. This results in larger archive files that may not fit into the remaining capacity on many cartridges. As a result, media is under-utilized overall. The -fillvsns parameter addresses this issue, but at the cost of more media mounts, positioning operations, and unmounts, all of which reduce archiving and staging performance.

- `-lock` mandates the use of locked buffers when making archive copies using direct I/O. Locked buffers prevent paging of the buffer and improve direct I/O performance.

  The `-lock` parameter can cause an out-of-memory condition if specified on systems that have limited memory available. By default, locked buffers are not mandated, and the file system retains control over the archiving buffer.

- `-offline_copy` *`method`* specifies how archive copies are made when files have already been released from the disk cache. The specified *`method`* can be `direct`, `stageahead`, `stageall`, or `none`.

  Files can be released as soon as a single archive copy is made, so the remaining copies must be made from an offline copy. A specified `-offline_copy` method lets you tailor the copy process to suit the number of drives that can be made available and the amount of space available in the disk cache.

  `direct` copies files directly from the offline volume to the archive volume, using two drives. To insure adequate buffer space, increase the value set by the `stage_n_window` mount option when using this method.

  `stageahead` stages the next archive file while writing an archive file to its destination.

  `stageall` stages all files to disk cache before archiving, using one drive. Make sure that the disk cache is large to hold the files when using this method.

  `none` (the default) stages files to the disk cache as needed before copying to the archive volume.

- `-sort` sorts files by *`criterion`* before archiving them, where *`criterion`* is `age`, `priority`, `size`, or `none`.

  `age` specifies sorting by modification time, from oldest to most recent.

  `path` (the default) specifies sorting by full path name and thus keeps files that reside in the same directories together on the archive media.

  `priority` specifies sorting by archiving priority, from highest to lowest.

  `size` sorts files by file size, from smallest to largest.

  `none` specifies no sorting and archives files in the order in which they are encountered in the file system.

- `-rsort` *`criterion`* sorts files by *`criterion`* like `-sort`, but in reverse order.

- `-recycle_dataquantity` *`size`* limits the amount of data that the recycler will schedule for rearchiving to *`size`* bytes, where *`size`* is an integer.

  The recycler schedules rearchiving when it need to drain archival volumes of valid archive files. Note that the actual number of volumes selected for recycling may also depend on the `-recycle_vsncount` parameter. The default is `1073741824` (1 gigabyte).

- `-recycle_hwm` *`percent`* sets the maximum percent media utilization (the high water mark or `hwm`) that initiates recycling of removable media. The parameter is ignored for disk media (see `-recycle_minobs` below). The default is `95`.

- `-recycle_ignore` prevents actual recycling of any media in the archive set, while allowing recycling processes to run normally. Used for testing.

- `-recycle_mailaddr` *`mail-address`* directs informational recycler messages to *`mail-address`*. Mail is not sent by default.

- -recycle_mingain limits selection of volumes for recycling to those which would increase their free space by at least the specified *percentage*. The default is 50.
- -recycle_vsncount limits the number of volumes that recycler schedules for rearchiving to *count*. Note that the actual number of volumes selected for recycling may also be dependant on the -recycle_dataquantity parameter. The parameter is ignored for disk media. The default is 1.
- -recycle_minobs sets the *percentage* of obsolete files in a disk-resident archive file that triggers rearchiving of the valid files and eventual deletion of the original tar file. The parameter is ignored for removable media (see -recycle_hwm above). The default is 50.
- -unarchage sets the reference time for computing the unarchive age to *time_ref*, where *time_ref* is either access for the file access time (the default) or modify for the modification time.
- -tapenonstop writes a single tape mark and an end-of-file (EOF) label at the end of the archive file without closing the removable media file. This speeds transfer of multiple archive files, but the tape cartridge cannot be unloaded until the entire archive set has been written to tape. By default, SAM-QFS software closes the tape file by writing two additional tape marks after the end-of-file label at the end of the archive file.
- -reserve *keyword* reserves a removable media volume for the exclusive use of a specified archive set. When a volume is first used to hold files from the archive set, the software assigns the volume a unique reserve name based on one or more specified keywords: fs, set, and/or one of the following: dir (directory), user, or group.

  fs includes the file system name in the reserve name: arset.1 -reserve fs.

  set includes the archive set name from the archive set assignment directive in the reserve name: allsets -reserve set.

  dir includes the first 31 characters of the directory path specified in the archive set assignment directive in the reserve name.

  user includes the user name associated with the archive file: arset.1 -reserve user.

  group includes the group name associated with the archive file: arset.1 -reserve group.

  Reserving volumes by set can be advantageous in some situations. But be aware that it is inherently less efficient than allowing the software to select the media. When volumes are reserved, the system must mount, unmount, and position cartridges more often, increasing overhead and reducing performance. Highly restrictive reservation schemes under-utilize available media and, in extreme cases, may cause archiving failures due to lack of available media.
- -priority *multiplier ranking* changes the archiving priority of files when used with the sort priority parameter listed above. The *ranking* is a real number in the range [(-3.400000000E+38)-3.400000000E+38] (-3.402823466x10^{38} to 3.402823466x10^{38}) and *multiplier* is the archive characteristic for which you are changing the relative *ranking*, selected from the following list: age, archive_immediate, archive_overflow, archive_loaded, copies, copy1, copy2, copy3, copy4, offline, queuewait, rearchive, reqrelease, size, stage_loaded, and stage_overflow.

  See the archiver and archiver.cmd man pages for more information on priorities.

Volume Serial Number (VSN) Pools Directives

The VSN pools section of the archiver.cmd file defines named collections of archival media volumes that can be specified as a unit in Volume Serial Number (VSN) Association Directives.

The section starts with a vsnpools directive and ends either with an endvsnpools directive or with the end of the archiver.cmd file. The syntax of a VSN pool definition is as follows:

```
vsn-pool-name media-type volume-specification
```

where:

- *vsn-pool-name* is the name that you assign to the pool.
- *media-type* is one of the two-character, SAM-QFS media type identifiers listed in Appendix A and in the mcf man page.
- *volume-specification* is a space-separated list of one or more regular expressions that match volume serial numbers. See the Solaris regcmp man page for details on regular expression syntax.

The example defines four VSN pools: users_pool, data_pool, proj_pool, and scratch_pool. A scratch pool is a set of volumes used when specific volumes in a VSN association are exhausted or when another VSN pool is exhausted. If one of the three specific pools is out of volumes, the archiver selects the scratch pool VSNs.

```
vsnpools
users_pool li ^VOL2[0-9][0-9]
data_pool li ^VOL3.*
scratch_pool li ^VOL4[0-9][0-9]
proj_pool li ^VOL[56].*
endvsnpools
```

Volume Serial Number (VSN) Association Directives

The VSN associations section of the archiver.cmd file assigns archival media volumes to archive sets. This section starts with a vsns directive and ends with an endvsns directive.

Volumes assignment directives take the following form:

```
archive-set-name.copy-number [media-type volume-specification] [-pool
vsn-pool-name]
```

where:.

- *archive-set-name* is the name that an archive set assignment directive assigned to the archive set that you are associating with the specified volumes.
- *copy-number* is the number that an archive copy directive assigned to the copy that you are associating with the specified volumes. It is an integer in the range [1-4].
- *media-type* is one of the two-character, SAM-QFS media type identifiers listed in Appendix A and in the mcf man page.
- *volume-specification* is a space-separated list of one or more regular expressions that match volume serial numbers. See the Solaris regcmp man page for details on regular expression syntax.
- -pool *vsn-pool-name* is a previously specified, named collection of archival media volumes that can be specified as a unit. See Volume Serial Number (VSN) Pools Directives.

The example illustrates various ways in which media can be associated with two lines of VSN specifications.

```
vsns
archiveset.1 lt VSN001 VSN002 VSN003 VSN004 VSN005
archiveset.2 lt VSN0[6-9] VSN10
archiveset.3 -pool data_pool
endvsns
```

Staging Directives

Staging is the process of copying file data from nearline or offline storage back to online storage.

The stager starts when the samd daemon runs. The stager has the following default behavior:

- The stager attempts to use all the drives in the library.
- The stage buffer size is determined by the media type, and the stage buffer is not locked.
- No log file is written.
- Up to 1000 stage requests can be active at any one time.

You can customize the stager's operations for your site by inserting directives into the /etc/opt/SUNWsamfs/stager.cmd file.

When an application requires an offline file, its archive copy is staged to disk cache unless the file was archived with the -n (*never stage*) option. To make the file available to an application immediately, the read operation tracks along directly behind the staging operation so that the access can begin before the entire file is staged.

Stage errors include media errors, unavailability of media, unavailability of an automated library, and others. If a stage error is returned, the SAM-QFS software attempts to find the next available copy of the file, if one exists and if there is a device available to read the archive copy's media.

The stager.cmd File

In the stager.cmd file, specify directives to override the default behaviors. You can configure the stager to stage files immediately, to never stage files, to staging partially, and to specify other staging actions. For example, specifying the never-stage attribute benefits applications that access small records from large files because the data is accessed directly from the archive media without staging the file online.

This section describes the stager directives. For additional information about stager directives, see the stager.cmd man page. If you are using the SAM-QFS File System Manager software, you can control staging from the File System Summary or File System Details page. You can browse the file system and see the status of individual files, use filters to view certain files, and select specific files to stage. You can select which copy to stage from or let the system choose the copy.

The example shows a stager.cmd file after all possible directives have been set.

```
drives=dog 1
bufsize=od 8 lock
logfile=/var/adm/stage.log
maxactive=500
```

`drives`: Specifying the Number of Drives for Staging

By default, the stager uses all available drives when staging files. If the stager keeps all the drives busy, it can interfere with the archiver's activities. The `drives` directive specifies the number of drives available to the stager. This directive has the following format:

```
drives=library count
```

where:

- *`library`* is the family set name of the library as it appears in the `mcf` file.
- *`count`* is the maximum number of drives used. By default, this is the number of drives configured in the `mcf` file for this library.

The example specifies that only one drive from the `dog` family set's library is used for staging files:

```
drives = dog 1
```

`bufsize`: Setting the Stage Buffer Size

By default, a file being staged is read into memory in a buffer before being restored from the archive media to disk cache. Use the `bufsize` directive to specify a buffer size and, optionally, to lock the buffer. These actions can improve performance. You can experiment with various *`buffer-size`* values. The directive has the following format:

```
bufsize= media-type buffer-size [lock]
```

where:

- *`media-type`* is one of the two-character, SAM-QFS media type identifiers listed in Appendix A and in the `mcf` man page.
- *`buffer-size`* is an integer in the range [`2-8192`]. This value is multiplied by the *`media-type`*`_blksize` value specified in the `defaults.conf` file. The higher the number specified for *`buffer-size`*, the more memory is used. The default is `16`.
- `lock` mandates the use of locked buffers for the duration of each staging operation. This avoids overhead associated with locking and unlocking the staging buffer for each I/O request and improves performance. The `lock` parameter can cause an out-of-memory condition if specified on systems that have limited memory available. By default, locked buffers are not mandated, and the file system retains control over the archiving buffer.

  The `lock` argument is effective only if direct I/O is enabled for the staged file. For more information about enabling direct I/O, see the `setfa`, `sam_setfa`, and `mount_samfs` man pages.

`logfile`: Specifying a Staging Log File

You can request that the SAM-QFS software collect file-staging event information and write it to a log file. By default, no log file is written. The `logfile` directive specifies a log file to which the stager can write logging information. The stager writes one or more lines to the log file for each file staged. This line includes information such as the name of the file, the date and time of the stage, and the volume serial number (VSN). The directive has the following format:

```
logfile=filename [event-list]
```

where *`filename`* is the full path name for the log file and *`event-list`* is a space-delimited list of the event types that are to be logged:

- `all` logs all staging events.
- `start` logs when staging begins for a file.
- `finish` (default) logs when staging ends for a file.
- `cancel` (default) logs when the operator cancels a stage.
- `error` (default) logs staging errors.

The following directive creates a stage log in the `/var/adm/` directory:

```
logfile=/var/adm/stage.log
```

Stager log entries take the following form:

```
status date time media-type volume position.offset inode filesize filename copy
user group requestor equipment-number validation
```

where:

- *`status`* is `S` for starting, `C` for canceled, `E` for error, `F` for finished.
- *`date`* is the date in the form *yyyy*/*mm*/*dd*, where *yyyy* is a four-digit number representing the year, *mm* is a two-digit number representing the month, and *dd* is a two-digit number representing the day of the month.
- *`time`* is the time in the form *hh*:*mm*:*ss* format, where *hh*, *mm*, and *ss* are a two-digit numbers representing the hour, minute, and seconds, respectively.
- *`media-type`* is one of the two-character, SAM-QFS media type identifiers listed in Appendix A and in the `mcf` man page.
- *`volume`* is the volume serial number (VSN) of the media that holds the file being staged.
- *`position.offset`* is a pair of hexadecimal numbers separated by a dot that represent position of the start of the archive (`tar`) file on the volume and the offset of the staged file relative to the start of the archive file.
- *`inode`* is the inode number and generation number of the staged file, separated by a dot.
- *`filesize`* is the size of the staged file.
- *`filename`* is the name of the staged file.
- *`copy`* is the archive copy number of the copy that contains the staged file.
- *`user`* is the user that owns the file.
- *`group`* is the group that owns the file.
- *`requestor`* is the group that requested the file.
- *`equipment-number`* is the equipment ordinal number defined in the `mcf` file for the drive from which the file was staged.
- *`validation`* indicates whether the staged file is being validated (`V`) or not validated (`-`).

The example shows part of a typical stager log:

```
S 2014/02/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1
root other root 0 -
F 2014/02/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1
root other root 0 -
S 2014/02/16 14:06:27 dk disk02 4.a68 1218.1387 519464 /sam1/testdir1/fileaq 1
root other root 0 -
```

```
S 2014/02/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1 root
other root 0 -
F 2014/02/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1 root
other root 0 -
```

`maxactive`: Specifying the Number of Stage Requests

The `maxactive` directive enables you to specify the number of stage requests that can be active at any one time. The directive has the following format:

```
maxactive=number
```

where *`number`* is an integer in the range [`1-500000`]. The default is `4000`.

The example specifies that no more than 500 stage requests can be in the queue simultaneously:

```
maxactive=500
```

`copysel`: Specifying the Copy Selection Order During Staging

The staging directive `copysel` sets the stager copy selection sequence per file system.

```
copysel=selection-order
```

where *`selection-order`* is a colon-delimited list of copy numbers in first-to-last order. The default selection order is `1:2:3:4`.

For more information, see the `stager.cmd` man page. The example shows a `stager.cmd` file that sets non-default copy-selection orders for file systems `samfs1` and `samfs2`:

```
logfile = /var/opt/SUNWsamfs/log/stager
drives = hp30 1
fs = samfs1
copysel = 4:3:2:1
fs = samfs2
copysel = 3:1:4:2
```

Preview Request Directives

When a SAM-QFS process requests a removable media volume that is not currently loaded into a drive, the request is added to the preview queue. Queued requests are satisfied in first-in-first-out (FIFO) order by default. But you can override the default behavior by editing the file `/etc/opt/SUNWsamfs/preview.cmd`. The SAM-QFS library-control daemon (`sam-amld`) reads these directives when it starts uses them until it stops. You cannot change queue priorities dynamically.

There are two types of directives:

- Global directives are placed at the top of the file and apply to all file systems.
- File-system directives take the form `fs=`*`directive`* and are specific to individual file systems

The following sections describe how to edit the `preview.cmd` file to control the preview queue:

- Global Directives
- Global and/or File System-Specific Directives
- Sample preview.cmd File

Global Directives

The following are purely global directives:

- vsn_priority: Adjusting Volume Priorities
- age_priority: Adjusting Priorities for Time Spent Waiting in the Queue

vsn_priority: Adjusting Volume Priorities

The vsn_priority directive increases the priority of volumes (VSNs) that are flagged as high-priority volumes by a specified value. The directive takes the following form:

```
vsn_priority=value
```

where *value* is a real number. The default is 1000.0.

You set the high priority flag on volumes using the command

```
chmed +p media-type.volume-serial-number
```

where *media-type* is one of the two-character, SAM-QFS media types listed in Appendix A and on the mcf man page and where *volume-serial-number* is the alphanumeric string that uniquely identifies the high-priority volume in the library. See the chmed man page for full information.

age_priority: Adjusting Priorities for Time Spent Waiting in the Queue

The age_priority directive changes the relative priority given to the amount of time that a request spends in the queue so that, for example, you can you keep older requests from being indefinitely superseded by newer, higher-priority, requests. The directive specifies a multiplier that changes the relative weighting of the time spent in the queue. It takes the following form:

```
age_priority=weighting-factor
```

where *weighting-factor* is a real number greater, less than, or equal to 1.0 and where:

- Values greater than 1.0 increase the weight given to time spent in the queue when calculating the aggregate priority.
- Values less than 1.0 reduce the weight given to time spent in the queue when calculating the total priority.
- Values equal to 1.0 do not change the relative weight given to time spent in the queue.

The default is 1.0.

Global and/or File System-Specific Directives

The following directives can be applied either globally or on a per-file system basis:

- hwm_priority: Adjusting Priorities When the Disk Cache is Nearly Full
- lwm_priority: Adjusting Priorities When the Disk Cache is Nearly Empty
- lhwm_priority: Adjusting Priorities as the Disk Cache Fills
- hlwm_priority: Adjusting Priorities as the Disk Cache Empties

`hwm_priority`: Adjusting Priorities When the Disk Cache is Nearly Full

The `hwm_priority` directive adjusts the relative weight given to archiving requests versus staging requests when file system utilization exceeds the high water mark (`hwm`), the point where the releaser process starts and begins reclaiming the disk space occupied by files that have copies on archival media. In this situation, increasing the relative weight given to archiving lets the releasing process free more space for staged archive copies and new files. The directive takes the following form:

```
hwm_priority=weighting-factor
```

where *`weighting-factor`* is a real number. The default is `0.0`.

`lwm_priority`: Adjusting Priorities When the Disk Cache is Nearly Empty

The `lwm_priority` directive adjusts the relative weight given to archiving requests versus staging requests when file system utilization drops below the low water mark (`lwm`), the point where the releaser process stops. In this situation, reducing the relative weight given to archiving and thereby raising the priority of staging requests places more files in the disk cache, reduces demand for media mounts, and increases file system performance. The directive takes the following form:

```
lwm_priority=weighting-factor
```

where *`weighting-factor`* is a real number. The default is `0.0`.

`lhwm_priority`: Adjusting Priorities as the Disk Cache Fills

The `hlwm_priority` directive adjusts the relative weight given to archiving requests versus staging requests when the disk cache is filling up, and cache utilization is between the low and high water marks (`lwm` and `hwm`). In this situation, increasing the relative weight given to archiving lets the releasing process free more space for staged archive copies and new files. The directive takes the following form:

```
lhwm_priority=weighting-factor
```

where *`weighting-factor`* is a real number. The default is `0.0`.

`hlwm_priority`: Adjusting Priorities as the Disk Cache Empties

The `hlwm_priority` directive adjusts the relative weight given to archiving requests versus staging requests when the disk cache is emptying, and cache utilization is between the high and low water marks (`hwm` and `lwm`). In this situation, reducing the relative weight given to archiving and thereby raising the priority of staging requests places more files in the disk cache, reduces demand for media mounts, and increases file system performance. The directive takes the following form:

```
hlwm_priority=weighting-factor
```

where *`weighting-factor`* is a real number. The default is `0.0`.

Sample `preview.cmd` File

The aggregate priority for any given media mount request is determined using the values set by all weighting factors, according to the following formula:

```
priority = vsn_priority + wm_priority + (age_priority * time-waiting-in-queue)
```

where *`wm_priority`* is the water mark priority currently in effect (`hwm_priority`, `lwm_priority`, `hlwm_priority`, or `lhwm_priority`) and *`time-waiting-in-queue`* is the number of seconds that the volume request has been queued. For a full explanation of priority calculation, see the PRIORITY CALCULATION section of the preview.cmd man page.

Under special conditions—when access to data is critically important or when removable media drives are in short supply—the directives in the preview.cmd file let you better match file-system activity to operational requirements and available resources. The integrity of stored data is unaffected by the settings in the preview.cmd file, so you can freely experiment until you find the proper balance between archiving and staging requests.

You may need to adjust the default priority calculation for either or both of the following reasons:

- to insure that staging requests are processed before archive requests, so that files are available when users and applications access them.
- to insure that archive requests gain top priority when a file system is about to fill up

The sample preview.cmd file below addresses the conditions highlighted above:

```
# Use default weighting value for vsn_priority:
vsn_priority=1000.0
age_priority = 1.0
# Insure that staging requests are processed before archive requests:
lwm_priority = -200.0
lhwm_priority = -200.0
hlwm_priority = -200.0
# Insure that archive requests gain top priority when a file system is about to
fill up:
hwm_priority = 500.0
```

Negative weighting values for lwm_priority, lhwm_priority, and hlwm_priority insure that stage requests have priority over archive requests whenever space is available in the disk cache, so that data is always accessible when requested. If several requests are sitting in the queue for 100 seconds and the file system is below the low water mark, then:

- An archiving mount request for a priority volume has the aggregate priority 1000+(-200)+(1x100)=900
- A staging mount request for a priority volume has the aggregate priority 1000+0+(1x100)=1100
- A staging mount request for a non-priority volume has the aggregate priority 0+0+(1x100)=100

But when the disk cache is near capacity, archiving requests need to take priority. If too few files are archived as the file system fills, there is no space available for staging archived files or ingesting new ones. If several requests are sitting in the queue for 100 seconds and the file system is above the high water mark, then:

- An archiving mount request for a priority volume has the aggregate priority 1000+500+(1x100)=1600
- A staging mount request for a priority volume has the aggregate priority 1000+0+(1x100)=1100
- A staging mount request for a non-priority volume has the aggregate priority 0+0+(1x100)=100

E

Product Accessibility Features

Users with low vision, blindness, color blindness, or other visual impairments can access the StorageTek Storage Archive Manager and StorageTek QFS Software (SAM-QFS) via the commandline interface. This text-based interface is compatible with screen readers, and all functions are controlled using a keyboard.

Glossary

This glossary focuses on terms specific to SAM-QFS software and file systems. For industry standard definitions, please refer to the dictionary maintained by the Storage Networking Industry Association at `http://www.snia.org/education/dictionary/`.

addressable storage

The storage space encompassing online, nearline, offsite, and offline storage that is user-referenced through a SAM-QFS file system.

admin set ID

A storage administrator-defined set of users and/or groups that share common characteristics. Admin sets are typically created to administer storage for projects that involve users from several groups and span multiple files and directories.

archival media

The media to which an archive file is written. Archival media includes both removable tape or magneto-optical cartridges and disk file systems configured for archiving.

archival storage

Data storage space created on archival media.

archive set

An archive set identifies a group of files to be archived, and the files share common criteria that pertain to the size, ownership, group, or directory location. Archive sets can be defined across any group of file systems.

archiver

The archive program that automatically controls the copying of files to removable cartridges.

associative staging

Staging a group of related files when any one member of the group is staged. When files inhabit the same directory and are frequently used together, file owners can associate them by setting the SAM-QFS associative-staging file attribute. Then if any files in the group are offline when one of them is accessed by an application, SAM-QFS stages the entire group from archival media to disk cache. This insures that all needed files re available at the same time.

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. See **volume serial number (VSN)**.

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.

backup

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

The size of the smallest addressable data unit on a block device, such as a hard disk or magnetic tape cartridge. On disk devices, this is equivalent to the *sector size*, which is typically 512 bytes.

cartridge

A container for data-storage media, such as magnetic tape or optical media. Also called a **volume**, *tape*, or *piece of media*. See **volume serial number (VSN)**.

catalog

A record of the removable media volumes in an automated library. There is one catalog for each automated library and, at a site, there is one historian for all automated libraries. Volumes are identified and tracked using a **volume serial number (VSN)**.

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.

data device

In a file system, a device or group of devices upon which file data is stored.

DAU

See **disk allocation unit (DAU)**.

device logging

A configurable feature that provides specific error information for the hardware devices that support a SAM-QFS file system.

device scanner

Software 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 SAM-QFS software.

direct I/O

An attribute used for large block-aligned sequential I/O. The `setfa` 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 (DAU)

In SAM-QFS file systems, the minimum amount of contiguous space that each I/O operation consumes, regardless of the amount of data written. The disk allocation unit thus determines minimum number of I/O operations needed when transferring a file of a given size. It should be a multiple of the **block size** of the disk device.

Disk allocation unit vary depending upon the SAM-QFS device type selected and user requirements. The `md` device type uses dual-allocation units: the DAU is `4` kilobytes for the first eight writes to a file and then a user-specified `16`, `32`, or `64` kilobytes for any subsequent writes, so that small files are written in suitably small blocks, while larger files are written in larger blocks. The `mr` and **striped group** device types use a DAU that is adjustable in increments of `8` within the range `[8-65528]` kilobytes. Files are thus written in large, uniform blocks that can closely approximate the size of the large, uniformly sized files.

disk buffer

In a SAM-Remote configuration, the buffer on the server system that is used for archiving data from the client to the server.

disk cache

The disk-resident portion of the file system software, 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 threshold

The maximum or minimum level of disk cache utilization, as defined by an administrator. 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. See also **striping**.

drive

A mechanism for transferring data to and from a removable media volume.

Ethernet

A packet-switched local-area network technology.

extent array

The array within a file's inode that defines the disk location of each data block assigned to the file.

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. See also **storage family set**.

FDDI

Fiber-distributed data interface, a standard for data transmission in a local area network that can extend in range up to 200 km (124 miles). The FDDI protocol is based on the token ring protocol.

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.

file system

A hierarchical collection of files and directories.

file-system-specific directives

Archiver and releaser directives that follow global directives in the `archiver.cmd` file, are specific to a particular file system, and begin with `fs =`. File-system-specific directives apply until the next **`fs =`** directive line or 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, a network protocol for transferring files between two hosts. For a more secure alternative, see **sftp**.

global directives

Archiver and releaser directives that apply to all file systems and that appear before the first **`fs=`** line.

grace period

In a **quota**, the amount of time that the file system allows the total size of files belonging to specified user, group, and/or **admin set ID**s to exceed the **soft limit** specified in the quota.

hard limit

In a **quota**, the absolute maximum quantity of storage resources that specified user, group, and/or **admin set ID**s can consume. See **soft limit**.

high-water mark

1. In an archiving file system, the percentage disk-cache utilization at which SAM-QFS file systems start the releaser process, deleting previously archived files from disk. A properly configured high-water mark insures that the file system always has enough space available for new and newly staged files. For more information, see the `sam-releaser` and `mount_samfs` man pages. Compare **low-water mark**.
2. In a removable media library that is part of an archiving file system, the percentage media-cache utilization that starts the recycler process. Recycling empties partially full volumes of current data so that they can replaced by new media or relabeled.

historian

The SAM-QFS historian is a catalog of volumes that have been exported from automated media libraries that are defined in the `/etc/opt/SUNWsamfs/mcf` file. By default, it is located on the SAM-QFS file-system host at `/var/opt/SUNWsamfs/catalog/historian`. For details, see the SAM-QFS `historian` man page.

hosts file

The hosts file contains a list of all of the hosts in a shared file system. If you are initializing a file system as a SAM-QFS shared file system, the hosts file, `/etc/opt/SUNWsamfs/hosts.fs-name`, must be created before the file system is created. The `sammkfs` command uses the hosts file when it creates the file system. You can use the `samsharefs` command to replace or update the contents of the hosts file at a later date.

indirect block

A disk block that contains a list of storage blocks. 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. Inodes are 512 bytes long. The inode file is a metadata file, which is separated from file data in the file system.

kernel

The program that provides basic operating system facilities. The UNIX kernel creates and manages processes, provides functions to access the file system, provides general security, and supplies communication facilities.

LAN

Local area network.

lease

A function that grants a client host permission to perform an operation on a file for a specified period of time. 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**.

local file system

A file system that is installed on one node of a Solaris Cluster system and is not made highly available to another node. Also, a file system that is installed on a server.

low-water mark

In an archiving file system, the percentage disk-cache utilization at which SAM-QFS file systems stops the releaser process and stops deleting previously archived files from disk. A properly configured low-water mark insures that the file system retains as many file in cache as possible, for best performance, while making space available for new and newly staged files. For more information, see the `sam-releaser` and `mount_samfs` man pages. Compare **high-water mark**.

LUN

Logical unit number.

mcf

Master Configuration File. The file that is read at initialization time that defines the relationships between the devices (the topology) in a file system environment.

media

Tape or optical disk cartridges.

media recycling

The process of recycling or reusing archive media with few active files.

metadata

Data about data. Metadata is the index information used 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 device

A device (for example, a solid-state disk or mirrored device) upon which file system metadata is stored. Having file data and metadata on separate devices 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

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 with the `mount` command. The single-writer host is specified with the `-o writer` option with the `mount` command. For more information, see the `mount_samfs` man page.

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 takes somewhat longer to access.

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 QFS file system interfaces with the vendor software using a SAM-QFS media changer daemon designed specifically for the automated library.

NFS

Network file system, a file system that provides transparent access to remote file systems on heterogeneous networks.

NIS

Network Information Service, a distributed network database containing key information about systems and users on the network. The NIS database is stored on the master server and all slave servers.

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, such as disk cache storage.

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. Preallocation can be specified only for a file that is size zero. For more information, see the `setfa` man page.

pseudo device

A software subsystem or driver with no associated hardware.

QFS

The SAM-QFS QFS Software product, a high-performance, high-capacity, UNIX file system that can be used on its own or as an archiving file system controlled by SAM-QFS Storage Archive Manager.

qfsdump

See **samfsdump (qfsdump)**.

qfsrestore

See **samfsrestore (qfsrestore)**.

quota

The amount of storage resources that specified user, group, or **admin set ID**s are allowed to consume. See **hard limit** and **soft limit**.

RAID

Redundant array of 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.

recovery point

A compressed file that stores a point-in-time backup copy of the metadata for a SAM-QFS file system.

In the event of a data loss—anything from accidental deletion of a user file to catastrophic loss of a whole file system—an administrator can recover to the last known-good state of the file or file system almost immediately by locating the last recovery point at which the file or file system remained intact. The administrator then restores the metadata recorded at that time and either stages the files indicated in the metadata to the disk cache from archival media or, preferably, lets the file system stage files on demand, as users and applications access them.

recycler

A SAM-QFS utility that reclaims space on cartridges that is occupied by expired archive copies.

regular expression

A string of characters in a standardized pattern-matching language that is designed for searching, selecting, and editing other character strings, such as file names and configuration files. For full details of the regular expression syntax used in SAM-QFS file-system operations, see the SAM-QFS Solaris `regex` and `regcmp` man pages.

release priority

The priority according to which a file in a file system is released after being archived. Release priority is calculated by multiplication of various weights of file properties and then summation of the results.

releaser

A 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 according to high and low thresholds.

remote procedure call

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

An **automated library** component 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. See also **disk striping** and **striping**.

RPC

Remote procedure call. The underlying data exchange mechanism used by NFS to implement custom network data servers.

SAM

See **SAM-QFS**.

SAM-Remote client

A SAM-QFS system with a client daemon that contains a number of pseudodevices, and can also have its own library devices. The client depends on a SAM-Remote server for archive media for one or more archive copies.

SAM-Remote server

Both a full-capacity SAM-QFS storage management server and a SAM-Remote server daemon that defines libraries to be shared among SAM-Remote clients.

SAM-QFS

1. A common abbreviation for the SAM-QFS product.
2. An adjective describing a **QFS** file system that is configured for archiving and managed by SAM-QFS Storage Archive Manager software.

samfsdump (qfsdump)

A program that creates a control structure dump and copies all the control structure information for a given group of files. It does not generally copy file data. With the `-U` option, the command also copies data files. If the Storage Archive Manager packages are not installed, the command is called `qfsdump`.

samfsrestore (qfsrestore)

A program that restores inode and directory information from a control structure dump. See also **samfsdump (qfsdump)**.

SAN

Storage Area Network.

SCSI

Small Computer System Interface, an electrical communication specification commonly used for peripheral devices such as disk and tape drives and automated libraries.

shared hosts file

When you create a shared file system, the system copies information from the hosts file to the shared hosts file on the metadata server. You update this information when you issue the **`samsharefs -u`** command

Small Computer System Interface

See **SCSI**.

soft limit

In a **quota**, the maximum amount of storage space that a specified user, group, and/or **admin set ID**s can fill for an indefinite period. Files can use more space than the soft limit allows, up to the hard limit, but only for a short **grace period** defined in the quota. See **hard limit**.

sftp

Secure File Transfer Protocol, a secure implementation of **ftp** based on **ssh**.

ssh

Secure Shell, an encrypted network protocol that allows secure, remote command-line login and command execution.

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.

stripe size

The number of disk allocation units (DAUs) to be allocated before writing proceeds to the next device of a stripe. If the `stripe=0` mount option is used, the file system uses round-robin access, not striped access.

striped group

A collection of devices within a file system that is defined in the `mcf` file as one or more `gXXX` devices. Striped groups are treated as one logical device and are always striped with a size equal to the disk allocation unit (DAU).

striping

A data access method in which files are simultaneously written to logical disks in an interlaced fashion. SAM-QFS file systems provide two types of striping: "hard striping," using stripe groups, and "soft striping," using the `stripe=x` mount parameter. Hard striping is enabled when a file system is set up, and requires the definition of stripe groups within the `mcf` file. Soft striping is enabled through the `stripe=x` mount parameter, and can be changed for the file system or for individual files. It is disabled by setting `stripe=0`. Hard and soft striping can both be used if a file system is composed of multiple stripe groups with the same number of elements. See also **round-robin**.

SUNW.qfs

A Solaris Cluster resource type that supports SAM-QFS shared file systems. The `SUNW.qfs` resource type defines failover resources for the shared file system's metadata server (MDS)

superblock

A data structure in the file system that defines the basic parameters of the file system. The superblock is written to all partitions in the storage family set and identifies the partition's membership in the set.

tar

Tape archive. A standard file and data recording format used 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).

timer

Quota software that keeps track of the period starting when a user reaches a soft limit and ending when the hard limit is imposed on the user.

transport

See **robot**.

`vfstab` file

The `vfstab` file contains mount options for the file system. Mount options specified on the command line override those specified in the `/etc/vfstab` file, but mount options specified in the `/etc/vfstab` file override those specified in the `samfs.cmd` file.

volume

1. On storage media, a single, accessible, logical storage area, usually addressed by a **volume serial number (VSN)** and/or volume label. Storage disks and magnetic tape cartridges can hold one or more volumes. For use, volumes are *mounted* on a file system at a specified **mount point**.
2. A magnetic tape **cartridge** that holds a single logical volume.
3. On a random-access disk device, a file system, directory or file that is configured and used as if it were a sequential-access, removable-media cartridge, such as a tape.

volume overflow

A capability that enables the system to span a single file over multiple **volume**s. Volume overflow is useful for sites using very large files that exceed the capacity of their individual cartridges.

volume serial number (VSN)

1. A serial number assigned to a tape or disk storage volume. A volume serial number can consist of up to six uppercase, alphanumeric characters, must start with a letter, and must identify the volume uniquely within a given context, such a tape library or partition. The volume serial number is written on the volume label.
2. Loosely, a specific storage **volume**, especially a removable media **cartridge**.

WORM

Write-Once-Read-Many. A storage classification for media that can be written only once but read many times.

Index