

IBM TotalStorage SAN File System
(based on IBM Storage Tank™ technology)



Maintenance and Problem Determination Guide

Version 2 Release 2

IBM TotalStorage SAN File System
(based on IBM Storage Tank™ technology)



Maintenance and Problem Determination Guide

Version 2 Release 2

Note

Before using this information and the product it supports, read the information in "Notices."

Third Edition (November 2004)

This edition applies to the IBM TotalStorage SAN File System and to all subsequent releases and modifications until otherwise indicated in new editions.

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

IBM welcomes your comments. A form for reader's comments is provided at the back of this publication. If the form has been removed, you may address your comments to:

International Business Machines Corporation
Design & Information Development
Department CGFA
PO Box 12195
Research Triangle Park, NC 27709-9990
U.S.A.

You can also submit comments by selecting Feedback at www.ibm.com/storage/support/.

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

© Copyright International Business Machines Corporation 2003, 2004. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this guide	v
Who should use this guide	v
Notices in this book	v
Publications	v
SAN File System publications	v
SAN File System related publications	vii
Web sites	vii
Chapter 1. Getting started	1
Chapter 2. Introduction to SAN File System	3
Components	3
Administrative server	5
Clients	6
Master console	19
Service alert	20
Remote access	21
SNMP	22
Chapter 3. Accessing SAN File System components	23
Accessing the master console remotely	23
Accessing the administrative server	25
Accessing an engine through a secure shell	26
Accessing the RSA II adapter	26
Accessing a client through a secure shell	27
Accessing a client through telnet	28
Accessing a client through a remote console utility	28
Chapter 4. Common errors	31
Chapter 5. Isolating problems with the SAN File System	33
Chapter 6. Diagnostic tools	37
Server diagnostic tools	37
Metadata server logs	37
Administrative server logs	40
Metadata server tracing	42
SAN File System dump capability	42
SAN File System server dump capability	42
Common errors	43
Client diagnostic tools	44
AIX client logging and tracing	44
AIX client dump capability	46
Linux client logging and tracing	47
Solaris client logging and tracing	48
Windows client logging and tracing	51
Windows client dump capability	52
One-button data collection utility	53
OBDC minimum C++ runtime requirements	54
Hardware Vital Product Data	54
Software Vital Product Data	55
Chapter 7. Isolating problems with the SAN File System	59
Chapter 8. Troubleshooting an administrative server	63
Troubleshooting user access to the console	64
Troubleshooting user access to the administrative command-line interface	66
Troubleshooting user access to a specific task or command	67
Resolution procedures	68
Verifying that the administrative agent is running	68
Verifying that the console is running	69
Replacing expired LDAP and CIMOM certificates	69
Configuring LDAP for SAN File System	69
Resetting an incorrect LDAP setting	71
Chapter 9. Troubleshooting the cluster	73
Troubleshooting a metadata server	74
Troubleshooting the local network	76
Resolution Procedures	79
Shutting down an engine from the RSA Web interface	79
Recovering from a lost RSA II adapter password	79
Taking a metadata server offline	79
Reassigning filesets to metadata servers	80
Changing a static fileset to a dynamic fileset	80
Bringing a metadata server online	80
Recovering metadata servers in the “Not Running” or “Not added” state	80
Adding a metadata server to the cluster	81
Repairing metadata	82
Metadata server does not start and no master disk is found in the standard log	82
Backing up system metadata	83
Chapter 10. Troubleshooting a SAN File System client	85
Troubleshooting client access to data	85
Troubleshooting client performance problems	89
Chapter 11. Troubleshooting the installation	91
Finding and correcting problems	91
Supplying power to metadata server engines	92
Chapter 12. Troubleshooting the master console	93
Resolution procedures	93
Performing a total software recovery	93
Recovering a hard disk drive	94

Replacing Fibre Channel cable and GBICs 94

Chapter 13. Managing disaster

recovery 97

Creating a recovery file 97
Creating recovery scripts 97
Deleting a recovery file 98
Listing recovery files 98
Restoring the master console 98
Restoring the engine hardware and operating system 98
Restoring SAN connectivity 99
Restoring SAN File System software. 100
Restoring SAN File System cluster configuration 100
Restoring SAN File System metadata 101
Restoring SAN File System clients 103
Restoring SAN File System user data 103

Chapter 14. Getting help, service, and information 105

Before you call for service 105
Getting help online 105
Getting help by telephone 106

Appendix A. Commands. 107

Administrative CLI overview 107
 Administrative command-line interface. 108
 Command modes 108
 Naming guidelines 109
 Standard format parameters 110
 Standard help parameters 112
 Standard listing parameters. 112
 Syntax diagram conventions 113

Common commands 115
Client commands 119
 Common client commands 119
 AIX-client commands. 123
 Linux-client commands 133
 Solaris-client commands. 141
 Windows-client command 148
Service commands and utilities 148
 disableConsoleTrace 150
 enableConsoleTrace 151
 mktruststore. 152
 obdc 153
 sanfstrace. 157
 startCimom 160
 startConsole 160
 stfsstat. 161
 stopCimom 164
 stopConsole 165
 tank extractbootrecord 165
 tank lscluster 166
 tank lsversion 168
 tank lsdisklabel. 168
 tank resetcluster 170
 tank resetversion 171
 tankpasswd 172

Appendix B. Accessibility 173

Appendix C. Notices 175

Trademarks 176

Index 179

About this guide

This topic informs support personnel about the information contained in the Maintenance and Problem Determination Guide.

This guide provides maintenance and problem determination information about the IBM® TotalStorage® SAN File System software.

Who should use this guide

This topic describes the audience for the Maintenance and Problem Determination Guide.

This guide is intended primarily for software service personnel who are familiar with the SAN File System.

The service commands and utilities listed in this document might cause a disruption in the SAN File System. Therefore, these commands should only be used by qualified software support personnel.

Notices in this book

This topic describes the notices used in the Maintenance and Problem Determination Guide.

The following notices are contained within this guide and convey these specific meanings:

Note: These notices provide important tips, guidance, or advice.

Attention: These notices indicate possible damage to programs, devices, or data. An attention notice appears before the instruction or situation in which damage could occur.

CAUTION:
These notices indicate situations that can be potentially hazardous to you. A caution notice appears before the description of a potentially hazardous procedure step or situation.

DANGER

<p>These notices indicate situations that can be potentially lethal or extremely hazardous to you. A danger notice appears before a description of a potentially lethal or extremely hazardous procedure step or situation.</p>

Publications

This topic describes the publications in the SAN File System library and in related libraries.

SAN File System publications

This topic describes the publications in the SAN File System library.

The following publications are available in the SAN File System library. They are provided in softcopy on the *IBM TotalStorage SAN File System Publications CD* and at www.ibm.com/storage/support. To use the CD, insert it in the CD-ROM drive. If the CD does not launch automatically, follow the instructions on the CD label.

Note: The softcopy versions of these publications are accessibility-enabled for the IBM Home Page Reader.

- *IBM TotalStorage SAN File System Release Notes*

This document provides any changes that were not available at the time the publications were produced. This document is available only from the technical support Web site: www.ibm.com/storage/support

- *IBM TotalStorage SAN File System Software License Information*

This publication provides multilingual information regarding the software license for IBM TotalStorage SAN File System Software.

- *IBM TotalStorage SAN File System Administrator's Guide and Reference, GA27-4317*

This publication introduces the concept of SAN File System, and provides instructions for configuring, managing, and monitoring the system using the SAN File System console and administrative command-line interfaces. This book also contains a commands reference for tasks that can be performed at the administrative command-line interface or the command window on the client machines.

- *IBM TotalStorage SAN File System Basic Configuration for a Quick Start, GX27-4058*

The document walks you through basic SAN File System configuration and specific tasks that exercise basic SAN File System functions. It assumes that the physical configuration and software setup have already been completed.

- *IBM TotalStorage SAN File System Maintenance and Problem Determination Guide, GA27-4318*

This publication provides instructions for adding and replacing hardware components, monitoring and troubleshooting the system, and resolving hardware and software problems.

Note: This document is intended only for trained support personnel.

- *IBM TotalStorage SAN File System Installation and Configuration Guide, GA27-4316*

This publication provides detailed procedures to set up and cable the hardware, install and upgrade the SAN File System software, perform the minimum required configuration, and migrate existing data.

- *IBM TotalStorage SAN File System Messages Reference, GC30-4076*

This publication contains message description and resolution information for errors that can occur in the SAN File System software.

- *IBM TotalStorage SAN File System Planning Guide, GA27-4344*

This publication provides detailed procedures to plan the installation and configuration of SAN File System.

- *IBM TotalStorage SAN File System System Management API Guide and Reference, GA27-4315*

This publication contains guide and reference information for using the CIM Proxy API, including common and SAN File System-specific information.

Note: This document contains information and procedures intended for only selected IBM Business Partners. Contact your IBM representative before using this publication.

SAN File System related publications

These publications are related to SAN File System.

- *IBM TotalStorage Subsystem Device Driver User's Guide, SC26-7637*

Web sites

This topic discusses any Web sites that offer additional, up-to-date information about SAN File System.

The following Web sites have additional information about SAN File System:

- www.ibm.com/storage/support/sanfs/
- www.ibm.com/storage/software/virtualization/sfs/

The following Web site has information about the languages that have International Components for UNICODE (ICU) converters:

oss.software.ibm.com/cgi-bin/icu/convexp/

Chapter 1. Getting started

This topic provides an overview of how to get started with troubleshooting SAN File System.

This topic is the starting point for all SAN File System problem determination actions. Using this topic, service representatives can quickly determine the appropriate chapter in this guide for their particular maintenance action. The information is organized as follows:

- Chapter 2, "Introduction to SAN File System," on page 3 provides an overview of SAN File System and related concepts.
- Chapter 3, "Accessing SAN File System components," on page 23 describes the various methods that are available for accessing SAN File System software components.
- Diagnostic tools provides information about the tools that you can use to diagnose problems with SAN File System components.
- Chapter 5, "Isolating problems with the SAN File System," on page 33 describes initial steps that you can take to begin isolating problems with the SAN File System, IP network, and SAN.
- Chapter 9, "Troubleshooting the cluster," on page 73 explains how to diagnose and resolve problems related to the SAN File System cluster, including the metadata servers and the IP network. In addition, it provides procedures that can assist you in resolving problems with the cluster.
- Chapter 8, "Troubleshooting an administrative server," on page 63 explains how to diagnose and resolve problems related to the administrative server. It includes information about problems related to administrative access to the SAN File System console, and the administrative command-line interface. In addition, it provides procedures that can assist you in resolving problems with the administrative server.
- Chapter 10, "Troubleshooting a SAN File System client," on page 85 explains how to diagnose and resolve problems related to client access to user data as well as client performance. It also provides procedures that can assist you in resolving problems with clients.
- Chapter 12, "Troubleshooting the master console," on page 93 explains how to diagnose and resolve problems related to the master console.
- Disaster recovery explains the disaster recovery procedure.
- Chapter 14, "Getting help, service, and information," on page 105 explains how to obtain help.
- The appendices provide the following additional information:
 - Accessibility features of the SAN File System console and help system
 - Notices

This topic also guides you through the process of determining the location of a failure, of preparing the SAN for maintenance, and of performing the repair.

You should perform the following steps:

1. Attach a keyboard and display to the SAN File System engine.

You can attach a keyboard and display to the specific engine or you can access the engine from the master console. See Chapter 3, "Accessing SAN File System components," on page 23 for the methods that you can use to access engine.

Note: Depending on the proximity of the master console to the engines in the SAN File System cluster, you might need to locally attach a keyboard, monitor, and mouse to the engine before attempting to service the engine.

2. Determine whether the problem is within the SAN File System subsystem.
To determine whether the problem is within SAN File System or elsewhere in the SAN, see Chapter 5, "Isolating problems with the SAN File System," on page 33 to make the determination.
If you determine that the problem is not within the SAN File System, follow the instructions in Chapter 5, "Isolating problems with the SAN File System," on page 33 to resolve the problem.
3. Determine whether the problem is within a SAN File System metadata server engine.
To determine whether the problem is within a metadata server engine, see Chapter 9, "Troubleshooting the cluster," on page 73.
 - If the problem is determined to be in the SAN File System master console, see Chapter 12, "Troubleshooting the master console," on page 93.
 - If the problem is determined to be neither the master console nor an engine, call your next level of support for assistance.
4. Determine which SAN File System engine is the cause of the problem.
To determine which metadata server engine is causing the problem, refer to your hardware documentation.
5. Prepare the engine for maintenance.
Before you begin working on an engine, it must be taken offline from the SAN File System cluster. Verify with that the engine has been taken offline before attempting to troubleshoot or replace any hardware components.
6. Perform the maintenance action.
See your server documentation for information about replacing hardware components in the engine. After the engine has been repaired, bring the engine back online within the SAN File System cluster.

Chapter 2. Introduction to SAN File System

IBM TotalStorage SAN File System is a multiplatform, scalable file system and storage management solution that works with a storage area network (SAN). It uses SAN technology, which allows an enterprise to connect large numbers of devices, such as client and server machines and mass storage subsystems, to a high-performance network.

IBM TotalStorage SAN File System is a multiplatform, scalable file system and storage management solution that works with a storage area network (SAN). It uses SAN technology, which allows an enterprise to connect large numbers of devices, such as client and server machines and mass storage subsystems, to a high-performance network.

On a SAN, storage is separated from the computers that use it. With SAN File System, heterogeneous clients can access shared data directly from large, high-performance, high-function storage systems, such as IBM TotalStorage Enterprise Storage Server[®]. The SAN File System is currently built on a fibre-channel network, and is designed to provide superior I/O performance for data sharing among heterogeneous computers. It also provides growth capability and simplified storage management.

SAN File System differs from conventional distributed file systems in that it uses a data-access model that requires *clients* to contact servers to obtain only the information they need to locate and access data on *storage devices*, not the data itself. SAN File System clients access data directly from storage devices using the high bandwidth provided by a fibre-channel network. Direct data access eliminates server bottlenecks and provides the performance necessary for data-intensive applications.

SAN File System presents a single, *global namespace* to clients where they can create and share data. Data consistency and integrity is maintained through SAN File System's management of distributed *locks* and the use of *leases*. SAN File System provides locks that enable file sharing among SAN File System clients, and when necessary, provides locks that allow clients to have exclusive access to files. A lease determines the maximum period of time that a server guarantees the locks it grants to clients. A client must contact the server before the lease period ends in order to retain its locks.

SAN File System also provides the benefits of automatic *file placement* through the use of *policies* and *rules*. Based on rules specified in a policy set, SAN File System automatically stores data on devices in *storage pools* that are specifically created to provide the capabilities and performance appropriate for how the data is accessed and used.

Components

Figure 1 on page 4 illustrates the major components of SAN File System.

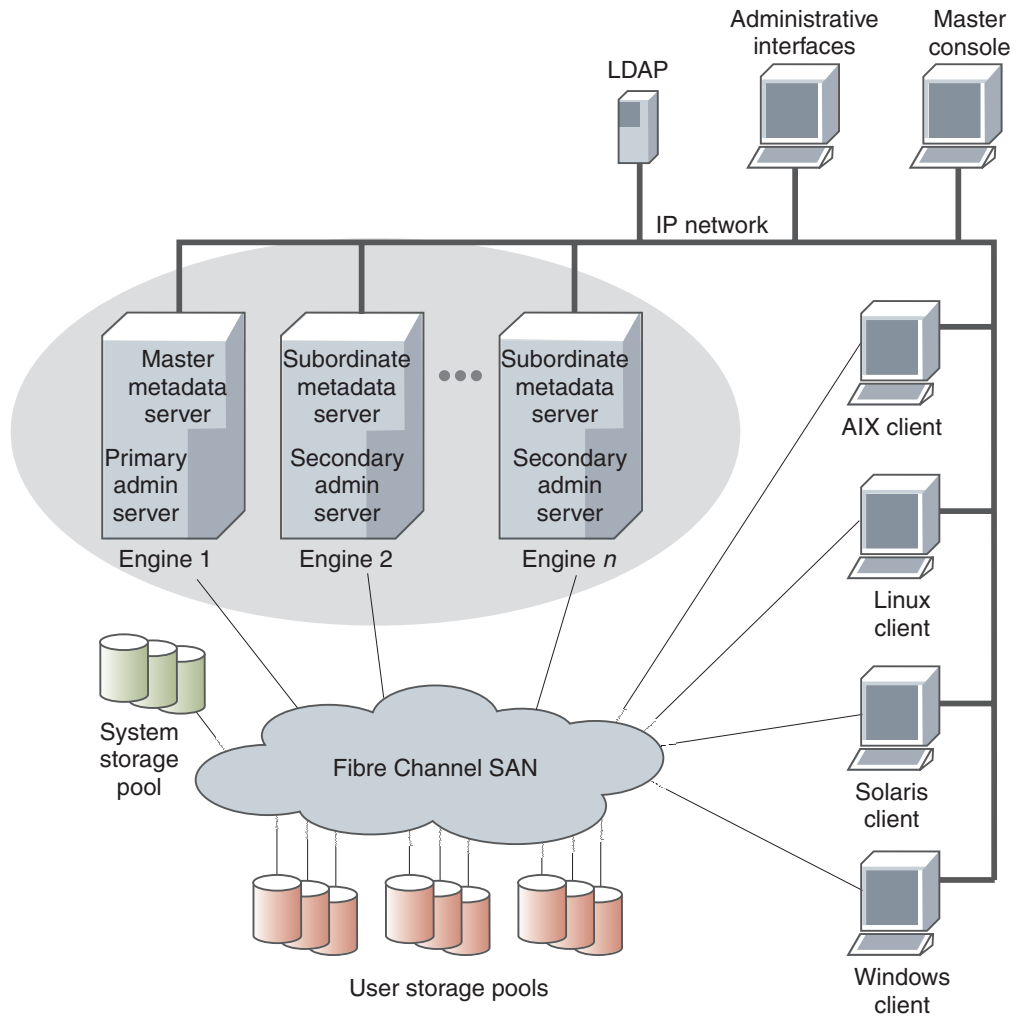


Figure 1. SAN File System components

The metadata servers and clients communicate over a private IP network and access data over a Fibre Channel storage attached network (SAN). SAN File System relies on networking hardware (including an IP network, SAN, network switches, and routers) that already exists in your environment.

The *metadata servers* run on separate physical machines (known as *engines*) and perform metadata, administrative, and storage-management services. The metadata servers are clustered for scalability and availability, and are referred to collectively as the *cluster*. In the cluster, there is one master metadata server and one or more subordinate metadata servers. Additional metadata servers can be added, as required, when the workload grows.

The metadata resides on private storage that is shared among all the metadata servers in the cluster. This storage is known as the *system storage pool*. A storage pool is a collection of SAN File System volumes in the SAN. The system storage pool contains the system metadata (such as system configuration and state information) and file metadata (such as file creation date and permissions). The actual file data is stored on the *user storage pools*, which may be shared among the clients.

The *administrative server* allows SAN File System to be remotely monitored and controlled through a Web-based user interface, called the *SAN File System console*. In addition, the administrative server processes requests issued from the administrative command-line interface, which can also be accessed remotely. The ability to access the SAN File System through these two types of interfaces allows you to administer SAN File System from almost any system with network connectivity. The machine that you use to access these interfaces is called the *administrative console*. The administrative server uses a *Lightweight Directory Access Protocol (LDAP) server* to look up authentication and authorization information about the administrative users. The primary administrative server runs on the same engine as the master metadata server. It receives all requests issued by administrators and also communicates with the administrative servers that run on each additional metadata server in the cluster to perform routine requests.

Computers that share data and have their storage centrally managed by SAN File System are known as *clients*. The SAN File System client software enables the clients to access a single, uniform global namespace through a virtual or installable file system. These clients can act as servers to a broader clientele, providing network file system (NFS) or common Internet file system (CIFS) access to the global namespace or hosting applications, such as database servers or Web-hosting services that use multiple servers.

The *master console* provides serviceability features, including the remote-support interface for remote access and service alert for call home capabilities. The master console is a required feature for SAN File System that can be shared with other IBM TotalStorage products, such as SAN Volume Controller.

Administrative server



Watch and learn

The *administrative server* processes all requests that are initiated from an administrative interface. Three major components of the administrative infrastructure include IBM Director Agent, a Web server, and the administrative agent.

IBM Director Agent enables remote administration and control of the storage engines.

The Web server interacts with the administrative agent and renders the Web pages that make up the SAN File System console. The console is a Web-based user interface, which can be accessed using a Web browser, that has network access to the engines that host the master metadata server in the cluster.

An administrative server interacts with a metadata server through an intermediary service, called the *administrative agent*. The administrative agent is based on the Common Information Model (CIM) standard to process all management requests from the SAN File System console and administrative command-line interface. When you issue a request, the administrative agent checks with the lightweight directory access protocol (LDAP) server to authenticate the user ID and password and to verify whether the user has the authority (is assigned the appropriate role) to issue a particular request. After authenticating the user, the administrative agent interacts with the metadata server to process the request. It also communicates with the operating system, the Remote Supervisory Adapter II (RSA-II) card, and

administrative agents on other engines when processing requests. This same system of authentication and interaction is also available to third-party CIM clients to manage SAN File System.

To ensure high availability, the administrative server resides on each storage engine. All requests that come from the SAN File System console are processed by the administrative server that runs on the same engine as the master metadata server. This server is known as the *primary administrative server*. However, requests that are initiated by the administrative command-line interface are processed by the administrative server that is running on the engine that you are logged in to. This can be the primary administrative server or a *secondary administrative server*, which is an administrative server that runs on an engine hosting a subordinate metadata server.

Clients



Watch and learn

SAN File System is based on a client-server paradigm. A SAN File System *client* is a computer system that accesses and creates data that is stored in the SAN File System global namespace. A client can act as a server to a broader clientele, providing Network File System (NFS) or Common Internet File System (CIFS) access to the global namespace for hosting applications (for example, database servers or Web-hosting services that use multiple servers).

The *SAN File System Protocol Specification* includes a description of the protocols that are used between a metadata server and clients running on application servers. It is available at www.ibm.com/storage/software/virtualization/sfs.

Clients access metadata (such as a file's location on a storage device) only through the metadata server, and then access data directly from storage devices attached to the SAN. This method of data access eliminates server bottlenecks and provides read and write performance that is comparable to that of file systems built on bus-attached, high-performance storage.

SAN File System supports clients that run several UNIX[®] and Windows[®] operating systems. You must install the SAN File System client software on each client machine. On UNIX clients, the client software presents the SAN File System as a local file system using the Virtual File System (VFS) interface. On Windows, the client software presents the SAN File System as a local file system using the Installable File System (IFS) interface. The VFS and IFS software provide clients with local access to the global namespace on the SAN. A VFS is a subsystem of a UNIX-based client's virtual file system layer, and an IFS is a subsystem of a Windows client's file system.

The SAN File System software directs all metadata operations to a metadata server and all data operations to storage devices attached to the SAN. The VFS or IFS software makes the metadata that is visible to a client's operating system, as well as any applications that are running on the client, look identical to metadata read from a native, locally-attached file system—that is, it emulates the local file system semantics. In this way, client applications do not need to change their access methods to use SAN File System.

When the global namespace is mounted on a UNIX-based client, it looks like a local file system. When the global namespace is mounted on a Windows client, it appears as another drive letter and looks like an NTFS file system. After the global

namespace is mounted, files can be shared between UNIX-based and Windows clients (permissions and suitable applications permitting).

Antivirus software

If more than one SAN File System client is running antivirus software that scans directories and files, shared files only need to be scanned by one SAN File System client. It is unnecessary to scan shared files more than once. When you run antivirus scans from more than one client, schedule the scans to run at different times, to allow better performance of each scan.

Tips:

- Consider using a single, designated client machine to perform all virus scans.
- On Windows 2003 clients, use antivirus software that does not use the mini-filter or filter manager model.

Authentication and authorization

SAN File System performs authorization checking for file-system operations on client machines based on the native operating system's user authentication mechanism. SAN File System does not restrict how authentication is performed, but it does assume that all UNIX clients share a common definition of users and groups; specifically, it assumes that any given identity using SAN File System has the same numerical value on all UNIX clients.

Client commands

There is a set of commands (issued from the operating-system command line) that you can use to set up, start and stop the client software and to migrate data. These commands are accessible from each client machine. The client commands are separate from the administrative command-line interface. The client commands allow users to perform operations on the client systems, and the administrative command-line interface allows administrators to administer all aspects of the metadata server.

Host-based clustering

This topic describes the cluster applications that you can run on SAN File System clients.

SAN File System works with clients that are in a clustered environment; however SAN File System is independent and not aware of any host-based clustering. SAN File System data volumes are owned and managed by SAN File System and must not be assigned as resources to the local operating system or cluster manager. Because cluster managers write on the volumes, configure the volumes as raw, unmanaged volumes to each member of the host-based cluster.

With SAN File System, you can use these clustering applications on the client machines:

- High availability cluster multi-processing (HACMP™) on AIX® platforms
- Sun Solaris clustering

Restriction: You cannot use Microsoft® clustering on Windows 2000 and Windows 2003 platforms at this time.

Data LUN configuration

You can configure the SAN so that all data LUNs are available to all clients (known as a *uniform configuration*) or so that only a subset of data LUNs are

available to some clients (known as a *nonuniform configuration*). In nonuniform configurations, a client must be able to access all LUNs in any storage pool that has been used or can be chosen by a fileset that is used by that client.

If a client tries to read or write data on a LUN that it cannot access, SAN File System returns an I/O error to that client. If the client performs a file-system operation that only involves metadata (such as changing a directory, or listing or creating files), SAN File System does not return an I/O error because the operation does not involve the data LUN.

The active policy determines which storage pool is selected when a new file is created. In a nonuniform configuration, the policy must ensure that a newly-created file is allocated to a storage pool that is accessible to the clients that need the file. When you change the active policy, the new policy must meet this consistency property.

Note: There is no automatic consistency check for a nonuniform configuration; however, when a client identifies itself to a metadata server, the metadata server inspects the client volume list to ensure that no incomplete storage pools are visible to the client. If an incomplete storage pool is visible, the metadata server logs an error in the metadata server log.

File sharing

When files are created and accessed from a Windows client, all the security features of Windows are available and enforced. When files are created and accessed from UNIX-based clients, all the security features of UNIX are available and enforced. When files created by a UNIX-based client are accessed by a Windows client, access is controlled using only the semantics and permissions of “other.” Similarly, when files created by a Windows-based client are accessed by a UNIX-based client, access is controlled using only the semantics and permissions of “everyone.”

Restriction: You can change the security settings of a system object (such as a file or directory) only from a client running the same platform as the client that originally created the object. For example, if you create a file on a UNIX client, you can change that file’s security settings only from a UNIX client, not from a Windows client. Instead, from a Windows client, you can create a new file by copying the original file.

File sharing in the SAN File System is classified as either homogenous or heterogeneous. File sharing is positioned primarily for homogenous environments. The ability to share files heterogeneously is recommended for read-only—that is, create files on one platform, and provide read-only access on the other platform. Therefore, set up filesets such that they have a *primary allegiance* to a single operating system. This means, for example, that certain filesets have files created in them only by Windows clients, and other filesets have files created in them only by UNIX clients.

File permissions: Newly created filesets are initially attached with a special dedicated user ID and group ID that lock out access to all clients. These are:

UNIX platforms

File permissions 000, userID/groupID 1000000/1000000

Windows platforms

Owner S-1-0-0

Note: For heterogeneous file sharing, the UNIX group ID is a fictitious number.

For clients to be able to access a fileset, a client must first take ownership of the fileset, by changing the fileset's owner to a valid user that can provide the required access. The take-ownership operation is only performed once for each fileset, and can only be done by a privileged client. A *privileged client* is a client on which root users in UNIX or users with administrator user in Windows are given those same privileges for the SAN File System global namespace. A root user logged in to a privileged client is granted full control over directories, files, and other file system objects created by clients in the SAN File System global namespace.

The concept of *root squashing* means that by default, when a root or Administrator user logs into a client that is not a privileged client, the user's privileges for the global namespace are reduced to that of "Other" in UNIX or "Everyone" in Windows. Therefore, in order to change the ownership and permissions on a fileset, one or more privileged clients must be created. You need at least one privileged Windows client if there are any Windows clients creating files, and at least one privileged UNIX client if there are any UNIX clients creating files.

In the current release of SAN File System, client files should be separated in filesets for each operating system — that is, a Windows client should create files only within filesets dedicated to Windows files, and a UNIX client should create files only within filesets dedicated to UNIX. This is referred to as the *primary allegiance* of a fileset — that is, either Windows or UNIX. There are several reasons for keeping the files separate:

- The presentation of access-control information for cross-platform files is approximate. Groups are not mapped; instead, they are derived from user identities, and access control on Windows is richer than on UNIX.
- Cross-platform access is limited to basic file operations. SAN File System does not support cross-platform attribute changes (for example, ownership and access control).
- Backup on either platform does not capture the access control attributes for foreign files.

The different client platforms can, however, share files in a common fileset if the permissions allow. Therefore, it is important to set up your access control lists (ACLs) on the clients and user maps in SAN File System to accomplish this goal.

To be able to take ownership and change permission on a new fileset, turn off root squashing for the client — that is, enable it as a privileged client to SAN File System.

Heterogeneous file sharing: In a heterogeneous environment (for example, both UNIX-based and Windows-based clients), SAN File System provides flexible yet secure sharing of files between UNIX and Windows platforms. It implements cross-platform access checking such that files created on Windows can be accessed by authorized users on UNIX, and files created on UNIX can be accessed by authorized users on Windows. SAN File System uses a *user map* to control cross-platform authorization. Each entry in the user map identifies a UNIX domain-qualified user and a Windows domain-qualified user that are to be treated as equivalent for the purpose of checking file-access permissions across platforms. The SAN File system cluster accesses your UNIX or Windows directory service to obtain user ID and group level information. Currently, you can use only a single NIS or LDAP directory service for UNIX clients and a single Active Directory controller for Windows clients. The mapping is defined between users in one UNIX domain and users in any Windows domain that is served by a single Active Directory instance.

The user map enables SAN File System to perform access checking and to provide a limited presentation of the security attributes for an object (file or directory) created on a different platform. Access to an object is determined in the context where the object was created. Presentation of ownership and permissions (for example, using the `ls -l` command on UNIX, or a Properties panel on Windows) must occur in the context where it is requested. Presentation might be approximate because permissions do not map perfectly between UNIX and Windows platforms. In some cases, presentation might not match the result of access checks, which are more precise.

When an object is accessed from the platform where it was created, no mapping is required, and the local platform rules apply. The following sections describe the behavior when user mappings have been established and the behavior in the absence of an appropriate mapping.

Access checking: The user map is consulted to translate the identity of the user requesting access into a user identity on the platform where the object was created. The UNIX or Windows directory service is consulted to determine the group membership for the translated user identity. The requested permissions are translated into permissions on the platform where the object was created. A single permission might be translated into several permissions, or several permissions might be translated to a single permission. Access to the object is then determined using the translated user and group identities and the translated required permissions, according to the access control on the object. Only file creation, deletion, and regular file access are permitted across platforms. Access control (ownership and permission) changes are only permitted on the platform where the object was created.

Group membership information for cross-platform accesses is retrieved from the UNIX or Windows directory when needed. Group identities are not translated from the requesting platform to the platform where the object was created. Group membership on the requesting side is not considered. Instead, group membership is determined by consulting the directory service for the platform where the object was created. All of the groups to which the translated user belongs are applied. There is no mechanism for operating with reduced group membership.

Group membership information is cached for some period of time. Information can be reused for many access checks without consulting the directory service for each one. Changes to group membership that are made at a directory service are not automatically reflected to SAN File System clients. If it is important that changes be reflected immediately, use the SAN File System administrative client to direct one or all clients to refresh their group information.

Permission translation: For cross-platform accesses, the permissions requested for a particular operation are determined by the requesting platform, and then translated to the platform where the object was created. This might result in broader permissions being required.

Permissions are translated as follows:

UNIX	Windows
read	READ_DATA(LIST_FOLDER)
execute	EXECUTE_DATA(TRVERSE_FOLDER)

UNIX	Windows
write	WRITE_DATA(CREATE_FILES), APPEND_DATA(CREATE_FOLDERS), and DELETE_SUBFOLDERS_AND_FILES (for directories)

Some permissions are not translated:

- UNIX clients do not restrict the ability to read attributes. The Windows READ_ATTRIBUTES, READ_EXTENDED_ATTRIBUTES, and SYNCHRONIZE permissions are not translated for access.
- UNIX clients do not request permission to delete a particular type of object (for example, a file). Instead, the request permission to delete any from the containing directory. The Windows per-object DELETE permission is not used.
- UNIX clients request permission to delete an object rather than a particular type of object (for example, a file). The Windows object-specific DELETE permission is not used.
- Cross-platform attribute changes are not permitted. Windows WRITE_ATTRIBUTES and WRITE_EXTENDED_ATTRIBUTES permissions do not apply.

Consider a UNIX client requesting write permission on a file or directory that was created on Windows. UNIX clients request write permission on a file when writing or appending, or on a directory when creating or removing a file or subdirectory. Because a write request could reflect any of these operations, any "write" request is translated to several Windows permissions: WRITE and APPEND for files; or, CREATE_FILES, CREATE_FOLDERS and DELETE_SUBFOLDERS_AND_FILES for directories. Access is granted only if the translated user has all of these permissions on the object, even though the operation actually being attempted would appear to match only one of them.

Presentation of security attributes: Security attributes (ownership and permission) for objects created on a platform are translated to allow presentation, but this translation is limited in several ways. Object ownership is translated according to the user map. Permissions are translated for the owner of the object and for the permission bits on UNIX or Everyone on Windows. The identity of the user requesting the translation does not affect presentation; all users use the same translation.

Permissions for the "owner" and "" are presented after translation such that they reflect the access check behavior. For example, a UNIX SAN File System client shows the "w" bit for the owner on a UNIX client if and only if the Windows file grants its owner both WRITE and APPEND permission.

There is no mapping of group information. The creating or owning group is not translated. Permissions for individual groups are not translated. The group is always shown as 999999 on UNIX for files created on Windows.

To satisfy the needs of some UNIX-based systems, the group EXECUTE permission is translated specially for presentation on UNIX. Whenever a user or group is granted EXECUTE permission for a file created on Windows, the group EXECUTE bit will be shown on UNIX.

Note: The difference in meaning between UNIX other permissions and Windows Everyone permissions is reflected during translation. The UNIX other permissions only apply when the requestor is not the owner and not a

member of the group associated with the object. On Windows, permissions granted to Everyone apply to the owner and group members; when viewed from UNIX, they are translated to both the owner and permission bits.

When UNIX object permissions are viewed on Windows:

- The entry for Everyone reflects the permission bits.
- One entry for the owner reflects the owner permission bits.
- An entry for the owner denies rights that reflect the permission bits that are not granted to the owner on UNIX.

Permissions that are not translated for access may still be translated for presentation. In particular, a UNIX object will appear to have READ_ATTRIBUTES and READ_EXTENDED_ATTRIBUTES permissions in its ACL for the owner and for Everyone, as UNIX does not restrict these operations.

Behavior when no user mapping exists: When no appropriate user mapping exists, cross-platform access and presentation is limited. UNIX access to Windows objects is granted only when Everyone has access. Windows access to UNIX objects is granted only when the other permission bits allow access. (In this case, there is no practical difference between Everyone and other.) Windows objects appear with a user and group ID of 999999 on UNIX. UNIX objects appear to belong to the null SID (S-1-0-0) on Windows. This behavior is consistent with earlier SAN File System releases that did not support heterogeneous user mapping.

This behavior also occurs when a user mapping exists but the metadata servers are not able to access the LDAP directory service.

Homogenous file sharing: In a homogenous environment (for example, either all UNIX-based or all Windows-based clients), SAN File System provides access and semantics that are customized for the operating system running on the client machines. When files are created and accessed from only Windows-based clients, all the security features of Windows are available and enforced. When files are created and accessed from only UNIX-based clients, all the security features of UNIX are available and enforced.

In homogenous file sharing, the permissions are all one type and are managed within the Windows or UNIX domain as appropriate. Therefore permissions propagate to all the sharing clients. Full support is provided for UNIX and Windows standard file access permissions; however, UNIX-extended ACLs are not currently supported.

In order to facilitate homogenous file sharing, you need UIDs and GIDs (UNIX) or SIDs (Windows) to be consistent in your operating system domains. For example, a UID number 2000 on one UNIX-based system must correspond to the same user with UID 2000 on every other UNIX-based system — and similarly for SIDs (security IDs) with Windows. To facilitate this, a common ID management system is required for each domain (Windows and UNIX), for example, Active Directory for Windows and Network Information Services (NIS) for UNIX, or LDAP, or manual synchronization of ID files. This ensures that permissions granted on one client map directly to other clients.

User maps: A user map is made up of a set of entries that define users in different domains that are to be considered equivalent. Each entry is defined as a tuple (*duid*, *sid*), where *duid* is the domain-qualified UNIX user ID, and *sid* is the domain-qualified Windows user ID. The tuple implies that the first identity is equivalent to the second identity. This relationship is symmetric, but not transitive.

For example, if user1 in domain1 is equivalent to user2 in domain2 and user3 in domain3, you must specify three user map entries.

You can specify mapped users in terms of user name or ID. IDs might simplify custom scripts that you have created to download mappings exported from third party mapping facilities.

The UNIX user name or ID must be qualified by the domain name of the UNIX directory service. Currently, you can configure SAN File System to access only a single UNIX directory service; therefore, the map entries are constrained to reference one common UNIX domain.

The Windows user name or ID must be qualified by a Windows domain. SAN File System permits any number of Windows domains provided that all of them are served by a single Active Directory instance. You can use trust relationships among Active Directory servers to allow a single Active Directory instance to serve information on multiple domains. The Windows domain and users are implicitly qualified by the single Active Directory instance that SAN File System is configured to access. SAN File System rejects Windows user names or IDs that are not known to the Active Directory instance.

The entries are constrained to a 1:1 relationship. In other words, a domain-qualified UNIX user name or ID cannot appear more than once on the UNIX side of the map, and a domain-qualified Windows user name or ID cannot appear more than once on the Windows side of the map.

When you add a new entry to the user map or refresh the user map, the master metadata server verifies that the user name or ID exists in the predefined UNIX and Windows directory services, translates the domain-qualified user names into IDs (if necessary), and stores the user IDs in the system pool.

Metadata, lock, and data caches

Caching allows a client to achieve low-latency access to both metadata and data. SAN File System supports caching of metadata, locks, and data.

A client caches metadata to perform multiple metadata reads locally. The metadata includes the mapping of logical file system data to physical addresses on storage devices that are attached to the SAN, name space, and file attributes.

A client caches locks to allow the client to grant multiple opens to a file locally without having to contact a metadata server for each operation that requires a lock. It also allows the client to perform reads and write on file data blocks when the appropriate locks and the logical physical address mappings are already cached .

A client caches data for files to eliminate I/O operations to storage devices attached to the SAN. A client performs all data caching in memory. If there is not enough space in the client's cache for all of the data in a file, SAN File System overwrites old data in the cache to make room for the new data.

Opportunistic locks

An *opportunistic lock* (or oplock) is a lock that is placed on a file in the global namespace. An oplock can be created and used by CIFS clients that use a SAN File System client, which acts as a CIFS server. The oplock is obtained by the CIFS client from the CIFS server and is bound to a SAN File System file lock. The CIFS lock and client caching of file data eliminates the need for the CIFS client to go to

the CIFS server across the network all the time. SAN File System enables the use of oplocks with appropriate synchronization of the CIFS client cache and SAN File System client cache.

A SAN File System client acting in the role of a CIFS server supports level 1, 2, batch, and filter locks.

Direct I/O

Some applications, such as database management systems, use their own cache management systems. For such applications, SAN File System provides a direct I/O mode, which allows these applications to bypass the data cache. In this mode, SAN File System performs direct writes to disk and does not cache data. This also allows distributed applications on different AIX or Linux™ client machines to write data to the same file at the same time. Using the direct I/O mode makes files act like raw devices. This gives database systems direct control over their I/O operations, while still providing the advantages of SAN File System, such as the FlashCopy® feature and file-level backup and restore processing. Applications need to be aware of, and configured for, direct I/O.

UNIX-based clients use existing operating-system interfaces to use direct I/O. That is, you must set the `O_DIRECT` flag to open a file in direct I/O mode.

Windows-based clients enforce full, native direct I/O, or *unbuffered I/O*, semantics. You must specify the `FILE_FLAG_NO_BUFFERING` flag to open or create a file in direct I/O mode. When using this flag, your application must meet the following requirements:

- The I/O buffers, offsets and transfer size must be integer multiples of the volume's sector size.
- Buffer addresses for read and write operations are not required to be sector aligned; however, the target offsets must be sector aligned.

You receive a return code of 87 (`ERROR_INVALID_PARAMETER`) if the requirements are not met.

Restrictions:

- You cannot use direct I/O processing on Linux clients.
- Applications that use direct I/O are responsible for managing concurrent writes to the same file.
- A process cannot use direct I/O on a file that is being used in cache mode by another process on the same client machine. Similarly, a process cannot use a file in cached mode that is being used in direct I/O mode by another process on the same client machine. In either case, you will receive an `EAGAIN` error.

However, a process can use direct I/O on a file that is being used in cache mode by a process on another client machine. Similarly, a process can use a file in cached mode that is being used in direct I/O mode by a process on another client machine.

Orphaned objects

An *orphaned object* is one that cannot be found in the directory tree. File system objects can become orphaned when the directories that referenced them are damaged or deleted. When an object becomes orphaned, it is moved to the *lost+found* directory. The *lost+found directory* is a special directory that is located at the root of each fileset directory, including the global fileset directory, and each

FlashCopy image. It is created automatically when the fileset or FlashCopy image is initialized. It is configured with owner/group 1000000/1000000 and permission 000. It is up to you to change the permissions, as appropriate, depending on whether the fileset is being used by Windows-based clients or UNIX-based clients.

You cannot rename or remove the lost+found directory. You can delete and move objects in the lost+found directory; however, you cannot create new objects in this directory.

Native client file-system security

For UNIX-based clients, SAN File System uses the POSIX definition of three sets of three file mode bits—one set each for user, group, and other. The bits in each group represent read, write, and execute or search permissions. It also uses the SETUID and SETGID bits, and the X/Open-specified restricted deletion mode (also known as “sticky”) bit used for directories. Therefore, the UNIX commands **ls** and **du** operate as usual when they are run against the global namespace of SAN File System.

If a file created by a UNIX-based client has the read and write bits set for user “other,” all UNIX and Windows users can read and write the file.

For Windows-based clients, SAN File System uses access control lists (ACLs), which are lists that define permissions for users and groups. An entry in an ACL is called an access control entry (ACE). If a Windows file creates an ACE for user “everyone,” all UNIX and Windows users can access that file.

Privileged clients

SAN File System includes a configurable list of privileged clients. A *privileged client* is a client on which root users in UNIX or users with administrator privileges in Windows are given those same privileges for the SAN File System global namespace. A root user that is logged in to a privileged UNIX-based client is granted full control over directories, files, and other file system objects that are created by UNIX-based clients. A user with administrator privileges who is logged in to a privileged Windows-based client is granted full control over the folders, files, and other file-system objects that are created by Windows-based clients.

If those same users log in to a client that is not a privileged client, their privileges for the global namespace are reduced to those of “everyone” for Windows users or “other” for UNIX users.

UNIX-based clients

A *UNIX-based client* is a SAN File System client that runs a UNIX operating system and has the SAN File System client code installed.

SAN File System supports clients running on these UNIX operating systems:

- AIX 5.1 (32-bit only)
- AIX 5.2 (32-bit and 64-bit)
- AIX 5.3 (32-bit and 64-bit)
- Red Hat Enterprise Linux Advanced Server 3.0 running either SMP or Hugesmem kernels in WS, ES or AS editions
- SUSE Linux Enterprise Server 8 (32-bit)
- Sun Solaris 9 (64-bit)

Restriction: SAN File System supports AIX client machines that have up to eight processors.

The SAN File System client code that is installed on a UNIX platform is called a Virtual File System (VFS). The VFS is a subsystem of the UNIX-based client's virtual file system layer. It directs all metadata operations to a metadata server and all data operations to storage devices that are attached to your SAN. The VFS makes the metadata that is visible to the client's operating system, as well as any applications that run on the client, look identical to metadata read from a native, locally-attached file system.

UNIX-based clients mount the global namespace on their systems. After the global namespace is mounted, you can use it just as you would any other file system to access data and to create, update, and delete files and directories. The following example shows an AIX mount point for SAN File System:

```
root@aix2:/# df
Filesystem 1024-blocks      Free %Used   Iused  %Iused  Mounted on
/dev                32768      23024   30%    1413     9%    /
/dev/hd1           950272      8096  100%   29103    13%   /usr
SANFS           16728064 16154624 4%     1     1%   /sanfs
```

UNIX-based clients use standard UNIX permission semantics (such as read, write, and execute bits, and owner and group IDs) that make the global namespace appear as if it were a local UNIX file system.

Named pipes: With SAN File System, you can create and use named pipes (or FIFO objects) in the global namespace. A *FIFO object* is a standard UNIX feature that is used to communicate and exchange data between processes. It has a directory name and is accessed by a path name. Its file size and a block size are always 0.

When a UNIX-based client creates a FIFO object, its file name becomes visible to all other clients, just like any other file. Users and applications on any UNIX-based client can perform standard file-system operations on a FIFO object; however the data-passing operation of the FIFO object is local to the client. In other words, data in a FIFO object is only readable to processes local to the same client that wrote the data. Data in FIFO objects is not passed between clients.

Although FIFO objects are visible to Windows clients (subject to file permissions), Windows-based clients cannot create, read from, or write to FIFO objects.

Limitations of UNIX-based clients: The following list describes the limitations of UNIX-based clients:

- UNIX-based clients cannot use user IDs or group IDs 999999 and 1000000 for real users or groups; these are reserved IDs that are used internally by SAN File System.

Tip: To avoid any conflicts with your current use of IDs, the reserved user IDs can be configured at installation time.

- You cannot use multibyte enablement for items referenced in a file-placement policy (such as storage pools, filesets, parts of file names).
- The **mkfs** command is not supported in UNIX.
- All AIX-based clients must reside in the same authentication domain for correct mapping of user and group IDs.
- This table lists the values for the maximum size of a file that SAN File System supports for each AIX platform:

AIX 5.1 (32-bit or 64-bit)	1 TB
----------------------------	------

AIX 5.2 and 5.3 (32-bit)	1 TB
AIX 5.2 and 5.3 (64-bit)	16 TB – 4KB

Windows-based clients

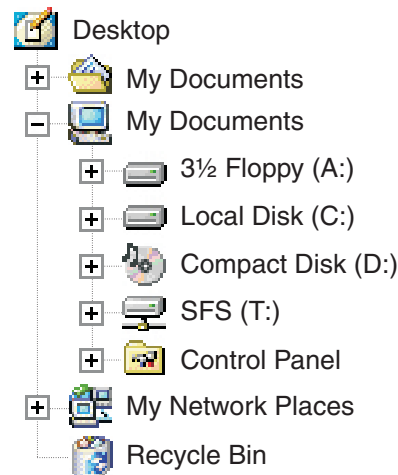
A *Windows-based client* is a client that runs a Windows operating system and has the SAN File System client code installed. In this release,

SAN File System supports clients that run on these Windows operating systems:

- Windows 2000 Advanced Server
- Windows 2000 Server
- Windows 2003, Standard Edition
- Windows 2003, Enterprise Edition

The SAN File System client code installed on a Windows-based client is an Installable File System (IFS). The IFS is a kernel-mode driver that extends the Windows I/O subsystem to support an additional file system. SAN File System client code directs all metadata operations to a metadata server and all data operations to storage devices attached to your storage area network (SAN). SAN File System client code makes the metadata that is visible to a client's operating system, as well as any applications that run on the client, look identical to metadata read from a native, locally attached file system.

Windows clients mount the global file system on their systems. After the global file system is mounted, users can use it just as they would any other file system to access data and to create, update, and delete files and directories. The following example shows the My Computer view from a Windows 2000 client. The T: drive (labeled SFS) is the attach point of SAN File System.



Windows-based clients use a subset of the Windows semantics. The allowed semantics are described to Windows as volume properties, which are visible, for example, as properties of the drive within Windows Explorer. The following volume properties are supported by SAN File System:

- NTFS-like access control lists (which requires all Windows-based clients to share a common Active Directory domain for users and groups)
- Long names and short names (eight-character names with three-character extensions)

- Unicode-based file names
- Case-sensitive file names

Case sensitivity: SAN File System natively is case-sensitive; however, Windows applications can choose to use case-sensitive or case-insensitive names. This means that case-sensitive applications, such as those making use of Windows support for POSIX interfaces, behave as expected. Native Win32 clients (such as Windows Explorer) get only case-aware semantics.

The case specified at file-creation time is preserved, but in general, file names are case-insensitive. For example, Windows Explorer allows you to create a file named "Hello.c," but an attempt to create "hello.c" in the same folder will fail because the file already exists. If a Windows-based client accesses a folder that contains two files that are created on a UNIX-based client with names that differ only in case, its inability to distinguish between the two files might lead to undesirable results. For this reason, UNIX-based clients should not create case-differentiated files in filesets that will be accessed by Windows-based clients.

Differences between SAN File System and NTFS: SAN File System differs from Microsoft Windows NT[®] File System (NTFS) in its degree of integration into the Windows administrative environment. The differences are as follows:

- Disk management within the Microsoft Management Console shows SAN File System disks as unallocated.
- SAN File System does not support reparse point or extended attributes.
- SAN File System does not support the use of the standard Windows write signature on its disks.
- The global namespace cannot be assigned a reserved drive letter.
- Disks used for the global namespace cannot sleep or hibernate.

SAN File System also differs from NTFS in its degree of integration into Windows Explorer and the desktop. The differences are as follows:

- Manual refreshes are required when updates to the SAN File System global namespace are initiated on the metadata server (such as attaching a new fileset).
- You cannot use the recycle bin.
- You cannot use distributed link tracing. This is a technique through which shell shortcuts and OLE links continue to work after the target file is renamed or moved. Distributed link tracking can help a user locate the link sources in case the link source is renamed or moved to another folder on the same or different volume on the same PC, or moved to a folder on any PC in the same domain.
- You cannot use NTFS sparse-file APIs or change journaling. This means that SAN File System does not provide efficient support for the indexing services accessible through the Windows "Search for files or folders" function. However, SAN File System does support implicitly sparse files.

File name considerations: File names created on UNIX-based clients using characters that are not valid for the Windows file systems (such as colons, slashes, back slashes, asterisks, question marks, double quotation marks, less than, greater than, and pipe) are transformed into valid short names. Windows-based applications can use the short name to gain access to files.

These rules are used to generate the short name, where the mapped UNIX file name is *prefix.suffix*:

- Each special character in the *prefix* and *suffix* is replaced with an underscore (_).

- If the mapped *prefix* is more than six characters in length, the first six characters are used and concatenated with a *~n*, where *n* is a unique number, to obtain a unique *prefix* on Windows with a maximum of eight characters.
- The first three characters of the *suffix* is used to generate the three-character extension on Windows.

For example,

UNIX	Windows
foo+bar.foobar	foo_ba~5.foo
foo bar.-bar	foo_ba~6._ba
foo bar.-bary	foo_ba~7._ba

Limitations of Windows-based clients: You cannot use the following features of NTFS when using SAN File System:

- File compression (on either individual files or all files within a folder)
- Extended attributes
- Encrypted files and directories
- Quotas (however, quotas are provided by SAN File System for filesets)
- Reparse points
- Defragmentation and error-checking tools
- Alternate data streams
- Assigning an access control list (ACL) for the entire drive
- Change journal for file activity
- Scan all files or directories owned by a particular SID (FSCTL_FIND_FILES_BY_SID)
- Security auditing or SACLs
- Windows sparse files

In addition, note these differences:

- Programs that open files using either the 64-bit file ID or the 128-bit object ID (the "FILE_OPEN_BY_FILE_ID" option) will fail. This applies to the NFS server bundled with Microsoft Services for UNIX.
- Symbolic links created on UNIX-based clients are handled specially by SAN File System on Windows-based clients; they appear as regular files with a size of 0, and their contents cannot be accessed or deleted.
- Any driver conforming to a newer Windows 2003 mini-filter driver or filter manager architecture might not work; however, all drivers that conform to the legacy filter driver architecture do.
- The SAN File System drive letter is not be visible until you perform a native port. File system functions have no impact.

Master console

The *master console* is a serviceability focal point for SAN File System and other IBM TotalStorage products. For SAN File System, the master console provides the key infrastructure for the remote access (through a virtual private network (VPN) and service alert features.

The master console is a software feature that includes the following software:

- Microsoft Windows 2000 Advanced Server edition
- IBM Director Server

- IBM Tivoli® Bonus Pack for SAN Management
- Adobe Acrobat
- The PuTTY openssh package
- Drivers for Qlogic QLA 2342 Fibre Channel Adapter
- VPN Connection Manager
- FASt Storage Manager

From the master console, you can access the following components:

- SAN File System console, through a Web browser.
- Administrative command-line interface, through a Secure Shell (SSH) session.
- Any of the storage engines in the cluster through an SSH session.
- The RSA II card for any of the storage engines running the SAN File System software through a Web browser. In addition, you can use the RSA II Web interface to establish a remote console to the engine, allowing you to view the engine desktop from the master console.
- Any of the SAN File System clients through an SSH session, a telnet session, or a remote display emulation package (such as VNC), depending on the configuration of the client.

Typically, you use the master console to access the SAN File System console or the administrative command-line interface as well as the storage engines. However, to perform service operations, you can attach a keyboard, monitor, and mouse to an engine if necessary.

Using the remote access feature, you can initiate a VPN connection to allow a support engineer to remotely access the master console. You can monitor that access and disconnect the session at any time.

Service alert

This topic provides an overview of the service alert.

For SAN File System, service alert notifies the system administrator of significant errors or failure conditions. SAN File System issues a Simple Network Management Protocol (SNMP) trap and sends that trap to the IBM Director Server running on the master console in response to a sever error condition. The IBM Director Server catches the trap and converts it into a Simple Mail Transfer Protocol (SMTP) e-mail message. The e-mail message is then sent to your SMTP mail server and then forwarded to the system administrator. The system administrator can then call the IBM Support Center.

This figure shows the service alert architecture for SAN File System:

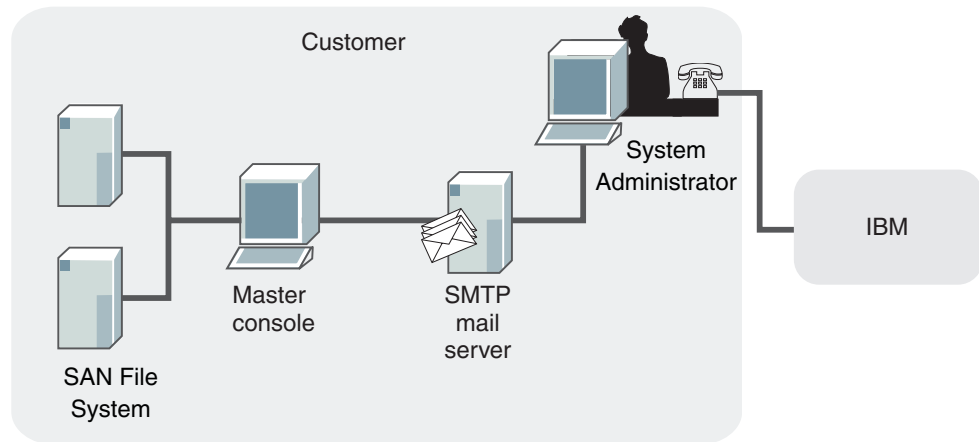


Figure 2. Service alert architecture for SAN File System

Remote access

This topic provides an overview of the Remote Access feature and explains the activities required to plan for it.

The master console provides *remote access* to the storage engines. Remote access allows IBM Support Center to diagnose problems with your system. Remote access support can help to greatly reduce service costs and shorten repair times, which in turn reduces the impact of any failures on your business.

Remote access gives IBM support personnel full access to the SAN File System or SAN Volume Controller through the master console, including querying and controlling the metadata servers and clients, and accessing metadata, log, dump, and configuration data. Remote access does not allow access without authentication. You must initiate a secure Virtual Private Network (VPN) connection, using VNC from the master console, to allow IBM support personnel to remotely access the master console. From the master console, the support personnel can establish a connection to the SAN File System metadata servers or SAN Volume Controller nodes. However, you can monitor that access and disconnect the session at any time.

In response to an error condition, you initiate a secure connection to the IBM VPN server using the VPN connection software on the master console called IBM Connection Manager. You must send the customer connection ID for the newly created connection to the IBM support personnel. The IBM support personnel initiates a secure connection to their VPN server, and then establishes a secure connection to the master console over the VPN tunnel using the customer connection ID and an account on the master console. Finally, IBM support personnel can access the SAN File System metadata server or SAN Volume Controller node through SSH.

The following figure shows the remote access architecture:

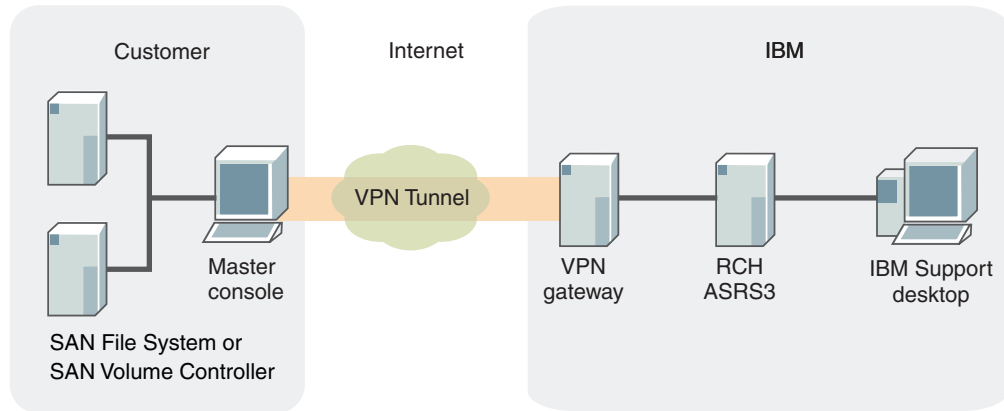


Figure 3. Remote access architecture

SNMP

The *Simple Network Management Protocol (SNMP)* is typically used to monitor network health, and performance and hardware, as well as to find and solve network problems. SNMP consists of two main components:

- SNMP agents, which are software components that reside on managed devices and collect management information (using Management Information Bases or MIBs). SNMP agents issue traps when SNMP events occur. These traps are sent through User Datagram Protocol (UDP) to an SNMP Manager.
- An SNMP manager, which is a network management application (for example, IBM Tivoli NetView[®]) that monitors and controls devices on which SNMP agents are running and can receive SNMP traps.

In SAN File System, each metadata server generates SNMP traps in response to certain events. SNMP traps are not issued from the operating system, hardware, or the administrative agent.

Tip: The RSA II cards can be set up to generate hardware traps as well.

You can configure which severity levels of events (informational, warning, error, or severe) should generate SNMP traps and you can define which SNMP managers in the SAN environment are to receive the traps. When an event occurs with a severity level that causes an SNMP trap, SAN File System sends the trap, and logs the event in the cluster log.

Note: SAN File System supports asynchronous monitoring through traps but does not support SNMP GETs or PUTs for active management. The SNMP Manager cannot manage SAN File System.

Not all events in SAN File System generate traps. Examples of events that might generate SNMP trap messages include:

- When a metadata server executes a change in state
- When a metadata server detects that another metadata server is not active
- When the size of a file set reaches a specified percentage of its capacity

Chapter 3. Accessing SAN File System components

This topic provides a summary of the various ways that you can access hardware and software components of the SAN File System.

There are two types of access that you can utilize to access the components of the SAN File System:

- Using the master console to access the SAN File System. From the master console, you can access the following components:
 - SAN File System console through a Web browser.
 - Any of the engines in the SAN File System cluster through a Secure Shell (SSH) session. From the SSH session, you can access the Administrative command-line interface.
 - The RSA II card for any of the engines in the SAN File System cluster through a Web browser. In addition, you can use the RSA II Web interface to establish a remote console to the engine, allowing you to view the engine desktop from the master console.
 - Any of the SAN File System clients through an SSH session, a telnet session, or a remote display emulation package (such as VNC), depending on the configuration of the client.
- Locally attaching a keyboard, monitor, and mouse (or KVM switch) to any of the hardware components of the SAN File System, including the engines in the cluster or any of the SAN File System clients.

The engines in the cluster are not shipped with a keyboard, monitor, or mouse. Typically, you will use the master console to access the engines as well as the SAN File System console or the Administrative command-line interface. However, if necessary, you can attach a keyboard, monitor, and mouse to an engine.

Note: Depending on the proximity of the master console to the engines in the SAN File System cluster, you may need to locally attach a customer-supplied keyboard, monitor, and mouse to the engine before attempting to service the engine.

Accessing the master console remotely

This topic describes how to access the master console remotely. You typically use this feature when you are working with a service representative, and the representative needs to access the master console from a remote location.

Before initiating a VPN connection with a service representative, the following requirements must be met:

- The master console must have a connection to the Internet.
- You must have a Windows user account for the service representative set up on the master console.
- You must have a remote display emulation package, such as Virtual Networking Computer (VNC) server, running on the master console if the service representative must access the SAN File System console or the RSA II Web interface remotely.
- You must provide a user ID and password for access.

- There must be a maintenance agreement between you and IBM, or the product must be under software warranty.

The master console is used to set up a VPN connection between you and a service representative. You initiate the connection and have the ability to monitor and control the connection.

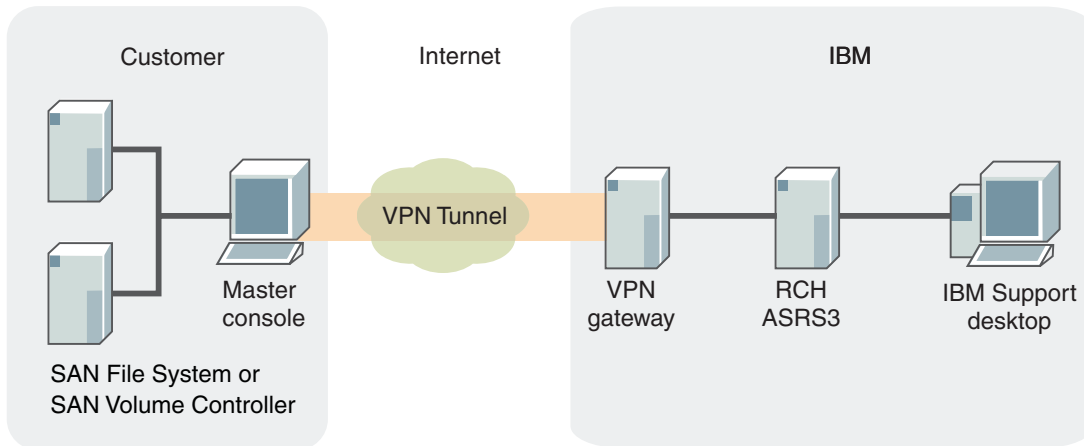


Figure 4. VPN connection between the customer and a service representative

1. Log into the master console. You can access the master console directly (using the keyboard, monitor, and mouse), or remotely through another computer on the same LAN.
2. Establish a secure connection from the master console through the VPN gateway to a designated VPN server within the IBM intranet. Establish the connection using the IBM connection manager and obtain a connection ID. The IBM connection manager icon is located on the master console desktop.
3. Provide the connection ID to the service representative. Each time you start a VPN session, a unique connection ID is created.
4. The service representative connects to the designated VPN server (this is the first of two connections), using either a Telnet client or a secure shell (SSH) client, such as PuTTY.
5. The service representative connects to an account on the master console. The service representative then establishes a second connection to the VPN server. The service representative can use the remote display emulator package connection to establish a remote console to the master console.

By accessing the master console remotely, a service representative can log on to the following devices or interfaces:

Each of the engines in the SAN File System cluster

The service representative can query and control the engines at the operating-system level by initiating an SSH session with the engine. This requires that a UNIX-based user account be set up on each of the engines in the cluster.

Administrative command-line interface

The service representative can query and control the SAN File System metadata servers, and access metadata, log, dump, and configuration data. This requires that a SAN File System administrative user account be set up for the service representative.

SAN File System clients

The service representative can query and control clients at the operating system level by initiating either an SSH session or a Telnet session with the client (if an SSH or Telnet application is installed and running on the client). This requires that an operating system user account be set up on each of the clients to which the service representative will need access.

SAN File System console and RSA II Web interface

These interfaces are available if a remote display emulation package is installed and running.

You can monitor all activity performed by the service representative. You can either run a remote desktop package from another machine to observe the master console desktop, view the master console SSH log file to see the results of all activity, or watch directly from the monitor on the master console. In addition, you can disconnect the VPN session at any time.

Accessing the administrative server

This topic describes how to access the administrative server through a Web browser from the master console (or from any computer that is on the same local area network (LAN) as the SAN File System cluster).

Before accessing the administrative server through a Web browser, the following requirements must be met:

- You must have a SAN File System administrative user account to sign on to the SAN File System console.
- If a service representative is accessing the master console remotely:
 - Make sure that you have already initiated a virtual private network (VPN) connection with the service representative.
 - The service representative must have established the VPN connection and used a remote display emulation package, such as VNC, to remotely view the desktop of the master console.

1. From the master console, open a Web browser and type the Web address for the primary administrative server

`https://primary_administrative_server:7979/sfs`

where *primary_administrative_server* is the host name or IP address of the engine hosting the primary administrative server and master metadata server.

If you enter a location in the Web browser for an administrative server other than the primary administrative server, the request is redirected to the primary administrative server. If the master administrative server is not available, the console for the secondary administrative server is displayed. However, some commands require that the primary administrative server be available; these commands do not complete successfully.

2. From the SAN File System console Welcome page, enter your SAN File System administrative user name and password to sign on.

From the SAN File System console, you can manage and view information about engines, metadata servers, and clients.

Accessing an engine through a secure shell

This topic describes how to access an engine using a secure shell (SSH) from the master console.

Before accessing an engine using SSH, the following requirements must be met:

- You must have a SAN File System administrative user account to sign on to the SAN File System console.
- You must have a UNIX-based operating system account on the engine to be accessed for use in signing on to the SSH session.
- If the service representative is accessing the master console remotely, you must have previously initiated a VPN connection with the service representative.

To access the engine using SSH, perform the following steps:

1. From the master console, use one of these methods to access the engine:
 - a. Open a shell prompt and type **putty.exe -ssh engine_IP_address**, where *engine_IP_address* is the IP address of the engine to be accessed.

Note: If you used SSH to establish a remote session with the master console, type this command from that session to establish an SSH session between the master console and the engine.
 - b. Click **Start** → **Programs** → **PuTTY** → **PuTTY**.
 - 1) Type the IP address of the engine to be accessed.
 - 2) Select SSH as the protocol.
 - 3) Click **Open**.
2. Log in using your UNIX-based user ID and password.

After connecting to the engine, you can perform these activities:

- You can access the SAN File System administrative command-line interface (CLI) to run SAN File System commands. Use these commands to manage engines, metadata servers, and administrative servers.
- You can access operating-system commands to enable or disable tracing, obtain dumps, and stop or start applications.

Accessing the RSA II adapter

This topic describes how to access the RSA adapter that is installed on each engine in the cluster.

Before you can access the RSA adapter for an engine, the following requirements must be met:

- You must have an RSA user and password to access the RSA II Web interface on the engine.
- If the service representative is accessing the master console remotely, you must have previously initiated a VPN connection with the service representative.

To access the RSA adapter for an engine, perform the following steps:

1. Open a Web browser from the remote master console and enter the Web address for an RSA adapter, as follows:
`http://RSA_II_web_address/`
2. Log on to the RSA II interface using your RSA user name and password.

From the left navigation pane, you can choose to perform tasks such as:

- Stop and restart the engine
- View vital product data for the engine
- Access the BIOS and firmware for the engine
- Update the firmware for the RSA card
- Access the RSA adapters of other engines in the SAN File System cluster

Note:

- If, when you view the System Health Summary page, all components are listed as unavailable, make sure the PCI-riser card assembly is firmly seated within the engine.
- You can start a remote video console that can be used to redirect the engine console to the master console. In addition, you can use the remote control to assign the CD-ROM or diskette drive from the master console to be used by the engine.
- To use the remote control functionality of the RSA adapter, you must be signed on with an ID defined in the RSA II card that has read/write access to the RSA II adapter.

For more information about using the RSA card, see the *Remote Supervisor Adapter II User's Guide*, which is available from this Web site (search for Remote Supervisor Adapter II from the Search Technical Support link):

www.ibm.com/storage/support/

Accessing a client through a secure shell

This topic describes how to remotely access a client using a secure shell (SSH) from the master console.

Before you can access a client using an SSH, the following requirements must be met:

- You must have an operating system account to log on to the client to be accessed.
- The PuTTY software package, or a similar SSH software package, must be installed and running on the client.
- If a service representative is accessing the master console remotely, you must have previously initiated a virtual private network (VPN) connection with the service representative.

1. From the master console, use one of these methods to access the client:
 - a. Open a shell prompt and type **putty.exe -ssh client_IP_address**, where *client_IP_address* is the IP address of the client to be accessed.

Note: If you used SSH to establish a remote session with the master console, type this command from that session to establish an SSH session between the master console and the client.

- b. Click **Start** → **Programs** → **PuTTY** → **PuTTY**.
 - 1) Type the IP address of the client to be accessed.
 - 2) Select SSH as the protocol.
 - 3) Click **Open**.
2. Log in using your operating system user ID and password.

After the connection is established, you can perform these activities:

- If you are accessing a UNIX client, you can run SAN File System commands to stop and start the client as well as list client status.
- If you are accessing a UNIX or Windows client, you can use the `migratedata` command.
- You can access operating-system commands to enable or disable tracing, obtain dumps, and stop or start applications.

Accessing a client through telnet

This topic describes how to access a client using telnet from the master console.

Before you can access a client using telnet, the following requirements must be met:

- You must have an operating system account for the service representative.
 - If the service representative is accessing the master console remotely, you must have previously initiated a virtual private network (VPN) connection with the service representative.
1. From the master console, open a shell prompt and type `telnet host`, where *host* is the IP address of the client to be accessed.
 2. Log in using your operating system user ID and password.

After the connection is established, you can perform these activities:

- If you are accessing a UNIX client, you can run SAN File System commands to stop and start the client as well as list client status.
- If you are accessing a UNIX or Windows client, you can also use the `migratedata` command.
- You can access operating-system commands to enable or disable tracing, obtain dumps, and stop or start applications.

Accessing a client through a remote console utility

This topic describes how to access a client using a remote console utility, such as Virtual Network Computing (VNC), from the master console.

Before you can access a client using a remote console utility, such as VNC, the following requirements must be met:

- You must have an operating system account on the client.
 - A VNC client must be installed and running on the master console.
 - A VNC server must be installed and running on the client. In addition, a session password must exist.
 - If the service representative is accessing the master console remotely, you must have previously initiated a VPN connection with the service representative.
 - The service representative must have established the VPN connection and used a remote display emulation package, such as VNC, to remotely view the desktop of the master console.
1. Double-click the VNC viewer icon on the master console.
 2. Enter the IP address or host name of the client to be accessed and click **OK**.
 3. Enter the session password and click **OK**.

After the connection is established, you see the remote desktop of the client. From the VNC connection, you can perform tasks and run commands as if you were physically at the client machine.

Chapter 4. Common errors

This topic lists some common errors that you might encounter when using SAN File System. While these errors are generally easy fixes, it is sometimes difficult to determine the cause.

Errors when running `setupsfsclient`

After running the SAN File System client setup utility, the utility fails with the following error messages:

```
HSTDR0029I The kernel extension was successfully loaded from file
/usr/tank/client/bin/stfs kernel module ID (kmid) = 63cc400.
HSTDR0030I File system driver is initialized and ready to handle file-system
type 20. /usr/tank/client/bin/stfsclient:
HSTCL0011E A -create parameter uses a maximum of two parameter values:
client name and server name. You specified 17.
HSTCS0008E Could not create SAN File System client
HSTDR0033E SAN File System driver shut down successfully.
HSTDR0035I The kernel extension 63cc400 was unloaded successfully.
```

This group of errors is displayed because of a syntax error in one of the configuration fields that you completed when you ran the `setupsfsclient` command.

An example of this can be seen in the following entry to the device candidate list:

```
pat=/dev/rvpath *
```

In this field, there should not be a space between `rvpath` and the asterisk (*). In this case, the `setupsfsclient` will fail and the previously mentioned errors will result.

Error: "Drive not found" when starting SAN File System

One startup test when starting SAN File System requires the system to discover the drives. If this test does not complete before a test that requires a specific drive to be available begins, the server will enter a 5 second pause and attempt the connection again.

The server should connect to the drive before its next attempt, but a Drive not found error might occur from the first attempt.

Chapter 5. Isolating problems with the SAN File System

This topic explains how to determine and isolate problems with the SAN File System, SAN, and LAN.

In most cases, you can use the logs provided by the SAN File System to begin isolating problems.

- For problems that seem to be related to clients, use the logs that are available with the client operating system (such as the system log on AIX clients and the Event Log on Windows clients) to determine the cause of the problem. In addition, you can use the information provided in Chapter 10, "Troubleshooting a SAN File System client," on page 85.
- For problems that seem to be related to administrative user access, use the security log and the administrative log to determine the cause of the problem. If you access these logs through the master metadata server, you see a consolidated view of the logs from each of the metadata servers in the cluster. If you access these logs through a subordinate metadata server, you see the logs for that particular metadata server.

In addition, you can use the information provided in Chapter 8, "Troubleshooting an administrative server," on page 63.

- For problems that seem to be related to the cluster, metadata servers, or metadata, use the server log to determine the cause of the problem.
In addition, you can use the information provided in Chapter 9, "Troubleshooting the cluster," on page 73. If you access this log through the master metadata server, you see a consolidated view of the logs from each metadata server in the cluster (called the cluster log). If you access these logs through a subordinate metadata server, you see the logs for that particular metadata server only.
- For problems that seem to be related to the installation, refer to Chapter 11, "Troubleshooting the installation," on page 91. The topics provide commands and suggestions that you can use to diagnose installation problems.
- For problems that seem to be related to the RSA (Remote Supervisor Adapter) II, refer to Troubleshooting the RSA II adapter. This topic provides information about problems you might encounter with the RSA II during the installation and upgrade processes.
- For problems that seem to be related to the engines in the SAN File System, refer to the server documentation to determine the cause of the problem.

If you are not sure if the problem is related to the SAN File System, SAN, or LAN, use the information in this section to begin isolating the problem.

Note: Certain events, such as restarting the operating system or disconnecting cables, can cause the SAN File System to lose connectivity to logical unit numbers (LUNs). If the logs indicate I/O failures for a client or metadata server, verify the following:

- Configured system LUNs should be visible only from metadata servers. Each metadata server must see all configured system LUNs. Configured user LUNs should be visible only from SAN File System clients. Not all user LUNs need to be seen by all clients. A client can see a subset of the total user LUNs, but it must see all LUNs configured for any pool to

which it has access. If the client cannot see all LUNs associated with a particular pool, a warning prints in the server log.

You can configure clients to only see a select group of user LUNs that it needs to access, or configure them to view all of the user LUNs.

- A SAN fabric switch has not lost the zoning configuration. If the operating system on the switch is restarted, it is possible for the fabric to lose the zoning configuration, which prevents metadata servers and SAN File System clients from reaching LUNs.

You might need to force SAN File System to remap LUNs if you lose connectivity. Refer to your SAN administrator for help with LUN rediscovery options specific to your operating environment.

Identifying SAN problems

Perform the following steps to determine if the problem is related to the SAN itself:

1. Determine whether the SAN configuration was recently changed, such as changing the Fibre Channel cable connections or switch zoning. If so, verify that the changes were correct and, if necessary, reverse those changes.
2. Verify that all switches and storage devices used by SAN File System are powered on and are not reporting hardware failures. If you find problems, resolve them before proceeding.
3. Verify that the Fibre Channel cables that connect the metadata servers to the switches are securely connected.
4. IBM Subsystem Device Driver (SDD) version 1.5.1 is provided with SAN File System and provides support for multipath environments. You can use the datapath query commands to view statistics, as well as information about paths and adapters. For information about using the datapath query commands, refer to the *Subsystem Device Driver User's Guide* provided on the SAN File System documentation CD-ROM.
5. If you are running a SAN Management tool that you are familiar with, use it to view the SAN topology and isolate the failing component. If you are not using a SAN Management tool, you can start IBM Tivoli SAN Manager on the master console and use it to view the SAN Topology and isolate the failure. For information about SAN problem determination with IBM Tivoli SAN Manager, contact the Tivoli Storage Area Network (SAN) support center.

Identifying IP networking problems

Perform the following steps to determine whether the problem is related to the IP network itself:

1. Verify that all switches used by the SAN File System are powered on and are not reporting any hardware failures. If problems are found, resolve them before proceeding further.
2. Verify that the Ethernet cables that connect the metadata servers to the switches are securely connected.
3. Verify that the metadata servers, clients, and storage devices are on the same network and subnet.

Identifying storage problems

Perform the following steps to determine whether the problem is related to the storage devices:

1. Determine whether any other hosts that may be attached to the storage devices are having the same problems.
2. Determine whether a single metadata server or client is having trouble accessing the storage device or all metadata servers and clients are experiencing I/O errors.
3. Refer to the documentation for the storage devices for more information about isolating problems with those devices.

Chapter 6. Diagnostic tools

This topic provides an overview of the tools available to diagnose problems with the SAN File System and its components.

Use these tools to assist you when troubleshooting the SAN File System.

Server diagnostic tools

This topic describes the diagnostic tools that are available to isolate and resolve problems with the metadata server or the administrative server.

You can use the following tools to diagnose problems that you are having with either the metadata server or the administrative server:

- **Logs.** The SAN File System provides several logs that you can use to view information about metadata servers and administrative servers.
- **Tracing.** The SAN File System provides the ability to trace both the metadata server and the administrative server to diagnose problems.
- **Dumps.** You can use dump capabilities provided by the SAN File System, as well as the dump capabilities of the UNIX-based operating systems, to diagnose problems.

Metadata server logs

This topic describes where the SAN File System metadata server logs are stored.

The following logs for the metadata server are stored on the engine hosting that server.

Table 1. SAN File System metadata server message log files

Log	File name	Location	Maximum file size
Audit log	log.audit	/usr/tank/server/log	250 MB
Dump log	log.dmp	/usr/tank/server/log	–
Server log	log.std	/usr/tank/server/log	250 MB
Trace log	log.trace	/usr/tank/server/log	250 MB
RSA command log	log.stopengine	/usr/tank/server/log	

Note:

1. Although the Audit Log (log.audit), Trace Log (log.trace), and Server Log (log.std) have a maximum file size of 250 MB, SAN File System actually stores 500 MB of data for each of these logs. When any of these logs reaches its maximum size, it is renamed to include the extension, **.old**. If a file by that name already exists, SAN File System overwrites the existing file. Then the log is cleared so that it can start accepting new messages again.
2. The log.dmp file starts over for either of these occurrences if the metadata server has restarted (for example, it restarted due to a server crash):

- The start of each day
- The file reaches a size of 1 MB.

When you display these logs from the master metadata server using either the administrative command-line interface or the SAN File System console, you see a consolidated view of all the logs from each engine in the cluster. The consolidated view of the server message log is called the Cluster log.

Note:

1. You can also display the Event Log. This log is actually a subset of the messages stored in the Cluster Log. It contains only messages with a message type of **event**.

Service-only logs

The Dump Log (log.dmp) and the Trace Log (log.trace) are not used for normal problem determination. They will not be discussed in detail.

Audit log

This topic describes the contents of the audit log.

The administrative audit log contains all administrative actions issued to the SAN File System. It contains a record of every command issued by a SAN File System administrator, from either the administrative command-line interface or the SAN File System console, that changes the state of the metadata server in some way.

Each administrative server has its own audit log. If, from the master metadata server, you display the audit log from either the administrative command-line interface or the SAN File System console, all audit logs on all engines in the cluster are consolidated into a single view.

Fields

Log entries contain the following fields:

Timestamp

A date followed by a local time.

Severity level

Set to a value of Informational (console) if the command succeeded. Otherwise, it is set to a value of Error.

Message ID

A unique identifier for the message.

Message type

Set to Audit. This field is contained in the audit log, but it is not displayed in the consolidated view.

Metadata server ID

A unique identifier for the metadata server on which the command was issued.

Message

Contains the user name of the SAN File System administrator who issued the command followed by a functional replica of the log message itself.

The following example shows a message from the consolidated view of the audit log that is displayed through the administrative command-line interface:


```
2003-04-21 18:36:32 INFORMATIONAL HSTAD0083I A mds_engine_0
User Name: root Command Name: MasterServiceAddServer
Parameters: SYSTEMCREATIONCLASSNAME=STC_Cluster
SYSTEMNAME=testnode CREATIONCLASSNAME=STC_MasterService
NAME=MasterService CLUSTERPORT=10001 IP=9.29.25.136
Command Succeeded.
```

Server log

The server message log contains operational information for the metadata server.

Each metadata server in the SAN File System cluster has its own log. If you display the server log from either the administrative command-line interface or the SAN File System console, all server logs on all engines in the cluster are consolidated into a single view. The consolidated view of the server message log is called the cluster log.

Fields

Log entries contain the following fields:

Timestamp

A date followed by a local time.

Severity level

Indicates whether the entry is an informational, warning, error, or severe message.

Message ID

A unique identifier for the message.

Message type

Specifies whether the message is a normal log entry or one that was generated as a result of an event on the metadata server.

Metadata server ID

A unique identifier for the metadata server on which the command was issued.

Message

A textual explanation of the message.

The following example shows a message from the cluster log that is displayed through the administrative command-line interface:

```
2003-04-16 12:55:50 INFORMATIONAL HSTPG0009I N
msd_engine_0 Using IP 9.38.203.26 port 10192
for Group Services messages.
```

Event log

The event log contains all messages from the cluster message log that have a message type of event.

This topic explains the fields on the event log.

Fields

Log entries contain the following fields:

Message ID

A unique identifier for the message.

Severity level

The severity indicates whether the entry is an informational, warning, or error message.

Metadata server ID

A unique identifier for the metadata server on which the command was issued.

Timestamp

The timestamp consists of a date followed by a local time.

Message

The message is a textual explanation.

The following example illustrates the event log that appears through the Administrative command-line interface:

```
ID          Level  Server  Date and Time          Message
=====
TANCM0393I Info   svc-mds1 May 16, 2003 2:08:01 AM ALERT: The server state
has changed from Initializing(2) to Joining(5).
TANCM0393I Info   svc-mds1 May 16, 2003 2:08:13 AM ALERT: The server state
has changed from Joining(5) to Online(10).
TANCM0389I Info   svc-mds1 May 16, 2003 2:08:13 AM ALERT: The cluster state
has changed from Forming(6) to Online(10).
TANCM0393I Info   svc-mds2 May 16, 2003 2:09:22 AM ALERT: The server state
has changed from NotAdded(4) to Joining(5)
```

Administrative server logs

This topic describes where the SAN File System administrative server logs are stored.

The following logs for the administrative server are stored on the engine hosting those servers.

Table 2. SAN File System administrative server message log files

Log	File name	Location	Maximum file size
Administrative log	cimom.log	/usr/tank/admin/log	–
Console log	console.log	/usr/tank/admin/log	–
Security log	security.log	/usr/tank/admin/log	–
Standard error	stderr.log	/usr/tank/admin/log	–
Standard out	stdout.log	/usr/tank/admin/log	–

When you display these logs from the master metadata server using either the administrative command-line interface or the SAN File System console, you actually see a consolidated view of all of the logs from each engine in the cluster.

Service only logs

The console.log, stderr.log, and the stdout.log are not intended for normal problem determination. They will not be discussed in detail.

Administrative log

The administrative log contains messages generated by the administrative server.

If, from the master metadata server, you display the administrative log from either the administrative command-line interface or the SAN File System console, all administrative logs on all engines in the cluster are consolidated into a single view.

Fields

Log entries contain the following fields:

Message ID

A unique identifier for the message.

Severity level

Indicates whether the entry is an informational, warning, error, or severe message.

Message type

Specifies whether the message is a normal log entry or one that was generated as a result of an event on the administrative server.

Administrative server ID

A unique identifier for the administrative server on which the command was issued.

Timestamp

A date followed by a local time.

Message

The log message.

The following example illustrates the consolidated view of the administrative log that is displayed through the administrative command-line interface:

```
CIMServer: Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  Namespace \root\cimv2 initialized
CMMOM0411I Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  Authorization is not active
CMMOM0901I Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  IndicationProcessor started
CMMOM0906I Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  No pre-existing indication subscriptions
CMMOM0404I Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  Security server starting on port 5989
CMMOM0402I Info Normal mds_engine_0 May 16, 2003 7:27:33 AM
  Platform is Unix
```

Security log

The security log displays the administrative user login activity for the administrative server.

If, from the master metadata server, you display the security log from either the administrative command-line interface or the SAN File System console, all security logs on all engines in the cluster are consolidated into a single view.

Fields

Log entries contain the following fields:

Message ID

A unique identifier for the message.

Severity level

Indicates whether the entry is an informational, warning, or error message.

Administrative server ID

A unique identifier for the administrative server on which the command was issued.

Timestamp

A date followed by a local time.

Message

The log message.

The following example illustrates the administrative log displayed through the administrative command-line interface:

```
CMMOM0302I Info mds_engine_0 May 19, 2003 9:21:17 AM
    User respey on client {1} could not be authenticated
CMMOM0302I Info mds_engine_0 Jun 13, 2003 1:51:40 PM
    User jkaminski on client {1} could not be authenticated
CMMOM0302I Info mds_engine_0 Jun 20, 2003 5:41:36 PM
    User fstock on client {1} could not be authenticated
```

Metadata server tracing

This topic describes how to enable tracing for metadata servers.

You can enable tracing for the metadata server through the administrative command-line interface using the trace command. This command enables you to control:

- When tracing begins and ends.
- The components within the metadata server for which tracing will occur.
- The level of detail (verbosity) to show during tracing.

Tracing generates a trace log called log.trace, that is stored in /usr/tank/server/log. Like other log files, this file has a maximum file size of 250 MB. When it reaches this size, the SAN File System creates a copy called log.trace.old and clears log.trace. If log.trace.old already exists, it is overwritten.

Note: A minimum level of tracing always occurs for the metadata server, so this file always exists.

SAN File System dump capability

This topic provides an overview of the tools that you can use to create system dumps for SAN File System servers.

The SAN File System provides three ways that you can gather diagnostic data about the engines in the cluster as well as the metadata servers that run on those engines:

- Run the **collectdiag** command from the administrative command-line interface. The **collectdiag** command allows you to collect information about an engine in the cluster.
- Click **Maintain System** → **Collect Diagnostics** from the SAN File System console. The Collect Diagnostics task is the console equivalent of running the **collectdiag** command from the administrative command-line interface.
- Use the One-Button Data Collection Utility.

SAN File System server dump capability

This topic describes how to create system dumps using the SAN File System server that is running the Linux operating system.

You can use the Linux operating system process dump capabilities to help with debugging and problem determination of the SAN File System. In addition, you may need to force a core dump of the metadata server process so that you can package the data and send it to IBM support personnel for review. Follow these steps to force a core dump:

1. Use the **ulimit** shell command to set the size of the allowable core dump file size to be unlimited.

Note: You can use the **ulimit** shell command with the **-a** parameter to verify the current allowable limit.

```
ulimit -c unlimited
```

2. On the command line, type **cat /usr/tank/server/config/Tank.PID** to view the contents of the file that contains the parent metadata PID that you need to kill. Use this PID value as the parameter to the **kill -6** command.

Note: This command terminates all SAN File System processes that are currently running on this engine.

3. Enter the command **kill -6** which produces a file called *core.PID* in the **/usr/tank/server** directory. The PID portion of the core file name matches the number of the parent PID that you just killed.

For example, if the parent **pid** value is 550, type **kill -6 550** and the core file **/usr/tank/server/core.550** is created.

The metadata server runs in user space. Therefore, a problem with the metadata server should not crash the Linux kernel. You should not need to analyze kernel dumps on Linux for the metadata server.

Common errors

This topic lists some common errors that you might encounter when using SAN File System. While these errors are generally easy fixes, it is sometimes difficult to determine the cause.

Errors when running **setupsfsclient**

After running the SAN File System client setup utility, the utility fails with the following error messages:

```
HSTDR0029I The kernel extension was successfully loaded from file
/usr/tank/client/bin/stfs kernel module ID (kmid) = 63cc400.
HSTDR0030I File system driver is initialized and ready to handle file-system
type 20. /usr/tank/client/bin/stfsclient:
HSTCL0011E A -create parameter uses a maximum of two parameter values:
client name and server name. You specified 17.
HSTCS0008E Could not create SAN File System client
HSTDR0033E SAN File System driver shut down successfully.
HSTDR0035I The kernel extension 63cc400 was unloaded successfully.
```

This group of errors is displayed because of a syntax error in one of the configuration fields that you completed when you ran the **setupsfsclient** command.

An example of this can be seen in the following entry to the device candidate list:

```
pat=/dev/rvpath *
```

In this field, there should not be a space between **rvpath** and the asterisk (*). In this case, the **setupsfsclient** will fail and the previously mentioned errors will result.

Error: "Drive not found" when starting SAN File System

One startup test when starting SAN File System requires the system to discover the drives. If this test does not complete before a test that requires a specific drive to be available begins, the server will enter a 5 second pause and attempt the connection again.

The server should connect to the drive before its next attempt, but a Drive not found error might occur from the first attempt.

Client diagnostic tools

There are several diagnostic tools that you can use to diagnose problems with SAN File System clients.

You can use the logging and tracing capabilities provided by SAN File System to perform problem determination with SAN File System clients on UNIX and Windows operating systems. In addition, both operating systems provide the ability to generate and analyze system dumps.

AIX client logging and tracing

This topic describes the logging and tracing capabilities available on SAN File System AIX clients.

Logs

In AIX, use the **syslog** utility to enable logging on an AIX client. The log messages are routed through the **syslog** utility on the AIX operating system. The **syslog** utility captures log output from the kernel, as well as other operating system services. By default, the **syslog** utility discards all kernel output. However, you can configure the **syslog** utility to specify a destination for the messages by modifying the `/etc/syslog.conf` file.

- **Specifying a file as the destination**

You can specify a file to receive kernel messages, such as `/var/adm/ras/messages`. Perform the following steps to specify that file:

1. Create the file `/var/adm/ras/messages` if it does not already exist. You can use the AIX **touch** command to create an empty file.
2. Edit the `/etc/syslog.conf` file.
3. Insert this line:

```
kern.debug /var/adm/ras/messages
```

You can also redirect the `kern.debug` to `/var/spool/mqueue/syslog` instead of `/var/adm/ras/messages` by specifying that file. Create the `/var/spool/mqueue/syslog` file first by using the **touch** command.

4. Restart the `syslogd` daemon.

```
kill -hup syslogd_PID
```

Refer to either *AIX 5L™ Version 5.1 Commands Reference, Volume 5* or *AIX 5L Version 5.2 Commands Reference, Volume 5, s-u* for more information about the `syslogd` daemon.

Specifying the console as the destination

Perform the following steps to specify the console as the destination for kernel messages:

1. Edit the `/etc/syslog.conf` file.
2. Insert this line:
`kern.debug /dev/console`
3. Restart the `syslogd` daemon.
`kill -hup syslogd_PID`

Refer to either *AIX 5L Version 5.1 Commands Reference, Volume 5* or *AIX 5L Version 5.2 Commands Reference, Volume 5, s-u* for more information about the `syslogd` daemon.

Log messages

When you specify `kern.debug` as shown in the previous examples, all levels of kernel output are routed because `debug` is the lowest priority level of kernel output. You can specify a different level of output, such as `kern.info` to show just informational messages.

The following example messages show the format of log messages:

```
Apr 21 07:43:50 aixclient1 unix: STFS: disk configuration process created
with PID = 13348
Apr 21 07:43:50 aixclient1 unix: STFS: cleaner process created with PID 12028
Apr 21 07:43:50 aixclient1 unix: STFS: CSM process created with PID 10860
```

Traces

The SAN File System client generates trace messages on the AIX operating system. Trace messages are recorded in a memory-based trace buffer. The trace buffer is a circular buffer. Trace messages are overwritten with new messages when the buffer overflows.

First failure tracing is enabled by default and cannot be disabled. All other tracing levels are disabled by default and you can enable them using the **sanfstrace set** utility.

Use the **sanfstrace log** utility to retrieve logged trace messages from the trace buffer and append them in text format to a specified log file. If you do not specify a file, SAN File System sends the messages to the standard output device.

You can use the **sanfstrace** utility to set and list levels of tracing.

Viewing SAN File System classes

Tracing in SAN File System is controlled by components called *classes*. A class loosely corresponds to a file system operation such as mounting or reading. Every trace class has an associated level of output. The higher you set the level of the trace class, the greater number of trace messages you receive from the class. A trace class set at level zero is disabled. When SAN File System encounters trace messages in the classes, it saves them to the internal ring buffer. No output is generated until you retrieve them using the **sanfstrace log** utility.

To view the classes, enter:

```
sanfstrace list
```

Enabling trace messages

By default, SAN File System maintains its messages in the internal ring buffer so that it can recover them after a system crash. To enable or disable tracing on the AIX client, use the **sanfstrace set** command. For example, to enable tracing on *all* classes at the same level, enter:

```
sanfstrace set -level=number
```

where *number* is a level of verbosity of the tracing that increases from 1 to 9.

To enable tracing for *selected* classes, enter:

```
sanfstrace set -class=classname -level=number
```

Where *classname* is the class for which you are enabling tracing, and *number* is the level of tracing that increases from 1 to 9. You can set as many classes as needed by separating them with a comma and no spaces.

Note: To disable tracing for a particular class, enter a zero as the level.

The following example messages show the format of trace messages:

```
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.439290 50337 STFS
  traceBuf_daemonize: going to sleep till shutdown
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448769 196267 STFS CSM
  OS-dependent services initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448827 196267 STFS Pager
  Strategy initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448875 196267 STFS GFS
  hooks initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448969 196267 STFS
  doInit(): system Initialized
```

AIX client dump capability

This topic describes how to create system dumps for SAN File System clients using the AIX operating system.

You may need to use the dump capabilities if the SAN File System client virtual file system (AIX) hangs, meaning that it is still running but will not respond to commands. Perform one of these steps to initiate a kernel dump.

Note: The SAN File System client runs in kernel space.

- If you can access the client machine remotely (using telnet or ssh) to obtain a shell prompt, issue the command `sysdumpstart`. For information about using the `sysdumpstart` command, refer to either *AIX 5L Version 5.1 Commands Reference, Volume 5* or *AIX 5L Version 5.2 Commands Reference, Volume 5, s-u* for more information about the `syslogd` daemon. These are available from the IBM Web site.
- If you cannot access the client machine remotely, you can initiate a kernel dump using the reset button on the front panel of the machine. See the documentation that was provided with the client machine to determine how to reset it.

By default, the dump file is saved as `/var/adm/ras/vmcore.n`, where *n* is a number that is incremented each time a dump file is created.

Linux client logging and tracing

This topic describes the logging and tracing capabilities available on SAN File System Linux clients.

Logs

In Linux, use the **syslog** utility to enable logging on a Linux client. The log messages are routed through the **syslog** utility on the Linux operating system. The **syslog** utility captures log output from the kernel, as well as other operating system services. By default, the **syslog** utility discards all kernel output. However, you can configure the **syslog** utility to specify a destination for the messages by modifying the `/etc/syslog.conf` file.

- **Specifying a file as the destination**

You can specify a file to receive kernel messages, such as `/var/log/messages`. Perform the following steps to specify that file:

1. Create the file `/var/log/messages` if it does not already exist. You can use the Linux **touch** command to create an empty file.
2. Edit the `/etc/syslog.conf` file.
3. Insert this line:
`kern.debug /var/log/messages`
4. Restart the `syslogd` daemon:
`/sbin/service syslog restart`

Refer to the *Linux Commands Reference* for more information about the `syslogd` daemon.

- **Specifying the console as the destination**

Perform the following steps to specify the console as the destination for kernel messages:

1. Edit the `/etc/syslog.conf` file.
2. Insert this line:
`kern.debug /dev/console`
3. Restart the `syslogd` daemon:
`/sbin/service syslog restart`

Refer to the *Linux Commands Reference* for more information about the `syslogd` daemon.

When you specify `kern.debug` as shown in the previous examples, all levels of kernel output are routed because `debug` is the lowest priority level of kernel output. You can specify a different level of output, such as `kern.info` to show just informational messages.

Traces

Trace messages are recorded in a memory-based trace buffer. The trace buffer is a circular buffer. Trace messages are overwritten with new messages when the buffer overflows.

First failure tracing is enabled by default and cannot be disabled. All other tracing levels are disabled by default and can be enabled using the **sanfstrace set** utility.

Use the **sanfstrace log** utility to retrieve the logged trace messages from the trace buffer and append them in text format to a specified log file. If you do not specify a file, the messages are sent to the standard output device.

You can also use the **sanfstrace** utility to set and list levels of tracing.

Viewing SAN File System classes

Tracing in SAN File System is controlled by components called *classes*. A class loosely corresponds to a file system operation such as mounting or reading. Every trace class has an associated level of output. The higher you set the level of the trace class, the greater number of trace messages you receive from the class. A trace class set at level zero is disabled. When SAN File System encounters trace messages in the classes, it saves them to the internal ring buffer. No output is generated until you retrieve them using the **sanfstrace log** utility.

To view the classes, enter:

```
sanfstrace list
```

Enabling trace messages

By default, SAN File System maintains its messages in the internal ring buffer so that it can recover them after a system crash. To enable or disable tracing on the Linux client, use the **sanfstrace set** command. For example, to enable tracing on *all* classes at the same level, enter:

```
sanfstrace set -level=number
```

where *number* is a level of verbosity of the tracing that increases from 1 to 9.

To enable tracing for *selected* classes, enter:

```
sanfstrace set -class=classname -level=number
```

Where *classname* is the class for which you are enabling tracing, and *number* is the level of tracing that increases from 1 to 9. You can set as many classes as needed by separating them with a comma and no spaces.

Note: To disable tracing for a particular class, enter a zero as the level.

The following example messages show the format of trace messages:

```
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.439290 50337 STFS
  traceBuf_daemonize: going to sleep till shutdown
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448769 196267 STFS CSM
  OS-dependent services initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448827 196267 STFS Pager
  Strategy initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448875 196267 STFS GFS
  hooks initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448969 196267 STFS
  doInit(): system Initialized
```

Solaris client logging and tracing

This topic describes the logging and tracing capabilities available on SAN File System Solaris clients.

Logs

In Solaris, use the **syslog** utility to enable logging on a Solaris client. The log messages are routed through the **syslog** utility on the Solaris operating system. The **syslog** utility captures log output from the kernel and other operating system services. By default, the **syslog** utility discards all kernel output; however, you can configure the **syslog** utility to specify a destination for the messages.

- **Specifying a file as the destination**

You can specify a file to receive kernel messages, such as `/var/log/messages`. Perform the following steps to specify that file:

1. Create the file `/var/log/messages` if it does not already exist. You can use the Solaris **touch** command to create an empty file.
2. Edit the `/etc/syslog.conf` file.
3. Insert this line:
`kern.debug /var/log/messages`
4. Restart the `syslogd` daemon:
`/sbin/service syslog restart`

Refer to the *Solaris Commands Reference* for more information about the `syslogd` daemon.

- **Specifying the console as the destination**

Perform the following steps to specify the console as the destination for kernel messages:

1. Edit the `/etc/syslog.conf` file.
2. Insert this line:
`kern.debug /dev/console`
3. Restart the `syslogd` daemon:
`/sbin/service syslog restart`

Refer to the *Solaris Commands Reference* for more information about the `syslogd` daemon.

Log messages

The following example messages show the format of log messages created by the Solaris client code that resides in the kernel:

```
Apr 29 19:30:08 gas sanfs: [ID 967454 kern.warning] WARNING: csmGetRequest:
\tmGetServerRequest() failed.
tmrc:7
Apr 29 19:33:39 gas sanfs: [ID 991888 kern.warning] WARNING: Lost lease with
server 0 lease 0x3000488cde0.
Apr 29 19:34:51 gas sanfs: [ID 733070 kern.notice] NOTICE: TmProcessIdentifyResp :
clusterId:15022 installationId:740883aae7649163
```

The following example messages show the format of log messages created by the Solaris FlexSAN daemon:

```
Apr 29 19:34:53 gas sanfsd[1007]: [ID 826785 daemon.error] Couldn't send the SCSI
command 0x12 to device /dev/dsk/c0t1d0s2: I/O error
Apr 29 19:34:54 gas sanfsd[1007]: [ID 518328 daemon.error] Failed to add the
disk /dev/dsk/c2t9d0s2 for /mnt/mwytank
Apr 29 19:34:54 gas sanfsd[1007]: [ID 484526 daemon.error] osDoDiscoverVols: failed
to add device /dev/rdisk/c2t9d0s2
```

Traces

The SAN File System client generates trace messages on the Solaris operating system. Trace messages are recorded in a memory-based trace buffer. The trace buffer is a circular buffer. Trace messages are overwritten with new messages when the buffer overflows.

First failure tracing is enabled by default and cannot be disabled. All other tracing levels are enabled by default and you can disable them using the **sanfstrace set** utility.

Use the **sanfstrace log** utility to retrieve logged trace messages from the trace buffer and append them in text format to a specified log file. If you do not specify a file, SAN File System sends the messages to the standard output device.

You can use the **sanfstrace** utility to set and list levels of tracing.

Viewing SAN File System classes

Tracing in SAN File System is controlled by components called *classes*. A class loosely corresponds to a file system operation such as mounting or reading. Every trace class has an associated level of output. The higher you set the level of the trace class, the greater number of trace messages you receive from the class. A trace class set at level zero is disabled. When SAN File System encounters trace messages in the classes, it saves them to the internal ring buffer. No output is generated until you retrieve them using the **sanfstrace log** utility.

To view the classes, enter:

```
sanfstrace list
```

Enabling trace messages

By default, SAN File System maintains its messages in the internal ring buffer so that it can recover them after a system crash. To enable or disable tracing on the Solaris client, use the **sanfstrace set** command. For example, to enable tracing on *all* classes at the same level, enter:

```
sanfstrace set -level=number
```

where *number* is a level of verbosity of the tracing that increases from 1 to 9.

To enable tracing for *selected* classes, enter:

```
sanfstrace set -class=classname -level=number
```

Where *classname* is the class for which you are enabling tracing, and *number* is the level of tracing that increases from 1 to 9. You can set as many classes as needed by separating them with a comma and no spaces.

Note: To disable tracing for a particular class, enter a zero as the level.

The following example messages show the format of trace messages:

```
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.439290 50337 STFS
  traceBuf daemonize: going to sleep till shutdown
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448769 196267 STFS CSM
  OS-dependent services initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448827 196267 STFS Pager
  Strategy initialized.
```

```
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448875 196267 STFS GFS
hooks initialized.
Apr 28 13:17:09 aixclient1 unix: STFS: 1051550182.448969 196267 STFS
doInit(): system Initialized
```

Windows client logging and tracing

This topic describes the logging and tracing capabilities available on SAN File System Windows clients.

Logs

On Windows clients, log messages are written to the standard event log. To view the log, click **Start** → **Programs** → **Administrative Tools** → **Event Viewer**.

Log messages provide information, warnings, and errors of general interest to administrators and support personnel.

Log messages are written to the standard system logging interface, the Windows Event Log. In addition to the operating system messages, the Windows Event Log contains messages generated by SAN File System.

You can use the Event Viewer to list messages from the Event Log. Double-click a message from the Event Viewer to find more information about that message. You can also use the Event Viewer to filter messages by message type, source of the message, or by a specified range of time. You can also dump events to a text file, which is useful for sending problem determination data to remote support personnel.

Log messages

The following example messages show the format of the log messages:

```
4/21/2003 7:32:03 PM Stfs Error      None 9 N/A WINCLIENT1
HSTCW0009E: Couldn't contact server at IP address <18.18.18.99:11190>
4/21/2003 7:32:36 PM Stfs Information None 8 N/A WINCLIENT1
HSTCW0008I: Contacted server at IP address <18.18.18.99:11190>.
4/21/2003 7:32:02 PM Stfs Information None 1 N/A
WINCLIENT1 HSTCW0001I: SAN File System client started successfully.
```

Traces

Tracing is started automatically on Windows clients. To disable the automatic start, edit the Windows registry and remove the **Stfs\Trace\Filename** setting. If tracing is not started automatically, you can manually start it using the **sanfstrace** utility. Use the Event Viewer to determine if tracing was started automatically.

Important: Do not initiate a trace session if a session is active. Multiple trace sessions will interfere with each other. You can, however, use the **sanfstrace** utility from the command-line interface (CLI) or the Microsoft Management Console (MMC) to set the trace levels or to list the trace classes while a trace session is active.

By default, Windows client tracing is written in the file `\Program Files\IBM\Storage Tank\Client\log\sanfstrace.log`. You can view the trace log using a standard text editor. To change the log file name, edit the Windows registry and modify the `Stfs\Trace\Filename` setting.

The automatic tracing feature on Windows now adds a date and time stamp to the filename specified in the registry. For example, the default filename, `sanfs.log`, now appears as `sanfs.log.YYYYMMDDHHMM` which indicates the complete date and time that the Windows client was last booted. Therefore, the filename `sanfs.log.200409021926` indicates that `sanfs.log` was last generated on September 2, 2004 at 7:26 p.m. Each time the client is restarted, a new trace file starts and has the same base name, but different date and time stamps. This filename convention prevents individual files from becoming too large.

Enabling detailed tracing

By default, minimal tracing is enabled. Use the `sanfstrace` command from the command-line interface or use Microsoft Management Console to modify and control the level of tracing for a client. Remember, however, that full tracing can significantly impact the performance of the SAN File System. To enable detailed tracing, you must provide a path and filename to a file to use for the detailed trace log:

1. Start the Windows registry editor.
2. Navigate to the following registry key:
`\\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Stfs\Trace`
3. In this key, there is a string value named **FileName** that is set to `\Program Files\IBM\Storage Tank\Client\log\sanfstrace.log`. Within the registry key, the **Categories** value defaults to Errors which enables just minimal tracing capability. Change this value to the level of tracing you want from SAN File System.
4. Click **OK**.
5. Restart for the changes to take effect.

Trace messages

The `sanfstrace.log` file contains tracing output only for the SAN File System client. It does not contain information for the operating system or other applications.

The following example messages show the format of trace messages:

```
#E 8125026652|80BF9DA0 Reassert TStreamSocket::Disconnect:1911
84860B68 192.168.10.6:10290 CheckStatus failed:
STATUS_CONNECTION_ACTIVE (C000023B)
8125026715|8122C9E0 ckground TBackground::Main:1301
F4A8EF88 Active count: 00000001
8125026715|8122C9E0 ckground TBackground::Main:1305
F4A8EF88 Active work items: 00000000
8125026715|8122C9E0 ckground TBackground::Main:1308
F4A8EF88 Active delayed work items: 00000000
8125031715|8122C9E0 ckground TBackground::Main:1301
F4A8EF88 Active count: 00000001
8125031715|8122C9E0 ckground TBackground::Main:1305
F4A8EF88 Active work items: 00000000
8125031715|8122C9E0 ckground TBackground::Main:1308
F4A8EF88 Active delayed work items: 00000000
8115545496|FE2D7020 TSc::Reference
FE484008 Fil:<23456.3.1342760.0> ReferenceCount 3, CsmHandle_Held
```

Windows client dump capability

This topic describes how to create system dumps for SAN File System clients using the Windows 2000 and Windows 2003 operating systems.

You can configure the operating system to generate a dump file if the SAN File System client on the Windows operating system terminates abnormally. By default, the file is C:\WINNT\memory.dmp. You can also configure where the file is saved.

In cases where the SAN File System client hangs, you can force the creation of a dump file. However, you must have previously configured the system to allow the creation of a dump file by:

- Making sure that the system is configured to generate a dump file. To verify the settings, click **Start** → **Control Panel** → **System** → **Advanced** → **Startup and Recovery** → **System Failure**.
- Using the registry editor (regedit) to modify the following registry setting:

Hive: HKEY_LOCAL_MACHINE

Key: System\CurrentControlSet\Services\i8042prt\Parameters

Name: CrashOnCtrlScroll

Data Type:
REG_DWORD

Value: 1

Note: A value of 1 enables the feature.

To create a dump file, press and hold the right **Ctrl** key while pressing **ScrLk** twice from the keyboard. The dump file is generated the next time you restart the machine.

One-button data collection utility

This topic provides an overview of the SAN File System one-button data collection utility.

You can use the one-button data collection utility to gather vital product data (VPD) about SAN File System hardware and software, as well as other pertinent information about the clients and metadata servers. This information can help you analyze a problem as well as collect the data to send to other support personnel.

You can invoke the one-button data collection utility in one of the following ways:

- For an engine in the cluster, perform one of these actions:
 - Start the SAN File System console. Then click **Maintain System** → **Collect Diagnostic**.
 - Access the engine and from a shell prompt, run `/usr/tank/server/bin/obdc` to collect the default data or add additional parameters to customize the data collection.
- For a client running UNIX, access the client and from a shell prompt, run `/usr/tank/client/bin/obdc` to collect the default data or add additional parameters to customize the data collection.
- For a client running Windows, access the client and from a shell prompt, run `C:\Program Files\IBM\Storage Tank\client\bin\obdc.exe` to collect the default data. In addition, you must specify some additional options to collect all of the data that OBDC can collect.

To access additional OBDC usage information, run:

```
/usr/tank/server/bin/obdc -help
```


OBDC minimum C++ runtime requirements

This topic describes the minimum C++ runtime levels that must be installed on a UNIX operating system for the One Button Data Collection (OBDC) utility to function correctly.

During installation, each client verifies that your operating system meets the minimum requirements. If it does not, the client package installation will fail.

Note: There are no C++ requirements for Windows 2000 and Windows 2003 operating systems.

C++ requirements for UNIX operating systems

The following table shows the minimum required C++ runtime for OBDC to function on UNIX operating systems.

Table 3. C++ minimum requirements

Operating System	Requirements
AIX 5.1, 5.2, and 5.3	vacpp.cmp.rte version 6.0.0.0 or xIC.rte version 6.0.0.0
Red Hat Linux 3.0	libstdc++-2.96-110
SuSE Linux 8	libstdc++-2.96-110
Sun Solaris 9	Minimum requirements installed with the operating system.

Hardware Vital Product Data

This topic provides an overview of the hardware VPD collected by the one-button data collection utility.

This section describes the hardware vital product data that is collected by the one-button data collection utility.

Engine

You can collect hardware information about an engine in the cluster, such as the following:

Table 4. Engine hardware VPD

Hardware area	Component	VPD collected
Processor/PCI devices	Machine	Type, model number, vendor, and serial number.
	Host bridge	Device, vendor, firmware version, and latency.
	ISA bridge	Device, vendor, and firmware version.
	Ethernet controller	For each device: adapter type, vendor, firmware version, latency, and memory usage.
	USB controller	
	VGA controller	
	IDE interface	
	Fibre-channel adapter	
	RSA II adapter	

Table 4. Engine hardware VPD (continued)

Hardware area	Component	VPD collected
Memory	Memory	Statistics for total memory available, used, free, shared, buffered, and cached. Additional usage statistics as well.
	Swap space	
LAN network	Ethernet interfaces	For each device: data received and transmitted, Internet address, network masks, packets, collisions, interrupts, errors, and base memory address.
	Loopback interface	Statistics about Internet address, network masks, packets, collisions, and errors.
	IP routing table	For each destination: gateway address, network masks, flags, and interfaces.
Local storage	SCSI devices	For each device: channel, ID, LUN, vendor, model, version, and type.
	File systems	Device, mount point type, and inodes (total, used, and free).
	Mount points	Device, file system, and read/write settings.

Software Vital Product Data

This topic provides an overview of the software VPD collected by the one-button data collection utility.

This section describes the software vital product data that is collected by the one-button data collection utility.

Engine

You can collect information about the software running on an engine in the cluster, such as the following:

Table 5. Engine software VPD

Software area	Component	VPD collected
Operating system	Machine	Name and version.
	Operating system	Version, build information, and installation date.
	Processes	Owner, ID, binary file, status, runtime parameters, and environment variables. Additional details as well.
	System log files	Collect in their entirety.
	Core files	Operating system core files and corresponding binary file, if present and requested.
Network	Active connections	Protocol (TCP/UDP), local and remote addresses, state, and receive and send queues.
	Active sockets	Type, state, flags, reference count, and full pathname.
	ARP	IP address, hardware address, and device.

Table 5. Engine software VPD (continued)

Software area	Component	VPD collected
Metadata server	Configuration files	Version, Tank.Bootstrap, and Tank.Config files.
	Log files	log.std, log.audit, log.trace, log.cim, log.stopengine, log.dmp, and log*.old.
	Core files	Server core file, if present.
	Server configuration	Dump information about: <ul style="list-style-type: none"> • Version of installed server code • Current server state • Current server role • Protocol (TCP or UDP) • IP address and network mask
	Server state	Dump information about: <ul style="list-style-type: none"> • Active threads, their state, and activity • Mutexes and current state of each one • Latches and current state of each one • Condition variables and information about each one • Write-ahead log writer thread information.
Administrative server	Cluster configuration	Dump information about: <ul style="list-style-type: none"> • Number of servers in cluster • Listing of all servers in cluster • Fileset (container) information • Global disk table and type of each disk (master, system, user) • List of registered clients and information about each one • Heartbeat interval between servers • Current state of High-Availability Manager
	Configuration files	tank.properties, cimom.properties, and tank_device_map files.
	Log files	cimom.log, console.log, security.log, and WebSphere® Application Server-based trace.log files.
	Core files	Administrative server core file, if present.

Client

You can collect information about the software running on a client, such as the following:

Table 6. Client software VPD

Software area	Component	VPD collected
Operating system	Machine	Name and version.
	Operating system	Version, build information, and installation date.
	Processes	Owner, ID, binary file, status, runtime parameters, and environment variables. Additional details as well.
	System log files	Collect in their entirety.
	Core files	Operating system core files and corresponding binary file, if present and requested.
SAN File System client	Log files and trace files	All client log and trace files
Network	Active connections	Protocol (TCP/UDP), local and remote addresses, state, and receive and send queues.
	Active sockets	Type, state, flags, reference count, and full pathname.
	ARP	IP address, hardware address, and device.

Chapter 7. Isolating problems with the SAN File System

This topic explains how to determine and isolate problems with the SAN File System, SAN, and LAN.

In most cases, you can use the logs provided by the SAN File System to begin isolating problems.

- For problems that seem to be related to clients, use the logs that are available with the client operating system (such as the system log on AIX clients and the Event Log on Windows clients) to determine the cause of the problem. In addition, you can use the information provided in Chapter 10, "Troubleshooting a SAN File System client," on page 85.
- For problems that seem to be related to administrative user access, use the security log and the administrative log to determine the cause of the problem. If you access these logs through the master metadata server, you see a consolidated view of the logs from each of the metadata servers in the cluster. If you access these logs through a subordinate metadata server, you see the logs for that particular metadata server.

In addition, you can use the information provided in Chapter 8, "Troubleshooting an administrative server," on page 63.

- For problems that seem to be related to the cluster, metadata servers, or metadata, use the server log to determine the cause of the problem.
In addition, you can use the information provided in Chapter 9, "Troubleshooting the cluster," on page 73. If you access this log through the master metadata server, you see a consolidated view of the logs from each metadata server in the cluster (called the cluster log). If you access these logs through a subordinate metadata server, you see the logs for that particular metadata server only.
- For problems that seem to be related to the installation, refer to Chapter 11, "Troubleshooting the installation," on page 91. The topics provide commands and suggestions that you can use to diagnose installation problems.
- For problems that seem to be related to the RSA (Remote Supervisor Adapter) II, refer to Troubleshooting the RSA II adapter. This topic provides information about problems you might encounter with the RSA II during the installation and upgrade processes.
- For problems that seem to be related to the engines in the SAN File System, refer to the server documentation to determine the cause of the problem.

If you are not sure if the problem is related to the SAN File System, SAN, or LAN, use the information in this section to begin isolating the problem.

Note: Certain events, such as restarting the operating system or disconnecting cables, can cause the SAN File System to lose connectivity to logical unit numbers (LUNs). If the logs indicate I/O failures for a client or metadata server, verify the following:

- Configured system LUNs should be visible only from metadata servers. Each metadata server must see all configured system LUNs. Configured user LUNs should be visible only from SAN File System clients. Not all user LUNs need to be seen by all clients. A client can see a subset of the total user LUNs, but it must see all LUNs configured for any pool to

which it has access. If the client cannot see all LUNs associated with a particular pool, a warning prints in the server log.

You can configure clients to only see a select group of user LUNs that it needs to access, or configure them to view all of the user LUNs.

- A SAN fabric switch has not lost the zoning configuration. If the operating system on the switch is restarted, it is possible for the fabric to lose the zoning configuration, which prevents metadata servers and SAN File System clients from reaching LUNs.

You might need to force SAN File System to remap LUNs if you lose connectivity. Refer to your SAN administrator for help with LUN rediscovery options specific to your operating environment.

Identifying SAN problems

Perform the following steps to determine if the problem is related to the SAN itself:

1. Determine whether the SAN configuration was recently changed, such as changing the Fibre Channel cable connections or switch zoning. If so, verify that the changes were correct and, if necessary, reverse those changes.
2. Verify that all switches and storage devices used by SAN File System are powered on and are not reporting hardware failures. If you find problems, resolve them before proceeding.
3. Verify that the Fibre Channel cables that connect the metadata servers to the switches are securely connected.
4. IBM Subsystem Device Driver (SDD) version 1.5.1 is provided with SAN File System and provides support for multipath environments. You can use the datapath query commands to view statistics, as well as information about paths and adapters. For information about using the datapath query commands, refer to the *Subsystem Device Driver User's Guide* provided on the SAN File System documentation CD-ROM.
5. If you are running a SAN Management tool that you are familiar with, use it to view the SAN topology and isolate the failing component. If you are not using a SAN Management tool, you can start IBM Tivoli SAN Manager on the master console and use it to view the SAN Topology and isolate the failure. For information about SAN problem determination with IBM Tivoli SAN Manager, contact the Tivoli Storage Area Network (SAN) support center.

Identifying IP networking problems

Perform the following steps to determine whether the problem is related to the IP network itself:

1. Verify that all switches used by the SAN File System are powered on and are not reporting any hardware failures. If problems are found, resolve them before proceeding further.
2. Verify that the Ethernet cables that connect the metadata servers to the switches are securely connected.
3. Verify that the metadata servers, clients, and storage devices are on the same network and subnet.

Identifying storage problems

Perform the following steps to determine whether the problem is related to the storage devices:

1. Determine whether any other hosts that may be attached to the storage devices are having the same problems.
2. Determine whether a single metadata server or client is having trouble accessing the storage device or all metadata servers and clients are experiencing I/O errors.
3. Refer to the documentation for the storage devices for more information about isolating problems with those devices.

Chapter 8. Troubleshooting an administrative server

This topic provides an overview of troubleshooting an administrative server.

Each metadata server in the SAN File System cluster runs an instance of the administrative server, which provides administrative access to the metadata server. The administrative server running on the engine that hosts the master metadata server is referred to as the primary administrative server. All other administrative server instances are referred to as secondary administrative servers.

Administrative server components

The administrative server contains three main components:

- SAN File System console – a set of servlets that run on WebSphere® Application Server. The console provides a Web browser interface to the SAN File System. Users access the console through a secure connection by going to:

`https://master_metadata_server_IP_address:7979/sfs`

If users point to the IP address of a subordinate metadata server and the master metadata server is online, they will automatically be redirected to the IP address of the master metadata server.

- Administrative command-line interface (CLI) – the program (called `sfscli`) that is available on each engine in the cluster. To access the Administrative CLI, users must initiate a secure shell (SSH) session with the master metadata server.

Users who initiate an SSH session with a subordinate metadata server are not automatically redirected to the master. However, most Administrative CLI commands must be run from the master metadata server, so users should typically initiate the SSH session with the master.

Many commands will provide output if run from a subordinate metadata server. However, the output may not be what you expect. For example, the `lsserver` command provides information about the metadata servers in the cluster if you run the command from the master metadata server. If you run this command from a subordinate metadata server, you will see details about that specific metadata server only.

- Administrative agent – part of the SAN File System implementation of the Common Information Model (CIM), which is a model for describing management and information interchange between agents and managers. The administrative agent is used for all native SAN File System operations, such as:
 - Interfacing with the Lightweight Directory Access Protocol (LDAP) server for user authentication
 - Automatically attempting to restart the metadata server in the event of a failure
 - Registering with the service location protocol (SLP), which is a CIM agent directory service

Startup sequence

If you have enabled the automatic restart service, the administrative agent performs these activities when it starts up:

1. Connects to the local metadata server.
2. Learns the location of the master metadata server.

3. Connects to the LDAP server.
4. Registers with the SLP daemon.

Troubleshooting user access to the console

Use the information in this topic to troubleshoot problems that you are having with a user attempting to access the SAN File System console.

Problem

A user was unable to access the SAN File System console from a Web browser.

Investigation

If the user received "access denied" or "unauthorized user" errors

Perform the following steps in order until the problem is resolved:

1. Attempt to sign on using a user name and password that you know are valid.
 - a. If you can sign on using a different user name and password, verify that the "unauthorized" user's user name and password are valid. The user name and password must be created on the Lightweight Directory Access Protocol (LDAP) server.
 - Run the **lsadmuser** command to list all of the administrative users. If the user is not listed, run the **ldapsearch** command from the bash shell to determine if the user is defined in the LDAP server (see the help that is available with **ldapsearch** to learn more about that command). In addition, you can run **ldapsearch -help** to view information about the **ldapsearch** command online.
 - Use the documentation that is provided with the LDAP server to verify that the user account was set up correctly with the LDAP server.
 - b. Attempt to use this user's user name and password with the other administrative access method. For example, if the user received the error while using the SAN File System console, attempt to sign on to the administrative command-line interface. If the user name and password work with the other administrative access method, suspect a problem with the original administrative access method (the console or the command-line interface).
2. Verify that the LDAP server is running and that there is no configuration problem between the LDAP server and the administrative agent. Check the administrative log to determine if you are receiving errors about the SAN File System not being able to connect to the LDAP server.

If the user received "page not found" errors

Perform the following steps until the problem is resolved:

1. Verify that you entered the correct URL for the primary administrative server.
2. Verify that you can access the engine hosting the primary administrative server.
 - a. From a shell prompt, attempt to ping the engine. If you cannot ping the engine, suspect either:
 - An IP network problem

- A hardware problem on the engine (see the engine documentation)
3. Verify that the administrative agent is running.

If the user received "administrative agent..." or "CIM agent not found" errors

Perform the following steps in order until the problem is resolved:

1. Run the **ps** shell command to verify that the administrative agent is running.

```
ps -ef w | grep -i cimom.cimom
```

You should see results similar to this:

```
root      3822      1  0 07:26 ?          S        0:00
/bin/bash /etc/rc.d/init.d/cimom
start /usr/tank/admin
root      4070    3822  0 07:26 ?          S        0:08
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4087    4070  0 07:26 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4088    4087  0 07:26 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4089    4087  0 07:26 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4090    4087  0 07:26 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4098    4087  0 07:27 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
ldap      4099      758  0 07:27 ?          S        0:00
/usr/sbin/slapd -u ldap
root      4100     946  0 07:27 ?          S        0:00
/opt/was/java/jre/bin/exe/java
-Xbootclasspath/p:/opt/was/java/jre/lib/ext/ibmorb
root      4101    4087  0 07:27 ?          S        2:11
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4102    4087  0 07:27 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
```

2. If the administrative agent is not running, run the **/usr/tank/admin/bin/startCimom** command to start the administrative agent. If you have problems starting the administrative agent, view the administrative log (**/usr/tank/admin/cimom.log**) or **/usr/tank/admin/log/stderr.log** and attempt to resolve the problems you find.

3. Verify that the console has been started.

```
/opt/was/bin/serverStatus.sh metadata_server
```

where *metadata_server* is the host name of the server that you are trying to access.

Note: Run the script with no parameters to obtain additional help on using the script.

4. If the console is not running, run the `/usr/tank/admin/bin/startConsole` command to start the console. If you have problems starting the console, view the log indicated by the error message and attempt to resolve the problems you find.
5. If both the administrative agent and console are running, suspect an IP networking problem.

Troubleshooting user access to the administrative command-line interface

Use the information in this topic to troubleshoot problems you might have when accessing the administrative command-line interface.

Problem

The following information describes scenarios and steps to take to access the administrative command-line interface (CLI).

Investigation

If you received access denied errors

Use the following steps to resolve the problem:

1. Using Secure SHell, access the engine that is having failures.
2. View the `tank.passwd` file to verify that the user name and password are correct:

```
# cat ~/.tank.passwd
```

3. Recreate the password file:

```
cd ~; /usr/tank/admin/bin/tankpasswd -u ldap_username -p ldap_passwd
```

where the `ldap_username` and `ldap_password` are for the user having problems.

4. Run the `sfscli>lserver` command to verify that you have access to the metadata server.
5. Verify that the LDAP server is running and there are no configuration problems between the LDAP server and the administrative agent. Check the administrative log to determine if you are received errors about SAN File System not connecting to the LDAP server.

If you received administrative agent or CIM agent not found errors

Use the following steps to resolve the problem:

1. Run the `ps` Secure SHell command to verify that the administrative agent is running:

```
ps -efw | grep -i cimom.cimom
```

The results should be similar to the following ones:

```
root      4070  3822  0 07:26 ?        S      0:08
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4087  4070  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4088  4087  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4089  4087  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
```

```

com.ibm.cimom.CIMOM -RootDir
root      4090  4087  0 07:26 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4098  4087  0 07:27 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
ldap      4099    758  0 07:27 ?          S        0:00
/usr/sbin/slapd -u ldap
root      4101  4087  0 07:27 ?          S        2:11
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4102  4087  0 07:27 ?          S        0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir

```

2. If the administrative agent is not running, use the `startCimom` command to start the administrative agent. If you have problems starting the administrative agent, view the administrative log or `/usr/tank/admin/log/stderr.log` and attempt to resolve the problems.
3. Verify that the console has been started:

```
/opt/was/bin/serverStatus.sh metadata_server
```

where `metadata_server` is the host name of the server you are trying to access.

Tip: To receive additional help on using the script, run the script with no parameters.

4. If the console is not running, run the `/usr/tank/admin/bin/startConsole` command to start the console. If you have problems starting the console, view the log indicated by the error message and attempt to resolve the problems.
5. If both the administrative agent and console are running, suspect an IP networking problem.

Troubleshooting user access to a specific task or command

Use the information in this topic to troubleshoot problems that you are having with a user attempting to access a specific task or run a specific command.

Problem

A user does not see a specific task listed from the SAN File System console or a user cannot run a specific command from the administrative command-line interface. The user receives authorization errors.

Investigation

Within SAN File System, authorization to perform tasks or run commands is based on the user role, which is defined in the LDAP server database. There are four valid roles in the SAN File System, ranging from Monitor, which provides a basic level of access, up to Administrator, which provides full access to the system. These roles determine what commands and tasks can be performed.

To resolve this problem, verify that the user's role is sufficient to perform the specified task or run the specified command. You can use the `lsadmuser` command to verify the roles for each administrative user. If the role is not sufficient, either

update the user's role to a higher level of access or use a different user name and password (one that has sufficient authorization) to access SAN File System.

Note: Use the documentation that came with the LDAP server to understand how to modify a role for a user.

Resolution procedures

This topic describes resolution procedures that can assist you in resolving problems with the administrative server.

Use the procedures in this section to help you resolve problems with the administrative server. These procedures include:

- "Verifying that the administrative agent is running"
- "Verifying that the console is running" on page 69
- "Replacing expired LDAP and CIMOM certificates" on page 69
- "Configuring LDAP for SAN File System" on page 69

Verifying that the administrative agent is running

This task describes how to verify that the CIM agent is running.

1. Access the engine hosting the master metadata server (by using SSH or the RSA II remote console interface).
2. From a bash shell prompt, enter the following command:

```
ps -efw | grep -i cimom.cimom
```

If the administrative agent is running, you will see output similar to the following:

```
root      4070  3822  0 07:26 ?        S      0:08
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4087  4070  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4088  4087  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4089  4087  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4090  4087  0 07:26 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4098  4087  0 07:27 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
ldap      4099   758  0 07:27 ?        S      0:00
/usr/sbin/slapd -u ldap
root      4101  4087  0 07:27 ?        S      2:11
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
root      4102  4087  0 07:27 ?        S      0:00
/opt/IBMJava2-131/jre/bin/exe/java -Xms128m -Xmx256m
com.ibm.cimom.CIMOM -RootDir
```

If the administrative agent is not running, you will not see any output.

Verifying that the console is running

This task describes how to verify that the console is running.

1. Access the engine hosting the master metadata server by using SSH.
2. From a shell prompt, enter the following command:

```
/opt/was/bin/serverStatus.sh metadata_server
```

If the console is not running, you will see error messages.

Replacing expired LDAP and CIMOM certificates

Expired CIMOM or LDAP certificates must be replaced.

CIMOM and LDAP certificates can expire. When this happens, they must be replaced. If you get an error saying: “Invalid key in truststore,” you must update your LDAP certificate.

To determine when your certificates expire, you can examine the truststore as a simple Java™ keystore. To use the tools provided on the metadata server, enter the following command:

```
/opt/IBMJava2-131/jre/bin/keytool -list -v -alias cimomserver -keystore /usr/tank/admin/truststore -storepass <truststore passwd from tank.properties>
```

The following example output shows a truststore that expires on May 25, 2005:

```
Alias name: cimomserver
Creation date: Tue May 25 16:27:43 PDT 2004
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=SANFS, OU=Software Development, O=IBM Corporation, C=US
Issuer: CN=SANFS, OU=Software Development, O=IBM Corporation, C=US
Serial number: 40b3d66f
Valid from: Tue May 25 16:27:43 PDT 2004 until: Wed May 25 16:27:43 PDT 2005
Certificate fingerprints:
  MD5: D0:04:24:BB:44:4F:BC:9C:05:EB:35:EA:61:1E:B3:AA
  SHA1: 3C:A8:71:C7:09:F0:49:6F:04:2C:97:8D:57:D7:F3:8C:CD:E1:67:2A
```

1. Obtain the current certificate. You can get these from the LDAP administrator. CIMOM certificates are created by the **mktruststore** command. See step 4.
2. On each engine, run **stopConsole**, then **stopCimom**.
3. On the master engine, change to `/usr/tank/admin`.
4. Run **bin/mktruststore**. As a parameter, use the path and file name of the LDAP certificate, if it exists.
5. Use **scp** to copy the truststore into `/usr/tank/admin`.

Important: Do not run the **mktruststore** command on each engine. You must copy the truststore to each engine.

6. On each engine, run `/usr/tank/admin/bin/startCimom`. Then run `/usr/tank/admin/bin/startConsole`.
7. If needed, you can now extract the CIMOM certificate for your third-party CIM application.

Configuring LDAP for SAN File System

There are several LDAP configuration settings in the `cimom.properties` file that must be set up during configuration.

1. Change to the `/usr/tank/admin/config/` directory.

2. Open the tank.properties file.
3. Modify the following parameters:

Table 7. LDAP parameters

Parameter name:	Sample values:	Description:
Port	example: 5989	The port on which CIMOM will listen. Set this to 5989.
TrustPassword		The password used when configuring the truststore.
Authorization		Set to false if you want everyone to be able to access the CLI without access controls. If this is set to false, the GUI is unusable.
ldap.cache.age=600		Maximum age of items in the LDAP Cache. Use 0 to disable the cache.
userrole.ldap.location		The IP address of the LDAP server.
userrole.ldap.bind.dn	example: cn=root	The distinguished name of an authorized LDAP user.
userrole.ldap.cred	example: password	The password of the LDAP user.
userrole.ldap.secured.connection		The flag to enable secured LDAP communication. Set to true, indicates that it uses SSL; set to false, indicates that it uses an open socket.
userrole.ldap.version		Set to 2 if the LDAP server does not support v3.
userrole.ldap.insecured.port	example: 389	The port on which the LDAP server should be listening for an insecure connection. 389 is standard.
userrole.ldap.secure.port	example: 636	The port on which the LDAP server should be listening for a secure connection. 636 is standard.
ldap.basedn.roles	example: ou=Roles,o=ibm,c=us	The base distinguished name to search for roles. This is the location in the LDAP hierarchy to find the role definitions.
ldap.basedn.users	example: ou=Users,o=ibm,c=us	The base distinguished name to search for users. This is the location in the LDAP hierarchy to find users.
ldap.user.filter	example: (&(uid=%v) (objectClass=inetOrgPerson))	The search filter to find a user.

Table 7. LDAP parameters (continued)

Parameter name:	Sample values:	Description:
ldap.user.id.attr	example: uid	The attribute that holds the User ID in the user's objectClass.
ldap.role.filter	example: (&(cn=%v) (objectClass=accessRole))	The role filter to find a role.
ldap.role.id.attr	example: cn	The attribute that holds the name of a role in the role's objectClass.
ldap.role.mem.id.attr	example: member	The attribute that holds the members of a role in the role's objectClass.
LogOnly		Setting this to true ensures that stdout.log in /usr/tank/admin/log is not a copy of cimom.log in the same place. This is recommended.
Note: Default values for parameters not listed in this table are acceptable.		

Resetting an incorrect LDAP setting

If there is an incorrect LDAP setting on the metadata server, all administrative functions will be denied.

If there is an incorrect LDAP setting defined on the metadata server, the administrative agent will not be able to authenticate with the LDAP server, thereby rendering the system unusable. The cause of the fault is listed in the cimom.log file.

If an incorrect LDAP configuration renders the administrative agent unusable, you can reset the configuration using this procedure:

1. Log into the engine hosting the master metadata server.
2. Stop the CIMOM using the **stopCimom** command.
3. Open the recovery.properties file located in /usr/tank/admin/config. Create the file if it does not exist.
4. Enter the appropriate overrides, each on its own line with no spaces before or after the entry. The overrides are listed in Table 8 on page 72.
5. Restart the CIMOM using the **startCimom** command.
6. Use the **chldapconfig** command to reset the internal LDAP settings. `sfscli> chldapconfig -user cn=manager, o=sanfs`. Administrative interfaces will not be usable until the LDAP server is modified to match. [y/n]: y.
CMMNP5406I The LDAP configuration was modified successfully.
7. Remove the recovery.properties override file and restart the CIMOM using **stopCimom** and **startCimom** commands.
8. Edit the tank.properties file with the same information.

Table 8. LDAP configuration overrides

Parameters	Description	Example
LDAP_SERVER	LDAP server IP address	LDAP_SERVER=192.168.1.1
LDAP_USER	Distinguished name of an authorized LDAP user	LDAP_USER=cn=manager or o=sanfs
LDAP_PASSWD	Password of the authorized LDAP user.	LDAP_PASSWD=PASSWORD
LDAP_SECURED_CONNECTION	Does the LDAP server require SSL connections?	LDAP_SECURED_CONNECTION=false
LDAP_BASEDN_ROLES	Base distinguished name to search for roles.	LDAP_BASEDN_ROLES=ou=sfsroles, o=sanfs
LDAP_ROLEMEM_ID_ATTR	The attribute that holds the members of a role.	LDAP_ROLEMEM_ID_ATTR=roleOccupant
LDAP_USER_ID_ATTR	The attribute that holds the user ID.	LDAP_USER_ID_ATTR=uid
LDAP_ROLE_ID_ATTR	The attribute that holds the name of the role.	LDAP_ROLE_ID_ATTR=cn

Chapter 9. Troubleshooting the cluster

This topic provides an overview of how to resolve problems with the SAN File System cluster.

A SAN File System cluster can contain from two to eight engines, each running a separate instance of a metadata server. A metadata server has one of the following roles:

- **Master**

The master metadata server manages system metadata for the entire cluster. It controls all operations involving system metadata such as allocation of storage space, coordination of most administrative operations, and access to the global namespace. In addition, the master metadata server can perform the same tasks that are performed by subordinate metadata servers: managing file metadata and workload for one or more filesets.

Only one metadata server at a time can act as the master in a cluster.

- **Subordinate**

Subordinate servers manage user metadata and workload for one or more filesets.

Note: A fileset can be managed by only one metadata server.

To access the user data in a specific fileset, clients communicate with the metadata server that manages that fileset.

Metadata server failures

Metadata servers in the cluster rely on a heartbeat mechanism to verify availability. When a metadata server becomes unresponsive or fails, there are several possible reasons:

- Operating system crashes or hangs
- Metadata server hangs
- Local network connection fails
- Network partition occurs
- Hardware failure occurs on the metadata server

If the operating system crashes or hangs, the metadata server hangs, or a hardware failure occurs, the failed metadata server is truly dead. It can no longer access metadata for filesets that it served.

If a local network connection fails or a network partition occurs, all metadata servers can function until they need to communicate with the master. When a metadata server continues to function but is not reachable from the cluster, the metadata server could potentially cause metadata corruption if not stopped. Such a metadata server is referred to as a *rogue* metadata server.

The cluster must guarantee that no server is rogue for all failure scenarios. To do this, SAN File System uses one of two possible containment scenarios:

1. SAN File System attempts to abort the failed metadata server, resulting in a server core file that goes to `/usr/tank/server`. This method is the least

disruptive way to handle a rogue metadata server because only the failed metadata server, and not the engine, is affected.

2. If the abort fails, SAN File System metadata servers use the RSA II cards and the RS-485 network to stop the engine on which the failed metadata server is running. This method is the most disruptive because the entire engine stops. The results of executing this operation are logged to `/usr/tank/server/log/log.stopengine`.

When the engine stops, the RSA II automatically tries to restart it. If the RSA II cannot restart the engine, you must manually fix the fault (as you must in a hardware failure).

While the failed metadata server is down, the filesets originally served by the failed metadata server fail over to a surviving metadata server. These filesets become available again to clients after a brief pause, typically from one to two minutes.

Note: During this time, active operations of some applications might time out. Whether additional errors occur is based on how client applications respond to a timeout situation.

If the failed metadata server was the master, SAN File System automatically elects a new master. When the engine successfully restarts, the metadata server automatically restarts and rejoins the cluster if the automatic restart service is enabled.

Note: Some failure scenarios might result in the automatic restart service being disabled. If these failure scenarios occur, it requires manual intervention by the administrator to re-enable the automatic restart service. Once enabled, the metadata server automatically restarts and rejoins the cluster.

Any filesets that were statically assigned to the restarted metadata server fail back to that server when it rejoins the cluster.

Troubleshooting a metadata server

Use the information in this topic to troubleshoot problems that you are having with a metadata server.

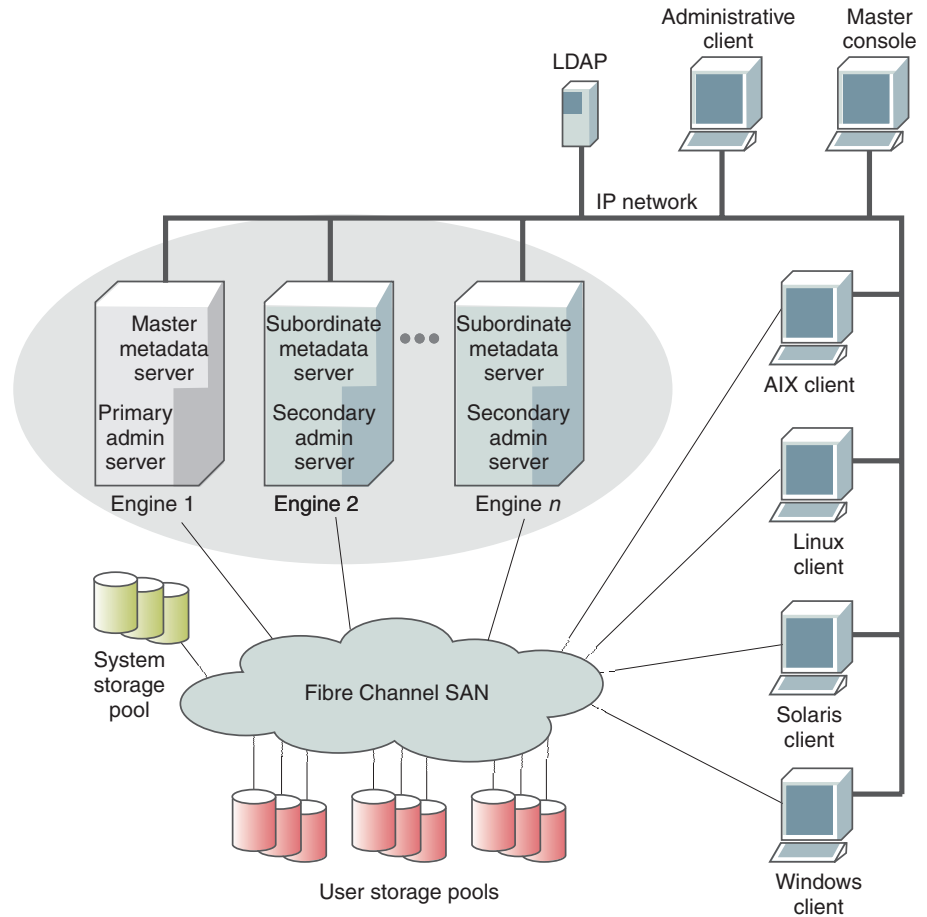
Problem

A metadata server fails with a hard fault or fails when auto-restart is disabled (for example, when the metadata server has not restarted). When a metadata server fails in this manner, SAN File System dynamically and automatically distributes the failed metadata server's filesets to one of the surviving metadata servers. If the metadata server that failed is the master, SAN File System automatically elects a new master allowing continued accessibility to the cluster. The client also pauses while the cluster reforms, a new master is detected, or filesets move. When it completes, clients continue to access file metadata as before.

A metadata server fails with a soft fault or fails when auto-restart is enabled (for example, when the metadata server restarts immediately). When a metadata server fails in this manner, there is no fileset movement and no mastership change. However, the client pauses briefly while the cluster reforms. Clients continue to access file metadata after the cluster reformation completes.

Investigation

If a metadata server fails



1. Use the SAN File System console or the administrative command-line interface (CLI) to view the status of the metadata server.
2. View the cluster message log to verify that SAN File System could not restart the metadata server. In addition, the messages in this log indicate a reason for the failure.
 - To view the cluster log from the master console, click **Monitor system** → **Cluster log**.
 - To view the cluster log from the CLI, enter this command on the master metadata server: `/usr/tank/admin/bin/sfscli catlog -log cluster -entries 25`.
3. Make sure that the metadata server is actually down and all restart attempts were unsuccessful. Use the `/usr/tank/admin/bin/sfscli lsautorestart` command to view the current state of the restart mechanism. If the Service state indicates "failed" and the Last Probe state of the metadata server is "absent," the auto-restart feature was unable to restart the metadata server.
 - a. Run the `/usr/tank/admin/bin/sfscli lsengine` command to verify that the engine hosting the metadata server is shut down.
 - b. Run the `/usr/tank/admin/bin/sfscli lsserver` command on the master metadata server to verify that the metadata server assigned the role of master is up and running.

4. Examine the RSA-II logs for the failed metadata server to find any hardware-related errors in the logs.

Note: The auto-restart feature might have restarted the server. If it did not restart automatically, continue with the following steps.

5. After repairing the failed metadata server, if the automatic restart service is enabled, the previously failed metadata server restarts automatically when you restart the metadata server. If the service is disabled, enable it to automatically restart the metadata server. The restarted metadata server comes up as a subordinate metadata server.
6. If there were any static filesets assigned to the restarted server, SAN File System automatically fails back the static filesets to the restarted server when it rejoins the cluster. Depending on the fileset load balance, SAN File System might also redistribute dynamic filesets.

Troubleshooting metadata server access to metadata

If a metadata server cannot access metadata LUNs, from the administrative command-line interface on the master metadata server, run the `/usr/tank/admin/bin/sfscli ls lun` command to verify that the LUNs are available. If the disks are not available, you can rediscover all disks by running the following commands from a metadata server:

1. Run the `/usr/tank/admin/bin/sfscli stopserver` command from the administrative command-line interface to stop a single server. This command might not work when trying to stop a server with an I/O problem.
2. Run the `rmmod qla2300` command.
3. Run the `modprobe qla2300` command.
4. Run the `/usr/tank/server/bin/device_init.sh` command to recreate raw devices needed by SAN File System.
5. Run the `/usr/tank/admin/bin/sfscli rediscoverluns` command from the administrative command-line interface.
6. Run the `/usr/tank/admin/bin/sfscli startserver` command from the administrative command-line interface to start the metadata server.

Troubleshooting the local network

Use the information in this topic to troubleshoot problems that you are having with the local network.

Note: For more information on metadata server failures, refer to *Troubleshooting the cluster*.

Problem

A problem exists with the local network on which the metadata servers communicate.

The problem might be a network fault or network partition:

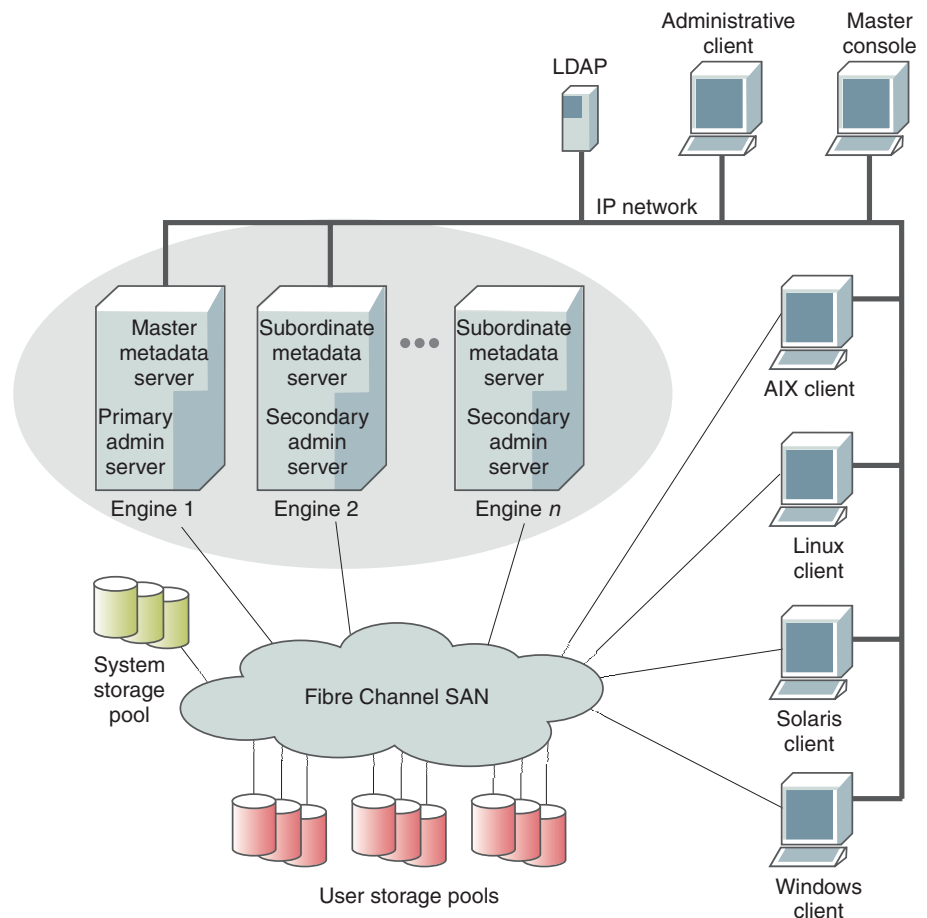
- **A network fault.** A local network fault can occur if there is a bad Ethernet adapter in an engine or the Ethernet cable is not connected between the Ethernet adapter and the IP network. In a local network fault, the cluster reacts as if the metadata server on which the fault occurred is down.

The master metadata server excludes the failed metadata server and reforms the cluster. The metadata server that was excluded is aborted, resulting in a server core file that goes to /usr/tank/server. If the abort fails, the RSA II stops and restarts the engine that is hosting the failed metadata server. Review the logs (log.std and log.stopengine) on the master and the log.std log on the failed metadata server.

- **A network partition.** A local network partition can occur if there is a problem in the Ethernet network that causes two or more metadata servers to lose communication with the master metadata server or with the rest of the cluster. Both sides of the partition attempt to take over, but the side of the partition that contains the majority of the metadata servers generally survives and reforms the cluster. If the partition results in an even number of metadata servers on each side of the partition, the side of the partition that contains the master survives and reforms the cluster. On the side of the partition that does not survive, the metadata servers are aborted or its engines are shutdown and restarted by the RSA II. Review the logs (log.std and log.stopengine) on the master of the surviving partition and the log.std log on all subordinate servers in the losing partition.

Investigation

If a local network fault occurs with one of the metadata servers, SAN File System performs the following actions to resolve the problem:

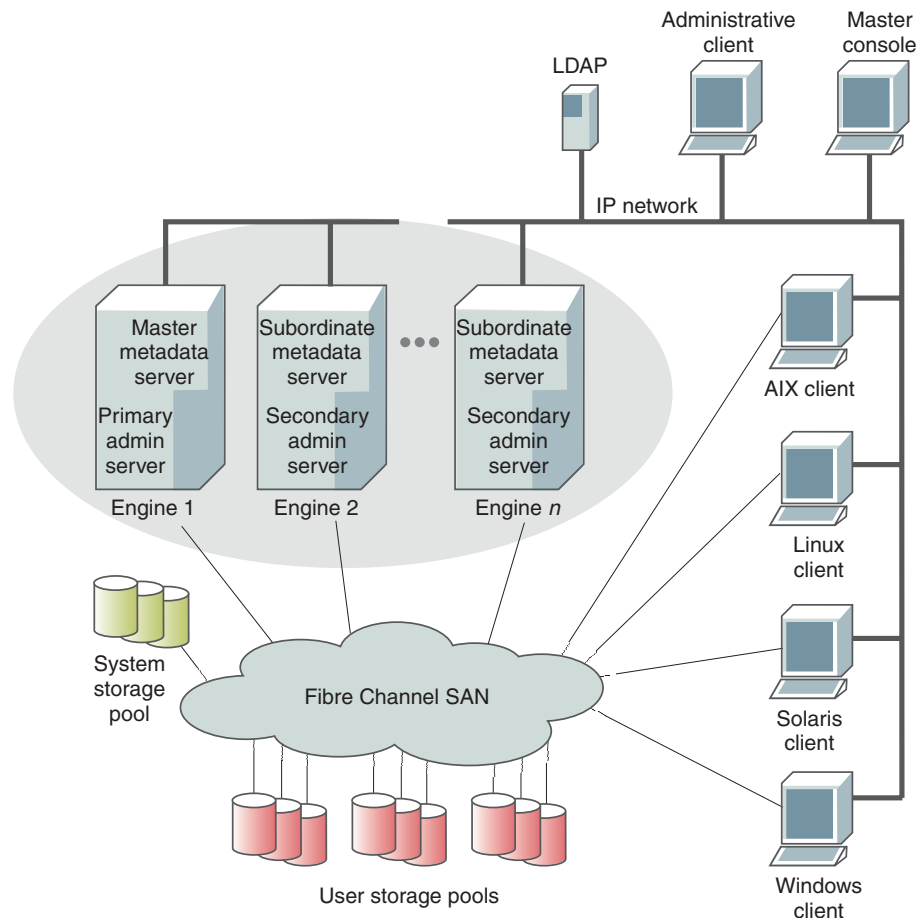


1. The master aborts the failed metadata server or, if the abort fails, the RSA II automatically shuts down and restarts the engine hosting the

failed metadata server. SAN File System fails over all filesets served by the failed subordinate metadata server to another metadata server.

2. After repairing the network fault, one of the following situations occurs:
 - a. After restarting, the failed metadata server attempts to restart and goes into Initializing state because it cannot communicate with the master. After the network partition is fixed, the metadata server completes the initialization and automatically rejoins the cluster.
 - b. If autorestart is disabled, run the following command from the command-line interface to restart the server:
`server:/usr/tank/admin/bin/sfsccli startautorestart`
 - c. Wait for the cluster to be reformed to include this metadata server.
 - d. Use the SAN File System console or the administrative CLI to verify that all metadata servers in the cluster are in an online state.
3. SAN File System automatically fails back any static filesets assigned to the restored metadata server.

If there is a network partition, SAN File System performs the following actions to resolve the problem:



1. The master attempts to abort the server. If the abort fails, the RSA II automatically shuts down and restarts the engine hosting the failed metadata server. SAN File System fails over all filesets served by the failed metadata server to another metadata server.
2. After repairing the network fault, one of the following situations occurs:

- a. After restarting, the failed metadata server attempts to restart and goes into Initializing state because it cannot communicate with the master. After the network partition is fixed, the metadata server completes the initialization and automatically rejoins the cluster.
- b. If autorestart is disabled, run the following command from the command-line interface to restart the server:
`server:/usr/tank/admin/bin/sfscli startautorestart`
- c. Wait for the cluster to be reformed to include this metadata server.
- d. Use the SAN File System console or the administrative CLI to verify that all metadata servers in the cluster are in an online state.

Resolution Procedures

This topic describes resolution procedures that can assist you in resolving problems with the metadata server.

You can use the procedures in this section to help you resolve problems with the metadata server. These procedures include:

- “Shutting down an engine from the RSA Web interface”
- “Taking a metadata server offline”
- Defining a new master metadata server
- “Reassigning filesets to metadata servers” on page 80
- “Bringing a metadata server online” on page 80

Shutting down an engine from the RSA Web interface

This task describes how to shut down an engine from the RSA Web interface.

1. Open a browser and point it to the URL for the RSA adapter that is located in the engine that you want to shut down.
2. From the Enter Network Password panel, type your RSA user name and password. Then click **OK**.
3. From the Welcome panel, select a session timeout value and click **Continue**.
4. From the left navigation pane of the System Health panel, click **Server** → **Tasks** → **Power/Restart**.
5. From the Power/Restart panel, Click **Power off server immediately**.
6. From the left navigation pane of the System Health panel, click **Sign off**.

Recovering from a lost RSA II adapter password

This topic provides the procedure for modifying the RSA II adapter settings when you have lost the password.

If you lose your RSA II password, and cannot access the adapter to make changes:

1. Log in to the master metadata server as root.
2. Start the Management Processor Command Line Interface.
3. Log in to the RSA II adapter using the Management Processor Command Line Interface.
4. Make the necessary changes to the RSA II configuration.

Taking a metadata server offline

This task describes how to stop a metadata server.

1. Go to the administrative command-line interface.

2. Run the **stopserver** command to stop the metadata server.

The metadata server has now stopped. SAN File System automatically redistributes the filesets of a failed or manually stopped metadata server and, if necessary, reassigns the master role to another metadata server in the cluster.

Reassigning filesets to metadata servers

This task describes how to change the assignment of a fileset to another metadata server.

1. Run **lsfileset** to list all of the filesets assigned to a specific metadata server.
`sfsccli lsfileset -server metadata_server_name`
2. Run **setfilesetserver**, specifying the new metadata server to which the filesets are to be assigned.
`setfilesetserver -server metadata_server_name fileset_name`

Note: You can assign multiple filesets to a metadata server from a single **setfilesetserver** command. However, you will need to run the **setfilesetserver** command for each metadata server to which you intend to assign filesets.

Changing a static fileset to a dynamic fileset

This topic describes how to change a fileset that is static to a metadata server dynamic fileset that is dynamically load balanced.metadata server

You must have Administrator privileges to perform this task.
Run the **autofilesetserver** command.

The following example converts two static filesets (*cnt_A* and *cnt_B*) to dynamic filesets:

```
sfsccli> autofilesetserver cnt_A cnt_B
Automatic server assignment for cnt_A is enabled.
Automatic server assignment for cnt_B is enabled.
```

Bringing a metadata server online

This task describes how to start a metadata server.

To start a metadata server, run the **startserver** command.

The metadata server is now started and managing filesets. Any static filesets assigned to that server, now fail back to it.

Recovering metadata servers in the “Not Running” or “Not added” state

This topic describes how to resolve the problem of a metadata server that is in the “Not running” or “Not added” state.

Use the following steps if, when you run the **sfsccli lsserver** command on the metadata server, it shows that any of the metadata servers are in either the “not running” or “not added” state:

1. Start or add the metadata server to the cluster using the appropriate **sfsccli** commands.

2. If the previous step does not apply to the situation or does not solve the problem, complete the following steps:
3. Stop all subordinate metadata servers in the cluster and the master using the **stopcluster** command.
4. On the master metadata server, determine if the Tank.Config and the Tank.Bootstrap files exist in the /usr/tank/server/config directory. If they exist, rename them for later use and recreate them by executing the following commands from the master metadata server.
 - a. Run the **Stop Cimom** command by entering `/usr/tank/admin/bin/stopCimom`.
 - b. Make sure no processes are running on the master, and run the **setupsfs -newserver** command by entering `/usr/tank/admin/bin/setupsfs -newserver`. This command creates a new Tank.Config file. The results of running the **lsserver** command should now indicate that the server is in the "not added subordinate" state.
 - c. Run the `sfscli stopserver master_server_name` command.
 - d. Change to the /usr/tank/server/bin directory.
 - e. To find the master disk, run the `tank lsdisklabel -device /dev/rvpathX` command. The disk with a Disk Type of "M" is the master.
 - f. Run the `tank extractbootrecord -device /dev/rvpathX` command, where /dev/rvpathX is the name of the master metadata LUN. This command creates a new Tank.Bootstrap file.
 - g. Run the `sfscli startserver master_server_name` command.
 - h. Run the `sfscli lsserver` command. The server is now the "online master."
5. From the master metadata server, start any subordinates by running the `sfscli startserver subordinate_server_name` command. At this point, the subordinate server might shutdown. If it does, it should restart automatically. If it does not restart automatically, start the server and run the **sfscli lsserver** command. The subordinate is now online. If it is not online, run the `setupsfs -newserver` command.

If there is no Tank.Config file in the /usr/tank/server/config directory of the subordinate, perform the following steps:

1. On the subordinate server, run the `setupTank -newserver` command.
2. On the master metadata server, run the `sfscli addserver -port cluster_port sub_IP_addr` command.

Adding a metadata server to the cluster

This topic describes how to add a metadata server to the cluster.

You must have Administrator or Operator privileges to perform this task.

Before adding the metadata server using the SAN File System console or the administrative CLI, you must have already started the metadata server using the **setupsfs -newserver** command.

The metadata server that you are adding must be in a state of "not added." The cluster to which you are adding the server must be online.

1. To add a metadata server using the SAN File System console:
 - a. Click **Manage Servers and Clients** → **Servers** from the My Work frame.
 - b. Click **Add a Server to Cluster** from the drop-down box in the table header.

- c. Click **Go**.
 - d. Enter the IP address of the metadata server in the IP Address field. If the cluster port is different than the default cluster port, you can append the port number to the IP address with a colon delimiter. For example, to specify an IP address of 9.67.101.202 and a port number of 1738, you can enter 9.67.101.202:1738.
 - e. Click **OK** to start the process.
2. To add a metadata server using the administrative CLI:
 - a. Add the metadata server to the cluster using the **addserver** command. For example: `addserver 10.30.30.104` . You should receive the following message: *metadata_server_name* was successfully added to the cluster.
 - b. Verify that the server is part of the cluster using the **lsserver** command.

Repairing metadata

This task describes how to repair metadata.

The metadata checker utility should be run in the following situations:

- Periodically, to validate the integrity of SAN File System metadata
 - After reverting a fileset to a FlashCopy image
 - Following the failure of a metadata server to ensure that no metadata corruption has occurred and to have the metadata checker utility attempt to repair any inconsistencies that are found
 - In response to error messages in the cluster log that indicate possible metadata inconsistency
 - In response to system behavior that implies possible metadata inconsistency
1. Use either the Administrative command-line interface or the SAN File System console to access the master metadata server.
 - a. If you use the Administrative command-line interface, run the **startmetadatacheck** command with the check option.
 - b. If you use the SAN File System console, click **Maintain system** → **Check metadata** to display the Check Metadata panel. Choose the type of metadata to check and the location of the metadata to be checked.
 2. After verifying the extent of the corruption, you can run the metadata checker utility again with the repair option specified.

Note: Before using the repair option, stop all applications that are using filesets that will be repaired. Applications may encounter errors during the salvage process, and you might have to recover some data from backup depending on what the metadata checker salvages.

Metadata server does not start and no master disk is found in the standard log

This topic explains how to identify when lost logical unit numbers (LUN) are preventing the metadata server from starting correctly.

This problem can occur in situations such as a recent complete power failure, and when the metadata LUNs were not available at the time that the metadata server started. Use the following steps if you attempted to start the metadata server using the **startserver** command, and the metadata server did not start:

1. View log.std to determine the cause of the problem.

2. Search for a message that reads, “**No master disk found.**” If you find the message, continue with the next step. Otherwise, this is not the problem.
3. Search for a message that begins, “Tank.Bootstrap file does not match.” If you find the message, continue with the next step. Otherwise, this is not the problem.
4. Make sure that you have Subsystem Device Driver installed on the metadata server.
5. Run the **lsvpcfg** command to determine if you have the required LUNs. If no LUNs are listed or the list is incomplete, continue with the next step.
6. Restart the system to find any available storage LUNs.
7. Contact your support representative if the LUNs are still not available.

Backing up system metadata

This task describes how to back up system metadata.

1. From the Administrative command-line interface, create a new system-metadata disaster-recovery dump file.

```
sfsccli> mkdrfile dr_file_1
```

2. Verify that the system-metadata disaster-recovery file was created.

```
sfsccli> lsdrfile
```

3. Exit the Administrative command-line interface and change directories to the directory where the recovery files are stored. List the files in this directory to verify that you have at least four files.

```
sfsccli>exit
# cd /usr/tank/server/DR
# ls
```

4. Save these files with your normal file data backup procedures. These files are used to recover system metadata as part of the disaster recovery procedures. In addition, you should also back up the `/usr/tank/server/config/Tank.Config` and `/usr/tank/server/config/Tank.Bootstrap` files from the master.

Chapter 10. Troubleshooting a SAN File System client

This topic provides an overview of how to resolve problems with SAN File System clients.

SAN File System supports clients running on these operating systems:

- AIX 5.1 (32-bit only)
- AIX 5.2 (32-bit and 64-bit)
- AIX 5.3 (32-bit and 64-bit)
- Red Hat Enterprise Linux Advanced Server 3.0 running either SMP or Hugesem kernels in WS, ES or AS editions
- SUSE Linux Enterprise Server 8 (32-bit)
- Sun Solaris 9 (64-bit)

SAN File System supports clients that run on these Windows operating systems:

- Windows 2000 Advanced Server
- Windows 2000 Server
- Windows 2003, Standard Edition
- Windows 2003, Enterprise Edition

The flexible SAN environment can be set up to allow clients to access only the storage pools assigned to it. Otherwise, all of the clients can be set up to access the same global namespace. File permissions are based on the operating system on which the files were created. The security features of the operating system on which the file was created are available and enforced. Across platforms, the following security rules apply:

- When files created by a UNIX client are accessed by a Windows client, access is controlled by the permissions of *Other*.
- When files created by a Windows client are accessed by a UNIX client, access is controlled by the permissions of *Everyone*.

Use the following topics to determine problem areas within SAN File System clients:

- "Troubleshooting client access to data"
- "Troubleshooting client performance problems" on page 89

Troubleshooting client access to data

Use the information in this topic to troubleshoot problems that you are having with client access to data.

Problem

This topic describes how to determine why a client cannot access or update user data.

Investigation

If a client cannot create a new file

Perform the following steps until you resolve the problem:

1. Verify that the master metadata server is online.
2. Verify that the client is connected to the master metadata server:

- From the client, attempt to ping or establish a Secure SHell (SSH) session with the metadata server.
 - From the Administrative command-line interface (CLI), run the **Isclient** command to see a list of all clients currently being served by the metadata servers in the cluster.
3. Use the **ls -l** command to list the directory in which the file is to be created to verify that the path is correct and accessible.
 4. Verify that you are logged into the client with a user name that has permission to write files to directories:
 - From a UNIX client, use the **id** command.
 - From a Windows client, right-click the directory. Then click **Properties** → **Security** .
 5. Verify that your user name has permission to write files to the specific directory.

Important: If you are logged into a UNIX client as root or a Windows client as Administrator, make sure that you are running on a privileged client.

6. Make sure you have space in your system pool.
7. If using hard quotas, check the quota of the fileset in which the parent directory resides to ensure that there is sufficient space to accommodate the new file and that the fileset is attached. If allocating blocks fails, check the quota. When it meets the hard quota, the client cannot allocate blocks for a file until some of the old blocks are free.
 - a. Access the master metadata server through either the SAN File System console or the CLI.
 - b. List the filesets to view the server to which the fileset is attached, the quota percentage and type, the attach point, and the directory:
 - From the CLI, run the **sfscli lsfileset -l** command.
 - From the SAN File System console, click **Manage Filing** → **Filesets**.
8. Make sure that the storage pool in which the file is stored has sufficient space. Check the active policy set to validate the placement rule that applies to your filename.
9. Verify that the client can access the storage device:
 - If the client is running Data Path Optimizer (DPO), use the **datapath query** command to ensure that the operating system can access the storage device.
 - On an AIX client, use the **stfsdisk** command to determine which LUNs can be accessed. Then, use the **lsvol -l** command to correlate the LUN with the device. You can also use the **lsdev -C** command to see the physical LUNs.
 - On a Linux client, enter **cat /proc/fs/sanfs/client/<client name>/disk/access** to determine which LUNs you can access. Then use the **lsvol -l** command to correlate the LUN with the device. Also, check the System Log for any messages about inaccessible LUNs and volumes.
10. Suspect a problem with corrupt SAN File System metadata.

If a client cannot access an existing file

Perform the following steps until you resolve the problem:

1. Verify that your user name has permission to read that specific file.

2. Verify that the master MDS is online.
3. Verify that the client has access to the cluster using SSH and the **Isclient** command.
4. Run the **Isclient** command on the server fileset to ensure that the client lease is valid with the server.
5. If the client is trying to access root-privileged files, ensure that the client has administrative privileges (AdmPriv) on the server.
6. Suspect a problem with corrupt SAN File System metadata.

If a client cannot update the attributes of an existing file

Perform the following steps until you resolve the problem:

1. Verify that your user name has permission to write to the file.
2. On a Windows client, you can change attributes on Windows domain files only. On a UNIX client, you can change attributes on UNIX domain files only.
3. Verify that the master MDS is online.
4. Verify that the client has access to the cluster using SSH and the **Isclient** command.
5. Run the **Isclient** command on the server fileset to ensure that the client lease is valid with the server.
6. If the client is trying to access root-privileged files, ensure that the client has administrative privileges (AdmPriv) on the server.
7. Make sure you are a local root superuser or a Windows administrator. If you are and the client does not have administrative privileges, you might not be able to update the attributes of an existing file.
8. Suspect a problem with corrupt SAN File System metadata.

If a client cannot access any data

Perform the following steps until the problem is resolved:

1. Verify that the client can access the cluster:

From the client, sign on to the SAN File System console. If you cannot sign on, suspect one of the following conditions:

- One or more metadata servers in the cluster are down. See Chapter 9, “Troubleshooting the cluster,” on page 73.

Tip: The administrative agent automatically attempts to restart a metadata server if it goes down for any reason. Therefore, if you cannot access a metadata server, wait a few minutes and try again to ensure that it is not in the process of restarting.

- An IP network problem occurred between the client and the cluster. See Chapter 5, “Isolating problems with the SAN File System,” on page 33.

View the system logs and look for errors that might indicate I/O errors. Attempt to resolve these errors.

- From a client running Windows, use the Event Viewer to view the Event Log or the use the **sanfstrace** utility to determine errors.
- From a client running on any supported UNIX platform, you can view logging with the **syslog** facility and tracing output using the **sanfstrace** utility.

2. From the CLI on the master metadata server, run the **lslun** command to verify that the LUNs are available. To see the LUNs that are visible to each client, use the command `lslun -client <client_name>` for each client.

If the LUNs are not available, you can rediscover all LUNs by running the following commands from the master metadata server:

- a. Run the **stopserver** command from the CLI to stop SAN File System.
 - b. Run the **rmmod qla2300** command.
 - c. Run the **modprobe qla2300** command.
 - d. Run the **/etc/rc.d/init.d/sanfs start** command to start SAN File System.
 - e. Run the **startserver** command from the CLI to start the metadata server.
3. Run the **thelsclient** command on the server to see if this client is recognized and has an active session with the server.
 4. Verify that the client can access the storage device:
 - If the client is running DPO, use the **datapath query device** command to ensure that the operating system can access the storage device.
 - On a client running AIX, use the **stfsdisk** command to determine which disks can be accessed. Then, from the CLI, use the **lsvol -l** command to correlate the disk with the device. You can also use the **lsdev -C** command to see the physical disks.
 - On a Linux client, enter `cat /proc/fs/sanfs/client/<client name>/disk/access` to determine which LUNs you can access. Then, run the **lsvol -l** command from the CLI to correlate the LUN with the device.

To rediscover all LUNs on a client running AIX, run the client command **stfsdisk -discover**.

To rediscover SAN File System LUNs on a Linux client, run the following command: `$ echo 1 > /proc/fs/sanfs/client/<client name>/disk/discover`

The LUNs should automatically be rediscovered on a client running Windows. If they are not, use the following steps:

- a. Right-click **My Computer**.
- b. Click **Manage**.
- c. Click **Storage** → **Disk Management**.
- d. Click **Action** → **Rescan Disks**.
- e. Restart clients.
 - For clients running on UNIX, using the following steps:
 - a. Run the **rmstclient** command to unmount the global namespace, remove the virtual client, and unload the file-system driver.
 - b. Run the **setupclient** command to load the file-system driver, create the virtual client, and mount the global namespace.
 - On clients running Windows, restart the system.
5. If the client cannot access user data, suspect the SAN route from the client. See Chapter 5, "Isolating problems with the SAN File System," on page 33.
6. Suspect a problem with corrupt SAN File System metadata.

If a client running Windows receives delayed write failure errors

A delayed write failure error might appear as a message box on the client desktop or in the Event Log. This message indicates that an error occurred when writing data from the local file system cache to a storage device. Perform the following steps until you resolve the problem:

1. The message includes the name of the file where the error occurred. Note the name of this file because the data it contains might have been corrupted, and the application using this file might encounter problems using this file's data.
2. If the file is not part of SAN File System, refer to your system documentation for resolving file system errors. If the file is part of SAN File System, view the Event Log and resolve errors that might relate to this problem.
3. Suspect a communication problem with either the SAN or between the client and the metadata server.

If a client running UNIX receives delayed write failure errors

A delayed write-failure error appears in the System Log. These errors are usually preceded in the log by SCSI or other input/output errors.

Troubleshooting client performance problems

Use the information in this topic to troubleshoot problems that you are having with client performance.

Problem

A client is timing out or taking an unusually long time to access data.

Perform the following steps to resolve the problem:

1. Verify that the metadata server managing the fileset being accessed is active. List the filesets to view the server to which the fileset being accessed is attached, the quota percentage and type, the attach point, and the directory.
 - From the administrative command-line interface (CLI), run the **sfscli lsfileset** command.
 - From the SAN File System console, click **Manage Servers and Clients** → **Filesets**.
2. Verify that the master metadata server is active. Attempt to access the master metadata server through either the SAN File System console or the administrative command-line interface. If you cannot, suspect one of the following:
 - The master metadata server is down. See Chapter 9, "Troubleshooting the cluster," on page 73.

Tip: The administrative agent automatically attempts to restart a metadata server that fails for any reason. Therefore, if you cannot access a metadata server, wait a few minutes and try again to ensure that it is not in the process of restarting.

- There is an IP network problem between the client and the cluster. See Chapter 5, "Isolating problems with the SAN File System," on page 33.

Chapter 11. Troubleshooting the installation

This topic describes technical problems that might occur during the installation of SAN File System.

The following topics describe the problems and provide solutions that help you troubleshoot the installation process.

Finding and correcting problems

This topic describes items to check to determine and correct problems in SAN File System.

1. In the `tank.properties` files, the `RSA_USER` and `RSA_PASSWD` entries must be the same on all servers. If the values do not match, correct the values using **vi editor** and save the file. Be careful not to space after the last letter.
2. Run the following commands: **stopCimom, startCimom, stopConsole, startConsole**. These commands are located in `/usr/tank/admin/bin`.
3. Retry the **tanktool lsengine** command to see if connectivity is restored. The default values are: `RSA_USER=USERID` and `RSA_PASSWD=PASSWORD` (zero not the letter O). You can change the values, but make the changes on all servers.
4. If the `RSA_USER` and `RSA_PASSWD` information match on all servers, try to log into each RSA card user interface using the userid and password in the `tank.properties` files. From the Configuration File of each RSA card, you can view the Login Profile entries to see if there is an entry to match the data in the `tank.properties` files.
5. If you cannot log in to a server or find a matching entry, add a new login profile.
6. As a test, log out of the RSA user interface and back in using the new Login information.
7. Run the following commands: **stopCimom, startCimom, stopConsole, startConsole**. These commands are located in `/usr/tank/admin/bin`.
8. Retry the **tanktool lsengine** command to see if connectivity is restored.

Note: Multiple Login profiles can exist, but at least one login profile must be the same on all servers; and each must match what is in all the `tank.properties` files. The Host operating system entry in the Configuration Summary File must say Linux. If it does not, the In Boot, Booting OS, and/or Wrong OS errors might appear and the `ibmasm` daemon might not start.

9. To correct, in the left panel, select Server->ASM Control->System Settings->ASM Information.
10. Select **Linux** from the Host operating system drop-down menu.
11. In the left panel, select **Restart ASM**.
12. Run the following commands: **stopCimom, startCimom, stopConsole, startConsole**. These commands are located in `/usr/tank/admin/bin`.
13. Retry the **tanktool lsengine** command to see if connectivity is restored.
14. If the command still gives an error, reboot the metadata server to start/restart the ASM daemon on the RSA card.

15. Verify that the RSA cabling and configuration is correct. You can verify the information using the RSA Configuration Summary File. Refer to the *SAN File System Planning, Installation, and Configuration Guide* for more information.

Supplying power to metadata server engines

This topic describes information about supplying power to metadata server engines.

Problem: Because the external Remote Supervisory Adapter II (RSA) power is not required to ensure SAN File System metadata server redundancy, you must ensure that the power supply units for each metadata servers' engine are connected to separate and independent power supply circuits.

Solution:To properly install and configure SAN File System, each power supply within a metadata server's engine must have a separate and independent power circuit. If servers are connected in this manner, there cannot be a single point of failure in the configuration and no external RSA II power is necessary. The external RSA II power supply is not a SAN File System requirement for high availability. If you want additional power-supply redundancy, use a third independent circuit for the RSA II. You can order an RSA II power supply in addition to SAN File System to connect the RSA II and the third independent circuit. For additional instructions on installing SAN File System metadata server engines, refer to the *SAN File System Planning, Installation, and Configuration Guide*.

Chapter 12. Troubleshooting the master console

This topic provides an overview of how to resolve problems with the master console.

The master console for the SAN File System is an IBM e(*logo*)server xSeries® 305 Type 8673 Model RA1 and provides support for Call Home and Remote Access.

Note: A Virtual Private Network (VPN) connection is required for Remote Access functionality. In addition, a remote display emulation package, such as Virtual Networking Computer (VNC), must be installed to access the SAN File System console or the RSA II Web interface.

The following software is used on the master console:

- Microsoft Windows 2000 Advanced Server edition

Note: The optional Simple Network Management Protocol (SNMP) extensions should not be installed or, if they are installed, the SNMP Trap service must be disabled.

- IBM Director Server, version 4.1
- IBM Tivoli Bonus Pack for SAN Management
- Adobe Acrobat, version 5.0
- The PuTTY openssh package

For troubleshooting information, refer to the *IBM e(*logo*)server xSeries® 305 Hardware Maintenance Manual and Troubleshooting Guide*. In addition, you can use the documentation that is available with each of the software packages loaded on the master console to troubleshoot software problems.

Resolution procedures

This topic describes resolution procedures that can assist you in resolving problems with the master console.

You can use the procedures in this section to help you resolve problems with the master console. These procedures include:

- “Performing a total software recovery”
- “Recovering a hard disk drive” on page 94
- “Replacing Fibre Channel cable and GBICs” on page 94

Performing a total software recovery

This task describes how to recover the master console software.

1. Power OFF the master console.
2. Insert recovery CD 1.
3. Power ON the master console and follow the on-screen instructions.
4. Check each software package, updating to the latest level where required. Use the supplied CD or download the particular software package from the Web site.

The master console software is now reset to manufacturing default settings. Refer to the *SAN File System Planning and Installation Guide* as well as the customer installation worksheets to update the master console to the current settings.

Recovering a hard disk drive

Use the information in this topic to recover a hard disk drive that has failed.

1. Right-click the **My Computer** icon on your desktop and select **Manage**.
2. Select **Disk Management**. The hard drives are displayed in the right panel.
3. If the failing disk drive is displayed, right-click the main volume of the drive and select **Break Mirror**.

Note: The mirror might have already been broken.

4. Shut down the master console and replace the failing disk drive using the procedures detailed in the xSeries 305 service documentation. Ensure that the new drive has its jumpers set the same as the drive that is being replaced. The new drive must be the same capacity or larger than the drive being replaced.

Note:

- a. It might not be obvious which of the two drives has failed. In this case, reboot with each drive connected in turn to isolate the failed drive.
 - b. If the replacement drive has a boot record present, erase it prior to use.
 - c. If the master console fails to boot because the Boot Record cannot be found, change the boot sequence in the BIOS to the other hard drive.
5. Disconnect the fibre-channel cables from the master console, making note of where they were connected.
 6. Restart the master console.
 7. Right-click the **My Computer** icon on your desktop and select **Manage**.
 8. Select **Disk Management**. The hard drives are displayed in the right panel.
 9. If a disk drive is displayed in the list marked "Missing," remove it by right-clicking the drive and selecting **Remove Disk**.
 10. If the new disk drive has a "no entry sign" displayed on it, right-click it and select **Write Signature** to remove the "no entry sign."
 11. Right-click the new disk drive and select **Upgrade to Dynamic Disk**.
 12. Right-click the volume that you want to mirror and select **Add Mirror**. This step starts the Add Mirror Wizard.
 13. Use the dialog boxes displayed to configure the second volume.
 14. A dialogue box with reference to making changes to the boot.ini file is displayed. You can safely ignore this dialog box.
 15. The status of both volumes, the existing drive, and the new drive will change to "Regenerating" and will, after a short period of time, start to show the percentage of regeneration completed. When the regeneration completes, the status changes to "Healthy."
 16. Reconnect the fibre-channel cables to the master console.

Replacing Fibre Channel cable and GBICs

Use the information in this topic to replace a Fibre Channel cable or Gigabit Interface Converter (GBIC).

1. Disconnect each end of the suspected failing Fibre Channel cable.
2. Fit a replacement Fibre Channel cable.
3. Check out the repair.
 - a. If the repair fixes the problem:
 - 1) Ensure that labels are fitted to each end of the new Fibre Channel cable with the same information that was on the original Fibre Channel cable.
 - 2) Follow customer procedures for the safe disposal of the original Fibre Channel cable.
 - b. If the repair does not fix the problem, remove the new Fibre Channel cable and reconnect the original Fibre Channel cable.
4. Replace the GBICs on each end of the failing link, one at a time, and checking to see if the problem is resolved with each replacement. If a new GBIC does not resolve the problem, refit the original GBIC.

Chapter 13. Managing disaster recovery

The SAN File System console enables you to create and delete files to assist in disaster recovery. In addition, there are several disaster recovery tasks that can be performed from the Administrative command-line interface.

Several of the restoring tasks depend on having first executed certain backup tasks. The method of restoration will depend a great deal upon whether you chose the LUN or API method of backup.

Creating a recovery file

This topic describes how to create a file for disaster recovery.

You must have Operator or Administrator privileges to perform this task.

1. Click **Maintain System** → **Disaster Recovery** from the My Work frame.
2. Click **Create** from the drop-down box in the table header.
3. Click **Go**.
4. Select the check box to **Create a new recovery file** or select the check box for a **Forced Create**, which will overwrite an existing file.

Attention: When you overwrite an existing file, metadata recovery of items from that file may not be possible.

5. Type a name for the newly created file, or select the **Existing Recovery File** to overwrite from the drop-down menu.
6. Click **OK** to confirm the creation of the new file or to overwrite the existing one.
7. Click **Maintain System** → **Disaster Recovery** from the My Work frame to verify that the recovery file was created.

Creating recovery scripts

This topic describes how to create recovery scripts from a recovery file. The recovery file and these scripts are required for restoring SAN File System metadata.

You must have Backup, Operator, or Administrator privileges to perform this task.

Note: The output for this command is written to the `/usr/tank/server/DR` directory. This command will overwrite any files that were created by a previous run of this command. If you want to preserve the existing files, copy them to another directory.

1. Use the **bulddrscript** command and specify the name of the recovery file, which you created using the **Maintain System** task in SAN File System console or the **mkdrfile** command in the administrative command-line interface:

```
sfsccli bulddrscript recovery-file-name
```

2. The **bulddrscript** command stores the recovery scripts in the `/usr/tank/server/DR` directory on the master metadata server. See “Restoring SAN File System metadata” on page 101 for information about editing and running these scripts to restore SAN File System metadata.

Deleting a recovery file

This topic describes how to delete a disaster recovery file.

You must have Operator or Administrator privileges to perform this task.

1. Click **Maintain System** → **Disaster Recovery** from the My Work frame.
2. Select a recovery file for deletion.
3. Click **Delete** from the drop-down box in the table header.
4. Click **Go**.

Attention: When you delete an existing recovery file, metadata recovery of items from that file may not be possible.

5. Click **Delete** to confirm the file deletion.

Listing recovery files

This topic describes how to display a list of all recovery files.

To display a list of all recovery files, click **Maintain System** → **Disaster Recovery** from the My Work frame.

Restoring the master console

This topic explains how to restore the hardware and the operating system for the master console.

1. Determine if the hardware for the master console is working properly. If so, review information about recovering the hard drives (if necessary) as well as recovering the software.
2. If the hardware for the master console is not working properly,
 - a. Refer to your server documentation to resolve problems with the hardware.
 - b. Refer to the *Planning, Installation and Configuration Guide* for information about installing the master console.

Restoring the engine hardware and operating system

This topic explains how to restore the hardware and the operating system for an engine.

1. Verify that there is no damage to the hardware and that the engine boots properly. If you suspect a problem with any of the hardware components, troubleshoot an engine to resolve the problem.
2. Verify that there is no damage to the master console and that it boots properly.
 - a. If you suspect a problem with any of the hardware components in the master console, refer to your server documentation to resolve those problems.
 - b. If you suspect a problem with the software or the hard disk drive, troubleshoot the master console to resolve the problem.
3. From the master console, point the Web browser to the URL of the RSA II adapter on the engine and access the RSA II adapter to set up a remote console to the engine. This interface allows you to use the master console as your display and keyboard for the engine.

Note: Instead of using the RSA II Web interface from the master console, you can directly attach a keyboard and display to the engine. However, make

sure that you attach the display to the VGA port of the RSA II card on the engine and not to the video port on the engine itself.

4. Determine if the boot drives for each engine hosting a metadata server still have an intact SAN File System configuration and executable files (undamaged).
5. If there are corrupt or damaged configuration and executable files, attempt to recover the damaged files from the mirrored boot drive. If you cannot recover the damaged files from the mirrored boot drive:
 - a. Load the Disaster Recovery CD into the CD-ROM drive on the engine.
 - b. Reboot the engine using one of the following methods:
 - 1) Open a bash shell prompt and enter **init 6**.
 - 2) Press the Reset button on the front panel of the engine.
 - 3) Power off the engine and then power it back on.
 - c. When you receive a warning prompt that the entire hard drive will be overwritten, respond by entering **y**.
 - d. After the operating system has been reloaded, the engine will eject the Disaster Recovery CD and automatically reboot.

Restoring SAN connectivity

This topic explains how to restore connectivity between the SAN File System and the SAN.

1. If the system was backed up using the LUN method, perform these steps on each engine in the SAN File System cluster to restore SAN connectivity:
 - a. Verify that the engines hosting the metadata servers are connected to the SAN in the same configuration that existed at the point of the last backup operation (make sure the metadata servers can see the same LUNs that existed prior to the unexpected outage).
 - b. If the LUN mapping has changed, use the device management tools for the storage subsystem or management tools for the SAN to recreate the old LUN map. After creating the old LUN map, reboot the metadata server so that the changes to the LUN map are visible to the metadata server.
 - c. If LUN contents were lost or corrupted, use the copy services facility of the storage subsystem to restore all LUN data (both metadata and user file data).
2. If the system was backed up using the API method, perform these steps on each engine in the SAN File System cluster to restore SAN connectivity.
 - a. Verify that the engines hosting the metadata servers are connected to the SAN in the same configuration that existed at the point of the last backup operation (make sure the metadata servers can see the same LUNs that existed prior to the unexpected outage).
 - b. If the LUN mapping has changed, use the device management tools for the storage subsystem or management tools for the SAN to recreate the old LUN map. You can also choose to restore data onto a new LUN map. However, if you do so, you will have to manually run some of the steps used to restore metadata.

Restoring SAN File System software

This topic explains how to restore the metadata server and administrative server software on an engine.

1. Reinstall the software for the metadata server.
 - a. Make sure that you are logged into the engine as root.
 - b. From a shell prompt on the engine, change to the directory where the metadata server software package is installed.

```
cd /usr/tank/packages
```
 - c. Install the metadata server software package using the following command:

```
rpm -ivh metadata_server_package_name.rpm
```
2. Reinstall the software package for the administrative server.
 - a. Install the administrative server software package using the following command:

```
rpm -ivh administrative_server_package_name.rpm
```

Restoring SAN File System cluster configuration

This topic explains how to restore the configuration for the SAN File System cluster.

1. If the system was backed up using the LUN method, and the entire cluster is down, perform these steps to restore the cluster configuration information:
 - a. If you have previously saved the configuration files to another location, copy these files onto the boot drive for the engine.
 - 1) Copy Tank.Bootstrap to /usr/tank/server/config.
 - 2) Copy Tank.Config to /usr/tank/server/config.

Note: If you have saved any other administrative configuration files, you can reference them when restoring the SAN File System metadata configuration.
 - b. If the cluster bootstrap file, Tank.Bootstrap, is corrupted or missing, you can attempt to recreate the contents of that file using information from the metadata LUNs:
 - 1) Use the `/usr/tank/server/bin/tank lsdisklabel -device` command to find the master volume. If you cannot remember which device is your master volume, this is an iterative process of searching all suspected master volume devices until the command indicates you have found a valid master volume.
 - 2) Use the `/usr/tank/server/bin/tank extractbootrecord` command to regenerate Tank.Bootstrap from the master volume.
 - 3) Use the `/usr/tank/server/bin/tank resetcluster` command to reinitialize the master volume for subsequent rebuilding of the cluster configuration.
 - 4) Use the `/usr/tank/server/bin/tank addserver` command for all subordinate metadata server engines to recreate the cluster definition.
2. If the system was backed up using the LUN method, and only the master metadata server is down, perform these steps to restore the cluster configuration information:
 - a. If you have previously saved the configuration files to another location, copy these files onto the boot drive for the engine.
 - 1) Copy Tank.Bootstrap to /usr/tank/server/config.

- 2) Copy Tank.Config to /usr/tank/server/config.

Note: If you have saved any other administrative configuration files, you can reference them when restoring the SAN File System metadata configuration.

- b. If the cluster bootstrap file, Tank.Bootstrap, is corrupted or missing, you can attempt to recreate the contents of that file using information from the metadata LUNs:
 - 1) Use the `/usr/tank/server/bin/tank lsdisklabel -device` command to find the master volume. If you cannot remember which device is your master volume, this is an iterative process of searching all suspected master volume devices until the command indicates you have found a valid master volume.
 - 2) Use the `/usr/tank/server/bin/tank extractbootrecord` command to regenerate Tank.Bootstrap from the master volume.
 - 3) Use the `/usr/tank/server/bin/tank resetcluster` command to reinitialize the master volume for subsequent rebuilding of the cluster configuration.
 - 4) Use the `/usr/tank/server/bin/tank addserver` command to add the original master metadata server to the cluster.
- c. Use the `/usr/tank/server/bin/tank resetcluster` command to reinitialize the master volume for subsequent rebuilding of the cluster configuration.
3. If the system was backed up using the API method, perform these steps to restore the cluster configuration information:
 - a. If you have previously saved the configuration files to another location, copy these files onto the boot drive for the engine.
 - 1) Copy Tank.Bootstrap to /usr/tank/server/config.
 - 2) Copy Tank.Config to /usr/tank/server/config.

Note: If you have saved any other administrative configuration files, you can reference them when restoring the SAN File System metadata configuration.

- b. If you suspect that the metadata LUNs are corrupted, you can perform these steps to recreate the cluster definition:
 - 1) Delete all Tank.Bootstrap and Tank.Config files from your metadata server engines.
 - 2) Start the `/usr/tank/server/bin/tank binary` on your master metadata server with the `install` option rather than `normal` option.

Attention: The existing metadata data server information will be overwritten.
This will create new Tank.Bootstrap and Tank.Config files on your master metadata server. Be sure to specify the same cluster name that was used prior to the disaster:
 - 3) Now start the master metadata server with `/usr/tank/server/bin/tank normal` command.
 - 4) Use the `addserver` command to add all subordinate metadata server engines. This will create new Tank.Bootstrap and Tank.Config files on the subordinates.

Restoring SAN File System metadata

This topic explains how to restore the metadata for the SAN File System cluster.

1. Verify that all metadata servers in the cluster are online and that the cluster is running.

```
sfsccli lsserver -state online
```

2. Copy the system-metadata disaster-recovery file (and the scripts) that you had previously backed up to /usr/tank/server/DR on the master metadata server.

3. Use the TankSysCLI.auto script:

- a. Edit the script TankSysCLI.auto for information about how the script is used and any changes that might need to be made to the script.

```
#####
# CLI Commands to create Storage Pools, Filesets, Service Classes and
# Policy Sets.
# These commands need NO manual intervention.
#####
```

- b. Run the script TankSysCLI.auto.

```
sfsccli -script /usr/tank/server/DR/TankSysCLI.auto
```

- c. If any errors occur while running the script, ensure that you resolve those errors before continuing.

4. Use the TankSysCLI.volume script:

- a. Edit /usr/tank/server/DR/TankSysCLI.volume and modify it to match your current SAN settings. It also contains usage information as well as information about any changes that might need to be made to the script.

```
#####
# CLI Commands to add Volumes to Storage pools.
# These commands need manual intervention.
# The device names were as they appeared during backup.
# Please make sure that the device names appearing here actually
# exist and have correct sizes and if not edit the device names to
# correct values.
# The System MASTER volume has to be specified in tank install command
# and therefore has no corresponding CLI.
# The other System Volumes can either be specified in tank install
# command, or, added using the CLI command, which appears inside comments
# for this reason.
#####
```

- b. Run the script TankSysCLI.volume.

```
sfsccli -script /usr/tank/server/DR/TankSysCLI.volume
```

- c. If any errors occur while running the script, ensure that you resolve those errors before continuing.

5. Use the TankSysCLI.attachpoint script:

- a. Edit /usr/tank/server/DR/TankSysCLI.attachpoint to verify the settings. It also contains usage information as well as information about any changes that may need to be made to the script.

```
#####
# CLI Commands to attach filesets.
# These commands need manual intervention.
# All the "mkdir" and "attachfileset" commands should be run in the
# order given.
# The "mkdir" command should be run on a client to recreate the directory
# path before running the following attachfileset CLI commands.
#####
```

- b. If all filesets are attached only to the root directories of other filesets, run the script TankSysCLI.attachpoint.

```
sfsccli -script /usr/tank/server/DR/TankSysCLI.attachpoint
```

Note: If you have any filesets attached to directories, you must reattach them manually.

- c. If any errors occur while running the script, ensure that you resolve those errors before continuing.
6. Grant privileges to those clients that require root or Administrator access to SAN File System using the **chclusterconfig -privclient** command.

Restoring SAN File System clients

This topic explains how to restore SAN File System clients.

SAN File System clients are access points to the SAN File System. Therefore, clients are not backed up. To restore SAN File System clients, you can perform the normal client installation procedure, which is described in the *Planning, Installation, and Configuration Guide*.

Restoring SAN File System user data

This topic explains how to restore SAN File System user data.

1. From a client, mount the SAN File System at its usual mount point. The top of the subdirectory tree (the portion of the subdirectory tree that consists of the fileset names) should be visible from the client.
2. Restore the files onto that mount point. Follow the procedures for the backup and recovery application to back up the files.
3. If you followed the guidelines in the *Planning, Installation, and Configuration Guide* for backup and recovery, restore files to the Windows filesets from a Windows client and restore files to the UNIX filesets from a UNIX-based client.

Chapter 14. Getting help, service, and information

If you need help, service, technical assistance, or just want more information about IBM products, you can find a wide variety of sources available from IBM to assist you.

Services available and telephone numbers listed are subject to change without notice.

Software Maintenance Agreement

All distributed software licenses include Software Maintenance Agreement (software subscription and technical support) for a period of 12 months from the date of acquisition providing a streamlined way to acquire IBM software and assure technical support coverage for all licenses. You can elect to extend coverage for a total of three years from date of acquisition. While your Software Maintenance is in effect, IBM provides you assistance for your 1) routine, short duration installation and usage (how-to) questions; and 2) code-related questions. IBM provides assistance by telephone and, if available, electronic access, only to your information systems (IS) technical support personnel during the normal business hours (published prime shift hours) of your IBM Support Center. (This assistance is not available to your end users.) IBM provides Severity 1 assistance 24 hours a day, every day of the year.

Before you call for service

This topic provides information you need to know before you call for service.

Some problems can be solved without outside assistance. You can use the online help by looking in the online or printed documentation that comes with the SAN File System, or by consulting the IBM Support Home Web site. Also, be sure to read the information in any README files and release notes that come with the SAN File System.

Getting help online

IBM maintains pages on the World Wide Web where you can get information about IBM products and services and find the latest technical information.

Table 9 lists some of these pages.

Table 9. IBM Web sites for help, services, and information

www.ibm.com/	Main IBM home page
www.ibm.com/storage/	IBM Storage home page
www.ibm.com/storage/support	IBM Support home page

Getting help by telephone

With the original purchase of the SAN File System, you have access to extensive support coverage. During the product warranty period, you can call the IBM Support Center (1 800 426-7378 in the U.S.) for product assistance covered under the terms of the software maintenance contract that comes with SAN File System purchase.

Have the following information ready when you call:

- SAN File System software identifier, which can be either the product name (SAN File System) or the Product Identification (PID) number
- Description of the problem
- Exact wording of any error messages
- Hardware and software configuration information

If possible, have access to your master console when you call.

In the U.S. and Canada, these services are available 24 hours a day, 7 days a week. In the U.K., these services are available Monday through Friday, from 9:00 a.m. to 6:00 p.m. In all other countries, contact your IBM reseller or IBM marketing representative.¹

1. Response time varies depending on the number and complexity of incoming calls.

Appendix A. Commands

SAN File System has two sets of commands: administrative and client commands.

Administrative commands

The administrative commands run on the storage engines that host the metadata server. Most commands must be run from the master metadata server. There are a few commands that must be run from subordinate metadata server for specific situations.

You run a majority of the administrative commands from the sfscli session to manage SAN File System. There are a few commands that must be run from the operating-system shell prompt.

To use the administrative commands, you must log in directly to the engine, or from another workstation through SSH, using the local operating system authentication mechanism. You must then log in to the administrative server on the engine using the same administrative user ID and password that you would use to log into the SAN File System console. You can specify the password in one of two ways:

- Set the password using the tankpasswd utility.
- Set the SFS_CLI_PASSWDFILE environment variable to the location of the password file.

When you run administrative commands that take a long time to complete, in a system with active applications, those applications that are sensitive to the response time of the system might experience timeout errors. An example of a possible long running command is **quiescecluster**.

Tip: The administrative commands are case sensitive. If you enter a command in uppercase, you receive an error.

Client commands

The client commands run on any client machine on which the client file-system driver has been installed. It provides a set of commands that you can use to manage your clients.

To use the client commands, you must log in directly to the client machine or from another workstation using SSH. You log in using the user ID and password for the client machine. You must have administrative (Windows) or root (UNIX-based) privileges to use the client commands.

Administrative CLI overview

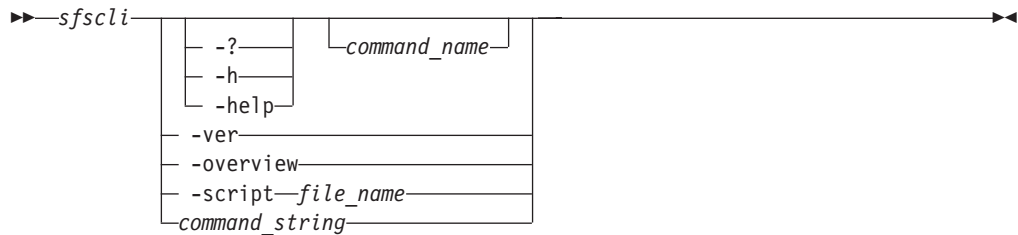
The SAN File System administrative command-line interface (CLI) provides a method for managing SAN File System.

Administrative command-line interface

The administrative command-line interface (CLI) utility allows you to run administrative commands in interactive mode. You can also run a single command or run a set of commands from a script without starting an interactive administrative CLI session.

If you run the administrative CLI utility with any of the valid parameters, an interactive session is not started. If you run the command with no parameters, the administrative CLI starts in a session. When you are in an interactive administrative CLI session, the *sfsccli* prompt is displayed.

This is the syntax for using the administrative CLI command:



Parameters

-? | -h | -help *command_name*

Displays help for the specified command (for example, **-h catlog** displays help for the **catlog** command). If a command name is not specified, this parameter displays a list of available commands in the administrative CLI.

-ver

Displays the current version and licensing information for this product.

-overview

Displays the overview information about the administrative CLI, including command modes, standard format, help and listing parameters, and syntax diagram conventions.

-script *file_name*

Runs the set of command strings in the specified file outside of an interactive administrative CLI session. If you specify this parameter, you must specify a file name.

The format options specified using the **setoutput** command apply to all commands in the script.

Output from successful commands routes to the standard output stream (stdout). Output from unsuccessful commands route to the standard error stream (stderr). If an error occurs while one of the commands in the script is running, the script will exit at the point of failure and return to the system prompt.

command_string

Runs the specified command string outside of an administrative CLI session.

Command modes

You can work with the administrative CLI in one of three modes: single-shot, interactive, and script.

Single-shot mode

If you want to run only a single command, specify the administrative CLI utility and the command that you want to run from the shell prompt, for example:

```
shell> sfscli lspool -l -type default
Name Type Size(MB) Used(MB) Used(%) Threshold(%) Volumes Partition Size(MB)
Allocation Size(KB) Description
-----
DEFAULT_POOL User Default 60272 336 0 80 1 16 Auto Default storage pool
```

Interactive mode

If you want to run several commands, start an administrative CLI session using the administrative CLI utility with no parameters, and then enter each command at the *sfscli*> prompt, for example:

```
shell> sfscli
sfscli> lspool -l -type default
DEFAULT,Default,10000,2500,25,80,10,64,Auto,Default Storage Pool
sfscli> exit
shell>
```

Script mode

If you want to run a set of commands that you defined in a file, use the administrative CLI utility with the **-script** parameter, for example:

```
shell> sfscli -script ~/bin/listpools.bat
```

You can add comments to the script file by placing a pound sign (#) in the first column, for example:

```
# This script file lists the default storage pool.
lspool -l -type default
```

Note: Output from successful commands routes to the standard output stream (stdout). Output from unsuccessful commands route to the standard error stream (stderr). If an error occurs while one of the commands in the script is running, the script will exit at the point of failure and return to the system prompt.

Naming guidelines

This topic provides guidelines to help you define objects and descriptions used in the administrative CLI.

Objects

Use the following guidelines when specifying names for objects:

- You can use alphanumeric characters, dashes (-), underscores (_), and periods (.) in the object names; however, object names cannot start with a dash or underscore character and must contain at least one alphanumeric character.
- The object name must not contain blank spaces.
- Most object names can contain up to 256 characters; the exception is cluster and metadata server names, which can contain up to 32 characters. Also, be aware that file names on AIX can contain up to only 255 characters.
- Object names are case-sensitive.

Descriptions

Use the following guidelines when specifying descriptions:

- A description can contain up to 256 characters and cannot start with a blank space.
- For administrative commands, if a description contains spaces, enclose the description in single quotation marks (') or double quotation marks (").
- For administrative commands, if a description contains quotation marks, enclose the description in opposite quotation marks (for example, 'This is a pool named \'Foo\>').
- For administrative commands, if a description that is enclosed in quotation marks also contains single quotation marks ('), double quotation marks ("), or asterisks (*) within the string, precede the character with a backslash (for example, "*\'This is a test\'*").

Ports

You can specify a value from 1 024 to 65 535 for port numbers.

Host and domain names

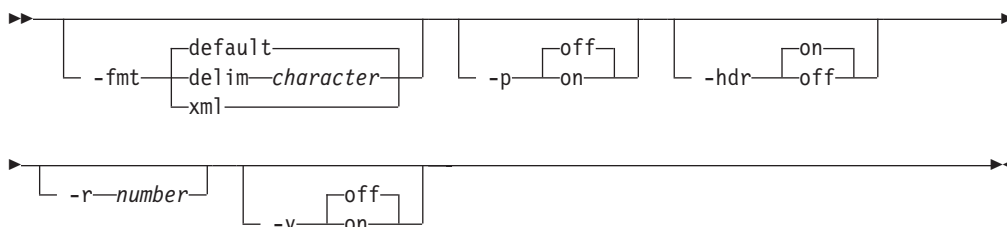
Machine host names must not include the underscore character (_). Internet standards dictate that domain names conform to the host name requirements described in Internet Official Protocol Standards RFC 952 and RFC 1123.

Domain names must contain only letters (upper or lower case) and digits. Domain names can also contain dash characters (-) as long as the dashes are not on the ends of the name.

Standard format parameters

The standard format parameters set the output format of listing commands (commands that start with ls*) in the administrative CLI. These parameters can be used either in a listing command syntax or in the **setoutput** command syntax. You can specify these parameters in addition to the parameters available for a specific listing command.

The format settings remain in effect for the duration of the administrative CLI session or until you reset the parameters either by specifying these parameters in a listing command or using the **setoutput** command.



Parameters

-fmt

Specifies the format of the output. You can specify one of the following values:

default

Specifies to display output in a tabular format using spaces as the delimiter between the columns. This is the default value. For example:

Name	Type	Size (GB)	Used (GB)	Used (%)	Alert (%)
DEFAULT	Default	10000	2500	25	80

Volumes	Partition Size (MB)	Allocation Size (KB)	Description
10	64	128	Default Storage pool

delim *character*

Specifies to display output in a tabular format using the specified character to separate the columns. If you use a shell metacharacter (for example, * or \t) as the delimiting character, enclose the character in single quotation marks (') or double quotation marks ("). A blank space is not a valid character. For example, **sfscli lspool -fmt delim ", "** produces the following output:

```
DEFAULT,Default,10000,2500,25,80,10,64,128,Default Storage Pool
```

xml

Specifies to display output using XML format, for example:

```
<IRETURNVALUE>
<INSTANCE CLASSNAME="STC_StoragePool">
<PROPERTY NAME="Name" TYPE="string"><VALUE>DEFAULT_POOL</VALUE>
</PROPERTY>
<PROPERTY NAME="PoolType" TYPE="uint32"><VALUE>1</VALUE>
</PROPERTY>
<PROPERTY NAME="PartitionSize" TYPE="uint64"><VALUE>16</VALUE>
</PROPERTY>
<PROPERTY NAME="AlertPercentage" TYPE="uint16"><VALUE>80</VALUE>
</PROPERTY>
<PROPERTY NAME="Size" TYPE="uint64"><VALUE>0</VALUE></PROPERTY>
<PROPERTY NAME="SizeAllocated" TYPE="uint64"><VALUE>0</VALUE>
</PROPERTY>
<PROPERTY NAME="SizeAllocatedPercentage" TYPE="uint16"><VALUE>0
</VALUE></PROPERTY>
<PROPERTY NAME="NumberOfVolumes" TYPE="uint32"><VALUE>0</VALUE>
</PROPERTY>
<PROPERTY NAME="Description" TYPE="string"><VALUE>Default storage pool
</VALUE></PROPERTY>
</INSTANCE>
</IRETURNVALUE>
```

-p Specifies whether to display one page of text at a time or all text at once.

off Displays all text at one time. This is the default value when the administrative CLI is run in single-shot mode.

on Displays one page of text at time. Pressing any key displays the next page. This is the default value when the administrative CLI is run in interactive mode.

-hdr

Specifies whether to display the table header.

on Displays the table header. This is the default value.

off Does not display the table header.

-r *number*

Specifies the number of rows per page to display when the **-p** parameter is on. The default is 24 rows. You can specify a value from 1 to 100.

-v Specifies whether to enable verbose mode.

- off** Disables verbose mode. This is the default value.
- on** Enables verbose mode.

Standard help parameters

The standard help parameters display user assistance for any command in the administrative CLI. Note that when you use a help parameter, all other parameters are ignored.

These are the help parameters:



Parameters

-? | -h | -help

Displays the detailed description of a specific command.

Standard listing parameters

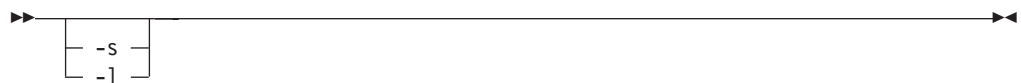
The standard listing parameters specify whether to display the default, long, or short output for listing commands (commands that start with ls*) in the administrative CLI. You can specify these parameters in addition to the parameters available for specific listing commands.

If you do not specify a listing parameter, the default listing displays all objects and the most vital column information, for example:

Name	User Role	Authorization
JohnDoe	Admin	Current
MaryBlack	Backup	Not Current
JimSmith	Operator	Current
TomJones	Monitor	Not Current

The format set using these parameters remains in effect for the duration of the command.

These are the listing parameters:



Parameters

-s Displays the list of objects with minimal information, for example,

```
Name
=====
JohnDoe
MaryBlack
JimSmith
TomJones
```

-l Displays the list of all objects with detailed information, for example:

Name	User Role	Authorization	Authorization Timeout (Secs)
JohnDoe	Admin	Current	300
MaryBlack	Backup	Not Current	0
JimSmith	Operator	Current	465
TomJones	Monitor	Not Current	0

Syntax diagram conventions

Syntax diagram conventions

To read syntax diagrams, follow the path of the line. Read from left to right, and top to bottom.

- The ► symbol indicates the beginning of the syntax diagram.
- The → symbol at the end of a line indicates that the syntax diagram continues on the next line.
- The ► symbol at the beginning of a line indicates that the syntax diagram continues from the previous line.

The →◄ symbol indicates the end of the syntax diagram.

Syntax diagrams use *position* to indicate required, optional, and default values for keywords, variables, and operands:

- On the line (required element)
- Above the line (default element)
- Below the line (optional element)

Abbreviations

The administrative CLI does not currently support aliases or abbreviations; however, you can use aliases that you set up in the shell environment.

Dash

A dash (-) indicates that in single-shot command mode you want to supply parameters to a command from an input stream (stdin) rather than entering parameters. In the following example, the command line gets input from the file /work/myfile:

```
$> sfsccli cmd - < work/myfile
```

In the following example, the command line gets input from returned data:

```
$> sfsccli lspool -s -hdr off -type user | sfsccli rmpool -quiet -
CMMNP5083I Storage Pool mypool1 was removed successfully.
CMMNP5083I Storage Pool mypool2 was removed successfully.
```

Defaults

Default values are above the main line. If the default is a keyword, it appears only above the main line. You can specify this item or allow it to default. In the following example, the keyword A is the default. You can override it by choosing B or C. You can also specify the default value explicitly.



If an operand has a default value, the operand appears both above and below the main line. A value below the main line indicated that if you specify the operand, you must also specify either the default value or another value shown. If you do not specify an operand, the default value above the main line is used. In the following example, the operand A=* is the default. You can override it by choosing A=C. You can also specify the default value explicitly.



Optional items

When one or more items are below the main line, all of the items are optional. In the following example, you can choose A, B, C, or nothing at all.



Repeatable items

An arrow returning to the left means you can repeat the item, for example:



A character or space within the arrow means you must separate repeated items with that character or space, for example:



A stack of items followed by an arrow returning to the left means that you can select more than one item or, in some cases, repeat a single item. In the following example, you can choose any combination of A, B, or C.



Required items

When a keyword, variable, or operand appears on the main line, you must specify that item. In the following example, you must choose A, B, and C.

▶▶ A—B—C ▶▶

When two or more items are in a stack and one of them is on the main line, you must specify one item. In the following example, you must choose A, B, or C.

▶▶ A
| B
| C ▶▶

Syntax fragments

Commands that contain lengthy groups or a section that is used more than once in a command are shown as separate fragments following the main diagram. The fragment name appears between vertical bars in the diagram. The expanded fragment also appears between vertical bars after the heading with the same fragment name.

▶▶ | The fragment name | ▶▶

The

| A
| B
| C |

Variables

Italicized, lowercase elements denote variables. In the following example, you must specify a variable name when you enter the keyword command:

▶▶ keyword—*variable* ▶▶

Common commands

These commands are common across other administrative command-line interfaces (CLIs).

exit

Ends an administrative command-line interface (CLI) session.

▶▶ exit
| -?
| -h
| -help ▶▶

Parameters

-? | -h | -help

Displays a detailed description of this command, including syntax, parameter descriptions, and examples. If you specify a help option, all other command options are ignored.

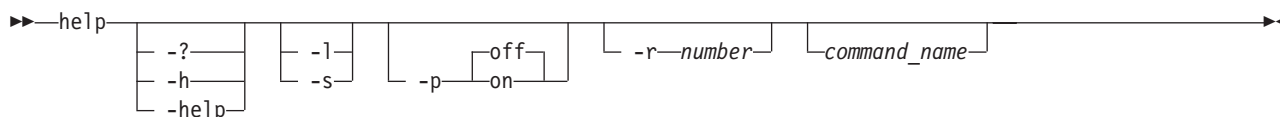
Example

Ends an administrative command-line interface (CLI) session The following example ends the CLI session:

```
sfscli> exit
shell>
```

help

Displays a list of commands available in an administrative command-line interface (CLI) and optionally displays the syntax or brief description of each command.



Parameters

-? | -h | -help

Displays a detailed description of this command, including syntax, parameter descriptions, and examples. If you specify a help option, all other command options are ignored.

-l Displays a list of available commands with the syntax diagrams for each. If you specify a command name with this parameter, this command displays the syntax for only the specified command.

-s Displays a list of available commands with a brief description of each. If you specify a command name with this parameter, this command displays a brief description for only the specified command.

-p Specifies whether to display one page of text at a time or all text at once.

off Displays all text at one time. This is the default value.

on Displays one page of text at time. Pressing any key displays the next page.

-r number

Specifies the number of rows per page to display when the **-p** parameter is on. The default is 24 rows. You can specify a value from 1 to 100.

command_name

Displays help information for the specified command, including the syntax diagram, parameter descriptions, return codes and errors, descriptions, examples, and miscellaneous remarks.

Description

If you specify this command with no parameters, this command displays only a list of available commands.

Example

Display a description of a command The following example displays the description of the **lspool** command:

```
sfscli>help -s lspool
lspool Displays a list of existing storage pools and their attributes.
```

quit

Ends an administrative command-line interface (CLI) session.



Parameters

-? | -h | -help

Displays a detailed description of this command, including syntax, parameter descriptions, and examples. If you specify a help option, all other command options are ignored.

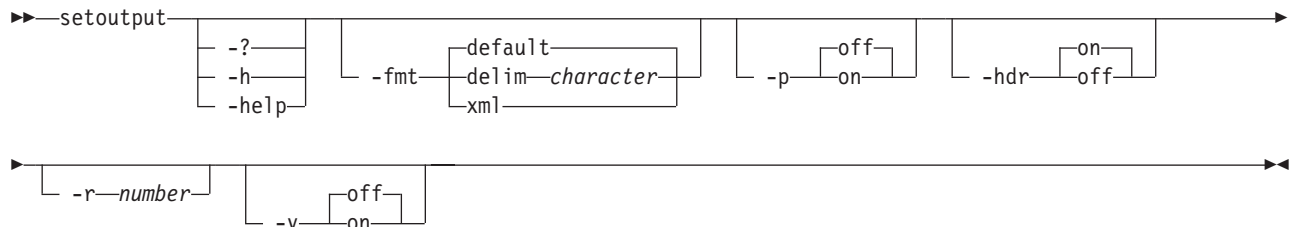
Example

Ends an administrative command-line interface (CLI) session The following example ends the CLI session:

```
sfscli> quit  
shell>
```

setoutput

Sets the output format for the administrative command-line interface (CLI).



Parameters

-? | -h | -help

Displays a detailed description of this command, including syntax, parameter descriptions, and examples. If you specify a help option, all other command options are ignored.

-fmt

Specifies the format of the output. You can specify one of the following values:

default

Specifies to display output in a tabular format using spaces as the delimiter between the columns. This is the default value. For example:

Name	Type	Size (GB)	Used (GB)	Used (%)	Alert (%)
DEFAULT	Default	10000	2500	25	80

Volumes	Partition Size (MB)	Allocation Size (KB)	Description
10	64	128	Default Storage Pool

delim character

Specifies to display output in a tabular format using the specified character to separate the columns. If you use a shell metacharacter (for

example, * or \t) as the delimiting character, enclose the character in single quotation mark (') or double quotation mark ("). A blank space is not a valid character. For example:

```
DEFAULT,Default,10000,2500,25,80,10,64,128,Default Storage Pool
```

xml Specifies to display output using XML format, for example:

```
<IRETURNVALUE>
<INSTANCE CLASSNAME="STC_StoragePool">
<PROPERTY NAME="Name" TYPE="string"><VALUE>DEFAULT_POOL</VALUE>
</PROPERTY>
<PROPERTY NAME="PoolType" TYPE="uint32"><VALUE>1</VALUE>
</PROPERTY>
<PROPERTY NAME="PartitionSize" TYPE="uint64"><VALUE>16</VALUE>
</PROPERTY>
<PROPERTY NAME="AlertPercentage" TYPE="uint16"><VALUE>80</VALUE>
</PROPERTY>
<PROPERTY NAME="Size" TYPE="uint64"><VALUE>0</VALUE></PROPERTY>
<PROPERTY NAME="SizeAllocated" TYPE="uint64"><VALUE>0</VALUE>
</PROPERTY>
<PROPERTY NAME="SizeAllocatedPercentage" TYPE="uint16"><VALUE>0
</VALUE></PROPERTY>
<PROPERTY NAME="NumberOfVolumes" TYPE="uint32"><VALUE>0</VALUE>
</PROPERTY>
<PROPERTY NAME="Description" TYPE="string"><VALUE>Default storage pool
</VALUE></PROPERTY>
</INSTANCE>
</IRETURNVALUE>
```

-p Specifies whether to display one page of text at a time or all text at once.

off Displays all text at one time. This is the default value when the **cli** command is run in single-shot mode.

on Displays one page of text at time. Pressing any key displays the next page. This is the default value when the **cli** command is run in interactive mode.

-hdr

Specifies whether to display the table header.

on Displays the table header. This is the default value.

off Does not display the table header.

-r number

Specifies the number of rows per page to display when the **-p** parameter is on. The default is 24 rows. You can specify a value from 1 to 100.

-v Specifies whether to enable verbose mode.

off Disables verbose mode. This is the default value.

on Enables verbose mode.

Description

The output format set by this command remains in effect for the duration of the administrative command-line interface (CLI) session or until the options are reset either by using this command or by specifying a output-format parameter as part of a command.

Running this command with no parameters displays the current output settings in the default output format, for example:

Paging	Rows	Format	Header	Verbose
off	-	default	on	off

Note: The output formats do not apply to help pages.

Example

Set the output format The following example sets the output format to display in tabular form using a comma as the delimiter without header information:

```
sfscli>setoutput -fmt delim , -hdr off
sfscli>lspool -l -type default
DEFAULT,Default,10000,2500,25,80,10,64,128,Default Storage Pool
```

Client commands

There is a set of commands for each client operating system that SAN File System supports.

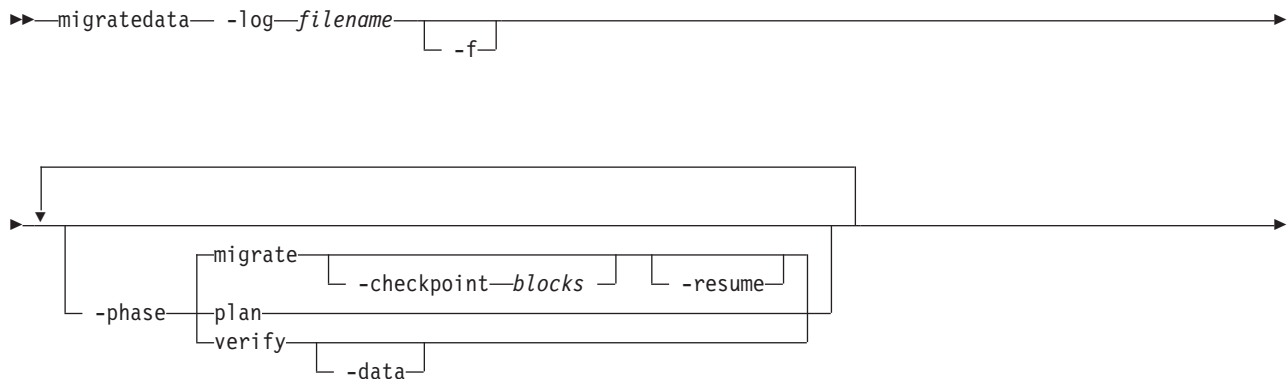
Common client commands

This topic provides a brief description for each command that has the same syntax on multiple operating systems.

Command	Operating System	Description
Migration		
"migratedata"	AIX, Linux, Solaris, and Windows	Migrates data to SAN File System.
Virtual client setup and removal		
"rmstclient" on page 122	AIX, Linux, and Solaris	Unmounts the global namespace, removes the SAN File System client, and unloads the file-system driver.
"setupstclient" on page 122	AIX, Linux, and Solaris	Configures and starts a client.

migratedata

Migrates data to SAN File System.





Parameters

-log *filename*

Specifies the location of a file in which to log migration activities, warnings, and errors. When used with the **-plan migrate -resume** parameter, the **-log** parameter specifies the log file from which to read information about the last completed block or file.

Attention: You must specify the correct log file with migrate **-resume** and verify that the source and destination directories specified on the command line match those in the log file.

- If you specify an incorrect log file and the **-f** parameter, **-resume** displays a warning, but migrates all of the data that you specify. However, the verification fails if any of the source directories specified are different than those listed in the incorrect log file.
- If you specify an incorrect log file, but do not specify the **-f** parameter, this command displays an error and exits.

-f Specifies that the migration should continue even if there is an error with a file. If specified with the **-phase migrate** parameter, this command skips any files with errors, and continues with the migration process. Examples of file errors include insufficient privileges to read the file, or not running as superuser, preventing the permissions on the SAN File System file to be set. If not specified, an error results in the entire migration being stopped before the file that caused the error. You can then restart the migration after fixing the error.

If specified with the **-phase verify** parameter, this command adjusts any missing metadata attributes, such as permissions and times. If there is a mismatch in size, however, this command does not try to readjust the metadata attributes.

-phase

Specifies the migration phase to run. Choices include:

plan Gathers information about the available system resources (available memory, number of CPUs, size of the source tree and space available on the destination file system), copies sample files from source directory to estimate transfer rates, and provides an estimated time for the migration of the data set. The copies of the sample files are then deleted, unless the process is interrupted, in which case the copies are not deleted.

migrate

Reads data from source file system and writes the data to the destination file system. Although not required, for large data sets, you should run this command in planning mode first. You can stop the migration process at any point (by pressing **Ctrl+C**) and resume from the last completed file or block (using the **-resume** parameter).

This is the default value.

verify Verifies the integrity of the migrated data using the Message Digest 5 verification algorithm on the contents of the file, as well as verifying

consistency of the metadata (such as owner and modification time stamp settings) between the source and destination files.

You can specify more than one phase. For example, to plan, migrate, and verify the data, specify **-phase plan -phase migrate -phase verify**. Although you can specify the phases in any order, this command always estimates the completion time, migrates data, and then verifies the migrated data.

If the **-phase** parameter is not specified, this command runs only the migration phase.

-checkpoint *blocks*

Shows the progress when migrating large files. If you specify this parameter, the **migratedata** command writes a checkpoint in the log file after each specified number of blocks of a file has been migrated. (The block size depends on the client platform.) For example, if you specify **-checkpoint 20**, this command makes an entry in the log file each time 20 blocks of file data is migrated. On a platform with a block size of 16 MB, this command writes to the log file after each 320 MB of data from a file has been migrated.

If the migration process is interrupted, this parameter allows you to resume the migration at the place it left off.

If unspecified, the **migratedata** command makes an entry in the log file after each complete file has been migrated. You can resume the migration at the point of the last migrated file.

-resume

Resumes the migration from the last completed block or file (logged in the log file specified by the **-log** parameter). If the log file indicates that some files in the source directory are migrated and this parameter is not specified, this command restarts the migration process from the beginning (performs a fresh migration).

-data

Verifies every block of source data (file data and metadata) with the destination data. If not specified, this command verifies only the metadata unless there is a mismatch in the file attributes, in which case this command then verifies the file data.

Note: Verifying all data is very time consuming and can take as long as the migration itself.

-destdir *dest_directory_name*

Specifies the name of the destination directory for the migrated data. The directory can either exist or be a new directory name. It is recommended that you create the directory before beginning the migration process. If the directory does not exist, this command creates the directory using the default permissions.

source_path

Specifies one or more paths of directories or files to migrate.

Prerequisites

You must have root privileges on a UNIX-based client or Administrative privileges on Windows to use this command.

All storage pools, all filesets, and at least one policy must be set up. All activity (from applications, such as database servers and application servers, or users) that

modifies data on the source and destination file systems must be stopped and remain stopped to guarantee consistency of the migrated data.

The destination directory must exist with correct set of permissions and appropriate storage policies must be configured.

Example

Migrating data This example migrates data from the work/capital directory on the client machine to the sanfs/cnt1 directory in the global namespace. A checkpoint is written to the mgrt_capital.log log file each time 20 blocks of file data is migrated.

```
migratedata -log /mgrtlogs/mgrt_capital.log -phase migrate -checkpoint 20  
-destdir /mnt/tank/sanfs/cnt1 work/capital
```

rmstclient

Unmounts the global namespace, removes the virtual client, and unloads the file system driver from the local client machine.



Parameters

`-prompt`

Prompts for required parameters, using values from the configuration file, if available.

`-noprompt`

Runs silently, using parameters from the configuration file (`/usr/tank/client/config/stclient.conf`). If a required parameter is not available, the command exits with an error.

Prerequisites

You must have root privileges to use this command.

You must unmount the SAN File System before invoking this command.

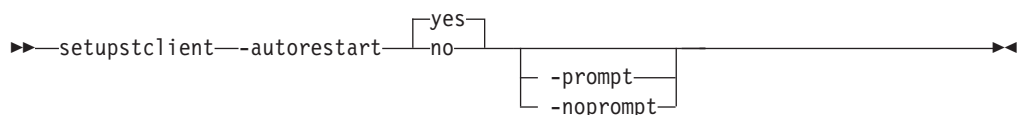
Example

Remove a client The following example removes the local SAN File System client without prompting:

```
rmstclient -noprompt
```

setupstclient

Configures and starts SAN File System clients.



Parameters

-autorestart

Specifies whether to restart the SAN File System client automatically at boot time.

yes Enables autorestart. The SAN File System client is restarted automatically at boot time. This is the default value.

no Disables autorestart.

-prompt

Forces the **setupstclient** command to prompt for all configuration values.

-noprompt

Runs silently, using parameters from the configuration file. If the configuration file does not exist or if a required parameter is not available or invalid, the command exits with an error.

Prerequisites

You must have root privileges to use this command.

Description

This command configures and starts or restarts a SAN File System client.

If you do not specify a parameter, this command runs silently using values from the configuration file as defaults. It only prompts for any required information, if a configuration file does not exist or if a value in an existing configuration file is not valid.

This command maintains any values given by the user in a configuration file in parameter=value format. The default configuration file is `/usr/tank/client/config/stclient.conf`.

Specify the `-prompt` parameter to force the command to prompt for all configuration values. In this case, if a configuration file exists, the command presents the value from the configuration file as the suggested default when the command displays a prompt. If a configuration file does not exist, the command presents the manufacturing default as the suggested default.

If you specify the `-noprompt` parameter, the command expects the configuration file to exist. If the file does not contain valid values, the command exits with an error.

Example

Setup a client The following example configures and starts SAN File System clients:

```
/usr/tank/client/bin/setupstclient
```

AIX-client commands

This topic provides a brief description for each AIX-client command.

Note: You must have root privileges to use these commands.

Command	Description
Migration	
"migratedata" on page 119	Migrates data to SAN File System.
Status	
"stfsstatus" on page 131	Displays the version of the file-system driver for the specified virtual client.
Volumes and LUNs	
"stfsdisk" on page 126	Scans the SAN File System for new and removed volumes.
Virtual client setup and removal	
"rmstclient" on page 122	Unmounts the global namespace, removes the SAN File System client, and unloads the file-system driver.
"setupstclient" on page 122	Configures and starts a client.
"stfsclient"	Creates or destroys a virtual client.
"stfsdriver" on page 129	Loads the file-system driver as a kernel extension.
"stfsmount" on page 130	Mounts the global namespace.
"stfsunmount" on page 132	Unmounts the global namespace.

stfsclient

Creates or destroys a virtual client.

```

▶▶ stfsclient -create client_name server_name server_IP_address :-port
▶ -kmname kernel_ext_name -converter 8859-1 -quiet

```

or

```

▶▶ stfsclient -destroy client_name -kmname kernel_ext_name
▶ -quiet

```

Parameters

-create

Creates a new virtual client.

-destroy

Destroys an existing virtual client.

client_name

Identifies the unique name of the virtual client that you want to create or

destroy. Each client connecting to the metadata servers must have a unique client name. The default client name is the host name of the client system.

server_name

Specifies the host name of a metadata server in the SAN File System. The metadata server that you specify informs the global namespace image of all other metadata servers.

This parameter is not required if this is not the first mount for a particular virtual client.

server_IP_address

Specifies the IP address, in dotted decimal notation, of a metadata server in the SAN File System.

port

Specifies the port number of the specified metadata server. The default is 1700.

-kmname *kernel_ext_name*

Identifies kernel-extension name of the file-system-driver instance associated with the virtual client.

The file-system driver is loaded as a kernel extension. To identify the instance of the file-system driver, you identify the kernel extension. The kernel-extension name is the same as name and location of the file-system driver that was used to load the driver (for example, /usr/tank/client/bin/stfs for AIX).

-devices

Determines which devices (also called disks or LUNs) that the virtual client considers as SAN File System volumes. The default is the value of the STFS_DEVICES environment variable or, if that is not set,

-devices=pat=/dev/rhdisk*. For SDD devices, specify

-devices=pat=/dev/rvpath*.

In addition to creating the virtual client, this command discovers which disks, or candidates, are available to the virtual client as volumes and transmits the candidate list to the virtual client. The **-devices** parameter controls the candidates list.

dir=*directory*

The candidates list is made up of those devices that have device special files in the specified directory (for example:

-devices=dir=/dev/stfsdisk).

The easiest way to mount the global namespace is to specify -devices=pat=/dev/rhdisk* , which looks at every SCSI-disk-like device in the system and whatever looks like a SAN File System disk is accessed when the metadata server refers to that disk's SAN File System disk identifier.

If you want the client to be more selective about what disks it considers available, you can create a /dev/stfsdisk directory, put device-special files (or symbolic links) for your candidates in it, and use -devices=dir=/dev/stfsdisk.

pat=*pattern*

The candidates list is made up of those devices that have device-special files whose file specifications match the specified pattern. You can use * wildcards in the last (filename) component but not in the directory components (for example, -devices=pat=/dev/rhdisk*).

none The candidates list is empty. Use this value when you want to establish the candidate list with a separate command, perhaps using a selection method more sophisticated than the stfsclient command offers.

-quiet

Turns off informational messages for this command. This parameter does not affect error messages.

Prerequisites

You must have root privileges to use this command.

Description

This command creates or destroys a virtual client. A *virtual client* is an entity that communicates with a metadata server and, indirectly, with other SAN File System clients. In this release, only one virtual client is supported per client machine. The terms virtual client and client can be used interchangeably.

A virtual client is associated with exactly one SAN File System. There is one file cache and one set of disk candidates per virtual client. Each virtual client running on the same system is as separate as if it were running on a different system. They share nothing except the file-system drive code that they execute.

A SAN File System virtual client is uniquely identified in the context of its file-system driver, and in the context of its SAN File System, by its client name.

To use the files in a global namespace, the virtual client must have a global namespace image. Creating a global namespace image makes the directory structure in the global namespace appear in the client's file structure. To create a global namespace image, use the **stfsmount** command.

A client can access and create data that is stored in a global namespace. Each virtual client can access data on multiple images in the same global namespace. It might be useful to have different mounts, where each mount has different options, for example, one mount might be read-only. Also, it might be useful to have multiple clients, where each client is communicating with a different SAN File System server cluster.

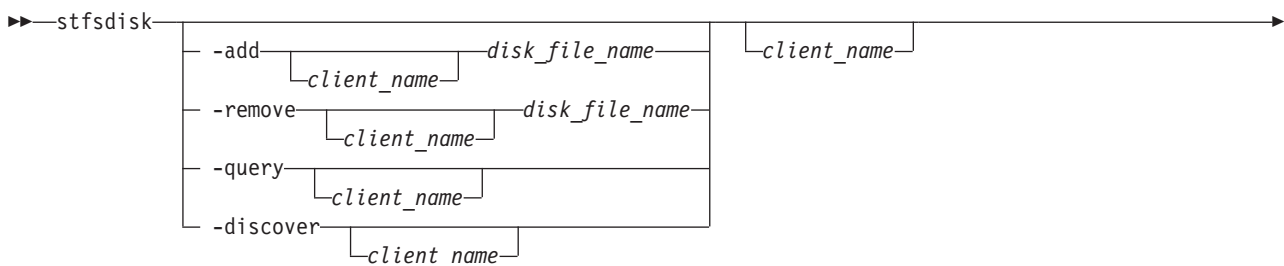
Example

Create a virtual client The following example creates a virtual client:

```
stfsclient -create MDS1:1700 -kname /usr/tank/client/bin/stfs
-converter 8859-1
```

stfsdisk

Controls the SAN File System volumes (disks) that a client can access.



Parameters**-add**

Adds the specified disk-specific file name to the disk-candidate list.

-remove

Removes the specified disk-specific file name from the disk candidate list. If the disk-specific file name is not in the list, this command does nothing, but does not consider it an error.

-query

Displays the list of disk-specific file names in the current disk-candidate list and the status of each. Possible status values are:

ACTIVE

Indicates that the disk is a valid SAN File System user-data volume and is available to the client to perform file reads and write operations.

INACTIVE

Indicates that the disk is not a valid SAN File System user-data volume and is not available to the client to perform file reads and writes. A disk can be inactive if the disk does not contain a SAN File System label or if the disk's SAN File System label says it is something other than a user-data volume.

-discover

Rebuild the database of usable disks by going through the current candidate-disk list and attempting to access each disk, determine if it is a valid SAN File System user-data volume, and read its SAN File System global disk ID. If a disk has become accessible or inaccessible, or changed its identity since the last time this disk-discovery procedure was run, the virtual client updates its candidate-disk list accordingly.

This parameter causes the disk-discovery procedure to start. The procedure typically ends before the disk-discovery procedure completes. While the disk-discovery procedures are in progress, any file-system access that would fail because the virtual client cannot find a specific disk will wait until the disk-discovery procedure completes, and then proceed on the basis of the new disk-accessibility information.

disk_file_name

Specifies the file name of the disk to add to or remove from the disk candidate list. This must be a raw disk file such as /dev/rhdisk5 or /dev/rvpath5.

client_name

Specifies the name of the virtual client whose disk-access you are controlling.

-kmname *kernel_ext_name*

Identifies kernel-extension name of the file-system-driver instance associated with the client.

The file-system driver is loaded as a kernel extension. To identify the instance of the file-system drive, you identify the kernel extension. The kernel-extension name is the same as the name and location of the file-system driver that was used to load the driver (for example, /usr/tank/client/bin/stfs for AIX). Note that the kernel extension name might not be unique.

Prerequisites

You must have root privileges to use this command.

Description

A client reads and writes files by accessing the disks on which the file data resides. To control which disks that a client can access, SAN File System identifies that disk by a SAN File System global disk identifier, and the disk-access subsystem associates that identifier with the name that the AIX operating system uses to identify that disk. The disk-access subsystem maintains a database that correlates global-disk identifiers with AIX device numbers. When the client needs to access a data block of a file, it consults that database.

The disk-access subsystem maintains the database by reading certain disks at certain times and looking for a SAN File System global disk identifier. If it finds the identifier, it determines whether the disk is a SAN File System user-data volume. If the disk is a volume, it adds the disk to its database.

The set of disks that the disk-access subsystem searches is called the *disk-candidate list*. The **stfsclient** command creates the disk-candidate list when it creates the virtual client. You can modify the list using the **-add** and **-remove** parameters.

The candidate-disk list is a list of unique disk-special file names. Because a disk can be referred to by more than one disk-special file name, the list is not strictly a list of unique devices. Actually examining disks and updating the database of valid user-data volumes is separate from maintaining the candidate-disk list.

When you add a disk to the candidate-disk list, the client immediately tries to read it and adds it to the database. But the disk becomes and stays a candidate regardless of the results of that operation.

You can force the client to rescan the entire list of candidate disks using the **-discover** parameter. The client updates its database of user-data volumes according to the results of this discovery, adding and removing disks as necessary. The results of the discovery do not affect the candidate-disk list, however.

Note that device file names can change as the client runs. Such a change has no effect on the client unless something causes a disk-discovery procedure to run. For example, if you add `/dev/rhdisk35` as a candidate disk, and the client successfully identifies it as a SAN File System user-data volume, and then you delete `/dev/rhdisk35`, the client continues accessing that disk as before. The disk `/dev/rhdisk35` continues to be a candidate. But the next time a disk-discovery procedure runs, the candidate will be found invalid and the client will no longer have access the disk.

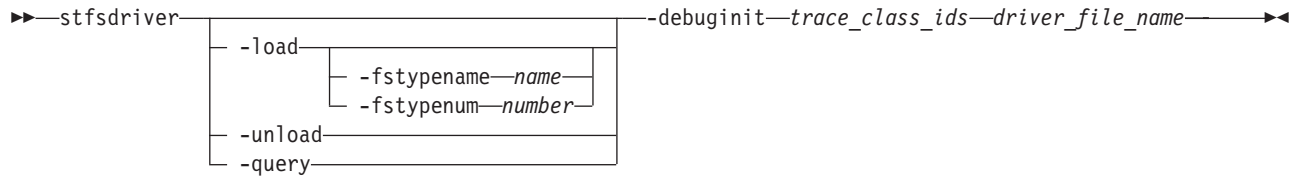
Example

Query the disk-candidate list The following example queries disk-candidate list and displays the status of each disk:

```
stfsdisk -query -kmname /usr/tank/client/bin/stfs
```

stfsdriver

Loads the file-system driver as a kernel extension.



Parameters

-load

Loads the kernel extension and create an instance of the file-system driver.

-unload

Unloads the kernel extension and destroys the instance of the file-system driver.

-query

Displays information about the kernel extension matching the specified criteria. For example, you can query the kernel extension ID to use in commands instead of the kernel module name.

-fstypename *name*

Specifies the name of the file-system type to use for the file-system-driver instance. This name relates to a specific file-system-type number. The file `/etc/vfs` maps the file-system-type name to the number.

If you do not specify a file-system-type name or number, the system defaults to the file-system-type named "sanfs". If there is no such type in the `/etc/vfs` file, the system defaults to the file-system-type number 20.

You will use this name to create the virtual client.

-fstypenum *number*

Identifies the number associated with the file-system type for the file-system-driver instance. All mount requests for a file system of this type are routed to this file-system-driver instance.

You would use this parameter only when you load multiple instances of the file-system driver on the same client system.

Restriction: Do not specify the number 1, 3, 16, any already loaded file-system type number, or a number greater than 39.

-debuginit *trace_class_ids*

Presets the tracing state of the driver during its initialization by specifying one or more IDs of trace classes that are automatically enabled when the driver initializes successfully.

driver_file_name

Specifies the name and location of the file-system driver that you want to load, unload, or query. The file name is typically "stfs".

The file-system driver is loaded as a kernel extension. To identify the instance of the file-system drive, you identify the kernel extension. The kernel-extension name is the same as the name and location of the file-system driver that was used to load the driver (for example, `/usr/tank/client/bin/stfs`).

Prerequisites

You must have root privileges to use this command.

Description

This command creates a file-system-driver instance by loading the file-system driver as a kernel extension. This command also unloads or queries the kernel extension.

After loading the file-system driver, you can use the **stfsclient** command to create a virtual client and then use the **stfsmount** command to mount the global namespace.

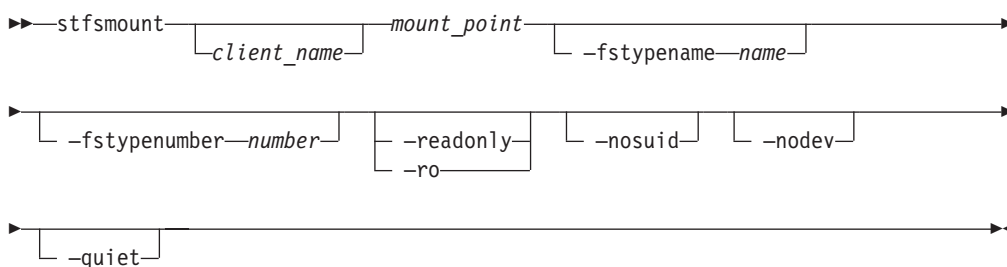
Example

Loads the file-system driver The following example loads the file-system driver on a client for AIX:

```
stfsdriver -load /usr/tank/client/bin/sanfs
```

stfsmount

Mounts the global namespace.



Parameters

client_name

Identifies the unique name of the virtual client to which you want to mount the global namespace. The client must be up and running. The default client name is the host name of the client system.

mount_point

Specifies the directory associated with the global namespace image that you want to mount.

-fstypename *name*

Specifies the name of the file-system type to use for the file-system-driver instance. This is the same name used to load the file-system driver.

This name relates to a specific file-system-type number. The file `/etc/vfs` maps the file-system-type name to the number.

If you do not specify a file-system-type name or number, the system defaults to the file-system-type named "sanfs." If there is no such type in the `/etc/vfs` file, the system defaults to the file-system-type number 20.

You would use this parameter only when you load multiple instances of the file-system driver on the same client system.

-fstypenumber *number*

The number that identifies the file-system type for the file-system-driver instance. All mount requests for a file system of this type are routed to this file-system-driver instance.

-readonly | **-ro**

Sets the global namespace image to read only. If specified, an attempt to update data or metadata in the global namespace will fail, and an attempt to access a file-system object will not update its access-time attribute.

-nosuid

Disallows any invocation of the `setuid` or `setgid` commands from this file-system image.

-nodev

Disallows any attempts to open device nodes in this file-system image.

-quiet

Turns off informational messages for this command. This parameter does not affect error messages.

Prerequisites

You must have root privileges to use this command.

Description

This command creates an image of the global namespace on the client system by mounting a directory. The global namespace maintains a list of its directories that are available to the clients. When a client mounts a directory in the global namespace, that directory and its subdirectories become part of the client's directory hierarchy.

Note: This command is used in place of the **mount** command to mount the global namespace.

Before you can mount the global namespace, you must have a virtual client running on the client system. To create the virtual client, use the **stfsclient -create** command.

Remounting the global namespace image is not the same as unmounting the global namespace and then mounting it again. Rather, it changes the attributes of an existing global namespace image, such as changing from read-write to read-only mode. To remount the global namespace image, use the **stfsmount** command. To see what global namespace images are currently mounted, specify the **stfsmount** command without any parameters.

To unmount the global namespace, use the **stfsumount** command.

Example

Mount the global namespace The following example mounts the global namespace:

```
stfsmount /mnt/sanfs -fstypename sanfs
```

stfsstatus

Displays the version of the file-system driver for the specified virtual client for AIX.

Parameters

-kmname *kernel_ext_name*

Identifies kernel-extension name of the file-system driver associated with the virtual client.

The file-system driver is loaded as a kernel extension. To identify the instance of the file-system driver, you identify the kernel extension. Each kernel extension has a name, but this name is not unique. This name is usually the file name of the object file from which you loaded the kernel extension (for example, /usr/tank/client/bin/stfs). To determine the kernel-extension name, use the `genkex | grep stfs` command.

Prerequisites

You must have root privileges to use this command.

Description

After issuing this command, if the client is running, the version of the file-system driver is displayed. If the file-system driver is not loaded, an error message is displayed stating that system could not determined the file-system driver instance.

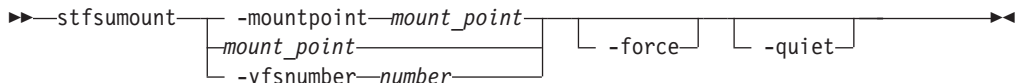
Example

Display the file-system-driver version The following example displays the version of the file-system driver for the local client:

```
stfsstatus -kmname /usr/tank/client/bin/stfs
```

stfsunmount

Unmounts the global namespace.



Parameters

-mountpoint *mount_point* | *mount_point*

Specifies the directory associated with the global namespace image that you want to destroy. This must be the same directory that you specified to mount the global namespace.

If you created multiple global namespace images over the same directory (mount point), this command chooses the most recently created directory.

-vfsnumber *number*

Identifies the virtual file-system (VFS) number associated with the global namespace image that you want to destroy.

In AIX, every global namespace image has a unique VFS number. The **stfsmount** command displays this number when it creates the global namespace image.

-force

Unmounts the file system even if it is in use.

-quiet

Turns off informational messages for this command. This parameter does not affect error messages.

Prerequisites

You must have root privileges to use this command.

Description

This command unmounts a global namespace image on the client system. It is used in place of the AIX **umount** command.

After you unmount all global namespace images that are linked to a specific virtual client, you can destroy the virtual client using the **stfsclient -destroy** command.

To see what global namespace images currently exist, use the AIX **mount** command with no parameters.

Example

Unmount the global namespace The following example unmounts the global namespace on the local client:

```
stfsumount -mountpoint /mnt/SANFS_MOUNTPT
```

Linux-client commands

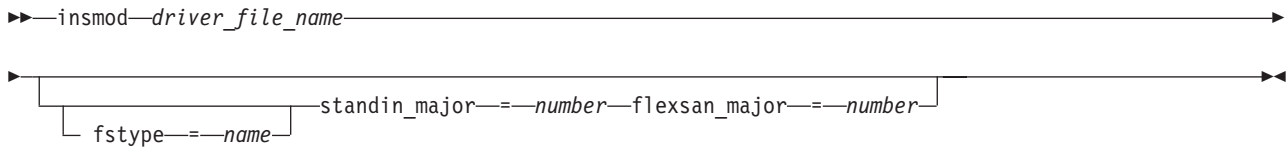
This topic provides a brief description for each Linux-client command.

Note: You must have root privileges to use these commands.

Command	Description
Migration	
"migratedata" on page 119	Migrates data to SAN File System.
Virtual client setup and removal	
"insmod" on page 134	Loads the file-system driver as a kernel module.
"rmmod" on page 134	Unloads the file-system driver as a kernel module.
"rmstclient" on page 122	Unmounts the global namespace, removes the SAN File System client, and unloads the file-system driver.
"setupstclient" on page 122	Configures and starts a client.
"stfsclient" on page 135	Creates or destroys a virtual client.
"stfsmount" on page 137	Mounts the global namespace.

insmod

Loads the file-system driver as a kernel extension.



Parameters

driver_file_name

Specifies the name and location of the file-system driver that you want to load. The file name is typically **stfs.o**.

fstype= *name*

Specifies the name of the file-system type. The file system must be of the STFS type, but you can choose any name for that type when you load the file-system driver on the client system. Use the same file-system type name when you create or destroy the client. The default name is **stfs**.

Use this parameter when you load multiple instances of the file-system driver on the same client system. The file-system type name connects the file-system driver instance with a global namespace image.

standin_major= *number*

Specifies the major device number for SAN File System. The default is 13.

flexsan_major= *number*

Specifies the major device number for the flexible SAN. The default is 15.

Prerequisites

You must have root privileges to use this command.

Description

This command creates a file-system-driver instance by loading the file-system driver as a kernel extension. After loading the file-system driver, you can use the **stfsclient** command to create a virtual client and then use the **stfsmount** command to mount the global namespace.

Example

Load the file-system driver The following example loads the file-system driver on a client for Linux:

```
insmod /usr/tank/client/bin/stfs.o
```

rmmod

Unloads the kernel extension and destroys the instance of the file-system driver.



Prerequisites

You must have root privileges to use this command.

Description

This command unloads the kernel extension and destroys the instance of the file-system driver. Use the **stfsclient** command to destroy all instances of the virtual client before unloading the module.

Example

Unload the file-system driver The following example unloads the file-system driver on a client for Linux:

```
rmmod stfs
```

stfsclient

Creates or destroys a virtual client.

```
▶▶ stfsclient -create [client_name] [server_name] [server_IP_address] [:port] [-fstype name]
▶ [quiet] [-nettype tcp|udp] [-devices dir=directory|pat=/dev/sd*[a-z]|pattern]
▶ -converter 8859-1
```

or

```
▶▶ stfsclient -destroy [client_name] [-fstype name]
```

Parameters

-create

Creates a new virtual client.

-destroy

Destroys an existing virtual client.

client_name

Identifies the unique name of the virtual client that you want to create or destroy. Each client connecting to the metadata servers must have a unique client name. The default client name is the host name of the client system.

server_name

Specifies the host name of a metadata server in the SAN File System. The metadata server that you specify informs the global namespace image of all other metadata servers.

This parameter is not required if this is not the first mount for a particular virtual client.

server_IP_address

Specifies the IP address, in dotted decimal notation, of a metadata server in the SAN File System.

port

Specifies the port number of the specified metadata server. The default is 1700.

-fstype *name*

Specifies the name of the file-system type. The file system must be of the STFS type, but you can choose any name for that type when you load the file-system driver on the client system. You must use the same name that you used to load the file-system driver. The default name is **stfs**.

Use this parameter when you load multiple instances of the file-system driver on the same client system. The file-system type name connects the file-system driver instance with a global namespace image.

-devices

Determines which devices, also called disks or logical unit numbers (LUNs), that the virtual client considers as SAN File System volumes. The default is **-devices=pat='/dev/sd*[a-z]**, where [a-z] represents any single alphabetic characters (a-z). Enclose the pattern in single quotes so the shell does not expand the asterisk characters.

In addition to creating the virtual client, this command discovers which disks, or candidates, are available to the virtual client as volumes and transmits the candidate list to the virtual client. The **-devices** parameter controls the candidates list.

dir=*directory*

The candidate list is made up of those devices that have device special files in the specified directory (for example:
-devices=dir=/dev/stfsdisk).

The easiest way to mount the global namespace is to specify **-devices=pat=/dev/sd*[a-z]**, where * represents any alphanumeric character (a-z, A-Z, 0-9). Specifying the parameter in this way causes the client to look at every SCSI-disk-like device in the system. Whatever looks like a SAN File System disk is accessed when the metadata server refers to that disk's SAN File System disk identifier. For SDD devices, specify **-devices=pat=/dev/vpath*[a-z]**.

If you want the client to be more selective about what disks it considers available, you can create a **/dev/stfsdisk** directory, put device-special files (or symbolic links) for your candidates in it, and use **-devices=dir=/dev/stfsdisk**.

pat=*pattern*

The candidate list is made up of those devices that have device-special files whose file specifications match the specified pattern. You can use * wildcards in the last (filename) component but not in the directory components (for example, **-devices=pat=/dev/sd*[a-z]**). For SDD devices, specify **-devices=pat=/dev/vpath*[a-z]**.

none The candidate list is empty. Use this value when you want to establish the candidate list with a separate command, perhaps using a selection method more sophisticated than the **stfsclient** command offers.

-quiet

Turns off informational messages for this command. This parameter does not affect error messages.

-nettype

Specifies the protocol to be used between the client and server (UDP or TCP). All clients must use the same protocol. The default is TCP.

Prerequisites

You must have root privileges to use this command.

Description

This command creates or destroys a virtual client. A *virtual client* is an entity that communicates with a metadata server and, indirectly, with other SAN File System clients. In this release, only one virtual client can be used per client machine. The terms *virtual client* and *client* can be used interchangeably.

A virtual client is associated with exactly one SAN File System. There is one file cache and one set of disk candidates per virtual client. Each virtual client that is running on the same system is as separate as if it were running on a different system. They share nothing except the file-system drive code that they execute.

A SAN File System virtual client is uniquely identified in the context of its file-system driver, and in the context of its SAN File System, by its client name.

To use the files in a global namespace, the virtual client must have a global namespace image. Creating a global namespace image makes the directory structure in the global namespace appear in the client's file structure. To create a global namespace image, use the **stfsmount** command.

A client can access and create data that is stored in a global namespace. Each virtual client can access data on multiple images in the same global namespace.

The client considers a file to be one file even if it appears with two different file names in two different global namespace images.

For Linux clients, to view the existing SAN File System virtual clients, look in the `proc/fs` file system, in the directory named after the file-system type (usually **stfs**). In that directory, there is a subdirectory for each virtual client. The name of the subdirectory is the same as the client name.

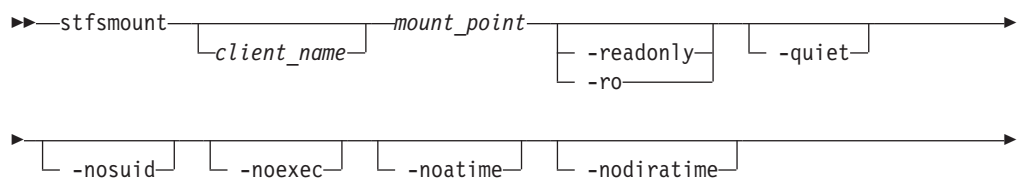
Example

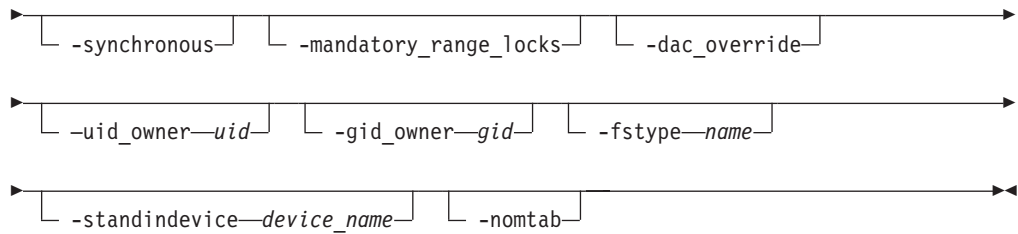
Create a virtual client The following example creates a virtual client with the candidates as volumes that have device-special files, with file specifications that match the pattern `sd*[a-z]` for non-SDD devices. (For SDD devices, the pattern would be `vpath*[a-z]`):

```
stfscient -create MDS1:1700 -devices=pat=/dev/sd*[a-z] -converter 8859-1
```

stfsmount

Mounts the global namespace.





Parameters

client_name

Identifies the unique name of the virtual client to which you want to mount the global namespace. The client must be up and running. The default client name is the host name of the client system.

mount_point

Specifies the directory associated with the global namespace image that you want to mount.

-readonly | -ro

Sets the global namespace image to read only. If specified, an attempt to update data or metadata in the global namespace will fail, and an attempt to access a file-system object will not update its access-time attribute.

-quiet

Turns off informational messages for this command. This parameter does not affect error messages.

-nosuid

Ignores the set user ID and set group ID flags on files that are accessed in the global namespace image. You cannot get special privileges by executing these files.

-noexec

Ignores the exec flag on files accessed in the global namespace image, except to indicate its existence. You cannot execute files.

-noatime

Specifies not to update the access-time attribute when a file-system object is accessed in the global namespace image.

Note: The virtual client does not update the access time of a file-system object when it accesses its attributes. It only updates the access time when it accesses its contents.

-nodiratime

Specifies not to update the access-time attribute when a directory is accessed in the global namespace image.

Note: The virtual client does not update the access time of a directory every time it is accessed. It updates the access time when you read the directory, but not when you look up a specific name in it.

-synchronous

Specifies not to return system calls that update file data or metadata in this global namespace image until after the updates have been committed to storage.

Without this parameter, system calls return as soon as the changes are stored in kernel-cache memory, and the changes are committed to storage some time later (automatically or by user request with the `/bin/sync` command).

-mandatory_range_locks

Allows mandatory range locking to be available in the global namespace image.

Without this parameter, all range locks set or seen through the global namespace image are advisory, as defined by POSIX. This means that you can lock a range of a file, but no one has to respect that lock, and you do not have to respect anyone else's locks.

With the parameter, some range locks can be mandatory to an extent. Range locks do not have an advisory or mandatory state. But when you access a range of a file that is locked, Linux treats the lock as mandatory or advisory depending on whether the file image that you access is in mandatory or advisory range lock mode. A file image is in mandatory-range-lock mode if it was created while the global namespace image had mandatory range locks allowed and the file had the set group ID flag on and the group execute flag off.

Mandatory range locks are considered mandatory only with respect to a particular global namespace image.

A client does not know whether another client considers a lock mandatory or advisory. So, if you access a file using multiple processes on the same system, through the same global namespace image, mandatory locks are fully mandatory. But another client, or even a process accessing the same client from a different global namespace image, is free to consider the locks advisory.

When the owning group of a file changes, its set group ID flag gets turned off, and range locks are no longer mandatory. This is normal behavior for the set group ID flag, but in many Linux file-system types, it is suspended for the special case of the set group ID flag that means mandatory range lock. SAN File System does make that special exception because the flag does not have the same meaning on non-Linux client. Note, that as with any Linux mandatory range lock, they do not restrict access to the file using a private mmap.

You can change the mandatory-range-locks-allowed status of a global namespace image by remounting.

The mandatory-range-lock status takes effect when a file is opened. Changing the mandatory-range-lock attribute for the global namespace image after you open the file does not change the file's status. For example, if you have a file open with mandatory range locks allowed, and then you disallow mandatory range locks, all locks on that file, including subsequent ones, are still mandatory until you close the file.

-dac_override

Enables capability to override regular permissions on a file, regardless of root client status.

-uidowner *uid*

Sets the owner user ID for each file accessed in the global namespace image. The owner user ID stored in the file system is ignored.

If not specified, the global namespace image uses the real owner user ID stored in the file system.

Note: You cannot set the owner user ID in the global namespace. An attempt to set it to user ID does succeed, but the global namespace does not actually get changed. An attempt to set it to anything else fails.

-gidowner *gid*

Sets the owner group ID for each file accessed in the global namespace image.

-fstype *name*

Specifies the name of the file-system type to use for the file-system-driver instance. This is the same name used to load the file-system driver.

If you do not specify a file-system-type name, the system defaults to the file-system-type named "stfs".

Use this parameter only when you load multiple instances of the file-system driver on the same client system.

-standindevice *device_name*

Specifies the device-specific file name of the standin block device for the global namespace image. If you specify a standin block device that is already being used for an existing global namespace image, this command does not create a new global namespace image. Instead, it mounts the existing global namespace image again, without disturbing the existing mounts.

In order to make Linux NFS export SAN File System files, the file-system type presents itself to Linux as a device-based file-system type, as opposed to a network file-system, such as NFS. But because the global namespace does not actually live on a device (it lives on a collection of devices and a metadata server), a block device must stand in for the global namespace wherever it makes a difference. Accordingly, every global namespace image is associated with a block device, known as the *standin block device*. The file-system driver contains a block-device driver and constitutes a set of very simple block devices whose only job is to be standin block devices. They cannot be used for anything. If you exports SAN File System files from NFS, make sure you do not use the same standin block device for two different SAN File Systems (at different times).

Note: You must specify one of the file-system-driver instance's standin block devices. You can see what these are in the standin proc file for the file-system type.

Note that the device is identified with the global namespace image by device number, not by the device-specific file name. You can delete the device-specific file after the **stfsmount** command completes.

The value of this parameter is what shows up as the file-system identifier in the Linux mount table listing in /proc/mounts.

You can create a block-device-specific file using the Linux **mknod** command.

If not specified, the **stfsmount** command uses an unused standin block device. It creates a temporary block-device-specific file for it and adds the name of that temporary file to the Linux mount table. It creates the file in the directory defined by the TMPDIR environment variable, or /tmp if TMPDIR is not defined. It uses a lock file to prevent two processes that are simultaneously running the **stfsmount** command from choosing the same unused standin block device. It creates that file in the same temporary directory as the block-device-specific file.

-nomtab

Specifies not to record the mount in the /etc/mstab file.

If not specified, the **stfsmount** command adds an entry to the `/etc/mtab` file describing the global namespace image if the mount is successful. It then locks the file using the `/etc/mtab~` lock file while it attempts the mount and updates `/etc/mtab`. If specified, this command does not attempt to lock or update `/etc/mtab`.

Note that `/etc/mtab` is obsolete and would only need to be updated if it might be referenced by existing applications. `/proc/mounts` contains more reliable information, straight from the kernel itself.

Prerequisites

You must have root privileges to use this command.

Description

This command creates an image of the global namespace on the client system by mounting a directory. When a client mounts a directory in the global namespace, that directory and its subdirectories become part of the client's directory hierarchy.

Note: This command is used in place of the **mount** command to mount the global namespace.

Before you can mount the global namespace, you must have a virtual client running on the client system. To create the virtual client, use the **stfsclient -create** command.

Remounting the global namespace image is not the same as unmounting the global namespace and then mounting it again. Rather, it changes the attributes of an existing global namespace image, such as changing from read-write to read-only mode. To remount the global namespace image you can use the Linux **mount** command.

To unmount the global namespace, use the Linux **umount** command.

Example

Mount the global namespace The following example mounts the global namespace:

```
stfsmount /mnt/sanfs -fstype sanfs
```

Solaris-client commands

This topic provides a brief description for each Solaris-client command.

Note: You must have root privileges to use these commands.

Command	Description
Migration	
"migratedata" on page 119	Migrates data to SAN File System.
Status	
"sanfs_ctl stats" on page 146	Displays statistics about the client.
Volumes and LUNs	

Command	Description
"sanfsd" on page 147	Starts the sanfsd program that responds to data LUN operation requests to the client. This program starts automatically after a successful mount. All data LUN operations are initiated from and coordinated by the metadata servers in the cluster but are actually performed by one or more clients.
"sanfs_ctl disk" on page 145	Adds, removes, and lists disks. This command is not necessary when the sanfsd program is running.
Virtual client setup	
"labellun"	Creates a Solaris label on a disk.
"mount"	Mounts the global namespace.
"rmstclient" on page 122	Removes the virtual client for Solaris.
"setupstclient" on page 122	Configures and starts a client.

labellun

Creates a Solaris label on a disk.

```

▶▶ labellun [-f] [-s] diskname ▶▶

```

Parameters

- f Forces the system to create a Solaris label on the disk even if a label already exists.
- s Runs silently, turns off confirmation messages for this command.

diskname

Identifies the disk on which you want a Solaris label.

Prerequisites

You must have root privileges to use this command.

Description

When a Solaris client first discovers a volume, a disk label is created in sector zero of the storage device to enable the Solaris driver to address the LUN as if it were a regular SCSI disk. The **labellun** command enables you to label your own LUN manually if you do not want to rely on it being done automatically.

Example

Label a disk The following example labels /dev/dsk/c2t9d5s2 as a Solaris disk:
labellun /dev/dsk/c2t9d5s2

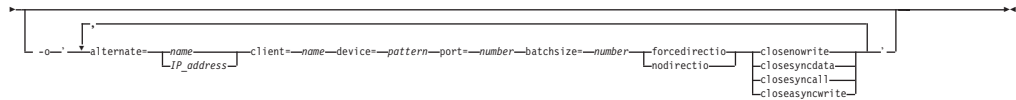
mount

Mounts a client instance of the SAN File System.

```

▶▶ mount -server_name mount_point [-r] [-m]
      server_ip_address

```

Parameters

server_name

Specifies the host name of a metadata server in the SAN File System. The metadata server that you specify informs the global namespace image of all other metadata servers.

server_IP_address

Specifies the IP address, in dotted decimal notation, of a metadata server in the SAN File System.

mount_point

Specifies the directory on which the client instance is mounted.

-r Sets the global namespace image to read only. If specified, an attempt to update data or metadata in the global namespace fails, and accessing a file-system object does not update its access-time attribute.

-m

Specifies not to record the mount in the /etc/mtab file.

If **-m** is not specified, the **mount** command adds an entry to the /etc/mtab file describing the global namespace image if the mount is successful. It then locks the file using the /etc/mtab~ lock file while it attempts the mount and updates /etc/mtab. If specified, this command does not attempt to lock or update /etc/mtab.

Note that /etc/mtab is obsolete and would only need to be updated if it might be referred to by existing applications. /proc/mounts contains more reliable information, straight from the kernel itself.

-o Specifies options that are specific to SAN File System. Enclose the options in single quotes and separate the options with a comma and no space.

alternate= *name* | *IP_address*

Specifies the host name or the IP address, in dotted decimal notation, of an alternate metadata server in the SAN File System if the metadata server specified by the **-server** parameter does not respond.

client= *name*

Identifies the unique name of the virtual client to which you want to mount the global namespace. The client must be up and running. The default client name is the host name of the client system.

device= *pattern*

Specifies the devices (also called disks or LUNs) that are usable for I/O operations. The default is /dev/dsk/c*t*d*s2. For SDD devices, specify /dev/dsk/vpath*c.

port= *number*

Specifies the port number of the specified metadata server to which the client should connect. The default is 1700.

batchsize= *number*

Specifies the maximum number of SAN File System messages that can be batched together in a single network request to the metadata server.

forcedirectio

Causes all file access on this mount instance to use direct I/O.

nodirectio

Inhibits the use of direct I/O on this mount instance.

closenowrite

Specifies that when a file is closed, the system not attempt to write any unwritten data back to disk at that time. Solaris flushing mechanisms or pageout operations will force the data to disk.

closesyncdata

Specifies that only file data, not metadata, be written back to disk when a file is closed. Metadata will be cached and written automatically or by user request using the `/bin/sync` command.

closesyncall

Specifies that file data and metadata be written back to disk when a file is closed.

closeasyncwrite

Specifies that file data and metadata be asynchronously written back to disk when a file is closed. The operations to write the data to disk start when the file is closed. However, the `close()` call does not wait for the operations to complete; it returns prior to all data being written to disk.

Prerequisites

You must have root privileges to use this command.

Description

This command creates an image of the global namespace on the client system by mounting a directory. The global namespace maintains a list of its directories that are available to the clients. When a client mounts a directory in the global namespace, that directory and its subdirectories become part of the client's directory hierarchy.

To unmount the global namespace, use the Solaris **umount** command.

Example

Mount the global namespace The following example mounts the global namespace:

```
mount MDS1 /mnt/sanfs -o 'alternate=MDS2,alternate=MDS3,device=/dev/dsk/c1t2d*s2'
```

rmstclient

Removes the virtual client for Solaris.

**Parameters****-prompt**

Prompts for required parameters, using values from the configuration file, if available.

-noprompt

Runs silently, using parameters from the configuration file (/usr/tank/client/config/stclient.conf). If a required parameter is not available, the command exists with an error.

Prerequisites

You must have root privileges to use this command.

You must unmount the SAN File System before invoking this command.

Example

Remove a client for Solaris The following example removes the local SAN File System client for Solaris without prompting:

```
rmstclient -noprompt
```

sanfs_ctl disk

Controls the SAN File System volumes (disks) that a client can access and lists accessible disks. The need to use this command is greatly reduced when the sanfsd program is running, because the sanfsd program responds to data LUN operations from the metadata servers. This command only provides another means to check and control the disks that a specific Solaris client can access.

```
▶▶ sanfs_ctl disk [add|remove] -instance mount_point [disk_name]
```

or

```
▶▶ sanfs_ctl disk [list|release] -instance mount_point
```

Parameters

add

Adds the specified disk file name to the list of disks that the Solaris client uses for I/O operations.

remove

Removes the disk from the list of disks that the Solaris client uses for I/O operations.

list

Lists all the disks that are accessible from the given instance.

-release

Removes all the disks that are accessible from the given instance.

-instance mount_point

Specifies the directory on which the client instance is mounted.

-disk disk_name

Specifies one or more names of the disks to add or remove. When you specify more than one name, separate the names with a comma and no space (for example, -disk /dev/dsk/disk1,/dev/dsk/disk2).

Prerequisites

You must have root privileges to use this command.

Description

The **sanfs_ctl disk** command enables manual control over the disks used by the Solaris client when it performs I/O operations. When the `sanfsd` program is running, there is little need to use this command.

Example

List accessible disks The following example lists all the disks that are accessible for the local client for Solaris that is mounted on `/mnt/sanfs`:

```
sanfs_ctl disk list -instance /mnt/sanfs
```

sanfs_ctl stats

Displays statistics about the Solaris client.

```
▶▶ sanfs_ctl stats get -instance mount_point [-data data_type] [-short]
```

or

```
▶▶ sanfs_ctl stats list
```

or

```
▶▶ sanfs_ctl stats reset -instance mount_point
```

or

```
▶▶ sanfs_ctl stats monitor -instance mount_point [-host host_name] [-p port]
```

```
▶ -interval interval
```

Parameters

get

Retrieves performance information for a single, multiple, or all data items.

list

Lists the names of all statistics that can be gathered.

reset

Resets the statistics for all the data types.

monitor

Specifies that the utility continuously generate statistics.

-instance *mount_point*

Specifies the directory on which the client instance is mounted.

-data *data_type*

Identifies the data types for which the utility gathers statistics. When you specify more than one data type, separate the data types with a comma and no space. The data types can be the following values:

- tm - Transaction Manager (TM) statistics per client related to TM data structures, including the number of messages sent and received (per message type), the maximum lengths and average lengths of the transaction queue, and the number of transactions, messages, and leases lost.
- net - Network statistics per metadata server with which this client has a lease.
- mc - Metadata-cache statistics.
- ho - The states of all objects in the object hash table, including object ID, flags, and attributes.
- lo - The states of all objects in the object LRU list, including object ID, flags, and attributes.
- ls - Mutex and latch statistics of metadata cache objects. Statistics include the number of attempts, wait time, and hold time.
- le - Mutex and latch state of all objects in the object hash table, including current holder, hold state, and waiters.

-short

Displays the minimal information for the specified data items.

-host *host_name* **-port** *port_number*

The statistics are sent as a string to the specified *host_name* on the specified UDP *port*. If no host or port is specified, the statistics are sent to stdout.

-interval *interval*

Specifies the *interval* (in seconds) at which the utility generates statistics when the monitor parameter is specified.

Prerequisites

You must have root privileges to use this command.

Description

The **sanfs_ctl stats get** command enables you to retrieve statistics about specific or all data types. You can use the **sanfs_ctl stats list** command to see a list of the available data types. The **sanfs_ctl stats reset** command enables you to reset the gathering of these statistics.

Example

Displays statistics from a client for Solaris The following example displays transaction-manager statistics and metadata-cache statistics for the local client for Solaris that is mounted on `/mnt/sanfs`:

```
sanfs_ctl stats get -instance /mnt/sanfs -data tm,mc
```

sanfsd

Starts the `sanfsd` program on a Solaris client. This program responds to data LUN operation requests from metadata servers. The `sanfsd` program is started automatically when the **mount** command succeeds so you would not need to ever invoke this command under normal conditions.

Parameters

- instance *mount_point*
Specifies the directory on which the client instance is mounted.
- device *pattern*
Determines which devices (also called disks or LUNs) that the client considers as SAN File System volumes. An example device pattern is /dev/dsk/c*t*d*s2. For SDD devices, specify /dev/dsk/vpath*c. Enclose the pattern in single quotes so the shell does not expand the asterisk characters.

Prerequisites

You must have root privileges to use this command.

Description

The sanfsd program starts automatically when the **mount** command successfully runs a mount invocation. A sanfsd program runs for each SAN File System mount. The sanfsd program stops when its accompanying mount point is unmounted. If the program is not running for a particular mount, the client cannot respond to data LUN operations until the program is restarted.

Example

Start the sanfsd program. The following example starts the sanfsd program for the local client for Solaris that is mounted on /mnt/sanfs and specifies non-SDD devices. For SDD devices, you would specify –device '/dev/dsk/vpath*c':
sanfsd –instance /mnt/sanfs –device '/dev/dsk/c*t*d*s2'

Windows-client command

The topic provides a brief description for the Windows-client command.

Note: You must have Administrator privileges on the Windows client to use this command.

Command	Description
Migration	
"migratedata" on page 119	Migrates data to SAN File System.

For information about changing Windows client parameters, see Changing Windows client properties.

Service commands and utilities

Service commands

Administrative commands

The following table provides a brief description and role for each command in the administrative CLI that is intended for use only by trained service personnel.

Command	Description	Role	Environment
Administrative server			
"startCimom" on page 160	Starts the administrative agent.	root	shell
"stopCimom" on page 164	Stops the administrative agent.	root	shell
Cluster			
"mktruststore" on page 152	Obtains a public certificate from LDAP and creates a truststore that is shared by the administrative infrastructure to certify secure connections. This command replaces any existing truststore on the local engine.	n/a	shell
"tank extractbootrecord" on page 165	Extracts the product (disk) label information contained in the master volume and creates a local copy of the Tank.Bootstrap file.	Administrator	shell
"tank lscluster" on page 166	Displays the cluster definition from a system master volume when the metadata servers are not running.	Administrator	shell
"tank lsdisklabel" on page 168	Displays the SAN File System product (disk) label for a specified device.	Administrator	shell
"tank lsversion" on page 168	Displays the version control information from a system master disk.	Administrator	shell
"tank resetcluster" on page 170	Erases the static cluster definition contained on the system master volume, without reinitializing the metadata, which in affect drops all metadata servers from the cluster at one time.	Administrator	shell
"tank resetversion" on page 171	Resets the version-control information on a system master disk.	Administrator	shell

Command	Description	Role	Environment
Engine			
“obdc” on page 153	Collects information for the first-failure data capture of SAN File System problems, from either a client machine or the storage engine.	Operator, Administrator	shell
SAN File System console			
“disableConsoleTrace”	Disables tracing and logging for the SAN File System console.	root	shell
“enableConsoleTrace” on page 151	Enables tracing and logging for the SAN File System console.	root	shell
“startConsole” on page 160	Starts the SAN File System console Web server.	root	shell
“stopConsole” on page 165	Stops the SAN File System console Web server.	root	shell

Client commands

The following table provides a brief description and role for each client command that is intended for use only by trained service personnel.

Command	Description	Role
Diagnostics		
“obdc” on page 153	Collects information for the first-failure data capture of SAN File System problems from the client machine.	root (AIX) or Administrator (Windows)
“sanfstrace” on page 157	Enables and disables tracing.	root or Administrator
“stfsdriver” on page 129	(AIX only) Loads the file-system driver as a kernel extension. Note: This command may be used by administrative users. Only specific options in this command are intended for use only by trained service technicians.	root
“stfsstat” on page 161	(AIX only) Display statistics about the client.	root

disableConsoleTrace

Disables tracing and logging for the SAN File System console.

▶▶—disableConsoleTrace—◀◀

Prerequisites

This task must be performed only by trained service technicians.

You must have root privileges to use the command.

The Web server must be running.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This command is case sensitive and must be entered as shown.

Tracing is automatically disabled after you restart the Web server.

Example

Disable tracingThe following example disables tracing and logging for the SAN File System console:

```
#disableConsoleTrace  
Disabling SAN File System console tracing...
```

enableConsoleTrace

Enables tracing and logging for the SAN File System console.

▶▶—enableConsoleTrace—◀◀

Prerequisites

This task must be performed only by trained service technicians.

You must have root privileges to use the command.

The Web server must be running.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This command is case sensitive and must be entered as shown.

Tracing is automatically enabled after you restart the Web server.

This command saves log and trace records to the /opt/was/logs/serverx/trace.log file.

Enabling tracing impacts the performance of SAN File System. Use this command only when necessary.

Example

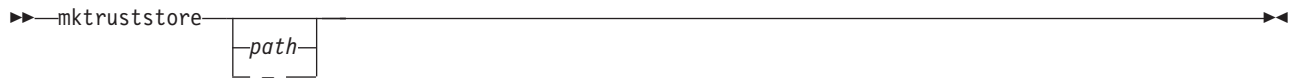
Enable tracing

The following example enables tracing and logging for the SAN File System console:

```
enableConsoleTrace  
Enabling SAN File System console tracing...
```

mktruststore

Obtains a public certificate from LDAP and creates a truststore that is shared by the administrative infrastructure to certify secure connections. This command replaces any existing truststore on the local engine.



Parameters

path

Specifies the full directory path to import an existing LDAP certificate into the new truststore file being created. The certificate cannot be added to an existing truststore because a new truststore will be created. If not specified, an LDAP certificate will not be added to the new truststore.

Note: An LDAP certificate is required to obtain secure communication with the LDAP server.

- Specifies that in single-shot command mode you want this command to receive from the input stream (stdin) the path to import an existing LDAP certificate.

Prerequisites

The `cimom.properties` file must be set up prior to using this command.

The `tank.properties` file must exist. It is used by the **mktruststore** command to determine the language being used.

This task must be performed only by trained service technicians.

Description

Note: This command is run from the shell prompt from the `/usr/tank/admin/bin` directory. It is not run inside of `sfcli`.

This command creates a new truststore file and places it in the `/usr/tank/admin` directory.

You would use the **mktruststore** command in these circumstances:

- During installation, the **mktruststore** command is run automatically by the **setupTank** script on the first engine. Because the truststore file must be exactly the same on every engine in the cluster, you must copy the truststore file from one engine to each remaining engine in the cluster before running `setupTank` on those engines.
- When replacing an expired truststore file. The truststore file is valid for one year.
- When you need to change the truststore (for example, if security is breached). You may change the truststore at any time; however, it must be the same on every engine in the cluster. The Administrative agent must be restarted any time

the truststore is changed by issuing the **stopcimom** and **startcimom** commands. The user interfaces (SAN File System console and Administrative command-line interface) connection will be broken when the Administrative agent is stopped.

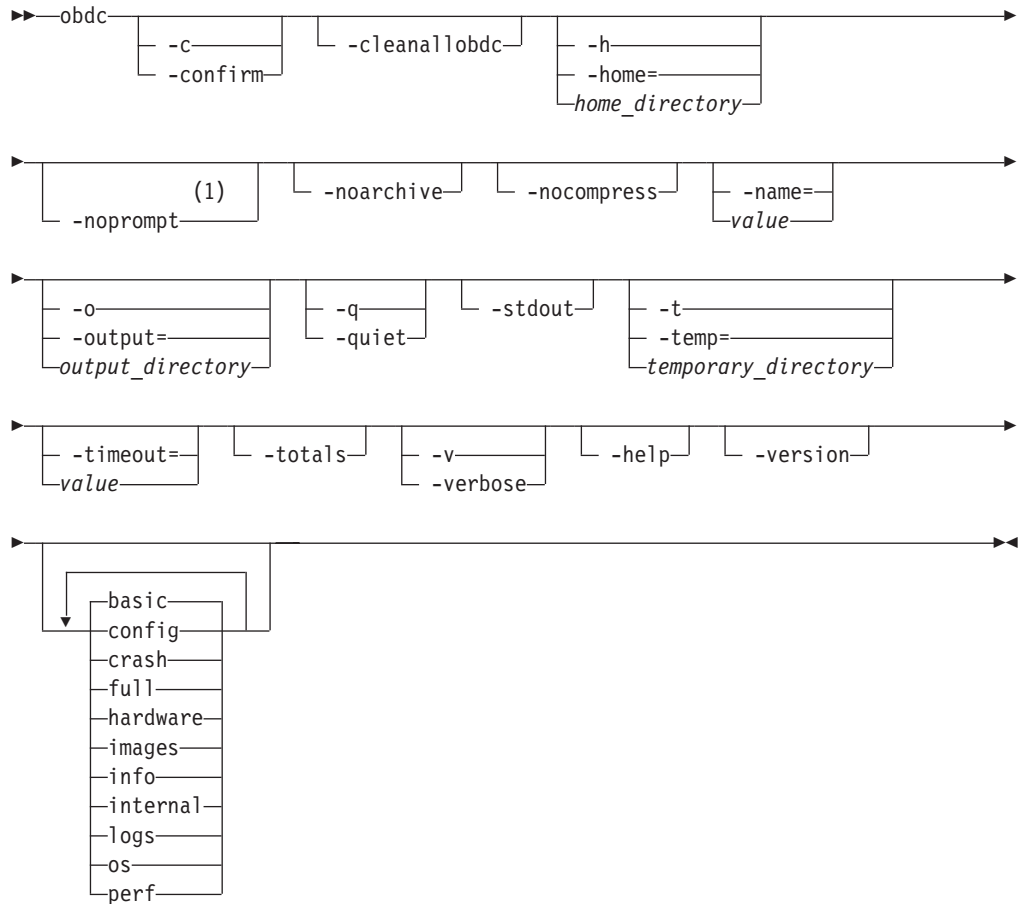
Example

Create truststore The following example creates a truststore and imports the LDAP certificate ldap.cert from the local directory:

```
mktruststore ldap.cert
Creating truststore file.
The truststore was created successfully
Certificate was added to keystore.
```

obdc

Collects information for the first-failure data capture of SAN File System problems, from either a client machine or the storage engine.



Notes:

- 1 When specifying the --noprompt option, the responses to all questions are automatically yes. Be sure that this will produce the expected results.

Parameters

-c | --confirm

Requests that the user be required to provide a confirmation for each piece of data that is collected.

--cleanallobdc

Removes all old one-button data collection (OBDC) archives and core files from the system. The user is asked to confirm each deletion.

-h | --home= *home_directory*

Specifies the root of the installation directory for the SAN File System component installed on the local machine. The default is /usr/tank. The user can specify the home directory as a fully qualified path name only.

--noprompt

Specifies that the command will not prompt the user for the answers to questions. Instead, it is to assume that the answers to all questions are affirmative.

Attention: This option applies to all other specified options, including the --cleanallobdc option, so be sure that you consider the results.

--noarchive

Specifies that the command is to generate the data in a directory tree below the target directory, instead of archiving the data into a zip file. The name of the target directory is **OBDC-*mmddyy-time-obdc_PID***, where *obdc_PID* is the process ID for the OBDC command instance. For example, OBDC-012204-1312-28494.

--nocompress

Specifies that the command not compress the archive file. If you specify --nocompress, the name of the archive file is **OBDC-*mmddyy-time-obdc_PID*.tar**, where *obdc_PID* is the process ID for the OBDC command instance. For example, OBDC-012204-1312-28494.tar. If you do not specify --nocompress, the name of the archive file is **OBDC-*mmddyy-time-obdc_PID*.tar.gz**, where *obdc_PID* is the process ID for the OBDC command instance. For example, OBDC-012204-1312-28494.tar.gz.

--name= *value*

Assigns a specific name to the resulting tar file.

-o | --output= *output_directory*

Specifies the directory in which to assemble the collected data. The default is /usr/tank/OBDC. For Windows clients, the output appears in Documents and Settings\Administrator\Application Data\IBM\Storage Tank\OBDC.

-q | --quiet

This mode does not display any output.

--stdout

Specifies that only information about how the command ran be sent to standard output.

-t | --temp= *temporary_directory*

Specifies an alternate directory in which to store temporary files. The default is /tmp.

--timeout *value*

Specifies the amount of time, in seconds, before stopping an attempt to collect a single piece of data. The default is 30 seconds.

--totals

Displays the size and number of files acquired.

-v | --verbose

Enables verbose mode.

--help

Displays a detailed description of this command, including syntax, parameter descriptions, and examples. If you specify a help option, all other command options are ignored.

--version

Displays information about the version of the OBDC program.

basic, config, crash, full, hardware, images, info, internal, logs, os, perf

Specifies the type and amount of data to be collected. You can specify more than one value, separated by a space between each type.

basic Collects only a small amount of fundamental system data, which is useful for addressing simple usage and configuration questions. This information includes configuration files, the internal state of the metadata server or client, operating system statistics, hardware specifications and log files.

This is the default collection option if no command-line options are specified.

config Collects the configuration files for a metadata server or client.

crash Collects crash dump files, such as operating system dumps, if they exist, or core files from the metadata server.

Tip: Using the **crash** value can cause the **obdc** command to collect a very large amount of data. For example, a client machine with 2 GB of memory produces a core file of the same size. (The binary file for the operating system can also be somewhat large). Therefore, you must take steps to ensure the availability of sufficient disk space in the output directory before invoking the **obdc** command with this value.

full Collects all of the data necessary to handle the majority of problems. It includes the information collected with the basic options plus crash dump files and performance information.

hardware

Collects hardware specifications.

images

Collects installation files.

info

Collects information about the metadata server.

internal

Collects different types of runtime state information for either the client or the metadata server.

logs

Collects log files from the metadata server or client.

os

Collects operating system statistics.

perf

Gathers statistics and metrics about the performance of a metadata server or client, which is useful in cases where system response has degraded.

Prerequisites

This task must be performed only by trained service technicians.

C++ requirements for UNIX operating systems

The following table shows the minimum required C++ runtime for OBDC to function on UNIX operating systems.

Table 10. C++ minimum requirements

Operating System	Requirements
AIX 5.1, 5.2, and 5.3	vacpp.cmp.rte version 6.0.0.0 or xIC.rte version 6.0.0.0
Red Hat Linux 3.0	libstdc++-2.96-110
SuSE Linux 8	libstdc++-2.96-110
Sun Solaris 9	Minimum requirements installed with the operating system.

Description

The **obdc** command gathers data for diagnosing errors or failures associated with metadata servers and clients. This command is intended primarily for first-failure data-capture capabilities useful for investigating problems upon their initial occurrence, without requiring problem recreation or subsequent tracing.

This command must be invoked natively on the metadata server or client machine from which you want to collect information. For a metadata server, this command can be invoked using the SAN File System console or the administrative command-line interface (CLI). The CLI command is named **collectdiag**. For a metadata server or client, the command can be invoked from a command shell (on UNIX) or a command prompt (on Windows). The SAN File System console provides true one-button running of the command against a specified metadata server, but it always issues the command with the default parameters for all parameters. Invoke the command using one of the alternate methods if you need to specify non-default parameters.

Privileges

The **obdc** command relies upon a number of SAN File System and operating system commands that require certain administrative privileges. To run the command, you must be logged in as root on a UNIX platform or Administrator on a Windows platform.

Collected data

The **obdc** command collects data for metadata servers, administrative servers, and clients of a SAN File System configuration, regardless of platform. The command relies both on SAN File System administrative commands and on operating-system commands to obtain much of the data it collects. It creates a GZIP-compressed tar file (by default) in the time-stamped output directory to record the results of the administrative and operating system commands that it invokes. In the event that the process from which data is requested is unavailable (for example, if the metadata server is unavailable or unresponsive), the command records a suitable message and continues without collecting data from the unavailable process. This command also assembles various configuration, log, trace, and core files in the output directory.

The **obdc** command collects information related to both hardware and software. The specific information gathered depends on the platform, the SAN File System configuration of the machine, and the parameters specified. In general, this command collects more data for metadata servers than for clients, due to the greater function and complexity of the metadata server configuration.

For hardware, this command collects information about:

- Processors
- Memory
- Network adapters (LAN and SAN)
- Storage devices (local and SAN)

For software, this command collects information about:

- Operating system
- Network configuration (LAN and SAN)
- Storage configuration (local and SAN)
- SAN File System configuration

Within these general categories, the command gathers essential component information (for example, name, version, and configuration), current status, logged messages, trace and diagnostic output, core files, and performance statistics and metrics, among other things. Not all types of information pertain to all hardware and software components about which information is collected.

Example

Collecting basic information plus installation files The following example collects the basic data and installation files. The collected data is archived and compressed into a single output file for delivery to remote support personnel.

```
/usr/tank/client/bin/obdc basic images
```

sanfstrace

Enables and disables tracing.

```
▶▶ sanfstrace help ◀◀
```

or

```
▶▶ sanfstrace list ◀◀
```

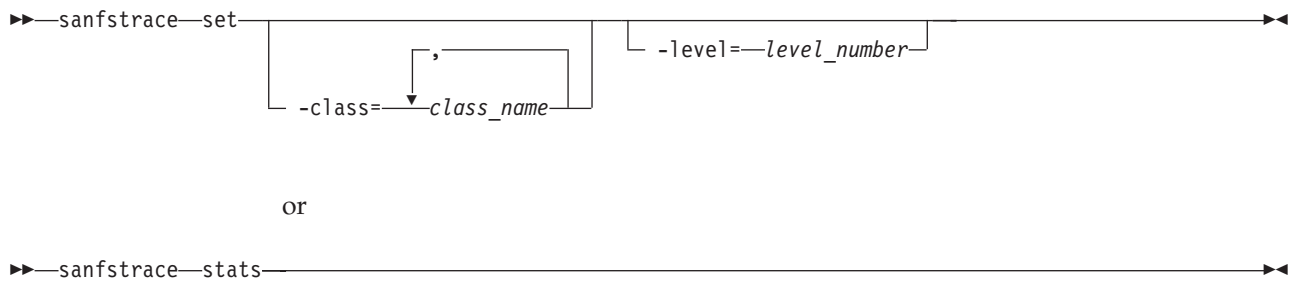
or

```
▶▶ sanfstrace log [ -delay=time ] [ -f=trace_log_file ] ◀◀
```

or

```
▶▶ sanfstrace ringsize size ◀◀
```

or



Parameters

help

Displays a description of this command.

list

Lists trace status, that is, the trace class names and IDs of client driver debugging messages and the current tracing level for each class. A tracing level of zero indicates that the trace class is disabled.

log

Specifies that the trace utility continuously log trace messages in a specified trace file or to the standard output stream. After the trace utility has logged a message, it will not be able to retrieve it again. To end this function use `Ctrl+C`, when the function is running in the foreground, or use the `kill -9 process_id` command, when it is running in the background.

ringsize *size*

Sets the size, in kilobytes, of the trace ring buffer. You can specify a size of 4 to 512.

set

Sets the tracing level for one or more specified trace classes of client driver debugging messages or all trace classes.

stats

Displays trace statistics.

`-class=class_name`

Identifies one or more classes for which to set the tracing level. The `-level` parameter must immediately follow a `-class` parameter. When you specify more than one class name, separate the names with a comma or a space. A maximum of 20 class names can be specified for each `-class` parameter.

`-delay=time`

Limits the frequency that the `sanfstrace log` command queries the driver for new messages. This parameter specifies the minimum time between successive retrieval requests in order to prevent the trace utility from affecting driver performance. It creates a delay that is at least as long as the specified delay time or as long as it takes to process previously retrieved data, whichever is longer. The default is one second. The minimum value is zero, which disables the delay option, and the maximum value is 360. All delay values are in seconds by default unless the 'ms' (millisecond) quantifier is specified.

Tip: If the log reports lost messages, try reducing the value of the `-delay` parameter.




`-f=trace_log_file`


Specifies the file path and output file name for trace logging. The default is the

standard output device. If the specified trace log file exists, the trace utility opens the file and appends messages to it. If the trace log file does not already exist, the utility creates the file.

-level=level_number

Specifies the tracing level of the classes that are specified by a -class parameter that immediately proceeds the -level parameter. If a -class parameter does not proceed the -level parameter, all classes are set to the specified level. Possible values for the level number are 0 through 9. Setting the level to 0 turns off tracing for the specified class.

Restriction:    Although values for the -level parameter can be specified in the range 0 through 9, the current implementation for CSM classes uses a maximum value of 5, and the non-CSM classes only differentiate zero (disabled) from non-zero (enabled).

Note:  A trace level of 0 disables tracing, and any nonzero trace level enables tracing and is reported as 1 (enabled).

Tip: You can specify multiple sets of -class and -level parameters in a single command. If a command affects the same class multiple times, the last occurrence in the command line applies.

Prerequisites

This task must be performed only by trained service technicians. You must have root or Administrator privileges to use this command. On UNIX clients, you must have super-user privileges to use this command.

Description

The **sanfstrace set** command enables you to set on or off trace classes of client driver debugging messages. The First Failure Data Capture (FFDC) class of driver trace messages is always enabled and cannot be disabled. On Windows, the DBG_ERROR class of trace messages are also enabled in a typical client configuration. All other trace classes are disabled by default when the client driver is loaded. Use the **sanfstrace list** command to view the status of trace classes.

The **sanfstrace log** command enables you to display or write to a log file accumulated trace messages from the client driver.

Example

List trace levels The following example lists traces on a Solaris client with no tracing enabled:

sanfstrace list

Class Name		Level
-----		-----
CSM:LEASE	(ID=0)	0
CSM_XMTRCV	(ID=1)	0
CSM:OBJECT	(ID=2)	0
CSM:CACHE	(ID=3)	0
CSM:MC_SES	(ID=4)	0
CSM:CSM_SES	(ID=5)	0
CSM:MC_DATA	(ID=6)	0
CSM:MC_RNGE	(ID=7)	0

CSM:MC_RNGE	(ID=8)	0
CSM:CSM_RNGE	(ID=9)	0
CSM:ALLOC	(ID=10)	0
CSM:OBJATTR	(ID=11)	0
CSM:SOCKET	(ID=12)	0
CSM:MISC	(ID=13)	0
CSM:MSG	(ID=141)	0
ULD:CONFIG	(ID=127)	0
ULD:MOUNT	(ID=128)	0
ULD:DISK	(ID=129)	0
ULD:PAGER	(ID=130)	0
ULD:VFSOP	(ID=131)	0
ULD:VNODEOP	(ID=132)	0
ULD:IO	(ID=133)	0
ULD:VNODE	(ID=134)	0
ULD:RNGLOCK	(ID=135)	0
ULD:RDWR	(ID=136)	0
ULD:ATTR	(ID=137)	0
ULD:LOOKUP	(ID=138)	0
ULD:CLEANER	(ID=139)	0
ULD:CAPTURE	(ID=140)	0
ULD:FFDC	(ID=143)	0

startCimom

Starts the administrative agent.

▶▶—startCimom—◀◀

Prerequisites

You must have root privileges to use this command.

This task must be performed only by trained service technicians.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This utility assumes that the required files are located in the /usr/tank/admin/config directory.

Example

Start the administrative agent The following example starts the administrative agent:

```
/usr/tank/admin/bin/startCimom
```

startConsole

Starts the SAN File System console Web server.

▶▶—startConsole—◀◀

Prerequisites

You must have root privileges to use the command.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This command is case sensitive and must be entered as shown.

The Web server must be running for you to open the SAN File System console. The Web server starts automatically when the storage engine boots up. This command allows you to manually restart the Web server if it is stopped.

Example

Starts the Web server The following example starts the SAN File System console Web server:

startConsole

Starting the SAN File System console...

The SAN File System console is operational at <https://189.24.15.162:7979/tank>

stfsstat

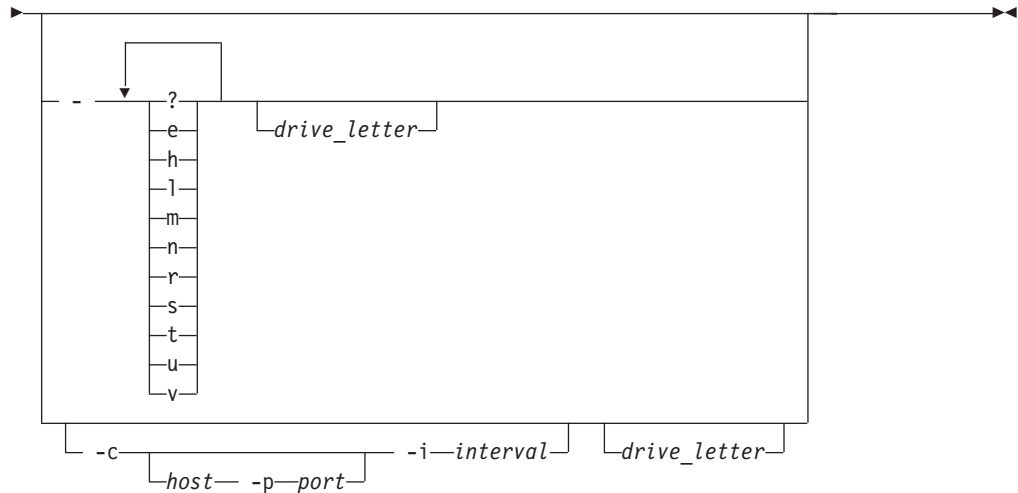
Displays statistics for the local client. Note that the **sanfs_ctl stats** command is the equivalent command on the Solaris operating system.

Clients for AIX and Linux

```
▶▶ stfsstat [ -all ] [ -cl ] [ -gio ] [ -help ] [ -ho ] [ -hos ]
[ -le ] [ -lo ] [ -los ] [ -ls ] [ -mc ] [ -net ] [ -pager ]
[ -reset ] [ -svfs ] [ -tm ]
[ -cm [ -host host_name -p port ] -i interval ]
[ -mountpoint mount_point ]
[ -vfsnumber vfs_number ]
```

Clients for Windows

```
▶▶ stfsstat
```



Parameters

Clients for AIX and Linux

-all

Displays all client statistics, except per-object statistics. This parameter is the same as specifying `-tm`, `-mc`, `-ls`, `-le`, `-svfs`, `-gio`, `-pager`, and `-cl` parameters.

-cl

Displays cleaner statistics.

-cm [-host *host_name* -p *port*] -i *interval*

Continuously monitors the specified statistics. The statistics are sent as a string to the specified *host_name* on the specified UDP *port* at every specified *interval* (in seconds). The minimum interval is 20 seconds. If no host or port is specified, the statistics are sent to stdout.

-gio

Displays I/O statistics per global disk (GDISK), including : blocks read and written, and the average number of bytes read and written.

-help

Displays a detailed description for this command.

-ho

Lists the states of all objects in the mcObject hash table, including object ID, flags, and attributes

-hos

Displays a short list of object hash table.

-le

Lists mutex and latch state of all objects in the mcObject hash table, including current holder, hold state, and waiters.

-lo

Lists the states of all objects in the mcObject LRU list, including object ID, flags, and attributes.

-los

Displays a short list of object LRU.

- ls**
Lists mutex and latch statistics of metadata cache objects such as mcInstance and mcObject. Statistics include the number of attempts, wait time, and hold time.
- mc**
Displays metadata-cache statistics.
- net**
Displays network statistics per metadata server with which this client has a lease.
- pager**
Displays pager statistics per client, including information about the page-in and page-out activity.
- reset**
Resets all statistics to zero. Some statistics, such as object and latch state, cannot be reset because no counters are incremented, but data structures are shown.
- svfs**
Displays file-system statistics. These metrics are mount-specific and distinguish between the different mount points. Statistics include the number of file, inode, address space, and super operations performed.
- tm**
Displays Transaction Manager (TM) statistics per client related to TM data structures, including the number of messages sent and received (per message type), the maximum lengths and average lengths of the transaction queue, and the number of transactions, messages, and leases lost. The statistics also include the number of transactions within certain time ranges, or buckets, for each transaction type.
- mountpoint** *mount_point*
Identifies the mount point for the file-system image for which you want to display statistics.
- vfsnumber** *vfs_number*
Identifies the virtual-file-system (VFS) number associated with the global namespace image for which you want to display statistics. In AIX, every global namespace image has a unique VFS number.

The **stfsmount** command displays this number when it creates the global namespace image.

Clients for Windows

- ?** Displays a detailed description for this command.
- c** Continuously monitors the specified statistics.
- e** Displays the latch state.
- h** Displays the object hashlist statistics.
- i** Displays the interval for sending statistics. This option must be used with the **-c** parameter.
- l** Lists mutex and latch statistics of metadata cache objects.
- m**
Displays metadata-cache statistics.

- n Displays the engine statistics.
- p Displays the port for sending statistics. This option must be used with the -c parameter.
- r Resets the CSM statistics
- s Displays the object LRU statistics.
- t Displays Transaction Manager (TM) statistics. This is the default option.
- u Displays statistics in the short-output format.
- v Displays file-system statistics

drive_letter

The drive letter mapped to SAN File System. The default letter is T.

Prerequisites

This task must be performed only by trained service technicians.

Description

On Windows machines running the client, the statistics parameters (*?*, *e*, *h*, *l*, *m*, *n*, *r*, *s*, *t*, *u*, and *v*) are appended together. For example, to choose to display the latch state and statistics, you would specify **-el**.

Note that the “sanfs_ctl stats” on page 146 command is the equivalent command on the Solaris operating system.

Example

Displays statistics from a client for AIX The following example displays transaction-manager statistics and metadata-cache statistics for the AIX client that is mounted on /mnt/sanfs:

```
stfsstat -tm -mc /mnt/sanfs
```

Displays statistics from a client for Windows The following example displays statistics for the object-hash list and latch state for the Windows client that is mapped to drive S:

```
stfsstat -he s
```

stopCimom

Stops the administrative agent.

▶▶—stopCimom—◀◀

Prerequisites

You must have root privileges to use this command.

This task must be performed only by trained service technicians.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This utility assumes that the required files are located in the /usr/tank/admin/config directory.

Example

Stop the administrative agent The following example stops the administrative agent:

```
/usr/tank/admin/bin/stopCimom
```

stopConsole

Stops the SAN File System console Web server.

▶▶—stopConsole—◀◀

Prerequisites

You must have root privileges to use the command.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

This command is case sensitive and must be entered as shown.

You would stop the SAN File System console Web server when you are upgrading software.

Example

Stops the Web server The following example stops the SAN File System console Web server:

```
stopConsole
Stopping the SAN File System console...
```

tank extractbootrecord

Extracts the product (disk) label information contained in the master volume and creates a local copy of the Tank.Bootstrap file.

▶▶—tank—extractbootrecord—◀◀

└─*-device device_path*─┘

Parameters

-device *device_path*

Specifies the device path for a valid master volume from where to read the product label.

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be stopped.

Description

This command is located in the `/usr/tank/server/bin` directory.

This command is used in disaster recovery situations or when attaching new hardware to an existing set of SAN devices.

Each metadata server in the cluster stores a local copy of the product label that is on the master volume. The local copy is stored in the file `/usr/tank/server/config/Tank.Bootstrap`, in binary format. SAN File System uses this information to locate the master volume on the SAN at boot time. After it is located, the cluster definition can be read. You would use this command when the local copy of the `Tank.Bootstrap` file is lost to extract the product label from the system master volume and recreate the local `Tank.Bootstrap` file.

This command is used to recreate the local `Tank.Bootstrap` file on the master metadata server in a single-engine cluster. The `Tank.Bootstrap` file is created on each remaining metadata server as they are added to the cluster (using the `addserver` command). It is not necessary or recommended to run this command on every metadata server or to copy this file from one metadata server to another.

Note: This command is run from the shell prompt. It is not run inside of `sfscli`.

This command allows a single-engine cluster to be brought up on new hardware in the case of disaster recovery or when you attach new hardware to an existing cluster. Recovering the `Tank.Bootstrap` file is required only when copies on all engine hosting metadata servers in the cluster are lost.

Example

Extract a boot record The following example extracts a local `Tank.Bootstrap` file from device `/dev/rsdc`. This device is a valid master volume of a SAN File System cluster instance.

```
#/usr/tank/server/tank extractbootrecord -device /dev/rsdc  
Label information from master disk /dev/rsdc was extracted and stored in  
Tank.Bootstrap.
```

tank lscluster

Displays the cluster definition from a system master volume when the metadata servers are not running.

▶▶—tank—lscluster—◀◀

Prerequisites

This task must be performed only by trained service technicians.

This command can be run in any cluster state.

Description

Note: This command is run from the shell prompt. It is not run inside of `sfscli`.

This command is used in disaster recovery situations, while troubleshooting the system, or to inspect the cluster definition.

This command is located in the `/usr/tank/server/bin` directory.

This command displays the following information:

- Master state of the state machine in the cluster services component.
- Subordinate state of the state machine in the cluster services component.
- Transport protocol used for group services. The value defaults to UDP and cannot be changed. Note that the group services includes cluster services.
- Transition state, which indicates whether a cluster transition is in progress at the time this command is issued.
- Started state, which indicates whether the group services subsystem was started.
- Joining state, which indicates whether the metadata server attempted to join the cluster.
- Forming state, which indicates whether the metadata server is attempting to reform the cluster as the master metadata server.
- Dynamic group ID, which identifies the subset of the cluster that is alive and well.
- Time when the cluster was last reformed (committed).
- Size of the current dynamic group. This is affected when the cluster or a metadata server starts, stops, crashes, or aborts.
- Size of the entire static cluster. This is affected by the **addserver** and **dropserver** commands.
- Cluster identifier, which is initialized at installation time.
- Cluster name (sanfs).
- Time of the installation.
- Timeout value used by group services.
- IP address of the network used by the cluster.
- Netmask used by the cluster.
- Current active version of the cluster boot record. Two copies are maintained. This value will be A or B.
- Identifier of the engine hosting the metadata server where this command was issued.
- A list of all engines in the cluster and attributes for each. The attributes include:
 - Engine identifier.
 - IP address.
 - Group services port number (gs)
 - SAN File System protocol port number (stp).
 - Heartbeat port number (hb).
 - Administrative port number (adm)
 - Administrative agent port number (agent)
 - Metadata server/engine name.
 - Last committed software version.

Example

Lists the static cluster definition The following example lists the static cluster definition before running the **tank resetcluster** command:

```
tank lscluster
Group information:
mast_state:      Microkernel
sub_state:       Invalid
protocol:        udp
in transition:   no
```

```

b_started:          0
b_joining:          0
b_forming:          0
group_id:           9
group commit time:  Jun 25, 2003 9:47:53 PM
group size:         4
cluster size:       4
cluster id:         1234
cluster name:       sanfs
install time:       Jun 25, 2003 4:23:45 PM
installation id:    835407414733488113
gs_timeout:         1000 (millisecs)
ip_network:         0.0.0.0 (interfaces: )
netmask:            0.0.0.0
active set:         B
this node id:       0

      id  ip addr      gs      stp      hb      gdm      agent
NODE: 0   192.168.10.88  11003   11001   11004   11002   5989
NODE: 1   192.168.11.88   11003   11001   11004   11002   5989

stnode name  SW Version
GR ST0       1.0.1
GR ST1       1.0.1

```

tank lsversion

Displays the version control information from a system master disk.

▶▶—tank—lsversion—◀◀

Prerequisites

This task must be performed only by trained service technicians.

This command must be used only under the direction of your IBM support representative.

Description

You would use this command only during troubleshooting or disaster recovery situations.

Example

Displays version control information The following example version control information from a system master disk:

```
/usr/tank/server/bin/tank lsversion
```

tank lsdisklabel

Displays the SAN File System product (disk) label for a specified device.

▶▶—tank—lsdisklabel—◀◀

└─ -device—device_path ─┘

Parameters

-device *device_path*

Specifies the device path of a valid master volume from which to read a product label. The specified device path can include the path of the raw device as an argument, that is, *rvpathn* where *n* is the *vpath* letter. If the **-device** parameter is not specified, the command displays the product label from a local `Tank.Bootstrap` file.

Prerequisites

This task must be performed only by trained service technicians.

This command can be run from any cluster state.

Description

This command is run from the shell prompt. It is not run inside of a `fscli` session.

This command is used in disaster recovery situations, when you attach new hardware to an existing cluster, or while troubleshooting the system.

This command is located in the `/usr/tank/server/bin` directory.

If you display the product label from the master volume, this command displays this information:

- Label identifier 1. This is always **SDISK** for all SAN File System devices. This should match the label identifier 2. It is used as an integrity check to detect valid or corrupted product labels.
- Label number.
- Disk type. Valid values are:
 - M** Master volume label
 - S** System volume label
 - U** User volume
- Global disk identifier. This is the ID of volume that is used and understood by the metadata servers and clients.
- Label date. This is the date when the volume was added.
- Owner identifier. This is the ID of the cluster that owns the volume.
- Installation identifier. This is the unique installation ID for the shared storage that is initialized during installation and changes only upon reinstallation.
- Disk epoch identifier.
- Product identifier.
- Label identifier 2. This is always **SDISK** for all SAN File System devices. This should match the label identifier 1. It is used as an integrity check to detect valid or corrupted product labels.

If you display the product label from a local `Tank.Bootstrap` file, this command displays this information:

- Disk type.
- Global disk identifier.
- Owner identifier.
- Installation identifier.
- Disk epoch identifier.

Tip: The owner ID, installation ID, and disk epoch ID for a given volume is the same as the owner ID, installation ID, and disk epoch ID contained in the Tank.Bootstrap file on all engines in the cluster that own that volume. An engine can be zoned in such a way that it sees devices from another cluster, but each cluster uses the product label information in the Tank.Bootstrap file to know which devices it owns.

Example

Displays the product label for the system master volume The following example displays the product label for the system master volume contained on device /dev/rvpatha (a is the vpath letter):

```
/usr/tank/server/bin/tank lsdisklabel -device /dev/rvpatha
```

```
-----  
Product Label on Device /dev/rvpatha:  
-----
```

```
Label ID 1           : SDISK  
Label Number        : 0001  
Disk Type           : M  
Global Disk ID      : 73EFA22D6A2355F1  
Label Date          : Jun 25, 2003 4:35:24 AM  
Owner ID            : 00000000000004D2  
Installation ID     : 73EFA22D6A2355F1  
Disk Epoch          : 0000000000000000  
Product ID          : STORAGETANK  
Label ID 2           : SDISK  
-----
```

Displays the information in Tank.Bootstrap The following example displays the product label information found in the local Tank.Bootstrap file:

```
/usr/tank/server/bin/tank lsdisklabel
```

```
-----  
Tank.Bootstrap information  
-----
```

```
Disk Type           : U  
Global Disk ID      : 73F02ABDBE27B00B  
Owner ID            : 00000000000004D2  
Installation ID     : 73F02ABDBE27B00B  
Disk Epoch          : 0000000000000000  
-----
```

tank resetcluster

Erases the static cluster definition contained on the system master volume, without reinitializing the metadata, which in affect drops all metadata servers from the cluster at one time.

▶▶—tank—resetcluster—◀◀

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be stopped.

Description

Note: This command is run from the shell prompt. It is not run inside of sfscli.

Attention: This command will wipe out the static cluster definition contained on the master volume. It is equivalent to issuing the **dropserver** command on all engines.

This command is used in extreme disaster recovery situations or when you attach new hardware to an existing cluster.

This command is located in the `/usr/tank/server/bin` directory.

This command must be run from an engine that has a `Tank.Bootstrap` file. If the engine does not have a `Tank.Bootstrap` file, this file can be recovered from the master volume using the **tank extractbootrecord** command. If the master volume is unknown, you can issue the **tank lsdisklabel** command to inspect the SAN File System product label and to locate the master volume.

After you run this command, the first metadata server that is started with a valid `Tank.Bootstrap` file becomes the new master metadata server. At this point, the cluster is a single-engine cluster, and all additional metadata servers must be added using the **addserver** command. In addition, it might be necessary to move filesets to these new engines (using the **setfilesetserver** command) if the added engines have different names than the original cluster's engine names.

Example

Reset the cluster The following example erases the static cluster definition on the master volume:

```
#/usr/tank/server/bin/tank resetcluster
Static cluster definition has been reset. cluster size = 0
```

tank resetversion

Resets the version-control information on a system master disk.

Syntax

▶▶—tank—resetversion—◀◀

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be stopped.

Description

Attention: This command should be used only by trained service technicians and only in the event of extreme disaster recovery or hardware refresh situations. Invoking this command at runtime can lead to unpredictable and destructive results.

Example

Resets the version-control information The following example resets the version-control information on a system master disk:

```
/usr/tank/server/bin/tank resetversion
```

tankpasswd

Generates a password file that enables you to log in to the administrative command-line interface, also known as sfscli.

```
►►—tankpasswd— -u —user_name— -p —password—◀◀
```

Parameters

-u *user_name*
Specifies the user name.

-p *password*
Specifies the password associated with the given user name.

Description

This command must be run before you attempt to log on to the administrative CLI for the first time and when your password changes.

This command creates a password file, named `tank.passwd`, in your home directory. By default, the `sfscli` utility expects this file to be in your home directory. You can change this location by modifying the `SFS_CLI_PASSWDFILE` environment variable.

Note: You must move the password file to the location expected by the `sfscli` utility.

The user ID and password that you specify in this command must be the same user ID and password that is specified in the LDAP server by your system administrator. The LDAP server is used to connect to the administrative agent, which authorizes or denies access to SAN File System each time you attempt to use the `sfscli` utility.

Example

Generate the password file The following example generates a password file for user `saki`:

```
$ pwd  
/home/saki  
$ tankpasswd -u saki -p mypassword  
The password file was successfully written to: /home/saki/.tank.passwd
```

Appendix B. Accessibility

This topic provides information about the accessibility features of SAN File System and its accompanying documentation.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully.

Features

These are the major accessibility features in SAN File System:

- You can use screen-reader software and a digital speech synthesizer to hear what is displayed on the screen.

Note: The SAN File System Information Center and its related publications are accessibility-enabled for the IBM Home Page Reader.

- You can operate all features using the keyboard instead of the mouse.

Navigating by keyboard

You can use keys or key combinations to perform operations and initiate many menu actions that can also be done with a mouse. You can navigate the SAN File System console and help system from the keyboard by using the following key combinations:

- To traverse to the next link, button or topic, press Tab inside a frame (page).
- To expand or collapse a tree node, press Right Arrow or Left Arrow, respectively.
- To move to the next topic node, press Down Arrow or Tab.
- To move to the previous topic node, press Up Arrow or Shift+Tab.
- To scroll all the way up or down, press Home or End, respectively.
- To go back, press Alt+Left Arrow
- To go forward, press Alt+Right Arrow.
- To go to the next frame, press Ctrl+Tab. There are quite a number of frames in the help system.
- To move to the previous frame, press Shift+Ctrl+Tab.
- To print the current page or active frame, press Ctrl+P.

Appendix C. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
MW9A/050
5600 Cottle Road
San Jose, CA 95193
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation or Tivoli Systems Inc. in the United States or other countries or both:

AIX	AIX 5L	DB2
Enterprise Storage Server	eServer	FlashCopy
HACMP	IBM	IBM logo

Storage Tank
WebSphere

Tivoli
xSeries

TotalStorage

Intel and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and Windows NT are trademarks or registered trademarks of Microsoft Corporation.

Red Hat and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc., in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks

of others. 
The Java logo consists of a stylized blue coffee cup with a red flame above it. Below the cup, the word "Java" is written in red, and the word "COMPATIBLE" is written in blue below it.

Index

A

- About the Maintenance and Problem Determination Guide v
- accessibility
 - disability 173
 - keyboard 173
 - shortcut keys 173
- administrative
 - log 41
 - server 5
- administrative agent
 - starting 160
 - stopping 164
 - verifying that it is running 68
- administrative CLI
 - logging in 172
 - starting a session 108
- administrative CLI (ACLI)
 - accessing 107
- administrative server
 - troubleshooting 63
- administrative server logs 40
- administrative server, accessing using a browser 25
- AIX client
 - commands 124
- AIX, client for
 - controlling disk access 126
 - creating 124
 - destroying 124
 - displaying status for 132
 - loading the file-system driver 129
 - mounting the global namespace 130
 - scanning SAN File System for new and removed volumes 126
 - statistics 161
- antivirus software 7
- Audit log 38
- authentication 7
- authorization 7

C

- cache 13
- capturing first failure data 153
- case sensitivity 18
- CD, publications vi, vii
- certificates, replacing expired 69
- checking access in a heterogeneous environment 10
- CIMOM certificate 69
- client
 - accessing using a remote console utility 28
 - accessing using SSH 27
 - accessing using telnet 28
 - commands 119
 - creating 122, 124, 135
 - data LUN configuration 7
 - debugging for AIX 157

- client (*continued*)
 - debugging for Solaris 157
 - destroying 124, 135
 - displaying status for 132
 - loading the file-system driver 122, 129, 134
 - mounting the global namespace 122, 130, 137, 142
 - privileged 8
 - removing 122, 144
 - setting up 122
 - unmounting the global namespace 132
- client CLI
 - accessing 107
- client commands 7
- client diagnostic tools 44
- client logging and tracing 44, 47, 49, 51
- clients
 - authentication 7
 - authorization 7
 - clustering 7
 - native file-system security 15
 - privileged 15
 - restoring 103
 - supported 6
 - troubleshooting performance 89
 - UNIX-based 15
- cluster
 - log 39
 - troubleshooting 73
- cluster configuration, restoring 100
- clustering 7
- command
 - disableConsoleTrace 150
 - enableConsoleTrace 151
 - exit 115
 - help 116
 - insmod 134
 - labellun 142
 - migratedata 119
 - mktruststore 152
 - mount 142
 - obdc 153
 - parameters, standard 110
 - quit 117
 - rmstclient 122, 144
 - sanfs_ctl 146
 - sanfs_ctl stats 161
 - sanfsctl 145
 - sanfsd 148
 - sanfstrace 157
 - setoutput 117
 - setupstclient 122
 - startConsole 160
 - stfsclient 124, 135
 - stfsdisk 126
 - stfsdriver 129
 - stfsmount 130, 137
 - stfsstat 161
 - stfsstatus 132

- command (*continued*)
 - stfsumount 132
 - stopConsole 165
 - tank extractbootrecord 165
 - tank lscluster 166
 - tank lsdisklabel 168
 - tank lsversion 168
 - tank resetcluster 170
 - tank resetversion 171
 - tankpasswd 172
 - user assistance for 112
- command-line interface
 - troubleshooting user access 66
- commands
 - AIX client 124
 - client 119
 - common client 119
 - Linux client 133
 - modes 109
 - service 148
 - Solaris client 141
 - troubleshooting user access 67
 - Windows client 148
- common client
 - commands 119
- components
 - accessing 23
 - SAN File System 3
- configuration
 - data LUN 7
- considerations
 - file name 18
- console
 - troubleshooting user access 64
 - verifying that it is running 69
- controlling disk access for a client for AIX 126
- conventions, syntax diagram 113
- creating
 - client 122
 - client for AIX 124
 - client for Linux 135
 - truststore 152

D

- data cache 13
- debugging client for AIX 157
- debugging client for Solaris 157
- deleting a recovery file 98
- destroying
 - client for AIX 124
 - client for Linux 135
- diagnostic tools 37
- direct I/O 13
- disableConsoleTrace 150
- disabling
 - SAN File System console Web server 150
- disaster recovery commands 165, 166, 168, 170, 171

- disk drive recovery
 - master console 94
- displaying
 - statistics for a client for AIX 161
 - statistics for a client for Solaris 161
 - Windows 161
 - status for a client for AIX 132
- dump capability
 - Linux 46
 - SAN File System 42, 43
 - Windows 53

E

- enableConsoleTrace 151
- ending a sfscli session 115
- ending an sfscli session 117
- engine
 - accessing using SSH 26
- event
 - log 39
- exit 115

F

- Fibre Channel cable replacement
 - master console 95
- FIFO objects 16
- file
 - sharing 8
- file names
 - considerations 18
- file permissions 8
- file-system drive
 - version of 132
- first failure data capture 153

G

- GBIC replacement
 - master console 95
- getting help 116
- guidelines, naming 109

H

- hardware
 - restoring 98
- hardware vital product data 54
- help 116
 - general 105
 - telephone 106
- help parameters, standard 112
- heterogeneous file sharing 9
 - checking accesst 10
 - security attributes 11
 - translating permissions 10
 - without user maps 12
- homogeneous file sharing 12

I

- insmod 134
- installation troubleshooting 91

- isolating problems 33, 59

L

- labellun 142
- LDAP certificate 69
- limitations
 - UNIX-based clients 16
 - Windows-based client 19
- limited warranty vi
- Linux client
 - commands 133
- Linux, client for
 - creating 135
 - destroying 135
 - loading the file-system driver 134
 - mounting the global namespace 137
- listing
 - recovery files 98
- listing parameters, standard 112
- loading the client file-system driver 122, 129, 134
- local network
 - troubleshooting 76
- lock cache 13
- log
 - administrative 41
 - cluster 39
 - event 39
 - security 41
- logging in to the administrative CLI 172
- logs, administrative server 40
- logs, metadata server 37
- lost+found directory 14
- LUN
 - configuration 7
- LUNs
 - troubleshooting 85

M

- master console
 - accessing remotely 23
 - recovering disk drive 94
 - replacing Fibre Channel cable 95
 - restoring 98
 - troubleshooting 93
- Master console 19
- metadata
 - repairing 82
 - restoring 102
- metadata cache 13
- metadata server
 - bringing online 80
 - reassigning filesets to 80
 - taking offline 79
 - troubleshooting 74
- metadata server logs 37
- metadata server, adding to the
 - cluster 81
- migratedata 119
- migrating data to SAN File System 119
- mktruststore 152
- modes, command 109
- mount 142

- mounting the global namespace on a client for AIX 130
- mounting the global namespace on a client for Linux 137
- mounting the global namespace on a client for Solaris 142
- mounting the global namespace on the client 122

N

- named pipes 16
- naming guidelines 109
- navigating by keyboard 173
- non-uniform data LUN configuration 7
- notices 175
- Notices v
- NTFS
 - differences 18
 - unsupported features 19

O

- obdc 153
- one-button data collection utility 53
- operating system
 - restoring 98
- opportunistic locks (oplocks) 13
- orphaned objects 14
- output format
 - setting 117
- Overview 3

P

- permissions, file 8
- privileged clients 8, 15
- publications vi, vii
- publications CD vi, vii

Q

- quit 117

R

- recovery file, deleting 98
- recovery files
 - listing 98
- release notes vi, vii
- remote access 21
- remote console utility
 - accessing a client using 28
- Remote Supervisor Adapter II,
 - accessing 26
- removing
 - client 122, 144
- resolution procedures 79, 93
- rmstclient 122, 144
- root squashing 8
- RSA II, accessing 26
- RSA Web interface, shutting down an engine using 79
- running administrative commands 108

S

- safety information vi, vii
- safety notices, translated vi
- SAN connectivity
 - restoring 99
- SAN File System
 - components 3
- SAN File System accessibility
 - features 173
- SAN File System console Web server
 - disabling 150
 - starting 160
 - stopping 151, 165
- sanfs_ctl 146
- sanfs_ctl stats 161
- sanfsctl 145
- sanfsd 148
- sanfstrace 157
- security
 - for native client file systems 15
 - log 41
- security-attribute presentation in a heterogeneous environment 11
- server
 - administrative 5
 - server diagnostic tools 37
 - service
 - phone 105
- Service alert 20
- setoutput 117
- setting output format 117
- setting up the clients 122, 132
- setupstclient 122
- sfscli
 - ending the session 115, 117
 - getting help 116
- sharing files 8, 12
- Simple Network Management Protocol (SNMP)
 - components 22
- SNMP (Simple Network Management Protocol)
 - components 22
- software
 - restoring 100
- software recovery 93
- software vital product data 55
- Solaris client
 - commands 141
- Solaris, client for
 - mounting the global namespace 142
 - statistics 161
- standard
 - command parameters 110
 - help parameters 112
 - listing parameters 112
- startCimom 160
- startConsole 160
- starting
 - administrative CLI session 108
 - SAN File System console Web server 160
- starting the administrative agent 160
- statistics
 - client for AIX 161
 - client for Solaris 161
 - client for Windows 161

- stfsclient 124, 135
- stfsdisk 126
- stfsdriver 129
- stfsmount 130, 137
- stfsstat 161
- stfsstatus 132
- stfsumount 132
- stopCimom 164
- stopConsole 165
- stopping
 - SAN File System console Web server 151, 165
- stopping the administrative agent 164
- support
 - general 105
 - telephone 106
- syntax diagram conventions 113

T

- tank extractbootrecord 165
- tank lscluster 166
- tank lsdisklabel 168
- tank lsversion 168
- tank resetcluster 170
- tank resetversion 171
- Tank.Bootstrap file 165, 170
- tankpasswd 172
- telnet, accessing the client using 28
- tracing
 - metadata server 42
- trademarks 176
- translating permissions in a heterogeneous environment 10
- troubleshooting
 - administrative server 63
 - client 85
 - client performance 89
 - cluster 73
 - local network 76
 - LUNs 85
 - master console 93
 - metadata server 74
 - user access, command-line interface 66
 - user access, commands 67
 - user access, console 64
- truststore
 - creating 152

U

- unbuffered I/O 13
- uniform data LUN configuration 7
- UNIX-based clients 15
 - limitations of 16
- unmounting the global namespace on a client for AIX 132
- user assistance for commands 112
- user data
 - restoring 103
- user maps 12
- utility
 - startCimom 160
 - stopCimom 164

V

- vital product data (VPD)
 - hardware 54
 - software 55
- volume
 - scanning SAN File System for 126
- VPD (vital product data)
 - hardware 54
 - software 55

W

- watch and learn
 - about administrative server 5
 - about clients 6
- Web sites vii, 105
- Who should use this guide v
- Windows client
 - commands 148
- Windows-based client
 - administrative privileges 17
 - case sensitivity 18
 - difference from NTFS 18
 - file management 17
 - limitations of 19
 - NTFS features 17
- Windows, client for
 - statistics 161

Readers' Comments — We'd Like to Hear from You

IBM TotalStorage SAN File System
(based on IBM Storage Tank™ technology)
Maintenance and Problem Determination Guide
Version 2 Release 2

Publication No. GA27-4318-02

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Complete	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to find	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to understand	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Well organized	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Applicable to your tasks	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? Yes No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.



Fold and Tape

Please do not staple

Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Dept. CGFA
PO Box 12195
Research Triangle Park, NC 27709-9990



Fold and Tape

Please do not staple

Fold and Tape



Printed in USA

GA27-4318-02

