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

Maintenance and Problem Determination Guide

Version 2 Release 1

GA27-4318-01

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

Maintenance and Problem Determination Guide

Version 2 Release 1

GA27-4318-01

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

Second Edition (June 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.

Contents

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

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.

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 version 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 theadministrative 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 theSAN 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
- www.ibm.com/storage/software/virtualization/sfs

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 19 describes the various methods that are available for accessing SAN File System software components.
- Chapter 4, "Diagnostic tools," on page 27 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 45 describes initial steps that you can take to begin isolating problems with the SAN File System, IP network, and SAN.
- Chapter 6, "Troubleshooting the cluster," on page 47 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 7, "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 8, "Troubleshooting a SAN File System client," on page 73 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 9, "Troubleshooting the master console," on page 79 explains how to diagnose and resolve problems related to the master console.
- Disaster recovery explains the disaster recovery procedure.
- Chapter 11, "Getting help, service, and information," on page 91 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 19 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 45 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 45 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 6, “Troubleshooting the cluster,” on page 47.

   - If the problem is determined to be in the SAN File System master console, see Chapter 9, “Troubleshooting the master console,” on page 79.
   - 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

The following figure illustrates the major components of SAN File System.

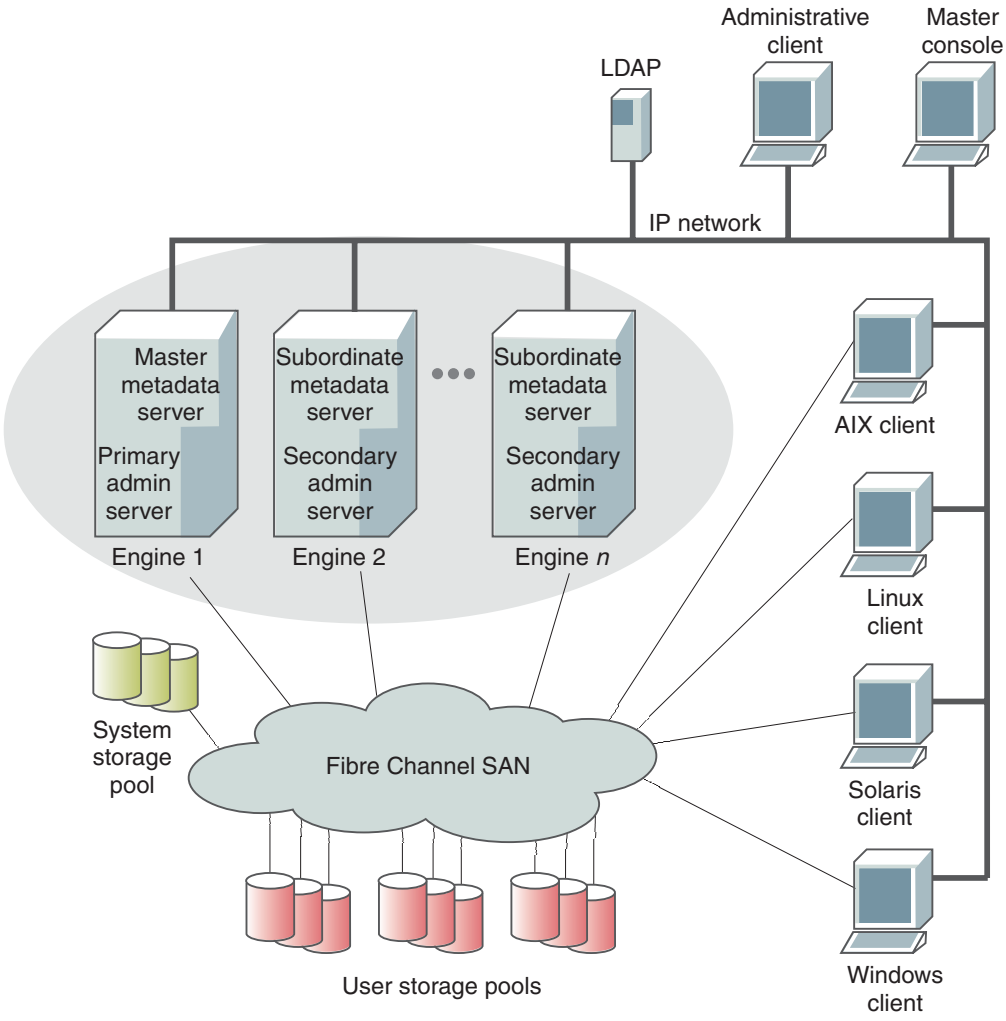


Figure 1. SAN File System components

The metadata servers and clients communicate over an 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 administrative server uses an *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 are going to share data and have their storage centrally managed are all connected to the SAN. In SAN File System, these computers 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 (or 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

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 is the part of the administrative infrastructure that 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 on behalf of that user to process the request. It also communicates with the operating system, the Remote Supervisory Adapter II (RSA) 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 always 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 is 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. All requests are redirected to a secondary administrative server if the primary administrative server is not available.

Clients

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 servers to a broader clientele, providing Network File System (NFS) or Common Internet File System (CIFS) access to the global namespace or 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 client software on each client machine. On UNIX-based clients, the software is a virtual file system (VFS). On Windows clients, it is an installable file system (IFS). 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 VFS or IFS 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 the client runs, 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. Therefore, 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.

Tip: Consider using a single, designated client machine as a backup client to perform all virus scans.

Authentication and authorization

Clients are authenticated using external services such as Active Directory for Windows-based clients and Network Information Services (NIS) for UNIX-based clients. SAN File System does not restrict how authentication is performed, but it does require that all Windows-based clients share a common definition of users and roles, and that all UNIX-based clients share a common domain (definition of users and groups). SAN File System does perform authorization for file access.

All systems that host a metadata server should be part of the shared domain. This makes it easier to perform client-based activities.

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 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. 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 cluster.

With SAN File System, you can use these clustering application on the clients:

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

Data LUN configuration

You can configure the SAN so that all data LUNs are available to all clients (known as *uniform configuration*) or so that a subset of data LUNs are available to some clients (known as a *nonuniform configuration*). In nonuniform configurations, a client must be able to see 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 see, 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 pool that is accessible to the clients that need the file. When you change the active policy, the new policy must meet the 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 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

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 file system, and can only be done by a privileged client. A *privileged client* is a client machine on which root or Administrator users have the same privileges on the global namespace as they have on other file systems available on their system. A root user logged in to a privileged client is granted full control over directories, files, and other file system objects created by clients.

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. Have at least one privileged client of each client OS type.

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 an AIX client should create files only within filesets dedicated to AIX. This is referred to as the *primary allegiance* of a fileset — that is, either Windows or UNIX. 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 list (ACLs) on the clients to accomplish this goal. You should limit your use of cross-platform or heterogeneous file sharing.

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.

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: After a directory, folder or file is created from a particular client type (UNIX or Windows), its security settings cannot be changed to another client type.

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-based clients, and other filesets have files created in them only by UNIX-based clients.

Homogenous file sharing

In a homogenous environment (for example, either all UNIX-based-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, currently UNIX-extended ACLs are not 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.

Heterogeneous file sharing

In a heterogeneous environment (for example, both UNIX-based and Windows-based clients), there is a restricted form of access. When files created on an UNIX-based client are accessed by a Windows-based client, access is controlled using only the semantics and permissions of the "Other" permission bits in UNIX. Similarly, when files created on a Windows-based client are accessed on an UNIX-based client, access is controlled using only the semantics and permissions of the "Everyone" user group in Windows.

Because the specific permissions do not match exactly between the two operating systems, translation is required. The following table shows the mapping of permissions types between UNIX and Windows. The permissions or ownership can

only be changed from the client machine (that is, Windows or UNIX) where the file or directory was created.

Windows permissions	UNIX permissions		
	Read	Write	Execute
Traverse Folder/Execute File			X
List Folder/Read Data	X		
Read Attributes		X	
Read Extended Attributes			
Create Files/Write Data		X	
Create Folders/Append Data		X (parent)	
Write Attributes		X	
Write Extended Attributes		X	
Delete Subfolders and Files		X (parent)	
Delete		X (parent)	
Read Permissions			
Change Permissions			
Take Ownership			
Synchronize		X	

For files created on a UNIX-based system, SAN File System stores the actual uid/gid numbers and shares them across all UNIX-based clients, but they all appear as SID S-1-0-0 on Windows. For files created on Windows, SAN File System stores the actual SID and shares it across Windows clients, but they all appear as 999999/999999 on UNIX. UIDs, GIDs, and SIDs are all mapped by the client to user, group, or owner according to whatever scheme is in use on the client.

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.

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.

A client caches data for small 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, the client reads the data from the shared storage device on which the file is stored. Data access is still fast because the client has direct access to all storage devices attached to the SAN.

Opportunistic locks

Opportunistic locks can be created and used by Windows-based clients. An *opportunistic lock* (or oplock) is a lock placed on a file in the global namespace. It

allows a Windows-based client to cache data locally, reducing network traffic and improving performance. SAN File System 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, does not cache data, and allows distributed applications on different computers 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. The I/O buffers, offsets and transfer size must be multiples of 512. You receive an `Invalid argument` error if this restriction is not met.

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 must be sector aligned.

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

Restriction: You cannot use direct I/O on a file that is being used in cache mode by another process.

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. 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 for each 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. SAN File System supports UNIX commands such as **ls** and **du** when they are run against the global namespace.

If a file created by an UNIX-based client has the read and write bits set for user “other,” all UNIX and Windows users can read and write to 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. In this release, SAN File System supports clients running on these platforms:

- AIX 5.1 (32-bit only)
- AIX 5.2 (32-bit and 64-bit)
- Red Hat Enterprise Linux Advanced Server 3.0, 2.4.21EL kernel
- Sun Solaris 9 (64-bit)

Tip: SAN File System allows AIX clients with up to 8 processors.

The SAN File System client code that is installed on a UNIX-based client is called a Virtual File System (VFS). The VFS is a subsystem of an 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 is 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 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 FIFO implementation is local to the client. In other words, data in a FIFO object is only readable 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.
- Clients running on the AIX 5.1 32-bit platform cannot create files that are larger than 1 TB.

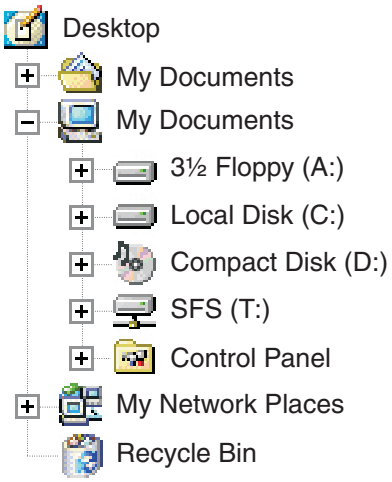
Windows-based clients

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 operating systems:

- Windows 2000 Server
- Windows 2000 Advanced Server

The SAN File System client code installed on a Windows-based client is called an Installable File System (IFS). The IFS is a subsystem of the Windows file system. It directs all metadata operations to a metadata server and all data operations to storage devices attached to your storage area network (SAN). An IFS 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 namespace on their systems. After the global namespace 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 the 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 the use of 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 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.

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. Applications can use the short name to gain access to files.

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)
- 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.

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
- FAStT 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.

Service alert is a feature of the master console that enables SAN File System or the SAN Volume Controller to proactively notifythe IBM Support Center of significant errors or failure conditions. This enables IBM to respond quickly to problems that occur, sometimes before the problem has been noticed by your system administrator.

In response to a sever error condition, SAN File System or the SAN Volume Controller issues an Simple Network Management Protocol (SNMP) trap and sends that trap to IBM Director Server running on the master console. 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 IBM support system, where it is converted into a problem record.

This figure shows the service alert architecture:

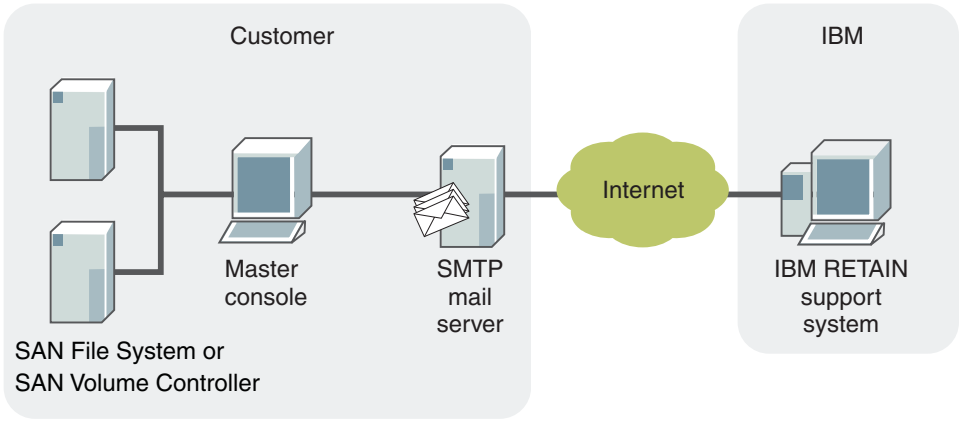


Figure 2. Service alert architecture

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:

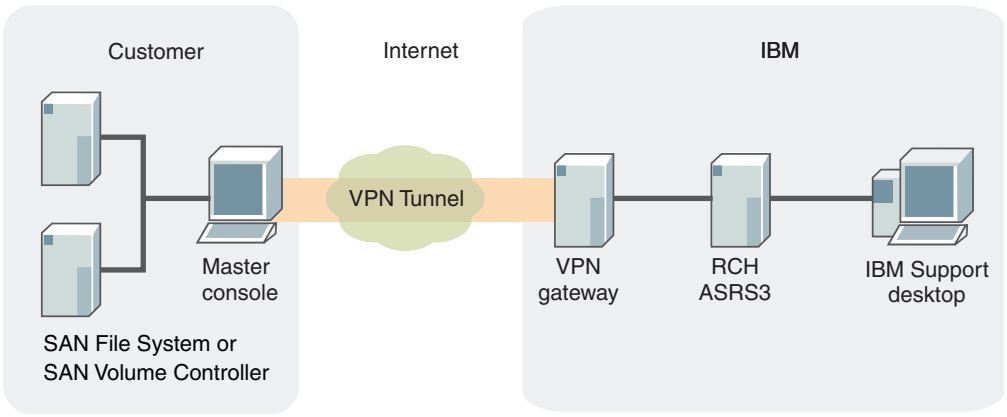


Figure 3. Remote access architecture

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.

Remotely accessing the master console

This task describes how to remotely access the master console.

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

- The master console must have a connection to the Internet.
- A Windows user account for the support representative must be set up on the master console.
- If the support representative has a need to access the SAN File System console or the RSA II Web interface remotely, a remote display emulation package, such as Virtual Networking Computer (VNC) server, must be installed and running on the master console.
- The customer must provide a user ID and password for access.
- A maintenance agreement must be established 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 IBM support representatives. You initiate the connection and have the ability to monitor and control the connection.

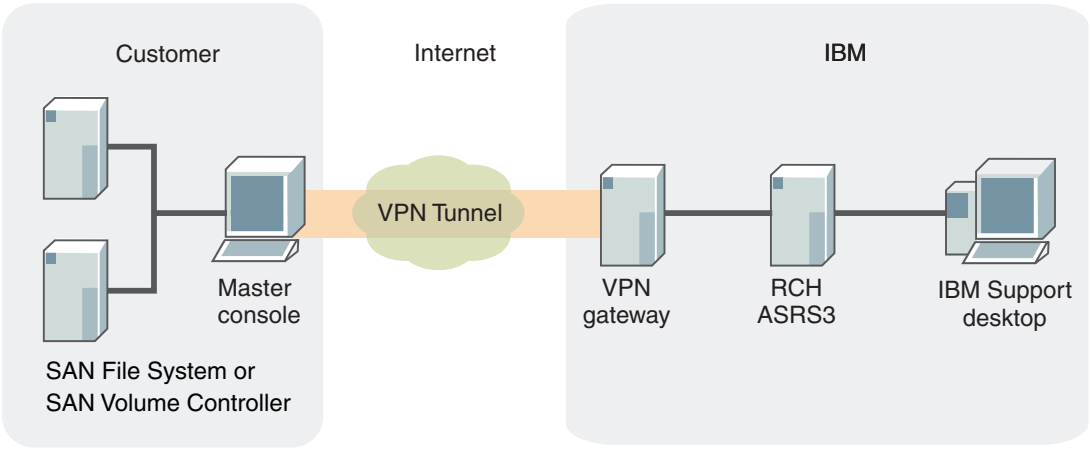


Figure 4. VPN connection between the customer and IBM support representative

1. Log into the master console. You can access the master console directly (using the keyboard, monitor, and mouse) or remotely though another computer on the same LAN.
2. Establish a secure VPN connection from the master console through the VPN gateway to a previously 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 previously designated VPN server within IBM using either a telnet client or a secure shell (SSH) client, such as PuTTY. The service representative uses the connection ID that you provide to access the active VPN tunnel.
5. The service representative connects to an account on the master console over the VPN connection to access to the master console. The service representative then establishes a second connection to the VPN server. If a remote display emulation package is available, the service representative can use this connection to establish a remote console to the master console.

This connection provides a service representative with the ability to log on to these devices or interfaces:

- Each of the engines in the SAN File System cluster. The support 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.
- The administrative command-line interface. The support 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 support representative.
- SAN File System clients. The support 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 support representative will need access.

- SAN File System console and RSA II Web interface (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 through a browser

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

Before accessing the administrative server through a browser from the master console, the following requirements must be met:

Note: You can access the SAN File System console from any computer that is on the same LAN as the SAN File System cluster.

- A SAN File System administrative user account must be set up for use in signing on to the SAN File System console.
- If the service representative is accessing the master console remotely:
  - Make sure that you have already 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. From the master console, open a Web browser and type the URL for the primary administrative server

   https://*primary_administrative_server*:7979/sfs

   where *primary_adminstrative_server* is the host name or IP address of the engine hosting the primary administrative server (as well as the master metadata server).

   Note: 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 will not complete successfully.

2. From the SAN File System console welcome page, enter a 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 SSH

This topic describes how to remotely access an engine using SSH from the master console.

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

- A SAN File System administrative user account must be set up for use in signing on to the SAN File System console.
- A UNIX-based operating system account must be set up 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.
  - The service representative must have established the VPN connection.

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. After the session is established, log in using a UNIX-based user ID and password.

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

- Access the SAN File System administrative command-line interface (CLI) to run SAN File System commands. These commands provide the ability to manage engines, metadata servers, and administrative servers.
- Access operating-system commands to enable or disable tracing, obtain dumps, and stop or start applications.

Accessing the RSA II adapter

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

Before accessing the RSA adapter for an engine, the following requirements must be met:

- An RSA user account must be set up for use in accessing the RSA II Web interface on the engine.
- If the service representative is accessing the master console remotely, the following conditions must be met:
  - 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. Open a browser from the remote console on the master console and type the URL for an RSA adapter:

   http://*RSA_II_web_address*/
2. Log on to the RSA II interface using a valid RSA user name and password.

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

- Stopping and restarting the engine
- Viewing vital product data for the engine
- Accessing the BIOS and firmware for the engine
- Updating the firmware for the RSA card
- Accessing 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 also 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 functionality to assign the CD-ROM or diskette drive from the master console to be used by the engine.

Note: 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 SSH

This task describes how to remotely access a client using SSH from the master console.

Before accessing a client using SSH, the following requirements must be met:

- An operating system account must be set up for use in signing on to the client to be accessed.
- An SSH software package, such as PuTTY, must be installed and running on the client to be accessed.
- If the service representative is accessing the master console remotely, you must have previously initiated a 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. After the session is established, you can log in using an operating system user ID and password.

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

- If you are accessing an AIX client, you can run SAN File System commands that provide the ability to stop and start the client as well as list client status. If you are accessing a Windows client, you can run the migratedata command.
- Access operating-system commands to enable or disable tracing, obtain dumps, and stop or start applications.

Accessing a client through telnet

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

Before accessing a client using telnet, the following requirements must be met:

- An operating system account for the support representative must be set up on the client to be accessed.
- If the service representative is accessing the master console remotely, the customer must have previously initiated a 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. After the session is established, you can log in using an operating system user ID and password.

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

- If you are accessing an AIX client, you can run SAN File System commands that provide the ability to stop and start the client as well as list client status. If you are accessing a Windows client, you can run the migratedata command.
- 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 task describes how to remotely access a client using a remote console utility, such as VNC, from the master console.

Before accessing a client using a remote console utility, such as Virtual Network Computing (VNC), the following requirements must be met:

- An operating system account must be set up on the client to be accessed.
- A VNC server must be installed and running on the client to be accessed. In addition, a session password must exist.
- A VNC client must be installed and running on the master console.
- 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 will 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. Diagnostic tools

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

You have several tools available to you when you attempt to diagnose problems with SAN File System components:

- Server. The SAN File System provides logs that you can use to view information about the metadata server and the administrative server. You can access these logs directly from the engine hosting a server or view a consolidated version of the logs through either the administrative command-line interface or SAN File System console.

  You can also use SAN File System tools as well as operating system tools to generate and analyze traces and dumps.
- Client. From a SAN File System client, you can use SAN File System commands and operating system commands to view, capture, and analyze log and trace data, as well as system dumps.

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	–
Failover log	log.failover	/usr/tank/server/log	–
Server log	log.std	/usr/tank/server/log	250 MB
Trace log	log.trace	/usr/tank/server/log	250 MB

Note:

1. Although log.audit, and log.trace have a maximum file size of 250 MB, the SAN File System actually stores 500 MB of data for each of these logs. When either of these logs reaches its maximum size, it is renamed to include the .old extension. If a file by that name already exists, the existing file is overwritten. 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:
   - 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 actually see a consolidated view of all of the logs from each engine in the cluster.

Note:

1. The consolidated view of the server message log is called the cluster log.
2. You can also display the event log. This log is actually a subset of the messages stored in the cluster log. It contains only those messages that have a message type of event.

Service only logs

The log.dmp, log.failover, and 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 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: jozvold  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.

The server log contains operational information. 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.

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

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 is displayed 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	–

Note:

1. Although log.std has a maximum file size of 250 MB, the SAN File System actually stores 500 MB of data for this log. This log reaches its maximum size, it is renamed to include the .old extension. If a file by

that name already exists, the existing file is overwritten. Then the log is cleared so that it can start accepting new messages again.

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.

Note: You can also display the event log. This log is actually a subset of the messages stored in the cluster log. It contains only those messages that have a message type of event.

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 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.

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

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. Use the Linux kill command to terminate the metadata server process:

   kill –6 *PID*

   where *PID* is the process ID for the metadata server process.

   Note: You will need to terminate all SAN File System processes that are currently running. Typically, using the kill command against the parent process will also terminate all child processes.

3. The kill command produces a file called core.*PID* in the directory where you started this process.

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.

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 the SAN File System to perform problem determination with SAN File System clients on the AIX operating system and the Windows 2000 operating system. In addition, both operating systems provide the ability 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.

Use the stfsdebug command and the syslog facility to enable tracing and logging on the SAN File System AIX client.

Syslog facility

The SAN File System client generates both log and trace messages, which are routed through the syslog facility on the AIX operating system. The syslog facility captures log and trace output from the kernel as well as other operating system services.

By default, the syslog facility discards all kernel output. However, you can configure the syslog facility to specify a destination for the messages by modifying /etc/syslog.conf.

- Specifying a file as the destination.

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

  1. Create /var/adm/ras/messages if it does not already exist. You can use the AIX touch command to create an empty file.
  2. Edit /etc/syslog.conf.
  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 directory instead. 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.

  To specify the console as the destination for kernel messages, perform the following steps:

  1. Edit /etc/syslog.conf.
  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.

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 could 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
```

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.

Configure the syslog facility and select one or more SAN File System classes to enable tracing and logging on the SAN File System Linux client.

Syslog facility

The SAN File System client generates both log and trace messages, which are routed through the syslog facility on the Linux operating system. The syslog facility captures log and trace output from the kernel as well as other operating system services.

By default, the syslog facility discards all kernel output. However, you can configure the syslog facility to specify a destination for the messages by modifying /etc/syslog.conf.

- Specifying a file as the destination.

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

  1. Create /var/log/messages if it does not already exist. You can use the Linux touch command to create an empty file.
  2. Edit /etc/syslog.conf.
  3. Make sure that this line exists and is not commented out:

     ```
     kern.debug /var/log/messages
     ```
  4. If you modified the file in step 3, then 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.

  To specify the console as the destination for kernel messages, perform the following steps:

  1. Edit /etc/syslog.conf.
  2. Make sure that this line exists and is not commented out:

     ```
     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 could specify a different level of output, such as kern.info to show just informational messages.

Selecting SAN File System classes

Tracing in the SAN File System is controlled according to components called *classes*. A class loosely corresponds to a file system operation such as mounting or reading.

The classes are grouped into two groups:

- Upper-driver classes
- CSM classes

Upper-driver classes are controlled by /proc/fs/stfs/debug. Here is an example of the initial contents of the /proc/fs/stfs/debug file:

```
$ cat /proc/fs/stfs/debug
INIT      OFF
MOUNT     OFF
DISK      OFF
PAGER     OFF
IO        OFF
INODE     OFF
FILEHAND  OFF
```

```
RNGLOCK   OFF
RDWR      OFF
CLEANER   OFF
PROC      OFF
FILEOP    OFF
INODEOP   OFF
SUPEROP   OFF
CSM       OFF
```

CSM classes are controlled by /proc/fs/stfs/csmdebug. Here is an example of the initial contents of the /proc/fs/stfs/csmdebug file:

```
$ cat /proc/fs/stfs/csmdebug
TM_LEASE         OFF
TM_XMIT_RECV     OFF
MC_OBJECT        OFF
CACHE_MANAGER    OFF
MC_SESSIONLOCK   OFF
CSM_SESSIONLOCK  OFF
MC_DATALOCK      OFF
CSM_DATALOCK     OFF
MC_RANGELOCK     OFF
CSM_RANGELOCK    OFF
BLKDISK          OFF
OBJATTR          OFF
NET_SOCKET       OFF
MISC             OFF
```

To enable debug messages for a particular class in either group:

1. Open the file that corresponds to the type of class that you are trying to change.
2. Change “OFF” to “ON” to enable any appropriate classes.
3. Change “ON” to “OFF” to disable any appropriate classes.
4. Save and exit the file.

The CSM debug classes also have different verbosity levels of messages. You can control the verbosity level of all messages in the class using /proc/fs/stfs/csmdebuglevel. Here is an example of the initial contents of the //proc/fs/stfs/csmdebuglevel file:

```
$ cat /proc/fs/stfs/csmdebuglevel
5
```

To change the verbosity level:

1. Open the /proc/fs/stfs/csmdebuglevel file.
2. Edit the file by changing the number to any number between 0 (least verbose) and 5 (most verbose).
3. Save and exit the file.

The following example messages show the format of log messages:

```
Apr 21 07:43:50 linuxclient1 unix: STFS: disk configuration process created
   with PID = 13348
Apr 21 07:43:50 linuxclient1 unix: STFS: cleaner process created with PID 12028
Apr 21 07:43:50 linuxclient1 unix: STFS: CSM process created with PID 10860
```

The following example messages show the format of trace messages:

```
Apr 28 13:17:09 linuxclient1 unix: STFS: 1051550182.439290  50337 STFS
   traceBuf_daemonize: going to sleep till shutdown
Apr 28 13:17:09 linuxclient1 unix: STFS: 1051550182.448769 196267 STFS CSM
   OS-dependent services initialized.
```

```
Apr 28 13:17:09 linuxclient1 unix: STFS: 1051550182.448827 196267 STFS Pager
   Strategy initialized.
Apr 28 13:17:09 linuxclient1 unix: STFS: 1051550182.448875 196267 STFS GFS
   hooks initialized.
Apr 28 13:17:09 linuxclient1 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.

Configure the syslog facility and select one or more SAN File System classes to enable tracing and logging on the SAN File System Solaris client.

Viewing SAN File System classes

Tracing in the SAN File System is controlled according to components called *classes*. A class loosely corresponds to a file system operation such as mounting or reading.

To view the classes, enter:

```
sanfs_ctl trace list
```

Enabling trace messages

The SAN File System client generates both log and trace messages, which are routed through the syslog facility on the Solaris operating system. The syslog facility captures log and trace output from the kernel as well as other operating system services.

By default, the SAN File System maintains all of its messages in an internal buffer to allow them to be recovered after a system crash. However, you can specify a destination for the messages by modifying /etc/syslog.conf.

To enable or disable logging and tracing on the Solaris client, use the sanfs_ctl command. For example, to enable tracing on all classes at the same level, enter:

```
sanfs_ctl trace set -level number
```

where *number* is a level of verbosity of the tracing that increases from 0 to 5.

You can also enable or disable trace messages for selected classes by entering:

```
sanfs_ctl trace set -class classname -level number
```

Where *classname* is the class for which you are enabling tracing, and *number* is the level of verbosity of the tracing that increases from 0 to 5. You can also list up to twenty classes by separating them with a comma and no spaces.

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/rdsk/c2t9d0s2
```

Windows 2000 client logging and tracing

This topic describes the logging and tracing capabilities available on SAN File System Windows 2000 clients.

A SAN File System Windows 2000 client provides two types of messages:

- Log messages that 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 the SAN File System.

  You can use the Event Viewer to list messages from the Event Log. If you double-click a message from the Event Viewer, you can find more detailed information about that message. You can also use the Event Viewer to filter messages by message type, source of the message, or according to 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.

  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.
  ```

- Trace messages that consist of extensive low-level tracing output about client functions and internal data.

  You can use the stlog command to enable and control tracing for a client. However, you should use the stlog command with care; enabling full tracing can significantly impact the performance of the SAN File System.

  If tracing is enabled, the SAN File System writes trace messages to a file named c:\Program Files\IBM\Storage Tank\client\log\sanfs.log. This file contains tracing output only for the SAN File System client. It does not contain information for the operating system or any other applications.

  Note: By default, minimal tracing is enabled.

  Enabling detailed tracing

  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 an empty string value named FileName. Double-click on FileName and change its value in the string editor to a valid path and file name for the trace log (for example, c:\stfs.log)

4. Click OK.
5. Reboot for the changes to take effect.

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 2000 client dump capability

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

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. Click **Start→Control Panel→System→Advanced→Startup and Recovery→System Failure** to verify the settings.
- 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** while pressing **ScrLk** twice from the keyboard. The dump file is generated the next time you power on 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 Data**.
  - 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
```

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 3. 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	
Memory	Memory	Statistics for total memory available, used, free, shared, buffered, and cached. Additional usage statistics as well.
	Swap space	

Table 3. Engine hardware VPD (continued)

Hardware area	Component	VPD collected
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 4. 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 4. 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.failover, and log.*.old.
	Core files	Server core file, if present.
	Server configuration	Dump information about: • 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: • 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.
	Cluster configuration	Dump information about: • 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
Administrative server	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	Server core file, if present.

Client

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

Table 5. 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 5. Isolating problems with the SAN File System

This topic explains how to begin problem determination by isolating 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 8, "Troubleshooting a SAN File System client," on page 73.
- 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 will 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 will see the logs for that particular metadata server.

  In addition, you can use the information provided in Chapter 7, "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 6, "Troubleshooting the cluster," on page 47. If you access this log through the master metadata server, you will see a consolidated view of the logs from each of the metadata servers in the cluster (called the cluster log). If you access these logs through a subordinate metadata server, you will see the logs for that particular metadata server.
- 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.

In cases where you are not sure whether 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 operating system reboots or cable disconnects, can cause the SAN File System to lose connectivity to LUNs. If the logs indicates I/O failures for a client or metadata server, verify the following:

- Configured LUNs are visible from the metadata servers and SAN File System clients. From the server, you should see both the user LUNs and system LUNs. From a client, you should see only the user LUNs.

  You can configure the clients to only see a select group of user LUNs which it needs to access, or you can 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 rebooted, 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 the SAN File System to remap LUNs in the event of lost connectivity. Please see your SAN administrator for help with LUN rediscovery options specific to your operating environment.

Identifying SAN problems

Perform the following steps to determine whether 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 RAID controllers that are 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.
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 the 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*, which is 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. 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. The metadata servers have 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 also perform the same tasks that are performed by subordinate metadata servers, managing file metadata and workload for one or more filesets.

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

Metadata server failures

When a subordinate metadata server becomes unresponsive or fails, such as when the operating system crashes or hangs, the engine is automatically restarted. In addition, if you have enabled the automatic restart service (enabled by default), the metadata server is also automatically restarted.

While the subordinate metadata server is in the process of restarting, it cannot respond to requests from clients:

- Client requests to the metadata server and client access of any files served by the metadata server will fail or be delayed.
- Client applications experience a pause in service while the metadata server is unavailable (typically, this will last approximately one or two minutes). During this time, active operations of some applications can begin to time out. Whether additional errors occur is based on how the client applications respond to a timeout situation.

When the master metadata server becomes unresponsive or fails, any clients attempting to access filesets managed by the master experience the same results as clients attempting to access filesets managed by subordinate metadata servers. In addition, subordinate metadata servers can be affected by the unavailability of the master. Metadata servers in the cluster rely on a heartbeat mechanism to verify availability. Depending on the length of time that the master metadata server is unavailable, subordinate metadata servers may detect the loss of the heartbeat mechanism and cease all activity until the master is available again (or you set a new master).

Troubleshooting a metadata server

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

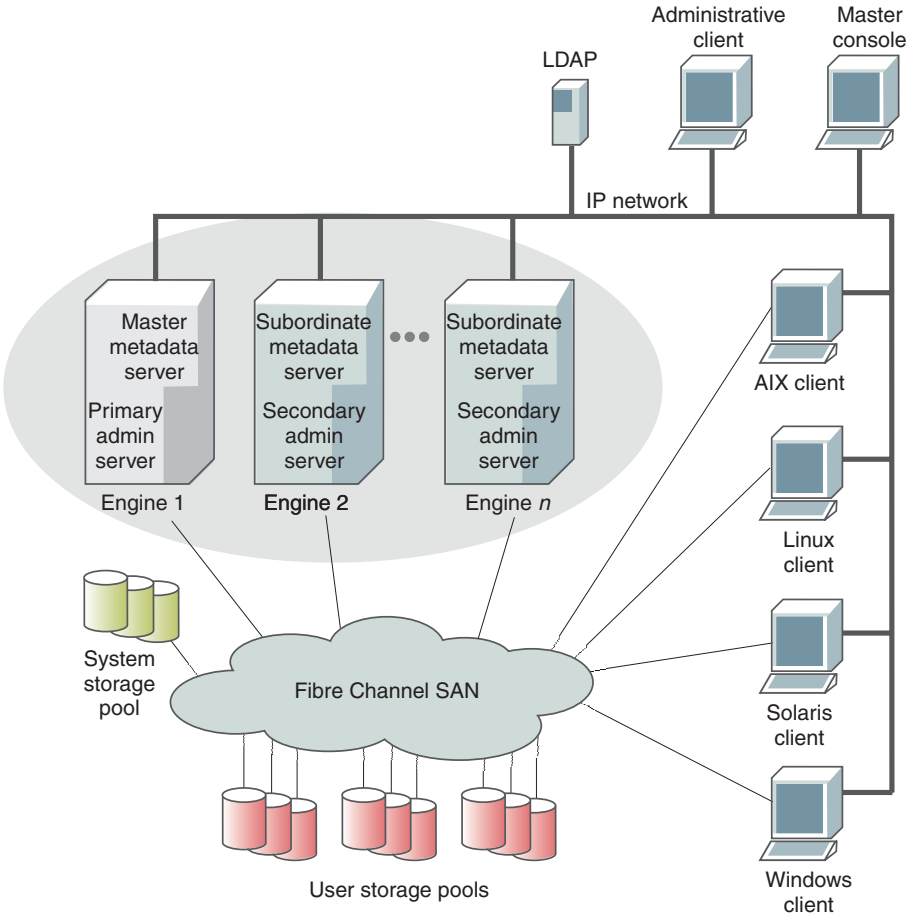
Problem

A metadata server has failed. Attempts by the SAN File System to automatically restart the metadata server have also failed and manual intervention will be required to restart the metadata server. Clients cannot access file metadata and, in the case of a master metadata server failure, the cluster itself is no longer available.

Note: If a metadata server loses power, you will need to reset the RSA II card on the engine hosting the metadata server to restore proper communication between the RSA II card and the metadata server. To reset the RSA II card, you will need to access the Web interface for the RSA II card and select the option for resetting the RSA II card.

Investigation

If a subordinate metadata server has failed, take the following actions:



Perform the following steps until the problem is resolved:

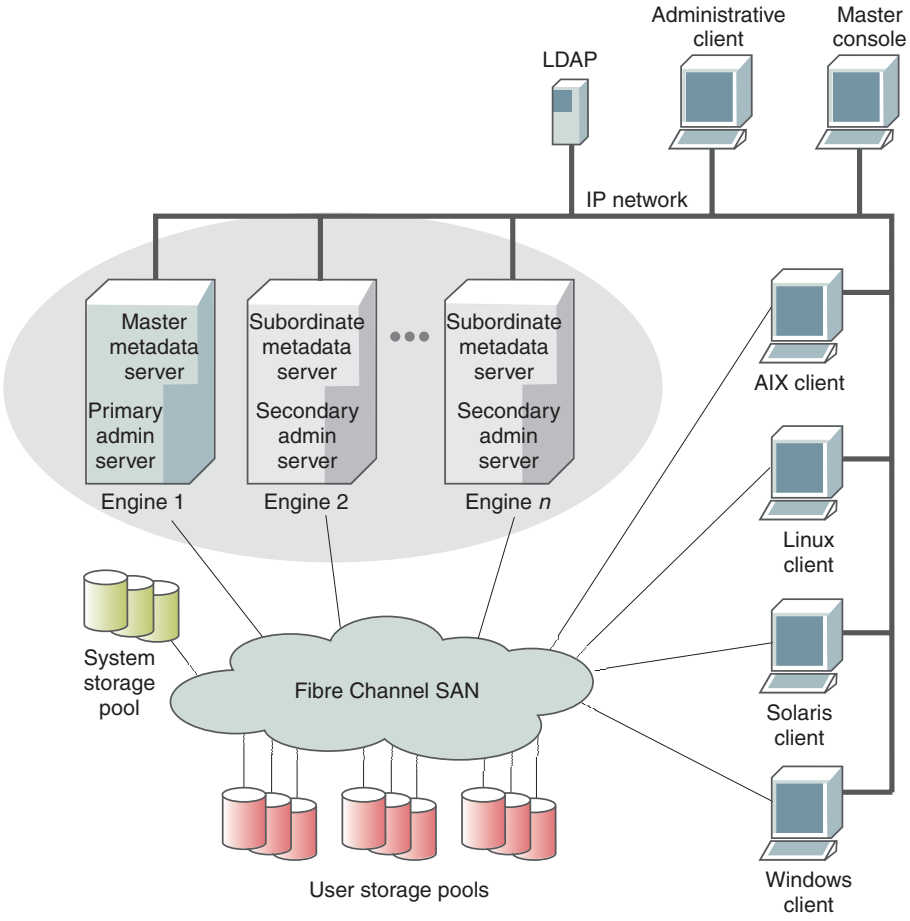
1. Use the SAN File System console or the administrative command-line interface to view the status of the subordinate metadata server.

2. View the cluster message log to verify that the SAN File System has not been able to restart the subordinate metadata server. In addition, the messages in this log can provide you with an indication of what the problem may be.
3. Reassign (move) the filesets from the subordinate metadata server to another metadata server that is online. The decision to reassign filesets will be based on the length of time it will take to repair the metadata server.

   Note: Make sure that this metadata server is actually down and all restart attempts were unsuccessful before reassigning filesets:

   a. Run lsserver to verify that the metadata server is shut down.
   b. Run lsengine to verify that the engine hosting the master metadata server is shut down.
   c. Run lsserver to verify that the metadata server to which you are going to assign the role of master is up and running.
4. Resolve any problems found in the cluster message log that are related to this metadata server. If the messages indicate a hardware error, use the RSA II Web interface to access the RSA II adapter for the engine hosting the subordinate metadata server. The RSA II Web Interface can assist you in isolating the hardware problem.
5. If there was an abnormal termination of the metadata server, you may begin to see errors on the clients, even after the problem with the metadata server has been resolved by itself with the automatic restart feature or by taking the failed metadata server offline and reassigning its filesets to another metadata server. If you begin seeing these types of problems, you will need to restart the affected clients:
   - On clients running the AIX operating system:
     a. Run rmstclient to unmount the global namespace, remove the virtual client, and unload the file-system driver.
     b. Run setupstclient to load the file-system driver, create the virtual client, and mount the global namespace.
   - On clients running the Windows operating system, reboot the system.
6. After repairing the failed metadata server, bring the server back online (use the startserver command from the administrative command-line interface).
7. If you previously reassigned the filesets for this metadata server to another server, you can now assign them back to this metadata server.

If the master metadata server has failed, take the following actions.



Perform the following steps until the problem is resolved:

1. Use the administrative command-line interface to view the server message log and verify that the SAN File System has not been able to restart the master metadata server. In addition, the messages in this log can provide you with an indication of what the problem may be.
2. Define a new master metadata server for the cluster.

   Note: Make sure that this metadata server is actually down and all restart attempts were unsuccessful before attempting to set a new master or before reassigning filesets:

   a. Run lsserver to verify that the cluster does not have a master metadata server.
   b. Run lsengine to verify that the engine hosting the master metadata server is shut down.
   c. Run lsserver to verify that the metadata server to which you are going to assign the role of master is up and running.
3. Reassign (move) the filesets from this metadata server to another metadata server that is online.
   a. List all of the filesets assigned to the metadata server to be upgraded.

      ```
      /usr/tank/admin/bin/sfscli lsfileset -server metadata_server_name
      ```

   b. Assign the filesets to different metadata server.

```
/usr/tank/admin/bin/sfscli setfilesetserver -server
new_metadata_server_name fileset1 fileset2 fileset3
```

4. Resolve any problems found in the cluster message log that are related to this metadata server. If the messages indicate a hardware error, use the RSA II Web interface to access the RSA II adapter for the engine hosting the former master metadata server. The RSA II Web Interface can assist you in isolating the hardware problem.
5. If there was an abnormal termination of the metadata server, you may begin to see errors on the clients, even after the problem with the metadata server has been resolved by itself with the automatic restart feature or by taking the failed metadata server offline and reassigning its filesets to another metadata server. If you begin seeing these types of problems, you will need to restart the affected clients:
   - On clients running AIX or Linux:
     a. Run rmstclient to unmount the global namespace, remove the virtual client, and unload the file-system driver.
     b. Run setupstclient to load the file-system driver, create the virtual client, and mount the global namespace.
   - On clients running Windows, reboot the system.
   - On clients running Solaris:
     a. Run umount to unmount the global namespace.
     b. Run mount to mount the global namespace again.
6. After repairing the failed metadata server, bring the server back online (use the startserver command from the administrative command-line interface).
7. If you choose, you can now set this metadata server to be the master metadata server once again. To set this metadata server to be the new master, you must first shut down the existing master metadata server and power off the engine hosting the master.
8. If you previously reassigned the filesets for this metadata server to another server, you can now assign them back to this metadata server.

Troubleshooting the local network

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

Problem

There is a problem with the local network on which the metadata servers communicate. The problem may be:

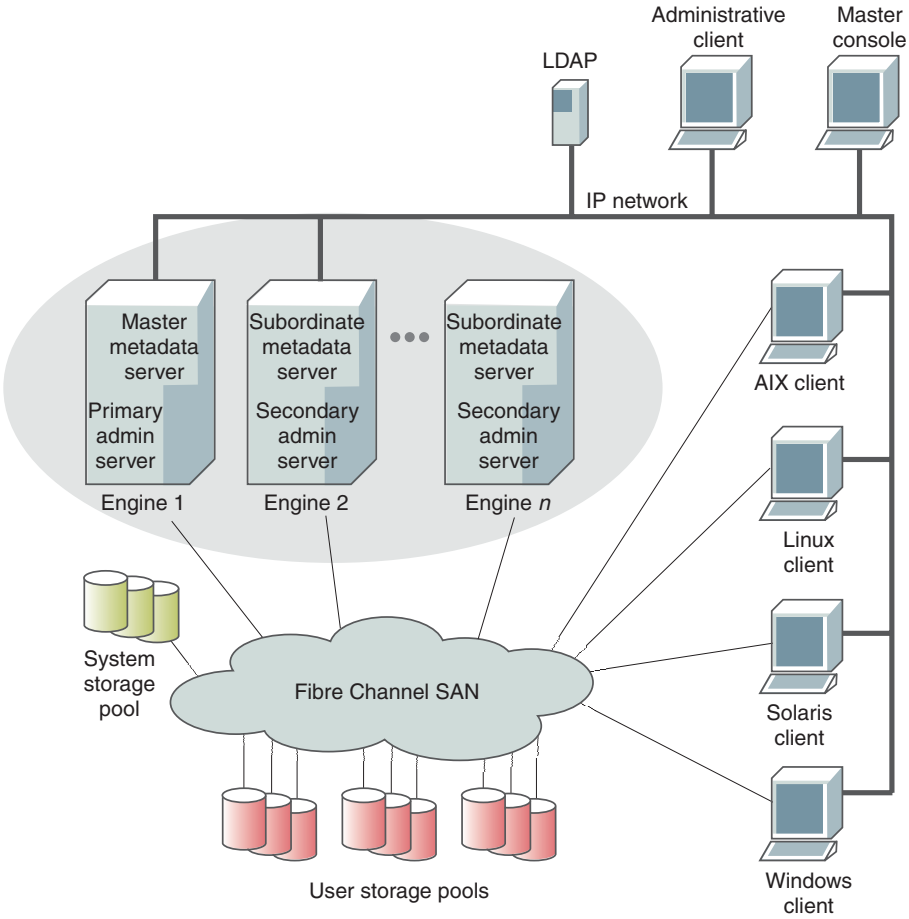
- A network fault. A local network fault can occur if there is a bad Ethernet adapter in an engine or the Ethernet cable is not connected between the Ethernet adapter and the IP network. In the event of a local network fault, the cluster will react as if the metadata server on which the fault occurred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

  The master metadata server reforms the cluster, excluding the failed metadata server. The metadata server itself will go into a wait state, and any filesets assigned to that metadata server will no longer be available to clients.
- 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 communications with the master metadata server. The partition containing the

master metadata server will react as if the metadata servers in the other partition are down. The metadata servers in the other partition will react as if the master metadata server is down.

Investigation

If there is a local network fault with one of the subordinate metadata servers, take the following actions.



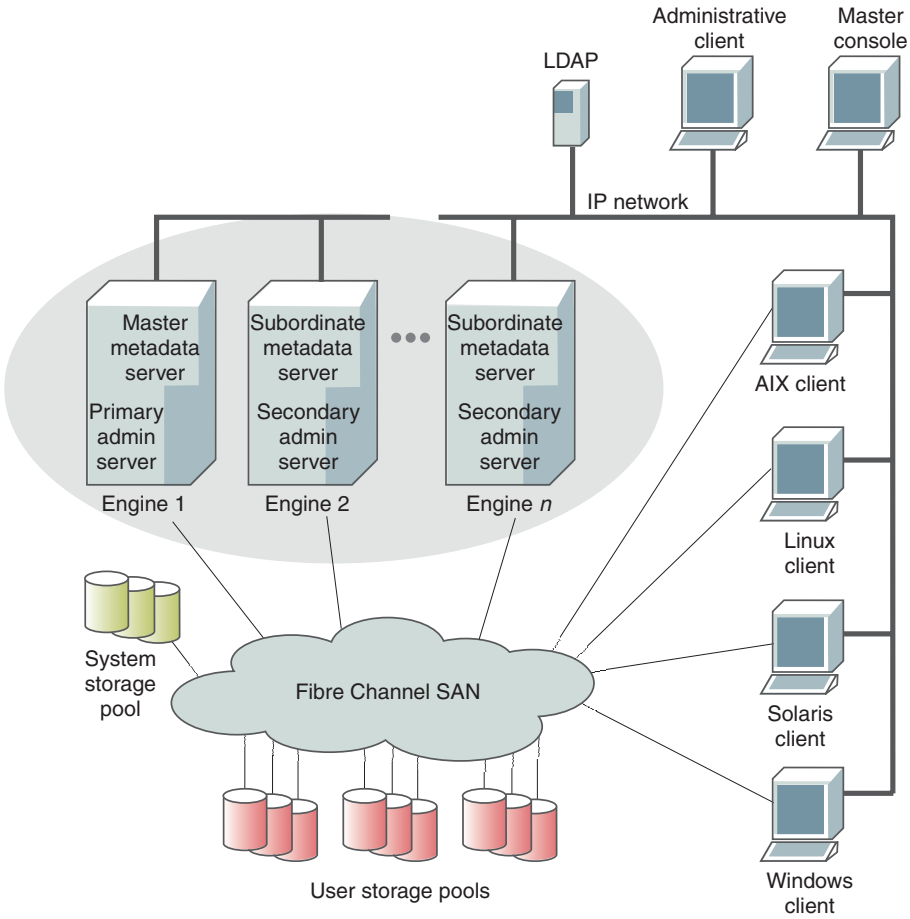
Perform the following steps in order until the problem is resolved:

1. Use the RSA II Web interface to access the RSA II adapter for the engine hosting the subordinate metadata server.
2. Shut down the engine from the RSA II Web interface. You will not be able to use the Administrative command-line interface or the SAN File System console to shut down the metadata server because the master metadata server already considers the server to be shut down.
3. Reassign (move) the filesets from the subordinate metadata server to another metadata server that is online. The decision to reassign filesets will be based on the length of time it will take to repair the metadata server.
   a. Run lsserver to verify that the metadata server to which you are going to assign the filesets is up and running.
   b. Refer to "Reassigning filesets to metadata servers" on page 58 for the procedure to reassign the filesets.
4. After repairing the network fault, you can have the metadata server rejoin the cluster:

a. Start the metadata server.
b. Wait for the cluster to be reformed to include this metadata server.
c. Use the SAN File System console or the Administrative command-line interface to verify that all metadata servers in the cluster are in an Online state.

5. If you previously reassigned the filesets for this metadata server to another server, you can now assign them back to this metadata server.
   a. Run lsserver to verify that the metadata server to which you are going to assign the filesets is up and running.
   b. Refer to "Reassigning filesets to metadata servers" on page 58 for the procedure to reassign the filesets.

If there is a network partition, take the following actions.



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

1. Use the RSA II Web interface to access the RSA II adapter for each of the partitioned engines.
2. Shut down each engine from the RSA II Web interface. You will not be able to use the Administrative command-line interface or the SAN File System console to shut down the metadata server because the master metadata server already considers the servers to be shut down.
3. Reassign (move) the filesets from the subordinate metadata server to another metadata server that is online. The decision to reassign filesets will be based on the length of time it will take to repair the metadata server.

a. Run lsserver to verify that the metadata server to which you are going to assign the filesets is up and running.
b. Refer to "Reassigning filesets to metadata servers" on page 58 for the procedure to reassign the filesets.

4. After repairing the network partition, you can have the metadata servers rejoin the cluster:
   a. Start each metadata server.
   b. Wait for the cluster to be reformed to include these metadata servers.
   c. Use the SAN File System console or the Administrative command-line interface to verify that all metadata servers in the cluster are in an Online state.
5. If you previously reassigned the filesets for these metadata servers to other servers, you can now assign them back to these metadata servers.
   a. Run lsserver to verify that the metadata server to which you are going to assign the filesets is up and running.
   b. Refer to "Reassigning filesets to metadata servers" on page 58 for the procedure to reassign the filesets.

States

This section describes the states of various SAN File System components.

Cluster states

This topic describes SAN File System cluster states.

SAN File System cluster states can be active, inactive, or unknown.

Table 6. SAN File System cluster states

State type	State name	State description
Inactive	Not running	The cluster cannot perform any functions because none of the servers in the cluster have completed the start up procedure to the point where the cluster is detected as running, (that is, in the "Not added" server state).
	Forming	The cluster has a cluster master and is in the process of forming. This state is always the initial one whenever a cluster is newly formed.

Table 6. SAN File System cluster states (continued)

State type	State name	State description
Active	Offline	One or more servers in the cluster are in the "Offline" state. This state terminates all current client sessions and prevents new client sessions from being started, but still allows all server I/O. The offline state is used for restricting the cluster from client access. **Characteristics:** • No cluster configurations are allowed in this state. Configuration changes will fail. • Metadata activity stops. • Existing locks are maintained. • File extensions (growing a file) are not allowed during this time, which guarantees a consistent metadata view for LUN-based backup. • Only existing USER data blocks from client side can be modified and assume client-application based techniques are used to make that data consistent later. • SAN File System FlashCopy is not available. • Administration command line interface client locks are revoked, caches flushed, and so on.
	Fully quiescent	One or more servers in the cluster are in the "Fully Quiescent" state. Quiescence puts the cluster in a "quiet" client communications mode to allow other operations to occur, such as configuring a cluster or creating a FlashCopy image. Full quiescence cuts off all client communications with the cluster. **Characteristics:** • No cluster configurations are allowed in this state. Configuration changes will fail. • SAN File System FlashCopy is not available. • Administration command line interface client locks are revoked, caches flushed, and so on.
	Partly quiescent	One or more servers in the cluster are in the "Partly Quiescent" state. Quiescence puts the cluster in a "quiet" client communications mode to allow other operations to occur, such as configuring a cluster or creating a FlashCopy image. Partial quiescence allows existing metadata activity and client communication to continue, but prohibits new communication.
	Online	All servers in the cluster are in the "Online" state, meaning that the cluster has fully initialized all of its subsystems, and is serving client requests.
Unknown	Unknown	The cluster state cannot be determined.

Metadata server states

This topic describes SAN File System metadata server states.

SAN File System metadata servers present different states depending on whether they are available or unavailable, or whether or not they are part of a cluster.

Table 7. SAN File System metadata server states

State type	State name	State description
Unavailable	Not running	The server is not running and cannot perform any functions.
	Failed initialization	A fatal error occurred during initialization and the server is suspended. The server remains suspended until an administrator can fix the problem that occurred during initialization.
Available, unclustered	Initializing	The server is running, but has not yet opened its communication ports.
	Not added	The server has not yet been added to the cluster. Because the master server does not know about servers that have not yet been added, this state is only available by logging into that server.
	Joining	The server is joining the cluster (has activated its communication).
Available, clustered	Offline	This state terminates all current client sessions and prevents new client sessions from being started, but still allows all server I/O. The offline state is used for restricting the cluster from client access. **Characteristics:** • No cluster configurations are allowed in this state. Configuration changes will fail. • Metadata activity stops. • Existing locks are maintained. • File extensions (growing a file) are not allowed during this time, which guarantees a consistent metadata view for LUN-based backup. • SAN File System FlashCopy is not available. • Administration command line interface client locks are revoked, caches flushed, and so on.
	Fully quiescent	Quiescence puts the server in a "quiet" client communications mode to allow other operations to occur, such as configuring a server or creating a FlashCopy image. Full quiescence cuts off all client communications with the server. **Characteristics:** • No cluster configurations are allowed in this state. Configuration changes will fail. • SAN File System FlashCopy is not available. • Administration command line interface client locks are revoked, caches flushed, and so on.
	Partly quiescent	Quiescence puts the server in a "quiet" client communications mode to allow other operations to occur, such as configuring a server or creating a FlashCopy image. Partial quiescence allows existing metadata activity and client communication to continue, but prohibits new communication.
	Online	The server has fully initialized all of its subsystems, is a member of a cluster, and is serving client requests.
Unknown	Unknown	The server state cannot be determined.

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 58
- "Bringing a metadata server online" on page 58

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 take a metadata server offline.

1. Perform these steps to take a subordinate metadata server offline. These steps assume that you are using the administrative command-line interface.
   a. Run the stopserver command to stop the metadata server.
   b. Reassign filesets that are currently assigned to the metadata server.
2. Perform these steps to remove a master metadata server from the cluster. These steps assume that you are using the administrative command-line interface.
   a. Run the stopserver command to stop the master metadata server.
   b. Run the setmaster command from a subordinate metadata server to set a new master.
   c. Reassign filesets that were assigned to the metadata server that you just stopped.

The metadata server is now offline.

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.

   ```
   sfscli 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.

Bringing a metadata server online

This task describes how to bring a metadata server online.

1. Run the startserver command to start the metadata server.
2. Reassign any filesets that need to be assigned to this metadata server.
3. Run the startserver command to start the metadata server.

The metadata server is now online and managing filesets.

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.

This procedure applies when one of more of the following occurs when you run the sfscli **lsserver** command on the metadata server:

- The master and subordinate metadata servers are shown as “not running subordinates.”
- The master metadata server only shows data for that particular server (it should show data for all subordinate metadata servers).
- The sfscli **lsserver** command shows that the metadata server is either in the “Not added” or “Not running” state.

1. Perform the following preliminary steps, listed by symptom, to try to solve the problem:
   a. If the metadata server is shown in the “Not added” or “Not running” state, start or add the metadata server to the cluster using the appropriate sfscli commands.

   If neither of these procedures applies to the situation or fails to solve the problem, perform the following steps:
2. 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 from the master metadata server:
   a. Stop Cimom by entering: `/usr/tank/admin/bin/stopCimom`

b. Run the command: `/usr/tank/admin/bin/setupsfs -newserver`. This 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 command: `sfscli stopserver` *`master_server_name`*

d. Change to the /usr/tank/server/bin directory.

e. Run: `tank extractbootrecord -device` */dev/rvpathX* where */dev/rvpathX* is that name of the master metadata LUN. This creates a new Tank.Bootstrap file.

f. To find the master disk, run the command: `tank lsdisklabel -device` */dev/rvpathX* The disk with a Disk Type of "M" is the master.

g. Run the command: `sfscli startserver` *`master_server_name`*

h. Run the command: `sfscli lsserver` The server should now be the "online master."

3. From the master metadata server, start any subordinates by running:`sfscli startserver` *`subordinate_server_name`* The subordinate server might shutdown at this point. If it does, it should restart automatically. If it does not restart automatically, start the server and run sfscli lsserver. The results should show that the subordinate is now online. If it does not, repeat step 3.

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

1. On the subordinate server, run `setupTank -newserver`.
2. On the master metadata server, run sfscli startserver *subordinate_server_name*.

If there is no Tank.Bootstrap file in the /usr/tank/server/config directory of the subordinate metadata server, then copy the file from the master metadata server.

Adding a metadata server to an existing cluster

This topic lists the steps required to add a metadata server to an existing cluster.

The cluster to which you are adding the server must be online.

1. Start the new metadata server using the **startserver** command.
2. Add the metadata server to the 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.
3. 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.

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

This topic explains how to identify when lost LUNs are preventing the metadata server from starting correctly.

This 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. If you attempt to start the metadata server using the **startserver** command, and the metadata server does 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 that message, continue with step 3. 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 step 4. 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 step 6.
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.

   ```
   sfscli> mkdrfile dr_file_1
   ```

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

   ```
   sfscli> 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.

   ```
   sfscli>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.

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. Change the master metadata server to one of the subordinate metadata servers:
      1) Run /usr/tank/server/bin/tank lsserver to verify that the cluster does not have a master metadata server.
      2) Run /usr/tank/server/bin/tank lsengine to verify that the engine hosting the master metadata server is shut down.
      3) Run /usr/tank/server/bin/tank lsserver to verify that the metadata server to which you are going to assign the role of master is up and running.
      4) Access the subordinate metadata server that you want to set as the new master metadata server.
      5) Run the /usr/tank/server/bin/tank setmaster command to set it as the new master metadata server.
      6) If the previous master metadata server was managing any filesets, reassign those filesets to other metadata servers in the cluster.
   b. 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.

c. 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.

d. Repeat steps 2a1 on page 61 through 2a6 on page 61 to change the master metadata server back to the original metadata server.
e. 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 metadata server master. 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.

Chapter 7. 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/tank

  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.

  Note: One exception to running a command from the master metadata server is the setmaster command, which is used to designate a new master metadata server in the event that there is a failure of the current master. This command is run from the subordinate metadata server that is going to become 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.
      - You can run the lsadmuser command to list all of the administrative users. If the user is not listed, you can 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 account for the user 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 previous 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 whether 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:
      - IP network problem
      - 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 /usr/tank/admin/bin/startCimom to start the administrative agent. If you run into 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 any problems that 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 /usr/tank/admin/bin/startConsole to start the console. If you run into problems starting the console, view the log indicated by the error message and attempt to resolve any problems that 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 that you are having with a user attempting to access the Administrative command-line interface.

Problem

A user was unable to access the Administrative command-line interface.

Investigation

If the user received access denied errors

Perform the following steps until the problem is resolved:

1. Using SSH, 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 that is having problems.
4. Run the lsserver command to verify that you have access to the metadata server.

   ```
   sfscli>lsserver
   ```
5. 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 whether you are receiving errors about the SAN File System not being able to connect to the LDAP server.

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

Perform the following steps until the problem is resolved:

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

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

   You should see results similar to this:

   ```
   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 run into problems starting the administrative agent, view the administrative log or /usr/tank/admin/log/stderr.log and attempt to resolve any problems that 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 **/usr/tank/admin/bin/startConsole** to start the console. If you run into problems starting the console, view the log indicated by the error message and attempt to resolve any problems that you find.
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 the 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 the 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.

You can 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.

1. Obtain the current certificate. LDAP certificates are obtained 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 to each engine in the cluster.

   Note: 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 8. 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.

Table 8. LDAP parameters (continued)

Parameter name:	Sample values:	Description:
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.
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.

Table 8. LDAP parameters (continued)

Parameter name:	Sample values:	Description:
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 9.
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.

Table 9. 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

Table 9. LDAP configuration overrides (continued)

Parameters	Description	Example
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 8. Troubleshooting a SAN File System client

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

SAN File System clients run the following operating systems:

- AIX 5.1 (32–bit only) – maintenance level 3 or higher
- AIX 5.2 (32-bit and 64-bit)
- Red Hat Enterprise Linux Advanced Server 3.0
- Sun Solaris 9 (64-bit)
- Windows 2000 Server or Advanced Server – minimum Service Pack 4

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 an AIX client are accessed by a Windows client, access is controlled by the permissions of the Other permission bits.
- When files created by a Windows client are accessed by an AIX client, access is controlled by the permissions of the Everyone group.
- When files created by a Linux client are accessed by a Windows client, access is controlled by the permissions of the Other permission bits.
- When files created by a Windows client are accessed by a Linux client, access is controlled by the permissions of the Everyone group.
- When files created by a Solaris client are accessed by a Windows client, access is controlled by the permissions of the Other permission bits.
- When files created by a Windows client are accessed by a Solaris client, access is controlled by the permissions of the Everyone group.

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

- "Troubleshooting client access to data"
- "Troubleshooting client performance problems" on page 76

Troubleshooting client access to data

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

Problem

A client cannot access or update user data.

Investigation

If a client cannot create a new file

Perform the following steps until the problem is resolved:

1. Verify that the master metadata server is online.

2. Verify that the client has connectivity to the master metadata server.
   - From the client, attempt to ping or establish an SSH session with the metadata server.
   - From the Administrative command-line interface, run the lsclient command to see a list of all clients currently being served by the metadata servers in the cluster.

     sfscli> lsclient
3. Use the ls -l command to list the directory in which the file is to be created to verify that the entire path is correct and accessible.
4. Verify that you are logged into the client with a user name that has authorization to write files to directories.
   - From an AIX client, use the id command.
   - From a Windows client, right-click the directory. Then click **Properties→Security**.
5. Verify that your user name has the authorization to write files in the specific directory.

   Note: If you are logged into an AIX client as root or a Windows client as Administrator, make sure that you are running on a privileged client.
6. 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.
   a. Access the master metadata server through either the SAN File System console or the Administrative command-line interface.
   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 Administrative command-line interface, run the sfscli lsfileset -l command.
      - From the SAN File System, click **Manage Filing→Filesets**.
7. Make sure that the storage pool in which the file will be stored has sufficient space to store the file.
8. Verify that the client can access the storage device.
   - Use the datapath query command to ensure that the operating system can access the storage device.
   - On an AIX client, you can use the stfsdisk command to determine which disks can be seen. Then you can use the lsvol -l command to correlate the disk back to the device.

If a client cannot access an existing file

Perform the following steps until the problem is resolved:

1. Verify that your user name has the authorization to read files and that it has authorization to read the specific file.
2. Verify that the client can access the storage device.
   - Use the datapath query command to ensure that the operating system can access the storage device.
   - On an AIX client, you can use the stfsdisk command to determine which disks can be seen. Then you can use the lsvol -l command to correlate the disk back to the device.
3. Suspect a problem with corrupt SAN File System metadata.

If a client cannot update an existing file

Perform the following steps until the problem is resolved:

1. Verify that your user name has the authorization to write to the specific file.
2. Verify that the client can access the storage device.
   - Use the datapath query command to ensure that the operating system can access the storage device.
   - On an AIX client, you can use the stfsdisk command to determine which disks can be seen. Then you can use the lsvol -l command to correlate the disk back to the device.
3. 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, attempt to sign on to the SAN File System console. If you cannot, suspect one of the following:

   - One or more metadata servers in the cluster are down. See Chapter 6, "Troubleshooting the cluster," on page 47.

     Note: The administrative agent will automatically attempt to restart a metadata server if it goes down for any reason. Therefore, if you cannot access a metadata server, you might want to wait a few minutes and try again just 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 45.
2. View the system logs and look for any errors that may indicate I/O errors. Attempt to resolve all of these errors.
   - From a client running Windows, you can use the Event Viewer to view the Event Log.
   - From a client running AIX, you can view logging and tracing output, assuming it was previously enabled with the syslog facility.
3. From the Administrative command-line interface on the master metadata server, run the **lslun** 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:

   a. Run **stopserver** from the Administrative command-line interface to stop the SAN File System.
   b. Run **rmmod qla2300**.
   c. Run **insmod qla2300** .
   d. Run **/etc/rc.d/init.d/sanfs start** to start the SAN File System.
   e. Run **startserver** from the Administrative command-line interface to start the metadata server.
4. Verify that the client can access the storage device:
   - 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 seen. Then, from the Administrative

command-line interface, you can use the lsvol -l command to correlate the disk back to the device.

To rediscover all disks from a client running AIX, run the client command stfsdisk -discover.

The disks should automatically be rediscovered on a client running Windows. However, it they are not:

a. Right-click My Computer.
b. Select Manage.
c. Select Storage\Disk Management.
d. From the Action menu, select Rescan Disks.

5. Verify that none of the metadata servers in the cluster has recently terminated abnormally. If there was an abnormal termination of a metadata server, you may begin to see errors on the clients even after problems with the failed metadata server have been resolved. If you begin seeing these types of problems, you will need to restart clients:
   - On clients running AIX:
     a. Run rmstclient to unmount the global namespace, remove the virtual client, and unload the file-system driver.
     b. Run setupclient to load the file-system driver, create the virtual client, and mount the global namespace.
   - On clients running Windows, reboot the system.
6. 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 45.
7. Suspect a problem with corrupt SAN File System metadata.

If a client running Windows receives delayed write failure errors

A delayed write failure error may appear as a message box on the client desktop or in the Event Log. This message indicates that there has been an error writing data from the local file system cache to a storage device. Perform these steps until the problem is resolved:

1. The message will include the name of the file where the error occurred. Note the name of this file, as the data it contains may have been corrupted and the application using this file may encounter problems using this file’s data.
2. If the file is not part of the SAN File System, refer to your system documentation for resolving file system errors.
3. If the file is part of the SAN File System:
   a. View the Event Log and resolve any errors that may be related to this problem.
   b. Suspect a communications problem either with the SAN or between the client and the metadata server.

Troubleshooting client performance problems

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

Problem

A client is either 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 that is 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, 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 master the 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 6, "Troubleshooting the cluster," on page 47.

     Note: The administrative agent will automatically attempt to restart a metadata server that fails for any reason. Therefore, if you cannot access a metadata server, you might want to wait a few minutes and try again just 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 45.

Chapter 9. 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 80
- "Replacing Fibre Channel cable and GBICs" on page 80

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 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 10. 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.

1. Use the **builddrscript** 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:

   ```
   sfscli builddrscript recovery-file-name
   ```
2. The **builddrscript** 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 88 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 serverrs 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:

      bash# rpm -ivh *metadata server_package_name*.rpm
   d. Install the Qlogic software using the following command:

      bash# rpm -i snia_qlogic_hba*rpm
2. Reinstall the software package for the administrative server.
   a. Install the administrative server software package using the following command:

      bash# 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. Change the master metadata server to one of the subordinate metadata servers:
   1) Run /usr/tank/server/bin/tank lsserver to verify that the cluster does not have a master metadata server.
   2) Run /usr/tank/server/bin/tank lsengine to verify that the engine hosting the master metadata server is shut down.
   3) Run /usr/tank/server/bin/tank lsserver to verify that the metadata server to which you are going to assign the role of master is up and running.
   4) Access the subordinate metadata server that you want to set as the new master metadata server.
   5) Run the /usr/tank/server/bin/tank setmaster command to set it as the new master metadata server.
   6) If the previous master metadata server was managing any filesets, reassign those filesets to other metadata servers in the cluster.

b. 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.

c. 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.

d. Repeat steps 2a1 on page 61 through 2a6 on page 61 to change the master metadata server back to the original metadata server.
e. 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 metadata server master. 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.

   ```
   sfscli 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.

   ```
   sfscli -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
# forthis reason.
#########################################################################
```

b. Run the script TankSysCLI.volume.

```
sfscli -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.

```
sfscli -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 a 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 11. Getting help, service, and information

If you need help, service, technical assistance, or just want more information about IBM products, you will find a wide variety of sources available from IBM to assist you.

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 10 lists some of these pages.

Table 10. 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

Services available and telephone numbers listed are subject to change without notice.

Software Maintenance

All distributed software licenses include Software Maintenance (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. Extending coverage for a total of three years from date of acquisition may be elected. While your Software Maintenance is in effect, IBM will provide you assistance for your 1) routine, short duration installation and usage (how-to) questions; and 2) code-related questions. IBM provides assistance via 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, by using the online help, by looking in the online or printed documentation that comes with the SAN File System, or by consulting the support Web page noted in Table 10. Also, be sure to read the information in any README files and release notes that come with the SAN File System.

Getting help online

Be sure to visit the support page for the SAN File System, complete with FAQs, parts information, technical hints and tips, technical publications, and downloadable files, if applicable. This page is at: www.ibm.com/storage/support.

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 may 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.

Please 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 will vary depending on the number and complexity of incoming calls.

Appendix A. 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.

Appendix B. 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 in the sclif.properties file, located in your home directory on the engine (for example, joe/sclif.properties), to your valid LDAP 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 runs 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.

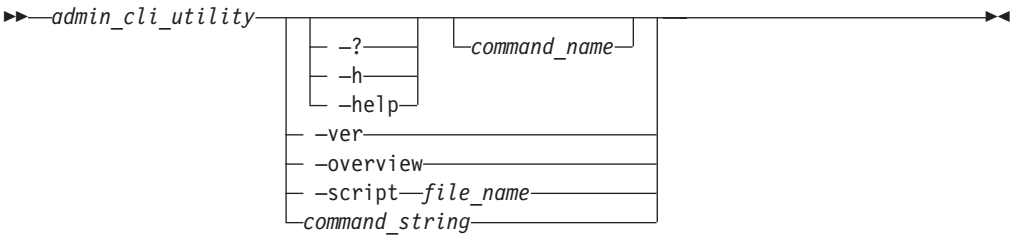
To use the client commands, you must log in directly to the client machine or from another workstation using SSH. You log in using the user ID and password for the client machine. You must have administrative (Windows) or root (UNIX-based) privileges to use the client commands.

Administrative 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 *admin_cli_utility*> 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.

Client commands

There is a set of commands for each client operating system that SAN File System supports.

AIX-client commands

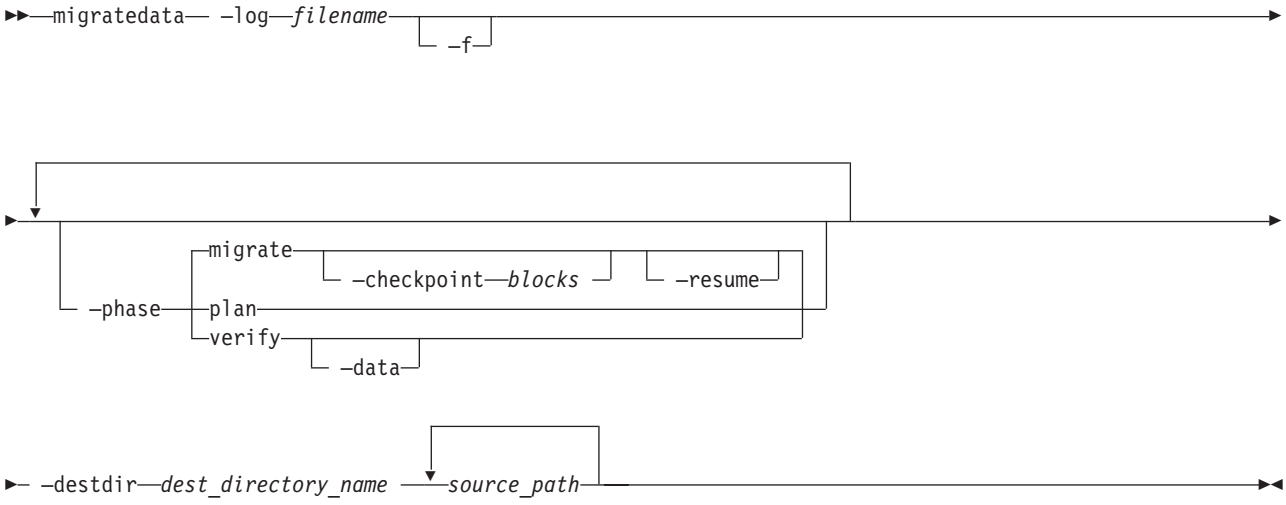
This topic provides a brief description for each AIX-client command.

Note: You must have root privileges to use these commands.

Command	Description
Migration	
"migratedata"	Migrates data to SAN File System.
Status	
"stfsstatus" on page 109	Displays the version of the file-system driver for the specified virtual client.
Volumes and LUNs	
"stfsdisk" on page 103	Scans the SAN File System for new and removed volumes.
Virtual client setup and removal	
"rmstclient" on page 99	Unmounts the global namespace, removes the SAN File System client, and unloads the file-system driver.
"setupstclient" on page 100	Configures and starts a client.
"stfsclient" on page 101	Creates or destroys a virtual client.
"stfsdriver" on page 105	Loads the file-system driver as a kernel extension.
"stfsmount" on page 107	Mounts the global namespace.
"stfsumount" on page 109	Unmounts the global namespace.

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 and overwrites the target file-system data with wrong information.
- 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, but continues with the migration process. 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 will 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.

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 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. IBM recommends 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.

You must run this command from the /usr/tank/migration/bin directory.

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.

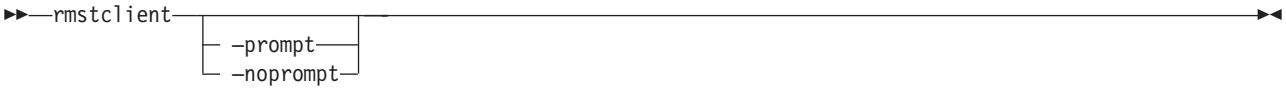
Example

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 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.

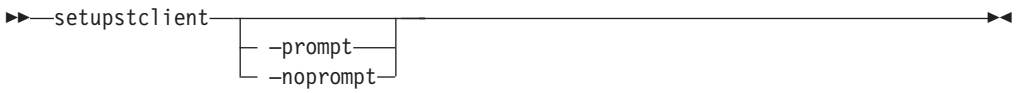
Example

Remove a client The following example removes the local SAN File System client without prompting:

```
rmstclient –noprompt
```

setupstclient

Configures and starts SAN File System clients.



Parameters

–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.

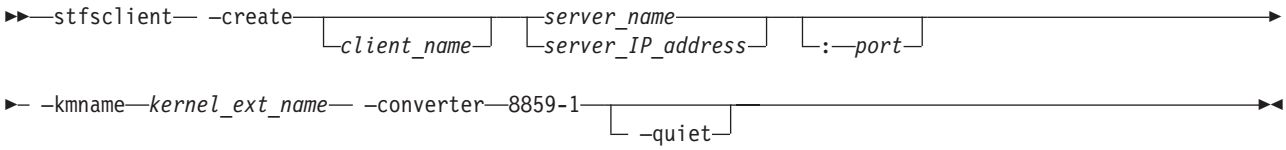
Example

Setup a client The following example configures and starts SAN File System clients:

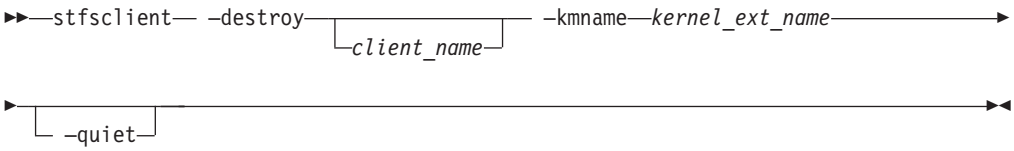
```
setupstclient
```

stfsclient

Creates or destroys a virtual client.



or



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. 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*.

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 just let –devices=dir=/dev/stfsdisk default.

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 client. 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.

The client considers a file to be one file even if it appears with two different file names in two different global namespace images.

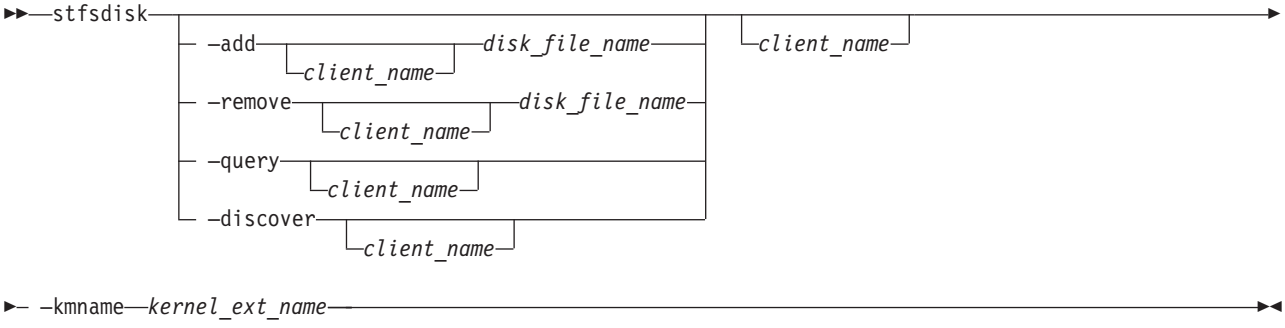
Example

Create a virtual client The following example creates a virtual client:

```
stfsclient –create MDS1:1700 –kmname /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. If the

disk-specific file name is already in the list, the client performs a disk discovery procedure on it, updating its database if indicated.

–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 there were I/O errors when the client tried to access the disk, 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/vpath0, not a logical volume file such as /dev/hdisk0.

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.

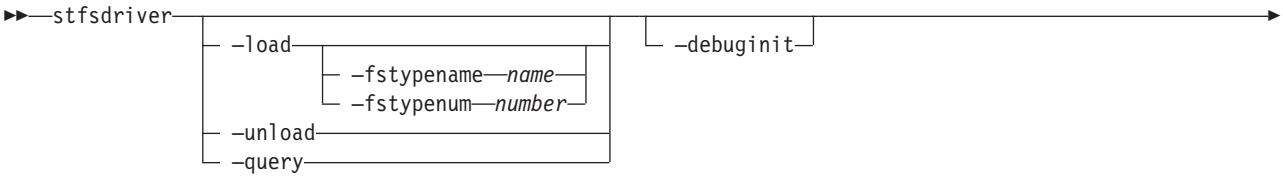
Example

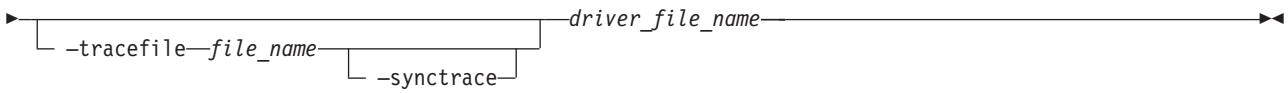
Query the disk-candidate list The following example queries disk-candidate list and displays the status of each disk:

```
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, an any already loaded file-system type number, or a number greater than 39.

–debuginit
Enables the file-system driver to issue diagnostic messages of the CONFIG class. Messages in this class are issued only during initialization.

If specified, the file-system driver does not issue diagnostic messages. You can turn the messages on after the file-system driver is running using the **stfsdebug** command.

Note: This parameter is intended for use only by trained service technicians.

–tracefile *file_name*
Specifies that the file-system driver is to write diagnostic information to the specified file.

Note: The specified file must already exist.

–synctrace
Specifies that the file-system drive is to write diagnostic information to the specified trace file synchronously rather than using buffered writes.

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 “sanfs”.

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 name and location of the file-system driver that was used to load the driver (for example, /usr/tank/client/bin/sanfs).

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.

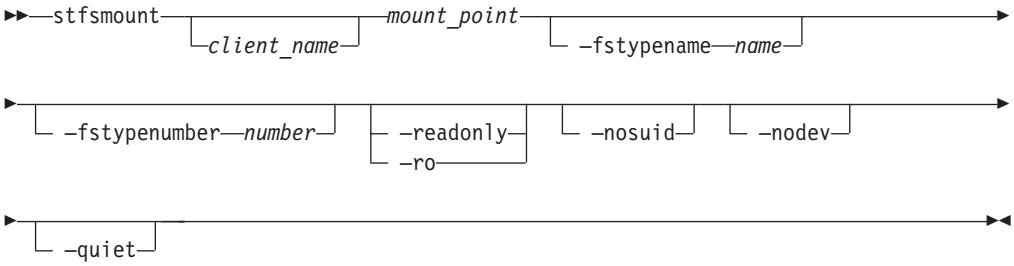
Example

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 or to see what global namespace images currently exist, use the **stfsmount** command.

To unmount the global namespace, use the **stfsumount** command.

Example

Mount the global namespace The following example mounts the global namespace:

```
stfsmount mnt/SANFS_MOUNTPT -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 drive, 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.

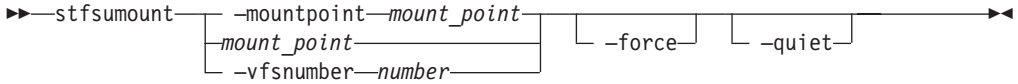
Example

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
```

stfsumount

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 destroys a global namespace image on the client system. It is used in place of the AIX **umount** command.

After you destroy 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 sanfs/cnt1
```

Linux-client commands

This topic provides a brief description for each Linux-client command.

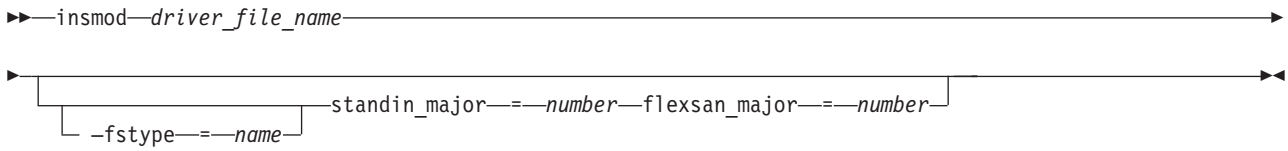
Note: You must have root privileges to use these commands.

Command	Description
Migration	
"migratedata" on page 97	Migrates data to SAN File System.
Virtual client setup and removal	
"insmod" on page 111	Loads the file-system driver as a kernel module.
"rmmod" on page 114	Unloads the file-system driver as a kernel module.
"rmstclient" on page 99	Unmounts the global namespace, removes the SAN File System client, and unloads the file-system driver.
"setupstclient" on page 100	Configures and starts a client.

Command	Description
"stfsclient" on page 114	Creates or destroys a virtual client.
"stfsmount" on page 117	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.

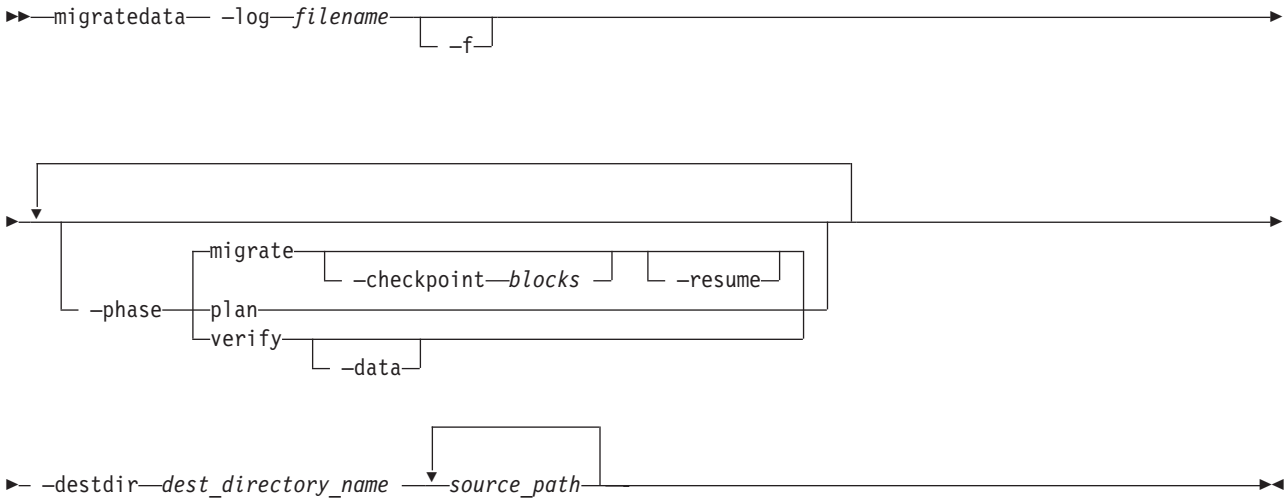
Example

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
```

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 and overwrites the target file-system data with wrong information.
- 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, but continues with the migration process. 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 will 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.

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 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. IBM recommends 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.

You must run this command from the /usr/tank/migration/bin directory.

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
```

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.

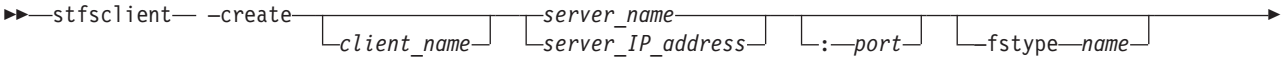
Example

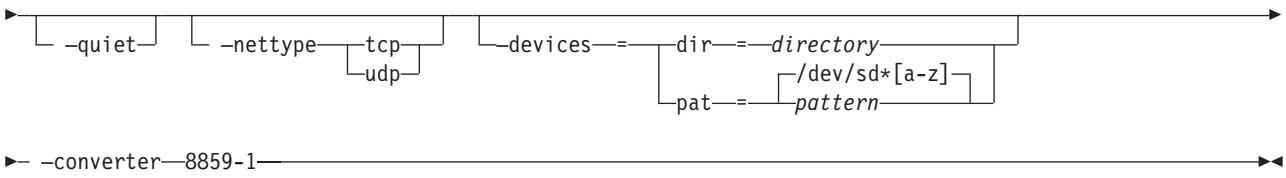
Unload the file-system driver The following example unloads the file-system driver on a client for Linux:

```
rmmod stfs
```

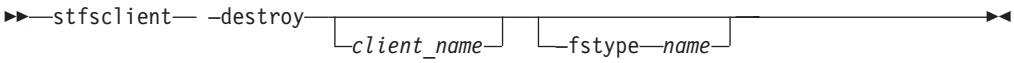
stfsclient

Creates or destroys a virtual client.





or



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. 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 * represents any alphabetic characters (a-z).

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.

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]**).

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.

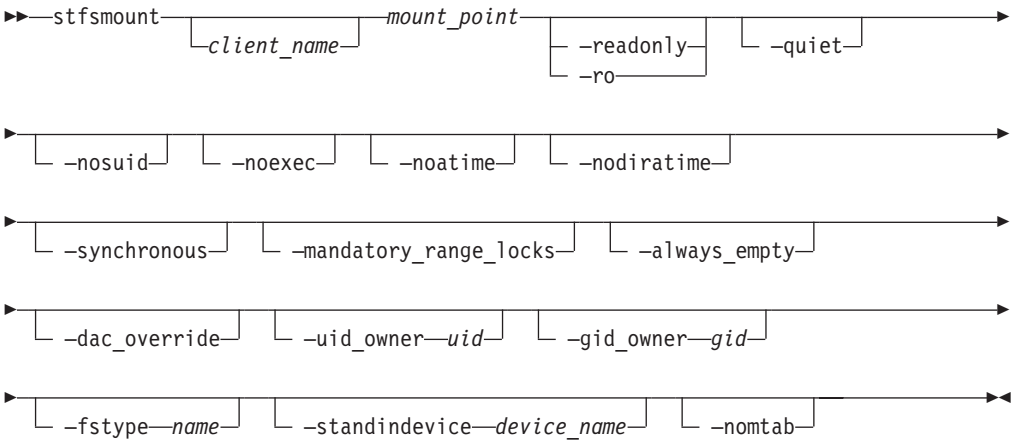
Example

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]:

```
stfsclient –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).

–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 uisng 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.

–always_empty
The Linux **statfs** command retrieves the approximate number of available blocks and free blocks in the global namespace. The number of *available blocks* is the total number of blocks in all storage pools presently in all the clusters that are reachable through the global namespace image. The number of free blocks is the number of blocks in all the clusters that are in partitions that are not assigned to any fileset.

Note: Based on these numbers, an application designed for a simpler file system might conclude that there is no space available to store new data when in fact there is plenty of space within partitions that are already assigned to the fileset, or that there is a large pool of storage not presently attached to a cluster but ready to be attached when demanded. If specified, the **statfs** command returns the largest number it can for the number of available blocks and returns that number minus one for the number of free blocks. If the number of blocks is too large for the **statfs** command to return, the command behaves as if you specified the –always_empty parameter. This is the case when you have more than 16 TB of space in the SAN File System.

–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/mtab 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. 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 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_MOUNTPT -fstype sanfs
```

Windows-client command

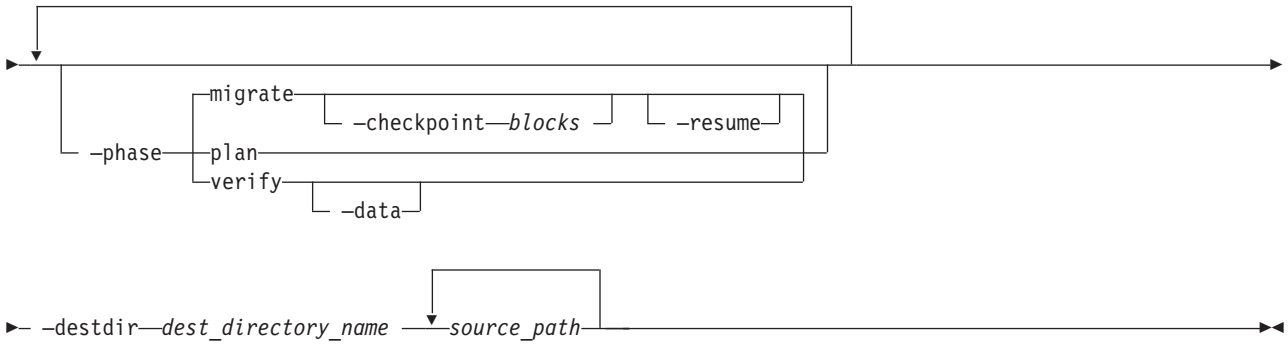
The topic provides a brief description for the Windows-client command.

Note: You must have Administrator privileges on the Windows client to use these commands.

Command	Description
Migration	
"migratedata" on page 97	Migrates data to SAN File System.

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 and overwrites the target file-system data with wrong information.
- 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, but continues with the migration process. 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 will 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.

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 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. IBM recommends 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.

You must run this command from the /usr/tank/migration/bin directory.

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
```

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 139	Starts the administrative agent.	root	shell
"stopCimom" on page 145	Stops the administrative agent.	root	shell
Cluster			
"mktruststore" on page 134	Creates a truststore that is shared by the administrative infrastructure to certify secure connections.	n/a	shell
"tank extractbootrecord" on page 146	Extracts a local Tank.Bootstrap file from a valid master volume.	Administrator	shell
"tank lscluster" on page 147	Displays the cluster definition from a system master volume when the metadata servers are not running.	Administrator	shell
"tank lsdisklabel" on page 149	Displays the a SAN File System product label for a specified device.	Administrator	shell

Command	Description	Role	Environment
"tank lsversion" on page 148	Displays the version control information from a system master disk.	Administrator	shell
"tank resetcluster" on page 151	Erases the static cluster definition contained on the system master volume.	Administrator	shell
"tank resetversion" on page 151	Resets the version-control information on a system master disk.	Administrator	shell
Engine			
"obdc" on page 135	Collects information for the first-failure data capture of SAN File System problems from the engine on which the metadata server resides.	Operator, Administrator	shell
Legacy CLI			
"legacy" on page 127	Runs commands in the legacy CLI.	Administrator	shell
SAN File System console			
"disableConsoleTrace" on page 126	Disables tracing and logging for the SAN File System console.	root	shell
"enableConsoleTrace" on page 126	Enables tracing and logging for the SAN File System console.	root	shell
"startConsole" on page 140	Starts the SAN File System console Web server.	root	shell
"stopConsole" on page 145	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 135	Collects information for the first-failure data capture of SAN File System problems from the client machine.	root (AIX) or Administrator (Windows)
"stfsdebug" on page 140	(AIX only) Enables or disables the logging of file-system-driver debug messages in the syslog for the specified virtual AIX client.	root

Command	Description	Role
"stfsdriver" on page 105	(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 141	(AIX only) Display statistics about the client.	root

disableConsoleTrace

Disables tracing and logging for the SAN File System console.

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 tracing The 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.

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.

This command save log and trace records to the /opt/was/logs/server*x*/trace.log file.

Enabling tracing will impact the performance of SAN File System. Use this command only when necessary.

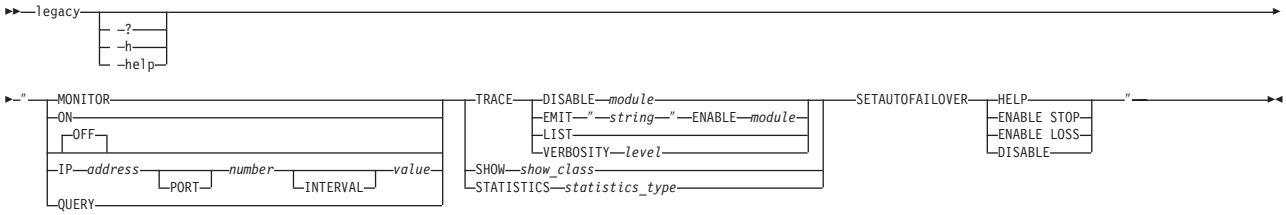
Example

Enable tracing The following example enables tracing and logging for the SAN File System console:

```
enableConsoleTrace
Enabling SAN File System console tracing...
```

legacy

Runs commands in the legacy command-line interface.



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.

MONITOR
Controls the real time monitor that runs as a part of a server and sends performance statistics to the central monitoring facility for the cluster.

Note: This parameter should be used only by trained service technicians.

ON Starts the real time monitor.

OFF Stops the real time monitor.

IP *address*
Specifies the IP address for the central monitoring facility.

PORT *number*
Specifies the port number for the central monitoring facility. The default is 2700.

INTERVAL *value*
Specifies the time duration in seconds after which the server collects and sends statistics. The default is 20 seconds.

QUERY
Displays real time monitor status and parameters.

TRACE
Configures tracing functions. If this parameter is specified without any additional options, this command prints online help about all the trace command variants

DISABLE *module*
Disables tracing on the specified module. If you do not specify a module, this command prints out all of the disabled trace modules. You can use single-wildcard (?) and multiple-wildcard characters (*) to specify more than one module

Example: TM.*

disables all of the TM modules

Example: WAL:WALF????:*

disables the WAL:WALFLUSH:WALWRITERREP and WAL:WALFORCE:WALWRITERREP modules.

EMIT *"string"*
Write the specified string to the trace log, /usr/tank/server/log/log.trace.

ENABLE *module*
Enables tracing on the specified module. If you do not specify a module, this command prints out all of the enabled trace modules. You can use single-wildcard (?) and multiple-wildcard characters (*) to specify more than one module (for example, TM:* would enable all of the TM modules, and WAL:WALF????:* would enable the WAL:WALFLUSH:WALWRITERREP and WAL:WALFORCE:WALWRITERREP modules.

Example: TM.*

enables all of the TM modules

Example: WAL:WALF????:*

enables the WAL:WALFLUSH:WALWRITERREP and WAL:WALFORCE:WALWRITERREP modules.

LIST Displays a list of all the trace modules that you can enable or disable in the metadata server. If a module is enabled, it is prefixed with an asterisk (*).

VERBOSITY *level*
Sets amount of output that is generated from tracing, based on the verbosity level. You can specify a value from 0 to 9. A value of 0, which is the default, emits only the most important messages, whereas a value of 9 emits all messages. If you do not specify a level, the current verbosity level is displayed.

SHOW *show_class*
Displays information about the specified show class.

Note: This parameter should be used only by trained service technicians. You can specify one of the following show classes:

ASMSESSIONS
Displays information about the administrative sessions.

BUFPOOL [*master***]**
Displays information about pages in the buffer pool, both the clean list and dirty list. Clean pages are pages available for reuse. The value for *master* is a flag that can be set to 0 or 1. The default is 0.

If the master metadata server is not specified, this command displays the buffer pool of the subordinate metadata server's workload. Specifying `SHOW BUFPOOL 1` displays the buffer pool of the master metadata server's workload.

CLEANLIST [*master***]**
Displays information about only clean pages in the buffer pool (see the **BUFPOOL** keyword).

CMATTACHPOINTS | CMATT
Displays the attach-points table.

CMARENAS
Displays the arenas table.

Tip: The output for this command can be quite large.

CMCP
Displays copy partition statistics.

CMFILESETS | CMCONT
Displays information about the filesets served by the subordinate metadata server.

CMLOADMAP [*detail***] | CMLOAD [***detail***]**
Displays the load map, which maps the write-ahead log to the metadata server. If the value for *detail* is greater than 0, this keyword displays additional information.

CMPOLSTAT *fileset*
Displays storage pool use by a fileset.

CMSMASTERREC | CMM
Displays the contents of the Cluster Manager master record.

CMSECTOR *sector_number*
Reads the raw contents of a Cluster Manager sector *sector_number* from Logical Volume Manager (LVM) and displays the data in dump format.

CMSPACERECLAIM | CMSPREC
Displays information about the space-reclamation thread on the master metadata server.

CMSTGPOOLS | CMSTG
Displays information about the storage pools.

CMSVCCLASSES | CMSVC
Displays the service-class table.

CMVOLUMES | CMVOL
Displays the volumes table.

CMCONTBIND | CMCB
Displays the fileset-bindings table, which maps the write-ahead-log volume to which each fileset is bound.

CMWALVOLS | MWALS
Displays the write-ahead-log volumes table.

CONDITION | COND
Displays information about the condition of the metadata server.

DBPAGE *space_ID page_address* **[***format***] [***master***]**
Displays the contents of a page.

DBPAGEHDR *space_ID page_address* **[***format***] [***master***]**
Displays the contents of a page (header only).

DBPAGETABLE [*master***]**
Displays information about locked pages (see the **BUFPOOL** keyword).

DBSPACE [*master***]**
Displays information about DB Spaces (see the **BUFPOOL** keyword).

DBTXNTABLE [*master***]**
Displays information about in-flight transactions (see the **BUFPOOL** keyword).

DIODISKS
Displays the open disk table.

DIRTYLIST [*master***]**
Displays information about only dirty pages in the buffer pool (see the **BUFPOOL** keyword).

DISPATCHERSTATS | DSPSTATS
Displays dispatcher and queue statistics.

FLASHCLEANER *fileset_ID* **[***flags***] [***dump_ID***]**
Displays information about a fileset's FlashCopy images.

FLASHTABLE [*fileset_ID***]**
Displays information about a fileset's FlashCopy image table.

FSCK Displays information about an active metadata check that is in progress.

FSCKSUB
Displays information about an active metadata check that is in progress on a subordinate metadata server.

GIODISKS
Displays information about the global disk.

GSNODES
Displays information about the engine.

INDEXSTATS *space_id page_address* **[***format***] [***master***] | INDEX** *space_id page_address* **[***format***] [***master***]**
Verifies the structural integrity of an index, and shows statistics on the index as a side-effect.

LATCH
Displays all active latches.

LATCHSTATS
Displays latch contention statistics.

LATCHX
Displays all active exclusive latches.

LATCHXSTATS
Displays exclusive-latch contention statistics.

LMCLIENTS
Displays a list of clients that are known to the lock manager.

LMALLLOCKS
Display all of the locks in the lock manager.

LMLOCKSBYCLIENT [*client_ID***]**
Display all of the locks held by the specified client. To display a list all registered clients, use the **TMCLIENTS** keyword.

LMLOCKSBYOID [*O***] [***I***] [***D***] | LMLOCKS [***O***] [***I***] [***D***]**
Display all of the locks held for the specified object.

LMSTATS
Displays summary statistics on the lock manager.

LVMDISKS
Displays the disk table.

LVMMASTERREC | LVMMR
Displays master control block, including:
- Global ID of master disk.
- Sector size of master disk.
- Size, in bytes, of the logical and physical partitions for this LVM installation.
- Installation common sector size, bytes.
- Number of sectors per partition.
- Number of physical partitions reserved to hold LVM persistent metadata.
- Starting sector numbers for the shadow copies of the LVM persistent tables.
- Index (0 or 1) of the shadow copy that contains the committed copy of the LVM persistent tables.
- Update sequence (version) number.

LVMPARTMAP | LVMPM *volume_ID*
Displays a volume's partition map.

LVMREP
Displays information about the volume manager on a subordinate metadata server.

LVMVOLDESC *volume_ID*
Displays information about a volume's description.

LVMVOLUMES
Displays the volume table, including the following information:
- Logical volume class.
- Logical volume ID.
- Size, in bytes, of logical pages.
- Capacity, in number of logical partitions.
- Number of formatted logical partitions.
- Node ID of subordinate that has locked this logical volume.
- Partition map for each logical partition, including the following data:
  - Logical partition.
  - LVM-assigned disk number of physical disk.

- Partition status flags.
- Physical partition number on disk.

MASTERWALSTATS | MWALSTATS
Displays write-ahead-log statistics for the master metadata server.

MUTEX
Displays all active mutexes (mutual exclusion locks).

MUTEXSTATS
Displays statistics about mutexes.

OMFILESETS | OMCONT [*fileset_ID***]**
Displays all fileset information.

OMFILESETSTATS | OMCS [*fileset_ID***]**
Displays statistics about a specific fileset.

OMOBJ [raw | pit | write | main] [*object***] [count** *count***] [start** *start***]**
Displays object.

OPTCONFIG
Displays information on the metadata server configurable parameters.

PMCONFLICTS [ALL | *command***]**
Displays a list of conflicting commands for each of all the existing commands or for a specified command.

PMTABLE
Displays information about long-running administrative commands.

SCARENASTATS | SCAS [*fileset_ID***] [***pool_ID***]**
Displays statistics about a fileset's free-space map as well as the free-space map cache.

SCSTSDSTATS | SCSS [*fileset_ID***]**
Displays statistics about a fileset's security descriptor table.

SERVERHANG
Displays the major data structures used for detecting when a metadata server hangs.

STATEINFO
Displays information about the state of the cluster and metadata servers.

SUBWALSTATS
Displays write-ahead-log statistics for the subordinate metadata server.

THREAD (STATE [force]) | (STACK [ALL | *thread_ID***]) | USAGE | HELP**
Displays information about threads.

TMATIMESTATS
Displays information about transaction manager access time attribute (tmatime) statistics.

TMATIMESTATSRESET
Resets information about tmatime statistics.

TMCLIENTS
Displays statistics for registered clients.

TMFILTERS
Displays information about metadata server transaction-manager filters.

TMPATH *cluster_ID fileset_ID local_OID* **[***epoch_ID***]**
Displays information about metadata server transaction-manager filters.

VERSION
Displays information about the metadata server version.

WALCKPT [*master***]**
Displays write-ahead-log context for the local node (see the **BUFPOOL** keyword).

WALSTATS [*master***]**
Displays write-ahead-log statistics (see the **BUFPOOL** keyword).

WALWRITER [*master*]
Displays write-ahead-log statistics (see the **BUFPOOL** keyword).

STATISTICS *statistics_type*
Displays information about the specified show class. You can specify one of the following types:

RESET
Resets the statistics buffer.

SHOWALL
Displays all statistics about the Cluster Manager master transactions and the SAN File System protocol received by the master and subordinate metadata servers.

SHOWBTREE
Displays B-tree statistics. A B-tree is a file access structure that provides a balanced search tree depth when accessing records. B-trees are used by SAN File System to store file system object and metadata information.

SHOWBUF
Displays statistics about the buffer pool.

SHOWCM
Displays statistics about the Cluster Manager master transactions received by the master and subordinate metadata server.

SHOWDSP
Displays dispatcher and queue statistics.

SHOWGIO
Displays statistics about the global disk table.

SHOWLATCH
Displays statistics about latches.

SHOWLATCHS
Displays statistics about extended latches.

SHOWLM
Displays statistics about the lock manager.

SHOWMUTEX
Displays statistics about mutexes.

SHOWSTP
Displays statistics about the SAN File System protocol by the message type that is received by the metadata servers.

SHOWWAL
Displays write-ahead-log statistics for the master and subordinate metadata server.

SETAUTOFAILOVER
Enables or disables the failover script.

HELP Displays a short help message.

ENABLE STOP
Enables the failover script to run for a **stopserver** command.

ENABLE LOSS
Enables the failover script to run for a server loss.

DISABLE
Disables all failover scripts from running.

Prerequisites

This task must be performed only by trained service technicians.

Description

If the legacy command string contains single quotation marks (') or double quotation marks (″), you must precede the character with a backslash (\).

Example

Running a legacy command The following example runs the show command from the legacy CLI:

```
sfscli> legacy "SHOW TMCLIENTS"
List of identified clients:
  ClientId=1010513811, ClientName="st-client-1", SendInProg=0, SendMsgNo=2,
  ccb_sendAcked=1, DeliveredMsgNo=1, IPaddress=192.168.1.10, IPport= 32769
```

mktruststore

Obtains a public certificate from LDAP to 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 you want to read the path from stdin (for example, – << /work/path.txt).

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 sfscli.

The truststore file resides in the /usr/tank/admin/ directory.

You would use the **mktruststore** command is used 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.

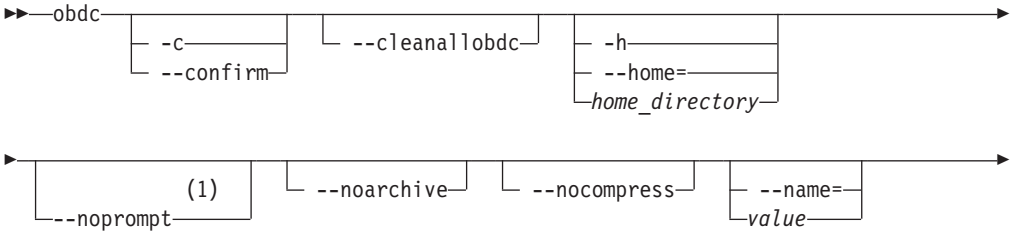
Example

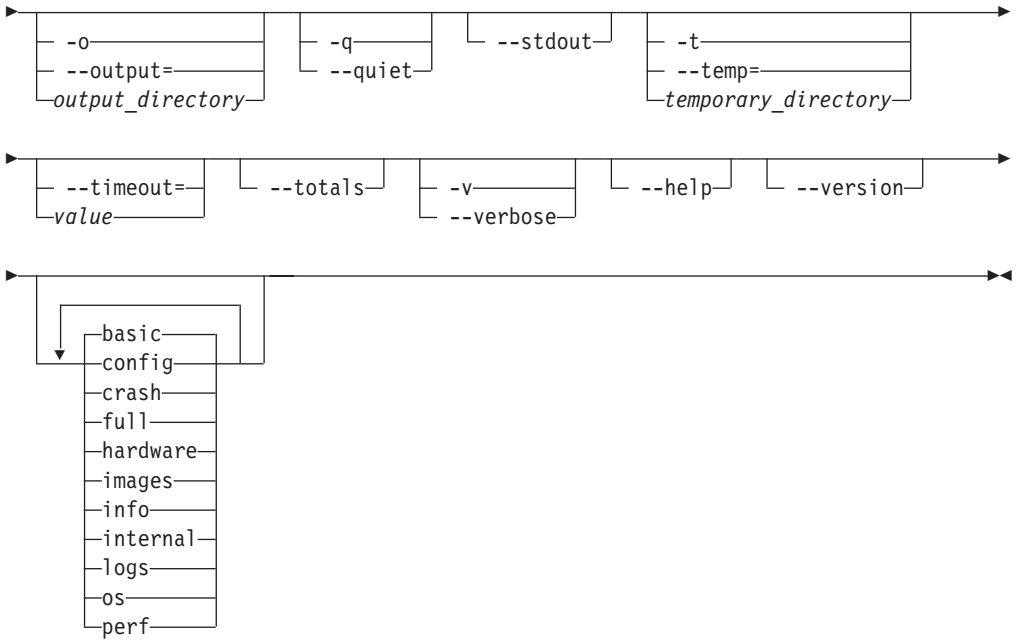
Create truststore The following example creates a truststore and imports the LDAP certificate ldap.cert from the local directory:

```
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 /user/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***-***odbc_PID*, where *odbc_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***-***odbc_PID*.tar, where for example, OBDC-012204-1312-28494.tar and if you do not specify

--nocompress, the name of the archive file is OBDC-*mmddyy-time-odbc_PID*.tar.gz, where *odbc_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 /user/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 as 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.

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
```

startCimom

Starts the administrative agent.

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 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.

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.

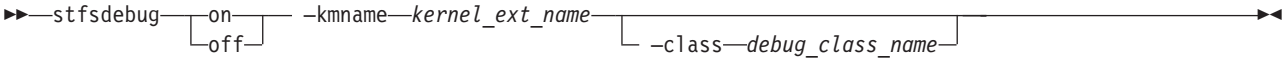
Example

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
```

stfsdebug

Enables or disables the logging of file-system-driver debug messages in the syslog for the specified virtual client for AIX. Note that the equivalent command for the Solaris operating system is the **sanfs_ctl trace** command.



Parameters

on Enables the logging debug messages in the syslog.

off
Disables the logging of debug messages in the syslog.

–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 drive, 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).

–class *debug_class_name*
Identifies the class name for debug messages. If not specified, this command applies all classes. You can specify any of the following values:

ATTR File attribute information.

CONFIG
Driver initialization, termination, and control.

DISK Disk configuration at client-creation time

IO I/O operations.

MOUNT
FSIs and clients creation and destruction.

PAGER
Pager and items that give work to the pager.

RDWR
Reading from and writing to file cache.

RNGLOCK
Range lock information.

VFSOP
vfs operations, excl get root vnode

VNODE
Management of vnodes, gnodes, and stfsnodes.

VNODEOP
vnode operations, excl lookup, hold and release vnode.

Prerequisites

This task must be performed only by trained service technicians.

Example

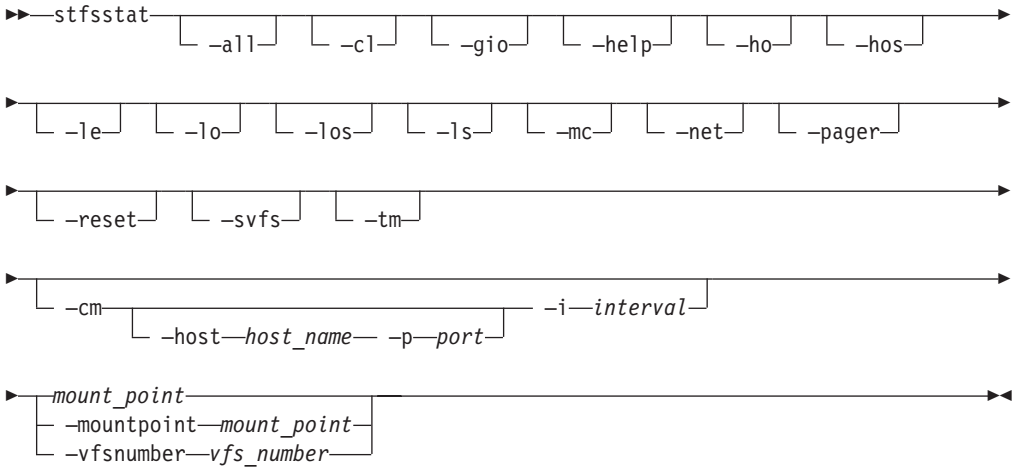
Enable debugging on a client for AIX The following example enables the logging of file-system-driver debug messages in the syslog for the specified virtual client:

```
stfsdebug on -kmname /lib/stfs
```

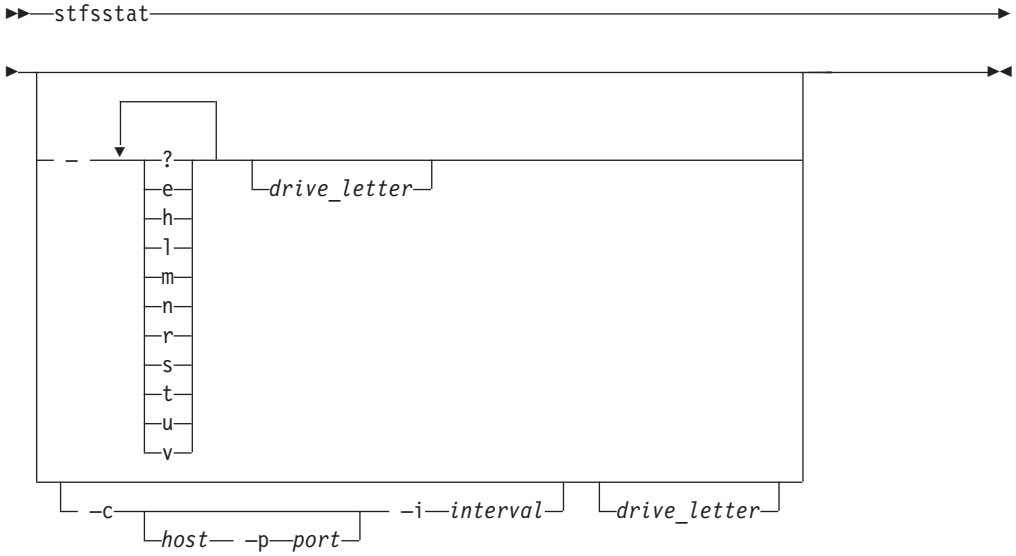
stfsstat

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



Clients for Windows



Parameters

Clients for AIX

–all
Displays all client for AIX 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.

mount_point | **–mountpoint** *mount_point*
The mount point where you want to mount the file-system image (for example, the directory above which you want the global namespace image's directory tree to appear).

–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**.

Example

Displays statistics from a client for AIX The following example displays transaction-manager statistics and metadata-cache statistics for the local client for AIX 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 the local client for Windows mapped to drive S:

```
stfsstat –he s
```

stopCimom

Stops the administrative agent.

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 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.

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.

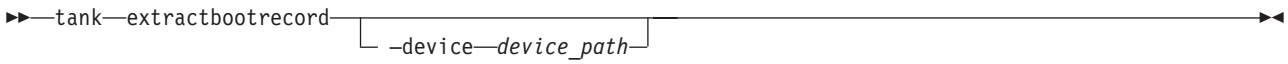
Example

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.



Parameters

–device *device_path*
Specifies the device path for a valid master volume from to read the product label.

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be offline.

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 /etc/tank/server/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 on all engines 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/srdc 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.

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be offline.

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.

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.

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, rvpath*n* where *n* is the vpath letter. If the device path is not specified, the command might return inaccurate information.

Prerequisites

This task must be performed only by trained service technicians.

The cluster must be offline.

Description

This command is run from the shell prompt. It is not run inside of a sfscli 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 offline.

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 offline.

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
```

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> admin_cli_utility lsport
DEFAULT,Default,10000,2500,25,80,10,64,Default Storage Pool
admin_cli_utility> exit
shell>
```

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 *admin_cli_utility*> prompt, for example:

```
shell> admin_cli_utility
admin_cli_utility> lspool -l -type default
DEFAULT,Default,10000,2500,25,80,10,64,Default Storage Pool
admin_cli_utility> 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> admin_cli_utility -script ~/bin/listpools.bat
```

You can add comments to the script file by placing a pound sign (#) in the first column, for example:

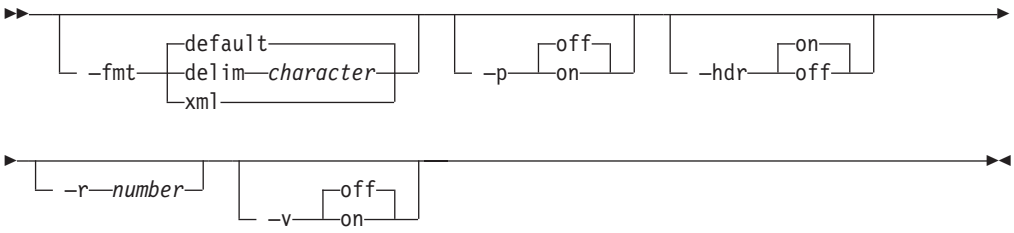
```
# This script file lists the default storage pool.
lspool -l -type default
```

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.

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)   Description
================================================
     10                    64   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:

```
DEFAULT,Default,10000,2500,25,80,10,64,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 listing parameters

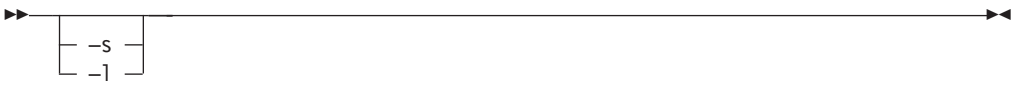
The standard listing parameters specify whether to display the default, long, or short output for listing commands (commands that start with ls*) in the administrative CLI. You can specify these parameters in addition to the parameters available for a specific listing commands.

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 you want to supply parameters from an input stream (stdin) rather than entering parameters. In the following example, the command line gets input from the file /work/myfile:

```
cmd - >> work/myfile
```

In the following example, the command line gets input from the keyboard:

```
cmd - >>
parameter_1
parameter_2
```

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.

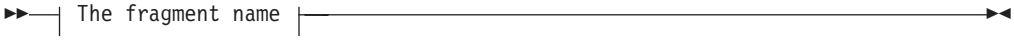
Required items

When a keyword, variable, or operand appears on the main line, you must specify that item. In the following example, you must choose A and B.

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.

Syntax fragments

Commands that contain lengthy groups or a section that is used more that 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

Variables

Italicized, lowercase elements denote variables. In the following example, you must specify a variable name when you enter the keyword command:

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 C. 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 D. 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, 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	Tivoli	TotalStorage
WebSphere	xSeries	

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.

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. 

Index

A

C

D

T

U

V

W

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 1

Publication No. GA27-4318-01

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

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. _____

Readers' Comments — We'd Like to Hear from You
GA27-4318-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

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

GA27-4318-01

Cut or Fold
Along Line

Printed in USA

GA27-4318-01