Veritas Access 7.2.1 Administrator's Guide

Linux

7.2.1

VERITAS™

Veritas Access Administrator's Guide

Last updated: 2017-05-18

Document version: 7.2.1 Rev 2

Legal Notice

Veritas Technologies LLC
500 E Middlefield Road
Mountain View, CA 94043

http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:

https://www.veritas.com/support

You can manage your Veritas account information at the following URL:

https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The document version appears on page 2 of each guide. The latest documentation is available on the Veritas website:

https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:

doc.feedback@veritas.com

You can also see documentation information or ask a question on the Veritas community site:

http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:

https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Section 1

Introducing Veritas Access

Chapter 1

Introducing Veritas Access

This chapter includes the following topics:

About Veritas Access

Veritas Access is a software-defined scale-out network-attached storage (NAS) solution for unstructured data that works on commodity hardware. Veritas Access provides resiliency, multi-protocol access, and data movement to and from the public or private cloud based on policies.

You can use Veritas Access in any of the following ways.

Table 1-1 Interfaces for using Veritas Access

Interface	Description
GUI	Centralized dashboard with operations for managing your storage. See the GUI and the online Help for more information.
RESTful APIs	Enables automation using scripts, which run storage administration commands against the Veritas Access cluster. See the *Veritas Access RESTful API Guide* for more information.
Command-line interface (CLI or CLISH)	Single point of administration for the entire cluster. See the manual pages for more information.

Table 1-2 describes the features of Veritas Access.

Table 1-2 Veritas Access key features

Feature	Description
Multi-protocol access	Veritas Access includes support for the following protocols: ■ Amazon S3 See "About the object server" on page 83. See the `objectaccess(1)` manual page. ■ CIFS See "About configuring Veritas Access for CIFS" on page 64. See the `cifs(1)` manual page. ■ FTP See "About FTP" on page 79. See the `ftp(1)` manual page. ■ NFS See "About using NFS server with Veritas Access" on page 59. See the `nfs(1)` manual page. ■ Oracle Direct NFS See "About using Veritas Access with Oracle Direct NFS" on page 74. See the `database(1)` manual page. ■ SMB 3 See the `cifs(1)` manual page.
Flexible Storage Sharing (FSS)	Enables cluster-wide network sharing of local storage.
Scale-out file system	The following functionality is provided for a scale-out file system: ■ File system that manages a single namespace spanning over both on-premises storage as well as cloud storage, which provides better fault tolerance for large data sets. See "About scale-out file systems" on page 111. See the `storage_fs(1)` manual page. ■ Highly available NFS and S3 shares. You use scale-out file systems if you want to store a large capacity of data in a single namespace (3 PB is the maximum file system size). See "Using the NFS-Ganesha server" on page 60. See the `storage_fs(1)` manual page.

Table 1-2 Veritas Access key features *(continued)*

Feature	Description
Cloud as a tier for a scale-out file system	Veritas Access supports adding a cloud service as a storage tier for a scale-out file system. You can move data between the tiers based on file name patterns and when the files were last accessed or modified. Use scheduled policies to move data between the tiers on a regular basis. Veritas Access moves the data from the on-premises tier to Amazon Glacier, Amazon Web Services (AWS) S3, or AWS S3-compatible directly based on automated policy management. You can also retrieve data archived in Amazon Glacier. See the `storage_cloud(1)` manual page. See "Configuring the cloud as a tier feature for scale-out file systems" on page 92. See the `storage_tier(1)` manual page. See the `storage_fs(1)` manual page if you want to create a policy with a schedule.
ISCSI target	ISCSI target support for block storage serving was introduced as a Technical Preview feature. See "About iSCSI targets for block storage" on page 81. See the `target(1)` manual page.
SmartIO	Veritas Access supports read caching on solid state drives (SSDs) for applications running on Veritas Access file systems. See "About SmartIO for solid-state drives " on page 175. See the `smartio(1)` manual page.
SmartTier	Veritas Access's built-in SmartTier feature can reduce the cost of storage by moving data to lower-cost storage. Veritas Access storage tiering also facilitates the moving of data between different drive architectures and on-premises. See "About Veritas Access SmartTier " on page 169. See the `storage_tier(1)` manual page.

Table 1-2 Veritas Access key features *(continued)*

Feature	Description
Snapshot	Veritas Access supports snapshots for recovering from data corruption. If files, or an entire file system, are deleted or become corrupted, you can replace them from the latest uncorrupted snapshot. See "About snapshots" on page 182. See the `storage_snapshot(1)` manual page.
Deduplication	You can run post-process periodic deduplication in a file system, which eliminates duplicate data without any continuous cost. See "About data deduplication" on page 161. See the `storage_dedup(1)` manual page.
Compression	You can compress files to reduce the space used, while retaining the accessibility of the files and having the compression be transparent to applications. Compressed files look and behave almost exactly like uncompressed files: the compressed files have the same name, and can be read and written as with uncompressed files. See "About compressing files" on page 166. See the `storage_compress(1)` manual page.
NetBackup integration	Built-in NetBackup client for backing up your file systems to a NetBackup master or media server. Once data is backed up, a storage administrator can delete unwanted data from Veritas Access to free up expensive primary storage for more data. Veritas Access as backup storage for NetBackup over S3 with OpenDedup. See "About Veritas Access as a NetBackup client" on page 187. See the `backup(1)` manual page.

Table 1-2 Veritas Access key features *(continued)*

Feature	Description
OpenStack plug-in	Integration with OpenStack: ■ OpenStack Cinder integration that allows OpenStack instances to use the storage hosted by Veritas Access. See "About the Veritas Access integration with OpenStack Cinder" on page 144. ■ OpenStack Manila integration that lets you share Veritas Access file systems with virtual machines on OpenStack Manila. See "About the Veritas Access integration with OpenStack Manila" on page 151. See the `openstack(1)` manual page.
Quotas	Support for setting file system quotas, user quotas, and hard quotas. See the `storage_quota(1)` manual page.
Replication	Periodic replication of data over IP networks. See "About Veritas Access file-level replication" on page 179. See the `replication(1)` manual page.
Support for LDAP, NIS, and AD	Veritas Access uses the Lightweight Directory Access Protocol (LDAP) for user authentication. See "About configuring LDAP settings" on page 26. See the `network_ldap(1)` manual page.
Partition Directory	With support for partitioned directories, directory entries are redistributed into various hash directories. These hash directories are not visible in the name-space view of the user or operating system. For every new create, delete, or lookup, this feature performs a lookup for the respective hashed directory and performs the operation in that directory. This leaves the parent directory inode and its other hash directories unobstructed for access, which vastly improves file system performance. By default this feature is not enabled. See the `storage_fs(1)` manual page to enable this feature.

Table 1-2 Veritas Access key features *(continued)*

Feature	Description
Isolated storage pools	Enables you to create an isolated storage pool, which contains its own configuration files. An isolated storage pool protects the pool from losing the associated metadata if a disk in another storage pool fails. See "About configuring storage pools" on page 30. See the `storage_pool(1)` manual page.
Performance and tuning	Workload-based tuning for the following workloads: ■ Media server - Streaming media represents a new wave of rich Internet content. Recent advancements in video creation, compression, caching, streaming, and other content delivery technology have brought audio and video together to the Internet as rich media. You can use Veritas Access to store your rich media, videos, movies, audio, music, and photos. ■ Virtual machine See "About creating a tuned file system for a specific workload" on page 119. See the `storage_fs(1)` manual page.

Accessing the Veritas Access CLI

To access the Veritas Access CLI

1. After installation, connect to the management console using the console IP address you assigned during the installation.
2. Log on to the management console node using the following credentials:

   User name: `master`

   Default password: `master`

   You are prompted to change your password after the initial logon.
3. For subsequent logons, use the user name `master` with the new password that you created.

   You can add additional users at any time.

   The End User License Agreement (EULA) is displayed the first time you log on to the Veritas Access CLI.

Navigating the Veritas Access CLI

All of the Veritas Access CLI commands are organized in different command modes depending on the operation you want to perform. You can get a list of the different command modes with descriptions of all the available modes by typing a question mark (`?`) at the CLI prompt.

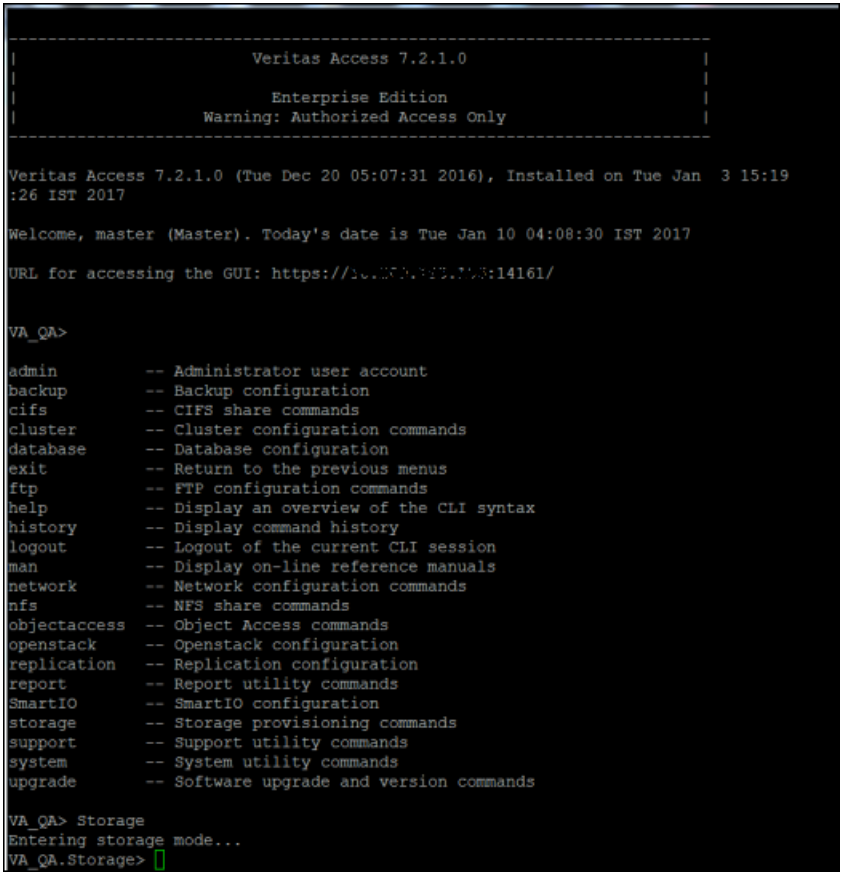
If you are using the support account to log on to Veritas Access, you can use `su - master` in the terminal of the console IP to access the Veritas Access CLI.

To navigate the Veritas Access CLI

1. After logging on to the Veritas Access CLI, type a question mark (`?`) to see the available command modes.
2. Enter the `Storage>` mode by typing `storage` for example.

   You can see that you are in the `Storage>` mode because the cluster name now displays with the command mode.

   clustername`.Storage`


Veritas Access 7.2.1.0
Enterprise Edition
Warning: Authorized Access Only
Veritas Access 7.2.1.0 (Tue Dec 20 05:07:31 2016), Installed on Tue Jan 3 15:19
:26 IST 2017
Welcome, master (Master). Today's date is Tue Jan 10 04:08:30 IST 2017
URL for accessing the GUI: https://:14161/
VA_QA>
admin -- Administrator user account
backup -- Backup configuration
cifs -- CIFS share commands
cluster -- Cluster configuration commands
database -- Database configuration
exit -- Return to the previous menus
ftp -- FTP configuration commands
help -- Display an overview of the CLI syntax
history -- Display command history
logout -- Logout of the current CLI session
man -- Display on-line reference manuals
network -- Network configuration commands
nfs -- NFS share commands
objectaccess -- Object Access commands
openstack -- Openstack configuration
replication -- Replication configuration
report -- Report utility commands
SmartIO -- SmartIO configuration
storage -- Storage provisioning commands
support -- Support utility commands
system -- System utility commands
upgrade -- Software upgrade and version commands
VA_QA> Storage
Entering storage mode...
VA_QA.Storage>

Section 2

Configuring Veritas Access

Chapter 2

Configuring users or roles

This chapter includes the following topics:

About user roles and privileges

Your privileges within Veritas Access are based on what user role (Master, System Administrator, or Storage Administrator) you have been assigned.

The following table provides an overview of the user roles within Veritas Access.

Table 2-1 User roles within Veritas Access

User role	Description
Master	Masters are responsible for adding or deleting users, displaying users, and managing passwords. Only the Masters can add or delete other administrators.
System Administrator	System Administrators are responsible for configuring and maintaining the file system, NFS sharing, networking, clustering, setting the current date/time, and creating reports.
Storage Administrator	Storage Administrators are responsible for provisioning storage and exporting and reviewing reports.

The `Support` account is reserved for Technical Support use only, and it cannot be created by administrators.

Chapter 3

Configuring the network

This chapter includes the following topics:

About configuring the Veritas Access network

Veritas Access has the following types of networks:

- Private network
  The network between the nodes of the cluster itself. The private network is not accessible to Veritas Access client nodes.
- Public network
  The public network is visible to all clients. Veritas Access uses static IP address for its public interface networking. Veritas Access does not support DHCP for public network configuration

Veritas Access supports the following operations to manage the networking settings:

- create bond
- vlan
- change IP addresses
- add or remove network interfaces
- swap or interchange existing network interfaces

About bonding Ethernet interfaces

Bonding associates a set of two or more Ethernet interfaces with one IP address. The association improves network performance on each Veritas Access cluster node by increasing the potential bandwidth available on an IP address beyond the limits of a single Ethernet interface. Bonding also provides redundancy for higher availability.

For example, you can bond two 1-gigabit Ethernet interfaces together to provide up to 2 gigabits per second of throughput to a single IP address. Moreover, if one of the interfaces fails, communication continues using the single Ethernet interface.

When you create a bond, you need to specify a bonding mode. In addition, for the following bonding modes: `802.3ad`, `balance-rr`, `balance-xor`, `broadcast`, `balance-tlb`, and `balance-alb`, make sure that the base network interface driver is configured correctly for the bond type. For type `802.3ad`, the switch must be configured for link aggregation.

Consult your vendor-specific documentation for port aggregation and switch set up. You can use the `-s` option in the Linux `ethtool` command to check if the base driver supports the link speed retrieval option. The `balance-alb` bond mode type works only if the underlying interface network driver enables you to set a link address.

Note: An added IPv6 address may go into a TENTATIVE state while bonding Ethernet interfaces with `balance-rr`, `balance-xor`, or `broadcast` bond modes. While bonding with those modes, Veritas Access requires the switch to balance incoming traffic across the ports, and not deliver looped back packets or duplicates. To work around this issue, enable EtherChannel on your switch, or avoid using these bond modes.

Table 3-1 Bonding mode

Index	Bonding mode	Fault tolerance	Load balancing	Switch setup	Ethtool/base driver support
0	balance-rr	yes	yes	yes	no
1	active-backup	yes	no	no	no
2	balance-xor	yes	yes	yes	no
3	broadcast	yes	no	yes	no
4	802.3ad	yes	yes	yes	yes (to retrieve speed)

Table 3-1 Bonding mode *(continued)*

Index	Bonding mode	Fault tolerance	Load balancing	Switch setup	Ethtool/base driver support
5	balance-tlb	yes	yes	no	yes (to retrieve speed)
6	balance-alb	yes	yes	no	yes (to retrieve speed)

Note: When you create or remove a bond, SSH connections with Ethernet interfaces involved in that bond may be dropped. When the operation is complete, you must restore the SSH connections.

About the IP addresses for the Ethernet interfaces

Internet Protocol (IP) commands configure your routing tables, Ethernet interfaces, and IP addresses, and display the settings.

The following sections describe how to configure the Ethernet interfaces:

- See "About Ethernet interfaces" on page 24.
-

About Ethernet interfaces

Each Ethernet interface must have a physical IP address associated with it. These are usually supplied when the Veritas Access software is installed.

Each Ethernet interface can be configured with a virtual IP address for clustering purposes in Veritas Access. This does not imply that each interface must have a virtual IP to communicate with the network.

The physical address must be present before adding a virtual address. To add an IPv6 address on an IPv4 cluster, you have to configure the IPv6 physical address and then add the virtual address for the given interface.

About configuring routing tables

Sometimes a Veritas Access cluster must communicate with network services (for example, LDAP) using specific gateways in the public network. In these cases, you must define routing table entries.

These entries consist of the following:

- The target network node's IP address and accompanying netmask.
- Gateway’s IP address.
- Optionally, a specific Ethernet interface via which to communicate with the target. This is useful, for example, if the demands of multiple remote clients are likely to exceed a single gateway’s throughput capacity.

Chapter 4

Configuring authentication services

This chapter includes the following topics:

About configuring LDAP settings

The Lightweight Directory Access Protocol (LDAP) is the protocol used to communicate with LDAP servers. The LDAP servers are the entities that perform the service. In Veritas Access, the most common use of LDAP is for user authentication.

For sites that use an LDAP server for access or authentication, Veritas Access provides a simple LDAP client configuration interface.

Before you configure Veritas Access LDAP settings, obtain the following LDAP configuration information from your system administrator:

- IP address or host name of the LDAP server. You also need the port number of the LDAP server.
- Base (or root) distinguished name (DN), for example:

  ```
  cn=employees,c=us
  ```

  LDAP database searches start here.
- Bind distinguished name (DN) and password, for example:

  ```
  ou=engineering,c=us
  ```

  This allows read access to portions of the LDAP database to search for information.

- Base DN for users, for example:

  ```
  ou=users,dc=com
  ```

  This allows access to the LDAP directory to search for and authenticate users.
- Base DN for groups, for example:

  ```
  ou=groups,dc=com
  ```

  This allows access to the LDAP database, to search for groups.
- Base DN for Netgroups, for example:

  ```
  ou=netgroups,dc=com
  ```

  This allows access to the LDAP database, to search for Netgroups.
- Root bind DN and password. This allows write access to the LDAP database, to modify information, such as changing a user's password.
- Secure Sockets Layer (SSL). Configures a cluster to use the Secure Sockets Layer (SSL) protocol to communicate with the LDAP server.
- Password hash algorithm, for example, `md5`, if a specific password encryption method is used with your LDAP server.

Section 3

Managing Veritas Access storage

Chapter 5

Configuring storage

This chapter includes the following topics:

About storage provisioning and management

When you provision storage, you want to be able to assign the appropriate storage for the particular application. Veritas Access supports a variety of storage types. To help the users that provision the storage to select the appropriate storage, you classify the storage into groups called storage pools. A storage pool is a user-defined way to group the disks that have similar characteristics.

Veritas Access supports a wide variety of storage arrays, direct attached storage as well as in-server SSDs and HDDs. During the initial configuration, you add the disks to the Veritas Access nodes. For a storage array, a disk is a LUN from the storage array. For best performance and resiliency, each LUN should be provisioned to all Veritas Access nodes. Local disks and fully shared disks have unique names, but partially shared disks across nodes may have the same name. Make sure that you do not assign LUNs from the same enclosure to different nodes partially.

Before you can provision storage to Veritas Access, the physical LUNs must be set up and zoned for use with the Veritas Access cluster. The storage array administrator normally allocates and zones the physical storage.

Veritas Access does not support thin reclamation disks.

After the disks are correctly discovered by Veritas Access, you assign the disks to storage pools. You create a file system on one or more storage pools. You can mirror across different pools. You can also create tiers on different pools, and use SmartTier to manage file system data across those tiers.

By default, all of the storage pools in Veritas Access share the same configuration. A copy of the configuration file resides on one of the disks in the configuration. To isolate configuration, an isolated storage pool must be created. Isolated storage pool contains its own configuration files, which protect the pool from losing the associated metadata if default configuration fails.

You can also use local disks that are shared over the network. Both DAS disks and SAN disks (LUNs) can be used by the same cluster, and you can have a mix of DAS and SAN disks in the same storage pool.

About configuring disks

Disks and pools can be specified in the same command provided the disks are part of an existing storage pool.

The pool and disk that are specified first are allocated space before other pools and disks.

If the specified disk is larger than the space allocated, the remainder of the space is still utilized when another file system is created spanning the same disk.

About configuring storage pools

Veritas Access uses storage pools to provision storage. Pools are more a logical construct rather than an architectural component. Pools are loosely collections of disks.

In the Veritas Access context, a disk is a LUN provisioned from a storage array. Each LUN should be provisioned to all Veritas Access nodes. Disks must be added to pools prior to use.

During the initial configuration, you create storage pools, to discover disks, and to assign them to pools. Disk discovery and pool assignment are done once. Veritas Access propagates disk information to all the cluster nodes.

You must first create storage pools that can be used to build file systems on. Disks and pools can be specified in the same command provided the disks are part of an existing storage pool.

The pool and disk specified first are allocated space before other pools and disks.

If the specified disk is larger than the space allocated, the remainder of the space is still utilized when another file system is created spanning the same disk.

By default, all of the storage pools in Veritas Access share the same configuration. A copy of the configuration file resides on one of the disks in the configuration. The first storage pool you create uses the default configuration. You can create additional storage pools to be part of that default configuration or to be isolated. An isolated storage pool contains its own configuration files, which protects the pool from losing the associated metadata if a disk in another storage pool fails. If isolated storage pools exist, you cannot remove the disks in the non-isolated pool or destroy the last non-isolated pool.

About quotas for usage

Disk quotas limit the usage for users or user groups. You can configure disk quotas for file systems or for CIFS home directories.

Note: Quota works over NFS, but quota reporting and quota details are not visible over NFS.

Users and groups visible through different sources of name service lookup (nsswitch), local users, LDAP, NIS, and Windows users can be configured for file systems or CIFS home directory quotas.

There are two types of disk quotas:

- Usage quota (numspace) - limits the amount of disk space that can be used on a file system.

  The numspace quota value must be an integer with a unit. The minimum unit is KB, because the block size in the underlying Veritas File System (VxFS) is 1KB, and VxFS calculates numspace quotas based on the number of KBs. The range for numspace is from 1K to 9007199254740991(2^53 - 1)K.

- Inode quota (numinodes) - limits the number of inodes that can be created on a file system.

  An inode is a data structure in a UNIX or UNIX-like file system that describes the location of some or all of the disk blocks allocated to the file.

  The numinodes quota value must be an integer without a unit, and the range is from 1 to 9999999999999999999(19bit).

  0 is valid for numspace and numinodes, which means the quota is infinite.

Veritas Access supports disk quota limits greater than 2 TB.

In addition to setting a limit on disk quotas, you can also define a warning level, or soft quota, whereby the Veritas Access administrator is informed that they are

nearing their limit, which is less than the effective limit, or hard quota. Hard quota limits can be set so that a user is strictly not allowed to cross quota limits. A soft quota limit must be less than a hard quota limit for any type of quota.

Note: The alert for when a hard limit quota or a soft limit quota is reached in Veritas Access is not sent out immediately. The hard limit quota or soft limit quota alert is generated by a cron job scheduled to run daily at midnight.

About quotas for CIFS home directories

You use `Storage> quota cifshomedir` commands to configure quotas for CIFS home directories. Users and groups visible through different sources of name service lookup (nsswitch), local users, LDAP, NIS, and Windows users can be configured for CIFS home directory quotas.

Default values are entered in a configuration file only. The actual application of the quota is done with the `set` and `setall` commands using the default values provided.

When a CIFS home directory file system is changed, quota information for a user's home directory is migrated from the existing home directory file system to the new home directory file system.

Quota migration results are based on the following logic:

- Case 1:
  In the case where the existing home directory file system is NULL, you can set the new home directory file system to be multiple file systems (for example, fs1, fs2). If the multiple file systems previously had different quota values, the quota status and values from the first file system are migrated to other file systems in the new home directory. The first file system is the template. Only the user/group quota values that existed on the first file system are migrated. Other user/group quota values remain the same on the other file system.

  For example, assume the following:

  - The new home directory file systems are fs1 and fs2.
  - user1, user2, and user3 have quota values on fs1.
  - user2, user3, and user4 have quota values on fs2.

  For the migration, user/group quota values for user1, user2, and user3 are migrated from fs1 to fs2. Quota values for user4 are kept the same on fs2, and user4 has no quota values on fs1.
- Case 2:
  When the existing home directory file systems are already set, and you change the file systems for the home directory, the quota status and values need to be

migrated from the existing home directory file systems to the new file systems. For this migration, the first file system in the existing home directory acts as the template for migrating quota status and values.

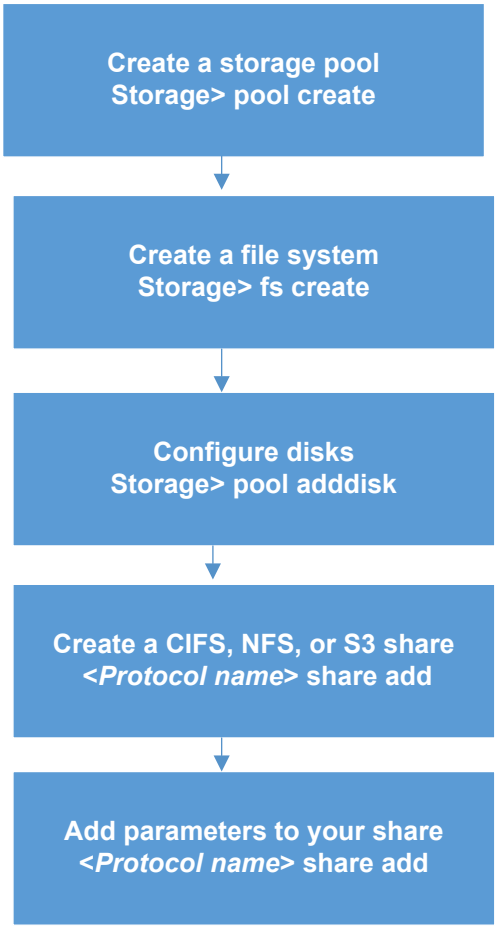
For example, if the existing home directory file systems are fs1 and fs2, and the file systems are changed to fs2, fs3, and fs4, then the user/group quota values on fs1 are migrated to fs3 and fs4. Other user/group values on fs3 and fs4 remain the same.

Workflow for configuring and managing storage using the Veritas Access CLI

Figure 5-1 describes configuring and managing storage using the Veritas Access CLI.

See the Veritas Access manual pages for the detailed syntax for completing the operations.

Figure 5-1 Workflow for configuring and managing Veritas Access storage using the CLI



Chapter 6

Configuring data integrity with I/O fencing

This chapter includes the following topics:

About I/O fencing

In the Veritas Access cluster, one method of communication between the nodes is conducted through heartbeats over private links. If the two nodes cannot communicate, the two nodes cannot verify each other's state. Neither node can distinguish if the failed communication is because of a failed link or a failed partner node. The network breaks into two networks that cannot communicate with each other but do communicate with the central storage. This condition is referred to as the "split-brain" condition.

I/O fencing protects data integrity if the split-brain condition occurs. I/O fencing determines which nodes retain access to the shared storage and which nodes are removed from the cluster, to prevent possible data corruption.

In Veritas Access, I/O fencing has the following modes:

- Disk-based I/O fencing uses coordinator disks for arbitration in the event of a network partition. Coordinator disks are standard disks or LUNs that are set aside for use by the I/O fencing driver. The coordinator disks act as a global lock device during a cluster reconfiguration. This lock mechanism determines which node is allowed to fence off data drives from other nodes. A system must eject a peer from the coordinator disks before it can fence the peer from the data drives. Racing for control of coordinator disks is how fencing helps prevent split-brain. Coordinator disks cannot be used for any other purpose. You cannot store data on them.

To use the disk-based I/O fencing feature, you enable fencing on each node in the cluster. Disk-based I/O fencing always requires an odd number of disks starting with three disks. You must also specify the three disks to use as coordinator disks. The minimum configuration must be a two-node cluster with Veritas Access software installed and more than three disks. Three of the disks are used as coordinator disks and the rest of the disks are used for storing data.

- Majority-based I/O fencing provides support for high availability when there are no additional servers or shared SCSI-3 disks that can act as coordination points. The cluster must have an odd number of nodes. In case a split-brain condition occurs, the sub-cluster with more than half of the nodes remains online. If a sub-cluster has less than half of the nodes, then it panics itself.

  For Veritas Access, majority-based fencing is used for Flexible Storage Sharing. Majority-based I/O fencing is administered only with the CLISH.

Chapter 7

Configuring ISCSI

This chapter includes the following topics:

About the iSCSI initiator

The Internet Small Computer System Interface (iSCSI) is an Internet protocol-based storage networking standard that links data storage facilities. By carrying SCSI commands over IP networks, iSCSI facilitates data transfers over Intranets and manages storage over long distances.

The iSCSI feature allows Veritas Access servers to use iSCSI disks as shared storage. Veritas Access can just act as a iSCSI initiator and not as a iSCSI target system.

Configuring the iSCSI initiator

To display the iSCSI initiator service

- To display the status of the iSCSI initiator service, enter the following:

  ```
  Storage> iscsi status
  ```

  For example:

  ```
  iSCSI Initiator Status on ACCESS_01 : ONLINE
  iSCSI Initiator Status on ACCESS_02 : ONLINE
  ```

To start the iSCSI initiator service

- To start the iSCSI initiator service, enter the following:

  ```
  Storage> iscsi start
  ```

  For example:

  ```
  Storage> iscsi start
  Storage> iscsi status
  iSCSI Initiator Status on ACCESS_01 : ONLINE
  iSCSI Initiator Status on ACCESS_02 : ONLINE
  ```

To stop the iSCSI initiator service

- To stop the iSCSI initiator service, enter the following:

  ```
  Storage> iscsi stop
  ```

  For example:

  ```
  Storage> iscsi stop
  Storage> iscsi status
  iSCSI Initiator Status on ACCESS_01 : OFFLINE
  iSCSI Initiator Status on ACCESS_02 : OFFLINE
  ```

Configuring the iSCSI initiator name

Veritas Access generates iSCSI initator names for each node.

You can set the prefix that Veritas Access uses to generate initiator names. Veritas Access names each initiator with this prefix followed by the node number of the node.

To display the iSCSI initiator names

- To display the iSCSI initiator names, enter the following:

  ```
  Storage> iscsi initiator name list
  ```

  For example:

  ```
  Storage> iscsi initiator name list
  Node              Initiator Name
  ----              --------------
  ACCESS_01         iqn.2009-05.com.test:test.1
  ACCESS_02         iqn.2009-05.com.test:test.2
  ```

To configure the iSCSI initiator name

- To configure the iSCSI initiator name prefix, enter the following:

  ```
  Storage> iscsi initiator name setprefix initiatorname-prefix
  ```

  where *initiatorname-prefix* is a name that conforms to the naming rules for initiator and target names as specified in RFC3721. Initiator names for nodes in the cluster are generated by appending the node number to this prefix.

  For example:

  ```
  Storage> iscsi initiator name setprefix iqn.2009-05.com.test:test
  ```

Configuring the iSCSI devices

The iSCSI initiator contains a list of network devices (network interfaces) from which connections are made to targets.

You can add or delete devices from this list.

When you add a device for use with the iSCSI initiator, iSCSI initiator connections use this device to connect to the target. If there are any existing targets, then the iSCSI initiator initiates a connection to all targets by using the newly set devices.

When you delete a `device` from the iSCSI configuration, any existing connections by way of the `device` to targets is terminated. If there are existing targets, you cannot delete the last device in the iSCSI initiator configuration.

To display the list of devices

- To display the list of devices, enter the following:

  ```
  Storage> iscsi device list
  ```

  For example:

  ```
  Storage> iscsi device list
  Device
  ------
  pubeth0
  pubeth1
  ```

To add an iSCSI device

- To add an iSCSI device, enter the following:

  ```
  Storage> iscsi device add device
  ```

  where *device* is the device where the operation takes place.

  For example:

  ```
  Storage> iscsi device add pubeth1
  Storage> iscsi device list
  Device
  ------
  pubeth0
  pubeth1
  ```

To delete an iSCSI device

- To delete an iSCSI device, enter the following:

  ```
  Storage> iscsi device delete device
  ```

  where *device* is the device where the operation takes place.

  For example:

  ```
  Storage> iscsi device add pubeth1
  Storage> iscsi device list
  Device
  ------
  pubeth0
  ```

Configuring discovery on iSCSI

The iSCSI initiator contains a list of iSCSI target discovery addresses.

To display the iSCSI discovery addresses

- To display the iSCSI discovery addresses, enter the following:

  ```
  Storage> iscsi discovery list
  ```

  For example:

  ```
  Storage> iscsi discovery list
  Discovery Address
  -----------------
  192.168.2.14:3260
  192.168.2.15:3260
  ```

To add a discovery address to the iSCSI initiator

1 To add a discovery address to the iSCSI initiator, enter the following:

```
Storage> iscsi discovery add discovery-address
```

where:

discovery-address The target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720.

You can specify either an IPv4 address or an IPv6 address. Optionally, you can specify a port with the IP address. For example:

```
192.168.0.4
192.168.0.4:3260
2001:c90::211:9ff:feb8:a9e9
[2001:c90::211:9ff:feb8:a9e9]:3260
```

If no port is specified, the default port `3260` is used. Verify that your firewall allows you to access the target location through the port. For example:

```
# telnet discovery-address 3260
```

For example:

```
Storage> iscsi discovery add 192.168.2.15:3260
Discovery CHAP credentials for ACCESS_1:
Outgoing CHAP Username : root
Outgoing CHAP Password : ********
Incoming CHAP Username :
Authentication succeeded.

Discovered Targets
------------------
iqn.2001-04.com.example:storage.disk2.sys3.xyz
iqn.2001-04.com.example:storage.disk3.sys3.xyz
iqn.2001-04.com.example:storage.disk4.sys3.xyz
iqn.2001-04.com.example:storage.disk5.sys3.xyz

Logging into target iqn.2001-04.com.example:storage.disk2.sys3.xyz
Logging into target iqn.2001-04.com.example:storage.disk3.sys3.xyz
Logging into target iqn.2001-04.com.example:storage.disk4.sys3.xyz
Logging into target iqn.2001-04.com.example:storage.disk5.sys3.xyz
```

2 To verify the addition of the discovery address, display the discovery addresses.

```
Storage> iscsi discovery list
```

For example:

```
Storage> iscsi discovery list
Discovery Address
-----------------
192.168.2.14:3260
192.168.2.15:3260
```

To delete an iSCSI discovery address

1 To delete the targets discovered using this discovery address, enter the following:

```
Storage> iscsi discovery del discovery-address
```

where:

discovery-address The target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720.

You can specify either an IPv4 address or an IPv6 address. Optionally, you can specify a port with the IP address. For example:

```
192.168.0.4
192.168.0.4:3260
2001:c90::211:9ff:feb8:a9e9
[2001:c90::211:9ff:feb8:a9e9]:3260
```

If no port is specified, the default port `3260` is used. Verify that your firewall allows you to access the target location through the port. For example:

```
# telnet discovery-address 3260
```

For example:

```
Storage> iscsi discovery del 192.168.2.15:3260
```

2 To verify the deletion of the discovery address, display the discovery addresses.

```
Storage> iscsi discovery list
Discovery Address
-----------------
192.168.2.14:3260
```

To rediscover an iSCSI discovery address

- To rediscover an iSCSI discovery address, enter the following:

```
Storage> iscsi discovery rediscover discovery-address
```

where:

discovery-address The target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720.

You can specify either an IPv4 address or an IPv6 address. Optionally, you can specify a port with the IP address. For example:

```
192.168.0.4
192.168.0.4:3260
2001:c90::211:9ff:feb8:a9e9
[2001:c90::211:9ff:feb8:a9e9]:3260
```

If no port is specified, the default port `3260` is used. Verify that your firewall allows you to access the target location through the port. For example:

```
# telnet discovery-address 3260
```

For example:

```
Storage> iscsi discovery rediscover 192.168.2.15:3260
Deleted targets
-----------------
iqn.2001-04.com.example:storage.disk5.sys3.xyz

New targets
-----------------
iqn.2001-04.com.example:storage.disk6.sys3.new.xyz

Logging into target iqn.2001-04.com.example:storage.disk6.sys3.new.xyz
```

To rediscover changes in targets or LUNs at a discovery address

- To rediscover changes in targets or LUNs at a discovery address, enter the following:

```
Storage> iscsi discovery rediscover_new discovery-address
```

where:

discovery-address The target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720.

You can specify either an IPv4 address or an IPv6 address. Optionally, you can specify a port with the IP address. For example:

```
192.168.0.4
192.168.0.4:3260
2001:c90::211:9ff:feb8:a9e9
[2001:c90::211:9ff:feb8:a9e9]:3260
```

If no port is specified, the default port `3260` is used. Verify that your firewall allows you to access the target location through the port. For example:

```
# telnet discovery-address 3260
```

New LUNs or targets discovered at *discovery-address* will be automatically added and logged into. This command does not discover any targets that have been deleted at *discovery-address*.

For example:

```
Storage> iscsi discovery rediscover_new 192.168.2.15:3260
14% [|] Checking for new targets

New targets
-----------------
iqn.2001-04.com.example:storage.disk7.sys3.new.xyz

100% [#] Updating disk list
```

Configuring the iSCSI targets

To display the iSCSI targets

- To display the iSCSI targets, enter the following:

```
Storage> iscsi target list
```

For example:

```
Storage> iscsi target list
Target
------
iqn.2001-04.com.example:storage.disk2.sys3.xyz
iqn.2001-04.com.example:storage.disk4.sys3.xyz
iqn.2001-04.com.example:storage.disk5.sys3.xyz
iqn.2001-04.com.example:storage.disk3.sys3.xyz
iqn.2001-04.com.example2:storage.disk2.sys3.xyz
iqn.2001-04.com.example2:storage.disk3.sys3.xyz
iqn.2001-04.com.example2:storage.disk4.sys3.xyz
iqn.2001-04.com.example2:storage.disk5.sys3.xyz

Discovery Address  State  Disk
------------------ ----- -----
192.168.2.14:3260  ONLINE disk_0
192.168.2.14:3260  ONLINE disk_2
192.168.2.14:3260  ONLINE disk_3
192.168.2.14:3260  ONLINE disk_1
192.168.2.15:3260  ONLINE disk_4
192.168.2.15:3260  ONLINE disk_5
192.168.2.15:3260  ONLINE disk_6
192.168.2.15:3260  ONLINE disk_7
```

To display the iSCSI target details

- To display the iSCSI target details, enter the following:

  `Storage> iscsi target listdetail target`

  where *target* is the name of the node you want to display the details for.

  This list also shows targets discovered at `discovery-address`, not only manually added targets.

  For example:

  `Storage> iscsi target listdetail iqn.2001-04.com.example:storage.disk2.sys3.xyz`

```
Discovery Address : 192.168.2.14:3260
Connections
===========
Portal Address           ACCESS_01   ACCESS_02
--------------           ---------   ---------
192.168.2.14:3260,1        2           2
```

To add an iSCSI target

- To add an iSCSI target, enter the following:

```
Storage> iscsi target add target-name portal-address
```

target-name	Name of the iSCSI target at which SCSI LUNs are available. `target-name` should conform to the naming rules defined in RFC3721.

portal-address	The location where the target is accessible.
	You can specify either an IPv4 address or an IPv6 address.
	For example:
	`192.168.0.4`
	`192.168.0.4,1`
	`192.168.0.4:3260`
	`192.168.0.4:3260,1`
	`2001:c90::211:9ff:feb8:a9e9`
	`2001:c90::211:9ff:feb8:a9e9,1`
	`[2001:c90::211:9ff:feb8:a9e9]:3260`
	`[2001:c90::211:9ff:feb8:a9e9]:3260,10`

For example:

```
Storage> iscsi target add iqn.2001-04.com.example:
storage.disk2.sys1.xyz 192.168.2.14:3260

Logging into target iqn.2001-04.com.example:
storage.disk2.sys1.xyz
Storage> iscsi target listdetail iqn.2001-04.com.example:
storage.disk2.sys1.xyz

Connections
===========
Portal Address     ACCESS55_01       ACCESS55_02
--------------     -----------       ------------
192.168.2.14:3260,1     1             1
```

To delete an iSCSI target

- To delete an iSCSI target, enter the following:

```
Storage> iscsi target del target-name
{discovery-address|portal-address}
```

target-name	Name of the iSCSI target at which SCSI LUNs are available. `target-name` should conform to the naming rules defined in RFC3721.
discovery-address	Target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720. If no port is specified with the discovery address, default port `3260` is used.
portal-address	The location where the target is accessible.

For example:

```
Storage> iscsi target del iqn.2001-04.com.example:
storage.disk2.sys3.xyz
```

To login to an iSCSI target

- To log in to an iSCSI target, enter the following:

```
Storage> iscsi target login target-name
{discovery-address | portal-address}
```

target-name	Name of the iSCSI target at which SCSI LUNs are available. `target-name` should conform to the naming rules defined in RFC3721.
discovery-address	Target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720. If no port is specified with the discovery address, default port `3260` is used.
portal-address	The location where the target is accessible.

For example:

```
Storage> iscsi target login iqn.2001-04.com.example:
storage.disk2.sys3.xyz
```

To logout from an iSCSI target

- To logout from an iSCSI target, enter the following:

```
Storage> iscsi target logout target-name
{discovery-address | portal-address}
```

target-name	Name of the iSCSI target at which SCSI LUNs are available. `target-name` should conform to the naming rules defined in RFC3721.
discovery-address	Target address at which an initiator can request a list of targets using a `SendTargets` text request as specified in iSCSI protocol of RFC3720. If no port is specified with the discovery address, default port `3260` is used.
portal-address	The location where the target is accessible.

For example:

```
Storage> iscsi target logout iqn.2001-04.com.example:
storage.disk2.sys3.xyz
```

To rescan targets for new LUNs

- To rescan a target for a new LUN, enter the following:

  `Storage> iscsi target rescan target-name`

  where *target-name* is the name of the iSCSI target that you want to rescan.

  You can use the `Storage> iscsi target rescan` command for both static targets and discovered targets.

  For example:

```
Storage> iscsi target rescan iqn.2001-04.com.example:storage.disk2
.sys3.xyz
100% [#] Updating disk list
Storage> iscsi target list
Target
------
iqn.2001-04.com.example:storage.disk2.sys3.xyz
iqn.2001-04.com.example:storage.disk4.sys3.xyz
iqn.2001-04.com.example:storage.disk5.sys3.xyz
iqn.2001-04.com.example:storage.disk3.sys3.xyz
iqn.2001-04.com.example2:storage.disk2.sys3.xyz
iqn.2001-04.com.example2:storage.disk3.sys3.xyz
iqn.2001-04.com.example2:storage.disk4.sys3.xyz
iqn.2001-04.com.example2:storage.disk5.sys3.xyz

Discovery Address   State    Disk
-----------------   -----    ----
192.168.2.14:3260   ONLINE   disk_0 disk_8 disk_9
192.168.2.14:3260   ONLINE   disk_2
192.168.2.14:3260   ONLINE   disk_3
192.168.2.14:3260   ONLINE   disk_1
192.168.2.15:3260   ONLINE   disk_4
192.168.2.15:3260   ONLINE   disk_5
192.168.2.15:3260   ONLINE   disk_6
192.168.2.15:3260   ONLINE   disk_7
```

Modifying tunables for iSCSI

You can set the values of the attributes on the targets. You can set or show the default values, the values for all targets, or the values for a specific target.

Table 7-1 shows the target attributes that you can modify.

Table 7-1 Attributes for iSCSI targets

Attribute	Description
`cmds_max`	The maximum number of SCSI commands that the session will queue. A session is defined as a connection between the initiator and target portal for accessing a given target. cmds_max defines the commands per target, which could be multiple LUNs. Valid values range from 2 to 2048 and should be a power of 2.
`fast_abort`	Defines whether initiator should respond to R2Ts (Request to Transfer) after sending a task management function like an ABORT_TASK or LOGICAL UNIT RESET. A value of Yes causes the initiator to stop responding to R2Ts after an ABORT_TASK request is received. For Equallogic arrays, the recommended value is No. Valid values are Yes or No.
`initial_login_retry_max`	The maximum number of times that the iSCSI initiator should try a login to the target during first login. This only affects the initial login. Valid values range from 1 to 16. During each login attempt, wait for login_timeout seconds for the login to succeed.
`login_timeout`	The amount of time that the iSCSI initiator service should wait for login to complete. The value of this attribute is in seconds. Valid values range from 10 to 600.
`logout_timeout`	The amount of time that the iSCSI initiator service should wait for logout to complete. The value of this attribute is in seconds. Valid values range from 10 to 600.
`noop_interval`	The time to wait between subsequent sending of Nop-out requests. The value of this attribute is in seconds. Valid values range from 5 to 600.
`noop_timeout`	The amount of time that the iSCSI initiator service should wait for response to a Nop-out request sent to the target, before failing the connection. Failing the connection causes the I/O to be failed and retried on any other available path. The value of this attribute is in seconds. Valid values range from 5 to 600.
`queue_depth`	The maximum number of SCSI commands queued per LUN, belonging to a target. The value for queue_depth cannot be greater than cmds_max. Valid values range from 1 to 128.

Table 7-1 Attributes for iSCSI targets *(continued)*

Attribute	Description
replacement_timeout	The amount of time to wait for session re-establishment before failing SCSI commands. The value of this attribute is in seconds. Valid values range from 10 to 86400.

To display the default value for target attributes

- To display the default value for target attributes, enter the following:

```
Storage> iscsi target attr showdefault
```

For example:

```
Storage> iscsi target attr showdefault
Attribute                Value
---------                -----
replacement_timeout      122
noop_timeout             5
noop_interval            13
login_timeout            10
logout_timeout           15
cmds_max                 128
queue_depth              32
initial_login_retry_max  10
fast_abort               No
```

To display values for target attributes of all known targets

- To display values for target attributes of all known targets, enter the following:

  ```
  Storage> iscsi target attr showall
  ```

  For example:

```
Storage> iscsi target attr showall

Attribute                Value  Target
---------                -----  ------
replacement_timeout      123    iqn.1992-08.com.iscsi:sn.84268871
noop_timeout             5      iqn.1992-08.com.iscsi:sn.84268871
noop_interval            121    iqn.1992-08.com.iscsi:sn.84268871
login_timeout            10     iqn.1992-08.com.iscsi:sn.84268871
logout_timeout           15     iqn.1992-08.com.iscsi:sn.84268871
cmds_max                 128    iqn.1992-08.com.iscsi:sn.84268871
queue_depth              32     iqn.1992-08.com.iscsi:sn.84268871
initial_login_retry_max  5      iqn.1992-08.com.iscsi:sn.84268871
fast_abort               No     iqn.1992-08.com.iscsi:sn.84268871
replacement_timeout      124    iqn.2009-01.com.example:storage.disk0.lun0
noop_timeout             5      iqn.2009-01.com.example:storage.disk0.lun0
noop_interval            121    iqn.2009-01.com.example:storage.disk0.lun0
login_timeout            10     iqn.2009-01.com.example:storage.disk0.lun0
logout_timeout           15     iqn.2009-01.com.example:storage.disk0.lun0
cmds_max                 128    iqn.2009-01.com.example:storage.disk0.lun0
queue_depth              32     iqn.2009-01.com.example:storage.disk0.lun0
initial_login_retry_max  10     iqn.2009-01.com.example:storage.disk0.lun0
fast_abort               No     iqn.2009-01.com.example:storage.disk0.lun0
```

To display the attribute values for a specific target

- To display the attribute values for a specific target, enter the following:

```
Storage> iscsi target attr show target-name
```

where *target-name* is the name of the iSCSI target to be displayed.

For example:

```
Storage> iscsi target attr show iqn.1992-08.com.iscsi:sn.84268871
Attribute               Value
---------               -----
replacement_timeout     123
noop_timeout            5
noop_interval           121
login_timeout           10
logout_timeout          15
cmds_max                128
queue_depth             32
initial_login_retry_max 5
fast_abort              No
```

To set the default value for a target attribute

- To set the default value for a target attribute, enter the following:

```
Storage> iscsi target attr setdefault attribute value
```

attribute	The attribute for which to set the value.
value	The default value to be set for the attribute.

The default value is inherited by any new targets that get added.

For example:

```
Storage> iscsi target attr setdefault login_timeout 10
Sucesss.
```

To set an attribute value for all known targets

- To set an attribute value for all known targets, enter the following:

```
Storage> iscsi target attr setall attribute value
```

attribute	The attribute for which to set the value.
value	The value to be set for the attribute.

This command does not change the default value as shown in the `Storage> iscsi target attr showdefault` command. Changes to values are effective after re-login.

For example:

```
Storage> iscsi target attr setall logout_timeout 20
Changes would be applicable after next login into the target.
Sucesss.
```

To set the attribute value for a specific target

- To set the attribute value for a specific target, enter the following:

```
Storage> iscsi target attr set target-name attribute value
```

target-name	The name of the specific iSCSI target.
attribute	The attribute of the specific target.
value	The value to be set for the target attribute.

For example:

```
Storage> iscsi target attr set iqn.1992-08.com.iscsi:sn.84268871 noop_interval 30
Changes would be applicable after next login into the target.
Sucesss.
```

Section 4

Managing Veritas Access file access services

Chapter 8

Configuring your NFS server

This chapter includes the following topics:

About using NFS server with Veritas Access

Veritas Access provides file access services to UNIX and Linux client computers using the Network File System (NFS) protocol. Veritas Access file systems can be exported over NFS v3 or NFS v4. Veritas Access provides the following NFS server support:

- Kernel-based NFS server
  See "Using the kernel-based NFS server" on page 60.
- NFS-Ganesha server
  See "Using the NFS-Ganesha server" on page 60.

At any time, either NFS-Ganesha or kernel NFS is active. The kernel NFS server is enabled by default. If required, you can switch the NFS server that you use.

Use the NFS-Ganesha server if you want NFS v4 support, or if you are using a scale-out file system.

Using the kernel-based NFS server

The kernel-based NFS server supports NFS version 3. The kernel NFS server is enabled by default. Kernel NFS supports Active-Active mode serving NFS version 3. Veritas recommends that you use the default kernel-based NFS server unless you require NFS version 4 support.

Using the NFS-Ganesha server

If you plan to use NFS version 4, you must use Veritas Access with an NFS-Ganesha server.

NFS-Ganesha provides support for both NFS version 3 and NFS version 4. NFS-Ganesha is a user-space implementation of the NFS server. The use of a NFS-Ganesha server is optional. NFS-Ganesha is not enabled by default.

For scale-out file systems with `largefs` layout, an NFS-Ganesha share is always exported from only one node in the cluster. This node can be any one of the nodes in the cluster. At the time of share export, the virtual IP address that is used for accessing the share is displayed. Different shares can be exported from different nodes. The shares are highly available in case of a node failure.

Certain limitations apply for NFS-Ganesha.

Since the kernel-based NFS server is the default, switch the NFS server to NFS-Ganesha.

Switching between NFS servers

If NFS v4 is your primary use case, we recommend that you use the NFS-Ganesha server. You should also use the NFS-Ganesha server if you require Kerberos authentication. The NFS-Ganesha server supports both NFS v3 and NFS v4, and Kerberos authentication is supported for both NFS v3 and v4.

If NFS v3 is your primary use case, then we recommend that you use the kernel NFS server.

A CLISH command is provided to switch from kernel NFS server to NFS-Ganesha, or vice versa. Before you switch between the NFS servers, the NFS server must be offline.

All of the available NFS shares are moved from the previous NFS server to the new NFS server; therefore, the operation may be time consuming.

To switch between NFS servers

1. Make sure that the NFS server is offline. You can view the status of the NFS server with the following command:

```
NFS> server status
```

2. Use the following command to switch the NFS server:

```
NFS> server switch
```

Recommended tuning for NFS-Ganesha version 3 and version 4

Veritas Access supports both the NFS kernel-based server and the NFS-Ganesha server in a mutually exclusive way. The NFS kernel-based server supports NFS version 3 only. The NFS-Ganesha server supports both NFS version 3 and NFS version 4.

The NFS-Ganesha server does not run in the kernel, instead NFS-Ganesha runs in user space on the NFS server. This means that the NFS-Ganesha server processes can be affected by system resource limitations as any other user space process can be affected. There are some NFS-server operating system tuning values that you should modify to ensure that the NFS-Ganesha server performance is not unduly affected. You use the NFS client mount option `version` to determine whether NFS version 3 or NFS version 4 is used. On the NFS client, you can select either the `version=3` or the `version=4` mount option. The NFS client is unaware of whether the NFS server is using kernel-based NFS or NFS-Ganesha. Only if NFS-Ganesha is enabled in Veritas Access, a client can perform an NFS mount using the mount option of `version=4`.

When you start a system, `kswapd_init()` calls a kernel thread that is called `kswapd`, which continuously executes the function `kswapd()` in `mm/vmscan.c` that usually sleeps. The `kswapd` daemon is responsible for reclaiming pages when memory is running low. `kswapd` performs most of the tasks that are needed to maintain the page cache correctly, shrink slab caches, and swap out processes if necessary. `kswapd` keeps freeing pages until the `pages_high` watermark is reached. Under extreme memory pressure, processes do the work of `kswapd` synchronously by calling `balance_classzone()`, which calls the `try_to_free_pages_zone()`.

When there is memory pressure, pages are claimed using two different methods.

- `pgscank/s` – The `kswapd` kernel daemon periodically wakes up and claims (frees) memory in the background when free memory is low. `pgscank/s` records this activity.
- `pgscand/s` – When `kswapd` fails to free up enough memory, then the memory is also claimed directly in the process context (thus blocking the user program execution). `pgscand/s` records this activity.
- The total pages being claimed (also known as page stealing) is therefore a combination of both `pgscank/s` and `pgscand/s`. `pgsteal/s` records the total activity, so (`pgsteal/s = pgscank/s + pgscand/s`).

The NFS-Ganesha user process can be affected when `kswapd` fails to free up enough memory. To alleviate the possibility of the NFS-Ganesha process from doing the work of `kswapd`, Veritas recommends increasing the value of the Linux virtual machine tunable `min_free_kbytes`.

Example of a default auto-tuned value:

```
sysctl -a | grep vm.min_free
vm.min_free_kbytes = 90112
```

You use `min_free_kbytes` to force the Linux VM (virtual memory management) to keep a minimum number of kilobytes free. The VM uses this number to compute a watermark value for each `lowmem` zone in the system.

Table 8-1 Recommended tuning parameters for NFS version 3 and version 4

Option	Description
NFS mount options	File system mount options for the NFS client: ■ `version=3/4` ■ `nordirplus` ■ `sharecache`
NFS server export options	NFS server export options: ■ `rw` ■ `sync` ■ `no_root_squash`

Table 8-1 Recommended tuning parameters for NFS version 3 and version 4 *(continued)*

Option	Description
Jumbo frames	A jumbo frame is an Ethernet frame with a payload greater than the standard maximum transmission unit (MTU) of 1,500 bytes. Enabling jumbo frames improves network performance in I/O intensive workloads. If jumbo frames are supported by your network, and if you wish to use jumbo frames, Veritas recommends using a jumbo frame size of 5000.
`min_free_kbytes`	On server nodes with 96 GB RAM or more, the recommended value of `min_free_kbytes` is 1048576 (=1 GB). On server nodes using the minimum of 32 GB RAM, the minimum recommended value of `min_free_kbytes` is 524288 (=512 MB).

About authenticating NFS clients

See “About managing NFS shares using netgroups” on page 135.

For the NFS-Ganesha server, you can also use Kerberos authentication.

Kerberos authentication is not supported with the kernel NFS server.

Configuration of Kerberos authentication is not supported in Veritas Access.

See the `nfs_server(1)` manual page for information on displaying NFS statistics.

Chapter 9

Using Veritas Access as a CIFS server

This chapter includes the following topics:

About configuring Veritas Access for CIFS

The Common Internet File System (CIFS), also known as the Server Message Block (SMB), is a network file sharing protocol that is widely used on Microsoft and other operating systems. Veritas Access supports the SMB3 protocol.

You can specify either an IPv4 address or an IPv6 address.

Veritas Access supports the following clustering modes:

- Normal
- Clustered Trivial Database (CTDB) - a cluster implementation of the TDB (Trivial database) based on the Berkeley database API

Veritas Access supports the following CIFS security modes:

- User
- ADS

Each clustering mode supports both of the CIFS security modes. The ctdb clustering mode is a different clustered implementation of Veritas Access CIFS, which supports almost all of the features supported by normal clustering mode as well as some additional features.

Additional features supported in ctdb clustering mode:

- Directory-level share support and also supported in normal clustering mode
- Multi-instance share export of a file system/directory
- Simultaneous access of a share from multiple nodes and therefore better load balancing

Veritas Access can be integrated into a network that consists of machines running Microsoft Windows. You can control and manage the network resources by using Active Directory (AD) domain controllers.

Before you use Veritas Access with CIFS, you must have administrator-level knowledge of the Microsoft operating systems, Microsoft services, and Microsoft protocols (including AD and NT services and protocols).

You can find more information about them at: www.microsoft.com.

When serving the CIFS clients, Veritas Access can be configured to operate in one of the operating mode environments described in Table 9-1.

Table 9-1 CIFS operating mode environments

Mode	Definition
Standalone	Information about the user and group accounts is stored locally on Veritas Access. Veritas Access also authenticates users locally using the Linux password and group files. This mode of operation is provided for Veritas Access testing and may be appropriate in other cases, for example, when Veritas Access is used in a small network and is not a member of a Windows security domain. In this mode of operation, you must create the local users and groups; they can access the shared resources subject to authorization control.
Active Directory (AD)	Veritas Access becomes a member of an AD security domain and is configured to use the services of the AD domain controller, such as DNS, LDAP, and NTP. Kerberos, NTLMv2, or NTLM authenticate users.

When Veritas Access operates in the AD domain mode, it acts as a domain member server and not as the domain controller.

About configuring CIFS for standalone mode

If you do not have an AD server, you can use Veritas Access as a standalone server. Veritas Access is used in standalone mode when testing Veritas Access functionality and when it is not a member of a domain.

Before you configure the CIFS service for the standalone mode, do the following:

- Make sure that the CIFS server is not running.
- Set security to user.
- Start the CIFS server.

To make sure that the configuration has changed, do the following:

- Check the server status.
- Display the server settings.

Configuring CIFS server status for standalone mode

To check the CIFS server status

1. To check the status of the CIFS server, enter the following:

   ```
   CIFS> server status
   ```

   By default, `security` is set to `user`, the required setting for standalone mode. The following example shows that `security` was previously set to `ads`.

   For example:

   ```
   CIFS> server status
   CIFS Status on test_01 : ONLINE
   CIFS Status on test_02 : ONLINE

   Homedirfs                 : fs1
   Security                  : ads
   Domain membership status  : Disabled
   Domain                    : VERITASDOMAIN.COM
   Domain Controller         : VRTSSERVER
   Domain User               : administrator
   Clustering Mode           : normal
   ```

2. If the server is running, enter the following:

   ```
   CIFS> server stop
   Stopping CIFS Server.....Success.
   ```

To check the security setting

1 To check the current settings before setting security, enter the following:

```
CIFS> show
```

For example:

```
Name                  Value
----                  -----
netbios name          mycluster
ntlm auth             yes
allow trusted domains no
homedirfs
aio size              1024
idmap backend         rid:10000-1000000
workgroup             VERITASDOMAIN
security              ads
Domain                VERITASDOMAIN.COM
Domain user           administrator
Domain Controller     VRTSSERVER
Clustering Mode       normal
```

2 To set security to `user`, enter the following:

```
CIFS> set security user
Global option updated. Note: Restart the CIFS server.
```

To start the CIFS service in standalone mode

1 To start the service in standalone mode, enter the following:

```
CIFS> server start
Starting CIFS Server.....Success.
```

2 To display the new settings, enter the following:

```
CIFS> show
```

For example:

```
Name                    Value
----                    -----
netbios name            mycluster
ntlm auth               yes
allow trusted domains   no
homedirfs
aio size                1024
idmap backend           rid:10000-1000000
workgroup               VERITASDOMAIN
security                user
Domain                  VERITASDOMAIN.COM
Domain user             administrator
Domain Controller       VRTSSERVER
Clustering Mode         normal
```

3 To make sure that the server is running in standalone mode, enter the following:

```
CIFS> server status
```

For example:

```
CIFS> server status
CIFS Status on test_01 : ONLINE
CIFS Status on test_02 : ONLINE

Homedirfs                 : fs1
Security                  : user
Clustering Mode           : normal
```

The CIFS service is now running in standalone mode.

About Active Directory (AD)

In order to provide CIFS services, Veritas Access must be able to authenticate within the Windows environment.

Active Directory (AD) is a technology created by Microsoft that provides a variety of network services including LDAP directory services, Kerberos-based authentication, Domain Name System (DNS) naming, secure access to resources, and more.

Veritas Access will not join the AD domain if its clock is excessively out-of-sync with the clock on the AD domain controller. Ensure that Network Time Protocol (NTP) is configured on Veritas Access, preferably on the same NTP server as the AD domain controller.

Configuring CIFS for the AD domain mode

To set the domain user for AD domain mode

1. To verify that the CIFS server is stopped, enter the following:

   ```
   CIFS> server status
   ```

2. If the server is running, stop the server. Enter the following:

   ```
   CIFS> server stop
   ```

3. To set the domain user, enter the following:

   ```
   CIFS> set domainuser username
   ```

   where *username* is the name of an existing AD domain user who has permission to perform the join domain operation.

   For example:

   ```
   CIFS> set domainuser administrator
   Global option updated. Note: Restart the CIFS server.
   ```

To set the domain for AD domain mode

- To set the domain for AD domain mode, enter the following:

```
CIFS> set domain domainname
```

where *domainname* is the name of the domain.

For example:

```
CIFS> set domain VERITASDOMAIN.COM
Global option updated. Note: Restart the CIFS server.
```

To set the domain controller for AD domain mode

- To set the domain controller, enter the following:

```
CIFS> set domaincontroller servername
```

where *servername* is the server's IP address or DNS name.

For example, if the server SYMSERVER has an IP address of 172.16.113.118, you can specify one of the following:

```
CIFS> set domaincontroller 172.16.113.118
Global option updated. Note: Restart the CIFS server.
```

or

```
CIFS> set domaincontroller SYMSERVER
Global option updated. Note: Restart the CIFS server.
```

To set security to ads

- To set security to ads, enter the following:

```
CIFS> set security ads|user
```

Enter `ads` for *security*.

```
CIFS> set security ads
Global option updated. Note: Restart the CIFS server.
```

To set the workgroup

- To set the workgroup name if the WORKGROUP or NetBIOS domain name is different from the domain name, enter the following:

  ```
  CIFS> set workgroup workgroup
  ```

  where *workgroup* sets the WORKGROUP name. If the name of the WORKGROUP or NetBIOS domain name is different from the domain name, use this command to set the WORKGROUP name.

  For example, if SIMPLE is the name of the WORKGROUP you want to set, you would enter the following:

  ```
  CIFS> set workgroup SIMPLE
  ```

  Though the following symbols $,( ), ', and & are valid characters for naming a WORKGROUP, the Veritas Access CIFS implementation does not allow using these symbols.

To start the CIFS server

1 To start the CIFS server, enter the following:

```
CIFS> server start

The skew of the system clock with respect to
Domain controller is:  -17 seconds

Time on Domain controller : Thu Dec  4 05:21:47 2008
Time on this system : Thu Dec  4 05:22:04 PST 2008

If the above clock skew is greater than that allowed by the server,
then the system won't be able to join the AD domain

Trying to become a member in AD domain VERITASDOMAIN.COM ...

Enter password for user 'administrator':
```

After you enter the correct password for the user administrator belonging to AD domain VERITASDOMAIN.COM, the following message appears:

```
Joined domain VERITASDOMAIN.COM OK
Starting CIFS Server.....Success.
```

2 To make sure that the service is running, enter the following:

```
CIFS> server status
CIFS Status on test_01 : ONLINE
CIFS Status on test_02 : ONLINE

Homedirfs                 : fs1
Security                  : ads
Domain membership status  : Enabled
Domain                    : VERITASDOMAIN.COM
Domain Controller         : VRTSSERVER
Domain User               : administrator
Clustering Mode           : normal
```

The CIFS server is now running in the AD domain mode. You can export the shares, and the domain users can access the shares subject to the AD authentication and authorization control.

Chapter 10

Configuring Veritas Access to work with Oracle Direct NFS

This chapter includes the following topics:

About using Veritas Access with Oracle Direct NFS

Veritas Access lets you create and manage storage for Oracle database clients. Oracle hosts access the storage using Oracle Direct NFS (DNFS).

Oracle Direct NFS is an optimized NFS (Network File System) client that provides faster access to NFS storage that is located on NAS storage devices. The Oracle Database Direct NFS client integrates the NFS client functionality directly in the Oracle software. Through this integration, the I/O path between Oracle and the NFS server is optimized, providing significantly better performance. In addition, the Oracle Direct NFS client simplifies and, in many cases, automates the performance optimization of the NFS client configuration for database workloads.

The Oracle Direct NFS client outperforms traditional NFS clients, and is easy to configure. The Oracle Direct NFS client provides a standard NFS client implementation across all hardware and operating system platforms.

Veritas Access creates different storage pools for different Oracle object types. Veritas Access has the following storage pools as described in Table 10-1.

Table 10-1 Veritas Access storage pools for Oracle object types

Pool Name	Database Object Type	Function
`ora_data_pool`	Oracle TABLE data files	Stores TABLE data of data files. This database object type uses striped volumes over four LUNs. The stripe size is 256K.
`ora_index_pool`	Oracle INDEX files	Stores INDEX data.
`ora_temp_pool`	Temporary files	Stores temporary files. The storage administrator should make sure the LUNs are from the fastest tier. Temporary files are used for sort, merge, or join queries.
`ora_archive_pool`	Archive logs	Stores archive logs. This database object type is a concatenated volume. Tier 2 LUNs can be used for this storage pool.
`ora_txnlog_pool`	REDO txnlog files	Stores REDO transaction logs. It is recommended to assign the fastest storage LUNs to this storage pool.

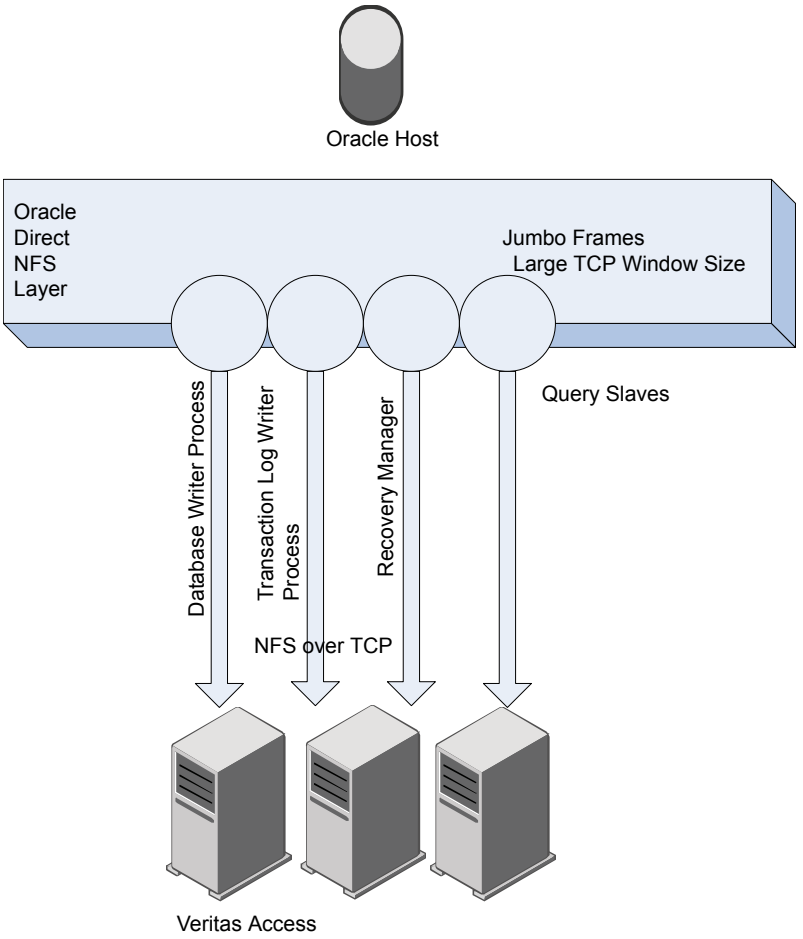
About the Oracle Direct NFS architecture

Oracle Direct NFS can issue 1000s of concurrent operations due to the parallel architecture. Every Oracle process has its own TCP connection.

See the *Veritas Access Installation Guide* for the supported Oracle operating systems.

Figure 10-1 describes the data flow from the Oracle host to Veritas Access.

Figure 10-1 Oracle Direct NFS architecture



Setting a larger frame size on an interface is commonly referred to as using jumbo frames. Jumbo frames help reduce fragmentation as data is sent over the network and in some cases, can also provide better throughput and reduced CPU usage.

You can configure jumbo frames for Oracle Direct NFS by setting the Maximum Transmission Unit (MTU) value.

Best practices for improving Oracle database performance

Oracle database performance depends on I/O bandwidth and latency.

The Oracle database has the following object types:

- DATA
- INDEX
- TXNLOG
- ARCHLOG
- TEMPFILES

See "About using Veritas Access with Oracle Direct NFS" on page 74.

The Oracle database transaction rate depends heavily on TXNLOG write latency.

Table 10-2 Best practices for improving Oracle database performance

Item	Action
DATA and INDEX files	Separate DATA and INDEX files into separate disk pools with a minimum of four LUNs.
REDOLOG files	Configure REDOLOG files in a separate file system (NFS share). The underlying storage LUNs should not be shared with other volumes or file systems.
TEMPFILES	For data warehouse applications, keep TEMPFILES in separate pools with a minimum of four LUNs.
TXNLOG files	Place TXNLOG files in a separate disk pool with dedicated fast LUNs.
NFS daemon threads	To get better performance, increase NFS daemon threads to 128 or more. If the network permits, you may want to enable jumbo frames. See "About the Oracle Direct NFS architecture" on page 75.
Veritas Access database-specific file systems	Access Veritas Access database-specific file systems from the Oracle host using dedicated virtual IPs. Do not use the same virtual IP for other applications.

Table 10-2 Best practices for improving Oracle database performance *(continued)*

Item	Action
Veritas Access file systems	Always use Oracle recommended mount options to mount Veritas Access file systems on the Oracle host. This mount option depends on the Oracle database version. See the *Veritas Access Installation Guide* for the supported Oracle database versions. See the Oracle documentation for the recommended mount options.

About Oracle Direct NFS node or storage connection failures

When a node or a storage connection fails, Veritas Access fails over the virtual IP (VIP) to the healthy node. If Oracle does active transactions, some I/O can fail during this VIP failover time window. Oracle generally issues several asynchronous I/O requests using Oracle Direct NFS in parallel. If Oracle detects some I/O failure waiting for completed I/Os, the result depends on the database file type. If the write fails for the REDO transaction log or the database control files, then the database instance fails. If I/O to a data file fails, then that particular data file is taken offline. The SYSTEM and the UNDO data files are considered critical data files. The Oracle database fails if I/O to these critical data files fail. When the database instance fails, the database administrator should wait until the VIP fails over to the healthy Veritas Access node. When the VIP is online, the database administrator can start up the database and then recover the database.

Chapter 11

Configuring your FTP server

This chapter includes the following topics:

About FTP

The file transfer protocol (FTP) server feature allows clients to access files on the Veritas Access servers using the FTP protocol. The FTP service provides secure/non-secure access by FTP to files in the Veritas Access servers. The FTP service runs on all of the nodes in the cluster and provides simultaneous read and write access to the files. The FTP service also provides configurable for anonymous access to Veritas Access.

By default, the FTP server is not running. You can start the FTP server using the `FTP> server start` command. The FTP server starts on the standard FTP port 21.

The Veritas Access FTP service does not support transparent failover. During failover due to either a shutdown or a restart of the server, the FTP client loses its connection to the server. As a consequence, any upload or download to the FTP service during the failover fails. Restart any upload or download to the FTP service from the beginning after the connection to the FTP service has been re-established.

Section 5

Managing Veritas Access block access services

Chapter 12

Configuring block access

This chapter includes the following topics:

About iSCSI targets for block storage

Veritas Access as an iSCSI target is added as a preview feature in the 7.2.1 release. Veritas Access can be configured as an iSCSI target to serve block storage. The iSCSI target service is hosted in active-passive mode in the Veritas Access cluster. Once configured, the cluster is available to any standard iSCSI initiator over a portal IP.

You can perform the following functions on an iSCSI target:

- Starting and stopping of the iSCSI target service
- Addition and deletion of targets
- Addition and deletion of LUNs
- Map and un-map initiators
- Addition and deletion of users

See the `target` manual pages for more information.

Section 6

Managing the Veritas Access object server

Chapter 13

Using Veritas Access as an object server

This chapter includes the following topics:

About the object server

The object server lets you store and retrieve the data that is stored in Veritas Access using the Amazon Simple Storage Service (Amazon S3) protocol.

A bucket maps to a directory inside Veritas File System and an object maps to a file. The object server in Veritas Access provides a RESTful interface. Buckets and objects are created, fetched, and deleted using standard HTTP requests.

See the object access (S3) user management APIs section in the *Veritas Access RESTful API Guide* for more information.

Features of the object access service:

- High availability
- Multipart uploading
- Concurrent access from multiple nodes
- Canned Access Control Lists

Configuring the object server

To configure the object server

1. Log on to Veritas Access using the CLISH.
2. Create a default storage pool (at least one) on the cluster.

   ```
   CLISH> storage pool create pool1 disk1,disk2,disk3,disk4
   ```

3. Use the storage pool that was created in Step 2 as the default object access pool.

   You need to set the default pool, as it is required for enabling the object access server.

   ```
   CLISH> objectaccess set pools pool1
   ```

   Note: Based on your storage requirements, you can configure different types of storage pools by using the object access group commands.

4. Verify the configured storage pool.

   ```
   CLISH> objectaccess show
   ```

5. Enable and start the object server.

   ```
   CLISH> objectaccess server enable
   ```

   ```
   CLISH> objectaccess server status
   ```

6. Configure the cluster using any authentication server (AD, LDAP, or NIS).

   See the following manual pages for more information on configuring AD, LDAP, or NIS:

   - `CLISH> network man ldap`
   - `CLISH> man cifs`
   - `CLISH> man nis`

7. Create the access and secret keys for the authorized user, any user in the authentication server.

   You have two options for creating the access and the secret keys, either using the Veritas Access RESTful APIs or by using the Veritas Access helper script.

   Create the access and secret keys using the Veritas Access RESTful APIs:

- See the *Veritas Access RESTful API Guide* on the SORT site for accessing the object access server (S3) user management APIs.
- Before using the Veritas Access RESTful APIs, set the host name resolution for the host as shown in the `objectaccess> show` output against ADMIN_URL.
- After creating your access and secret key, you can create a bucket using the AWS S3 API.

Create the access and the secret keys using the Veritas Access helper script:

- Location of the helper script:
  `/opt/VRTSnas/scripts/utils/objectaccess/objectaccess_client.py`
- The Veritas Access helper script can be used from any client system that has Python installed.
- To run the script, your S3 client needs to have the `argparse` and `requests` Python modules.
  If these modules are missing, install both these modules using `pip` or `easy_install`.
- Add the ADMIN_URL name in your `/etc/hosts` file.
- Create the access and the secret key using the Veritas Access helper script by providing the user name, password, and ADMIN_URL (check the online Help of the Veritas Access helper script for all of the provided operations like `list key` and `delete key`).
  Create a secret key:

```
clus_01:~ # ./objectaccess_client.py --create_key
--server admin.clus:8144 --username localuser1 --password root123
--insecure
UserName                : localuser1
AccessKeyId             : Y2FkODU2NTU2MjVhYzV
Status                  : Active
SecretAccessKey         : ODk0YzQxMDhkMmRjM2M5OTUzNjI5OWIzMDgyNzY
```

  List a secret key for the specified user:

```
clus_01:~ # ./objectaccess_client.py --list_key --server
admin.clus:8144 --username localuser2 --password root123 --insecure
```

  Delete a secret key for the specified user:

```
clus_01:~ # ./objectaccess_client.py --delete_key
ZTkyNDdjZTViM2EyMWZ --server admin.clus:8144 --username localuser2
--password root123 --insecure
```

- If the object server is enabled without the `SSL` option, you need to add the `--insecure` option.

  ```
  clus_01 ~# ./objectaccess_client.py --server
  admin.clus:8144 --username <uname> --create_key --insecure
  ```

8 Use the following `objectaccess` command to see all the existing access and secret keys in the Veritas Access cluster:

```
CLISH> objectaccess account user show
```

Setting other object server options and configuring groups

1. If required, you can change the other object server defaults, such as `fs_type`, `fs_size`, and other options.

   After setting the defaults, you can verify whether the proper value is assigned or not.

```
vmdellr> objectaccess set fs_type
mirrored  mirrored-stripe  simple  striped striped-mirror

vmdellr> objectaccess set fs_type simple
ACCESS ObjectAccess INFO V-288-0 Set fs_type successful.

vmdellr> objectaccess set fs_size 2G
ACCESS ObjectAccess INFO V-288-0 Set operation successful.

vmdellr> objectaccess show
Name            Value
=============   ========================

Server Status   Enabled
Admin_URL       http://admin.vmdellr:8144
S3_URL          http://s3.vmdellr:8143
admin_port      8144
s3_port         8143
ssl             no
poollist        ['pool1']
fs_size         2G
fs_blksize      8192
fs_pdirenable   no
fs_encrypt      off
fs_type         simple
```

2. If you have multiple users, and you want to set different default values for different sets of users, you can use the `group` option.

3. Create a group.

```
[root@vmdellr_01 ~]# groupadd -g 512 VRTS-grp
```

4 Add users to the group (If AD, LDAP, or NIS users are not available, then add only local users).

```
[root@vmdellr_01 ~]# useradd -g 512 user_3
[root@vmdellr_01 ~]# useradd -g 512 user_4
[root@vmdellr_01 ~]#
[root@vmdellr_01 ~]# id user_3
uid=512(user_3) gid=512(VRTS-grp) groups=512(VRTS-grp)

[root@vmdellr_01 ~]# id user_4
uid=513(user_4) gid=512(VRTS-grp) groups=512(VRTS-grp)
```

5 Set or unset the defaults per your requirements.

```
vmdellr> objectaccess group set fs_type simple VRTS-grp
ACCESS ObjectAccess INFO V-288-0 Group set fs-type successful.

vmdellr> objectaccess group set pool VRTS-grp pool1
ACCESS ObjectAccess INFO V-288-0 Success.

vmdellr> objectaccess group show
Group Name   Fs Sharing   Fs Size   Fs Type   Pool(s)
===========  ===========  ========  ========  ========
VRTS-grp     -            -         simple    pool1

vmdellr> objectaccess group show
Group Name   Fs Sharing   Fs Size   Fs Type   Pool(s)
===========  ===========  ========  ========  ========
VRTS-grp     -            -         -         pool1
```

About buckets and objects

Amazon S3 provides cloud storage for the Internet. To upload your unstructured data (photos, videos, and documents), you first need to create a bucket in one of the AWS regions. You can then upload any number of objects to the bucket. You can then upload any number of objects to the bucket.

Buckets and objects are resources, and Amazon S3 provides APIs for managing the buckets and the objects.

Veritas Access supports the following methods for accessing the buckets and the objects:

- Path-style method
- Virtual-hosted-style method

When using the virtual hosted-style method, the ***bucket_name*.s3.*cluster_name*** should be DNS resolvable.

See the `objectaccess_bucket(1)` manual page for more information.

See the `objectaccess` manual pages for all of the Veritas Access object server operations that can be performed.

Operations on buckets

Buckets are created by AWS S3 clients by calling the standard AWS S3 APIs to the Veritas Access S3 server. For creating a bucket, you need the endpoint of the Veritas Access server, access key, and the secret key. The endpoint of the Veritas Access object server is `s3.cluster_name:8143`.

The Veritas Access object server can also be accessed using the fully qualified domain name:

`s3.cluster_name.fqdn:8143`

Make sure that you associate one (or more) of the VIPs of the Veritas Access cluster to `s3.cluster_name.fadn` in the client's DNS server.

The following S3 bucket operations are supported:

- Setting and removing S3 storage pools.
- Starting and stopping the object server.

Operations on objects

Amazon S3 enables you to store, retrieve, and delete objects. You can retrieve an entire object or a portion of an object.

Section 7

Configuring cloud storage

Chapter 14

Configuring the cloud gateway

This chapter includes the following topics:

About the cloud gateway

You can configure Veritas Access as a gateway to cloud storage. You can register Amazon AWS S3 subscriptions to your Veritas Access cluster. Multiple cloud subscriptions can be attached, so you need to assign a service name to each subscription. You can then use the service name to attach the S3 subscription to a scale-out file system as a storage tier.

The cloud as a tier feature lets you have hybrid storage that uses both on-premises storage and public cloud storage. After the gateway and tier are configured, you can use the cloud as a tier feature to move data between the cloud and the on-premises storage. The files in the Amazon S3 cloud, like the files on the on-premises storage, are accessible using the NFS protocol. Access of the data present in the cloud tier is transparent to the application.

See “Configuring the cloud as a tier feature for scale-out file systems” on page 92.

Before you provision cloud storage, you set up Amazon AWS S3 subscriptions. To set up the cloud gateway, you attach Amazon AWS S3 subscriptions to your Veritas Access cluster. You need to have the subscription credentials to add the AWS S3 subscriptions. You need the Amazon AWS S3 access and secret keys.

Chapter 15

Configuring cloud as a tier

This chapter includes the following topics:

Configuring the cloud as a tier feature for scale-out file systems

You can move data to the Amazon S3 cloud using a tiering mechanism if you use a scale-out file system. A cloud container is used as a storage tier in a scale-out file system. The terms cloud container and cloud tier are used interchangeably.

Warning: When an Amazon S3 account is used as a cloud tier for a file system, Veritas Access exclusively owns all the buckets and the objects created by Veritas Access. Any attempt to tamper with these buckets or objects outside of Veritas Access corrupts the files represented by those modified objects.

See "Moving files between tiers in a scale-out file system" on page 93.

See the `storage_cloud(1)` man page for detailed examples.

See the `storage_tier(1)` man page for detailed examples.

Warning: Veritas Access cannot add a cloud tier if the clock on the Veritas Access system is more than 15 minutes out-of-sync with the actual time. To ensure that the Veritas Access clock time is accurate, configure an NTP server or use the `System> clock set` command.

To configure the cloud as a tier for scale-out file systems

1 Display the available cloud services.

```
Storage> cloud listservice service_name
```

If the cloud service is not listed, you may need to add the cloud subscription to the cluster.

See "About the cloud gateway" on page 91.

2 Add the cloud as a tier to a scale-out file system.

```
Storage> tier add cloud fs_name tier_name service_name
region S3 | Glacier
```

Amazon AWS has standard regions defined. Based on the region you choose in Veritas Access, AWS storage is served through that region. To get better performance, always select the closest geographic region.

3 Verify that the cloud tier is configured on the specified scale-out file system.

```
Storage> tier list fs_name
```

To remove the cloud tier

- Remove the cloud tier.

```
Storage> tier remove fs_name tier_name
```

If there are any files present in the cloud tier, the remove cloud tier operation fails. Move the files back to the primary tier before removing the cloud tier.

Moving files between tiers in a scale-out file system

By default, a scale-out file system has a single primary tier, which is the on-premises storage for the scale-out file system. You can add a cloud service as an additional tier. After a cloud tier is configured, you can move data between the tiers of the scale-out file system as needed.

Use the commands in this section to move data as a one-time operation. For example, if you have just set up a cloud tier, and you want to move some older data to that tier.

If you want to specify repeatable rules for maintaining data on the tiers, you can set up a policy for the file system.

You can specify the following criteria to indicate which files or directories to move between tiers:

- file or directory name pattern to match
- last accessed time (`atime`)
- last modified time (`mtime`)

Because a scale-out file system can be large, and the size of the files to be moved can be large as well, the `Storage> tier move` command lets you perform a dry run.

See the `storage_tier(1)` man page.

To move data between storage tiers in a scale-out file system

1 (Optional) Perform a dry run to see which files would be moved and some statistics about the move.

```
Storage> tier move dryrun fs_name source_tier destination_tier pattern
[atime condition] [mtime condition]
```

The dry run starts in the background. The command output shows the job ID.

2 Move the files that match *pattern* from *source_tier* to *destination_tier* based on the last accessed time (*atime*) or the last modified time (*mtime*).

```
Storage> tier move start fs_name source_tier destination_tier pattern
 [atime condition] [mtime condition]
```

pattern is required. To include all the files, specify `*` for *pattern*.

The *condition* for `atime` or `mtime` includes an operator, a value, and a unit. Possible operators are the following: `<, <=, >, >=`. Possible units are m, h, and d, indicating minutes, hours, and days.

The name of the default tier is `primary`. The name of the cloud tier is specified when you add the tier to the file system.

The move job starts in the background. The command output shows the job ID.

Examples:

Move the files that match *pattern* and that have not been accessed within the past 100 days to the cloud tier.

```
Storage> tier move start lfs1 primary cloudtier1 pattern
atime >100d
```

Move the files that match *pattern* and that have been accessed recently in the past 30 days to the specified tier.

```
Storage> tier move start lfs1 cloud_tier primary pattern atime <=30d
```

Move the files that match *pattern* and that have not been modified within the past 100 days to the cloud tier.

```
Storage> tier move start lfs1 primary cloud_tier pattern
mtime >=100d
```

Move only the files that match *pattern* and that were modified within the last three days from the cloud tier to the primary tier.

```
Storage> tier move start lfs2 cloud_tier primary pattern mtime >=3d
```

Move all files to the primary tier.

```
Storage> tier move start lfs2 cloud_tier primary *
```

3 View the move jobs that are in progress in the background. This command lists the job IDs and the status of the job.

```
Storage> tier move list
Job         Fs name   Source Tier  Destination Tier  Pattern        Atime   Mtime  State
========== ========  ============ ================= ===========    ======  ====== ========
1473684478 largefs1  cloudtier    primary           /vx/largefs1/* >120s   -      not running
1473684602 largefs1  cloudtier    primary           /vx/largefs1/* -       -      scanning
```

4 View the detailed status of the data movement for the specified job ID.

```
Storage> tier move status jobid
```

For example:

```
Storage> tier move status 1473682883
Job run type:         normal
Job Status:           running
Total Data (Files):   4.0 G (100)
Moved Data (Files):   100.0 M (10)
Last file visited:    /vx/fstfs/10.txt
```

5 If required, you can abort a move job.

```
Storage> tier move abort jobid
```

About policies for scale-out file systems

When a scale-out file system includes a cloud tier, you can use policies to control the movement of data between the on-premises storage and the cloud tier. A policy is a set of rules defined for a file system for deleting data or moving data. If you want to specify repeatable rules for maintaining data on the tiers, you can set up a policy for the file system.

Each rule defines the following criteria:

- what action should be taken (move or delete)
- when the data should be moved or deleted based on the access time or modified time of the file
- which data should be moved based on the pattern matching for the files and directories.
  See “About pattern matching for data movement policies” on page 98.

Each file system can have more than one rule, though you should be careful not to create rules that conflict or cause looping.

To move or delete the data, you run the policy. When you run a policy, the rules are applied to the data at the time the policy is run. The policy does not run automatically. You can attach a schedule to a policy to have it run automatically at the specified times.

See “Creating and scheduling a policy for a scale-out file system ” on page 100.

About pattern matching for data movement policies

Within a policy, you can use a pattern to specify that the rule applies to file names or directory names that match the pattern. By using a pattern, you do not need to know the exact file name in advance; the files that match the pattern are selected dynamically.

A pattern uses special characters, which are case sensitive. There are the following types of patterns:

- Directory patterns
  A pattern that ends with a slash (/) is matched only against directories.
- File patterns
  A pattern that does not end with a slash (/) is matched only against files.

The following is a list of supported special characters and their meanings:

* (asterisk)	Matches any character any number of times.
? (question mark)	Matches any single character.
** (two asterisks)	Matches across child directories recursively. The pattern `fs1/*/*.pdf` will match `.pdf` file names present after first sub-directory of `fs1/`. For example, if the following files exist: fs1/dir1/a.pdf fs1/dir2/b.pdf fs1/dir3/dir4/c.pdf then the pattern `fs1/*/*.pdf` will match only `a.pdf` and `b.pdf`. It will not match c.pdf. The pattern `fs1/**/*.pdf` will match `.pdf` files in any directory after `fs1`. For the above file list, it will match all of the files: `a.pdf`, `b.pdf`, and `c.pdf`.

[ ] (square brackets)	Matches either range or set of characters. [0-5] will match any character in range of 0 to 5. [a-g] will match any character in range of a to g. [abxyz] will match any one character out of a,b,x,y,z.
! (exclamation point)	Can be used as the first character in a range to invert the meaning of the match. [!0-5] will match any character which is not in range of 0 to 5.
\ (backslash)	Can be used as an escape character. Use this to match for one of the above pattern matching characters to avoid the special meaning of the character.

About schedules for running policies

A schedule is specified in a format similar to the UNIX `crontab` format. The format uses five fields to specify when the schedule runs:

minute	Enter a numeric value between 0-59, or an asterisk (*), which represents every minute. You can also enter a step value (*/x), or a range of numbers separated by a hyphen.
hour	Enter a numeric value between 0-23, or an asterisk (*), which represents every hour. You can also enter a step value (*/x), or a range of numbers separated by a hyphen.
day_of_the_month	Enter a numeric value between 1-31, or an asterisk (*), which represents every day of the month. You can also enter a step value (*/x), or a range of numbers separated by a hyphen.
month	Enter a numeric value between 1-12, or an asterisk (*), which represents every month. You can also use the names of the month. Enter the first three letters of the month (you must use lower case letters). You can also enter a step value (*/x), or a range.
day_of_the_week	Enter a numeric value between 0-6, where 0 represents Sunday, or an asterisk (*), which represents every day of the week. You can also enter the first three letters of the week (you must use lower case letters). You can also enter a step value (*/x), or a range.

A step value (*/x) specifies that the schedule runs at an interval of x. The interval should be an even multiple of the field's range. For example, you could specify */4 for the hour field to specify every four hours, since 24 is evenly divisible by 4. However, if you specify */15, you may get undesired results, since 24 is not evenly divisible by 15. The schedule would run after 15 hours, then 7 hours.

A range of numbers (two values separated by a hyphen) represents a time period during which you want the schedule to run.

Examples: To run the schedule every two hours every day:

```
0 */2 * * *
```

To run the schedule on 2:00 a.m. every Monday:

```
* 2 * * 1
```

To run the schedule at 11:15 p.m. every Saturday:

```
15 23 * * 6
```

Creating and scheduling a policy for a scale-out file system

By default, a scale-out file system has a single disk tier, which is the on-premises storage for the scale-out file system. You can add a cloud service as an additional tier. After a cloud tier is configured, you can move data between the tiers of the scale-out file system as needed.

Use policies to define a set of data movement rules for the scale-out file system. Each file system can include a policy for deletion and a policy for data movement between tiers.

Be careful when specifying the criteria for moving files. Conflicting policies may cause data to move from one tier to another tier. A best practice is to use policies with a smaller data set first before applying those policies to file systems using a schedule.

A data movement policy can use the following criteria to indicate which files or directories to move between tiers:

- pattern
- atime
- mtime

You can also perform a dry run of a policy.

See the `storage_fs(1)` man page for detailed examples.

To create a policy

1. View the policies that are associated with the file system.

   ```
   Storage> fs policy list fs_name
   ```

2. Create a move policy or a delete policy.

   For a move policy:

   ```
   Storage> fs policy add policy_name fs_name operation=move
   from_tier to_tier pattern {atime|mtime}
   ```

   For a delete policy:

   ```
   Storage> fs policy add policy_name fs_name operation=delete from_tier
   pattern {atime|mtime}
   ```

3. If you want to test the policy, you can perform a dry run:

   ```
   Storage> fs policy dryrun policy_name
   ```

4. To move or delete the files according to the policy, run the policy manually or attach a schedule to the policy.

   To run the policy manually:

   ```
   Storage> fs policy run policy_name
   ```

   To attach a schedule to the policy:

   ```
   Storage> fs policy schedule create fs_name minute hour day_of_the_month
   month day_of_the_week
   ```

5. Check on the status of the currently running policy or dry run of the policy.

   ```
   Storage> fs policy status
   ```

6. If necessary, you can abort a policy.

   ```
   Storage> fs policy abort
   ```

Obtaining statistics on data usage in the cloud tier in scale-out file systems

You can find the following information for data stored in the cloud tier in a scale-out file system:

- Storage utilization in the cloud
- Number of the objects that are stored in the cloud
- Number of the files that are stored in the cloud
- Number of GET, PUT, and DELETE requests

See the `storage_tier(1)` man page for detailed examples.

To display the number of GET, PUT, and DELETE requests

- Show the number of GET (read requests), PUT (update or replacement requests), and DELETE (deletion requests).

  ```
  Storage> tier stats show fs_name
       tier_name
  ```

  These statistics are maintained in memory and so are not persistent across reboots.

  Example:

  ```
  Storage> tier stats show largefs1 cloudtier
  GET       168
  GET bytes 174.5MB
  PUT       918
  PUT bytes 10.3GB
  DELETE    20
  ```

To monitor the usage of data in the cloud tier

- Monitor the usage of data in the cloud tier.

```
Storage> tier stats monitor fs_name
     tier_name [interval]
```

Example:

```
Storage> tier stats monitor largefs1 cloudtier
GET     GET bytes        PUT     PUT bytes       DELETE
6       384.0MB          4       256.0MB         6
0          0             0       0               0
0          0             0       0               0
0          0             2       128.0MB         0
0          0             0       0               0
```

The default interval is five seconds.

To show the total data usage in the cloud tier

- Show the total data usage in the cloud tier.

```
Storage> tier stats usage fs_name
     tier_name
```

Unlike the `Storage> tier stats show` command, these statistics are persistent across reboots.

Example:

```
Storage> tier stats usage largefs1 cloudtier
Storage Utilized   223.1GB
Number of objects  488
Number of files    231
```

This example shows that 223.1 GB is used in the cloud tier. Based on the size of the file, each file is chunked to multiple objects when moved to the cloud, so 231 files were stored as 488 objects in the cloud.

To reset the in-memory statistics of the cloud tier to zero

- Reset the statistics of the specified cloud tier to zero.

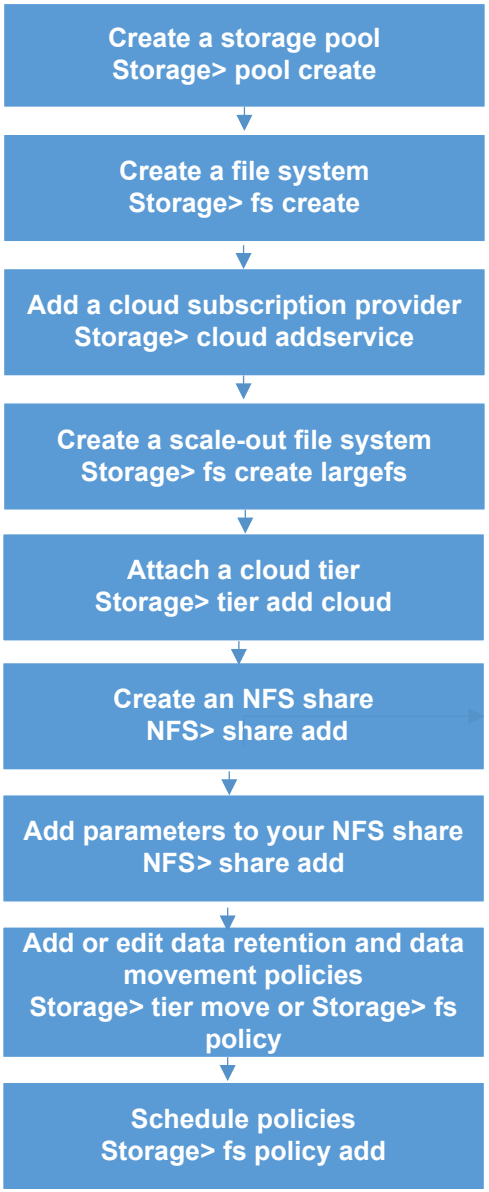
```
Storage> tier stats reset fs_name tier_name
```

After executing the `Storage> tier stats reset` command, the output for the `Storage> tier stats show` command is reset to zero.

Workflow for moving on-premises storage to cloud storage for NFS shares

Figure 15-1 describes the workflow for moving your on-premises storage to cloud storage for NFS shares for a scale-out file system.

Figure 15-1 Workflow for moving on-premises storage to cloud storage for NFS shares for a scale-out file system



Section 8

Monitoring and troubleshooting

Chapter 16

Configuring event notifications and audit logs topics

This chapter includes the following topics:

About configuring event notifications

Veritas Access monitors the status and health of various network and storage components, and generates events to notify the administrator. Veritas Access provides a mechanism to send these events to external event monitoring applications like syslog server, SNMP trap logger, and mail servers. This section explains how to configure Veritas Access so that external event monitoring applications are notified of events on the Veritas Access cluster.

About severity levels and filters

Veritas Access monitors events of different severity levels. Set the severity to a particular level to specify the severity level to include in notifications. Notifications are sent for events having the same or higher severity.

Table 16-1 describes the valid Veritas Access severity levels in descending order of severity.

Table 16-1 Severity levels

Valid value	Description
emerg	Indicates that the system is unusable
alert	Indicates that immediate action is required
crit	Indicates a critical condition
err	Indicates an error condition
warning	Indicates a warning condition
notice	Indicates a normal but a significant condition
info	Indicates an informational message
debug	Indicates a debugging message

Veritas Access also classifies event notifications by type. Set the event filter to specify which type of events to include in notifications. Notifications are sent only for events matching the given filter.

The filter is set to one of the following options:

- Network - for networking events
- Storage - for storage-related events. For example, events related to file systems, snapshots, disks, and pools.
- All - resets the filter to show all events.

For example, if the filter is set to `network`, a network event triggers a notification. A storage-related event would not trigger a notification.

About SNMP notifications

Simple Network Management Protocol (SNMP) is a network protocol to simplify the management of remote network-attached devices such as servers and routers. SNMP is an open standard system management interface. Information from the Management Information Base (MIB) can also be exported.

SNMP traps enable the reporting of a serious condition to a management station. The management station is then responsible for initiating further interactions with the managed node to determine the nature and extent of the problem.

See “About severity levels and filters” on page 107.

Section 9

Provisioning and managing Veritas Access file systems

Chapter 17

Creating and maintaining file systems

This chapter includes the following topics:

About creating and maintaining file systems

A Veritas Access environment consists of multiple nodes that can access and update files in the same Veritas file system at the same time. Many file systems can be supported at the same time. You create file systems on groups of disks called storage pools.

File systems consist of both metadata and file system data. Metadata contains information such as the last modification date, creation time, permissions, and so on. The total amount of the space that is required for the metadata depends on the number of files in the file system. A file system with many small files requires more space to store metadata. A file system with fewer larger files requires less space for handling the metadata.

When you create a file system, you need to set aside some space for handling the metadata. The space that is required is generally proportional to the size of the file system. For this reason, after you create the file system, a small portion of the space appears to be used. The space that is set aside to handle metadata may increase or decrease as needed. For example, a file system on a 1 GB volume takes approximately 35 MB (about 3%) initially to store metadata. In contrast, a file system of 10 MB requires approximately 3.3 MB (30%) initially for storing the metadata.

File systems can be increased or decreased in size. SmartTier functionality is also provided at the file system level.

See “About Veritas Access SmartTier ” on page 169.

Any file system can be enabled for deduplication.

About scale-out file systems

Veritas Access provides a scale-out file system that manages a single namespace spanning over both on-premises storage as well as cloud storage, which provides better fault tolerance for large data sets. Unlike a standard file system, a scale-out file system is Active/Passive, which means that the file system can be online on only one node of the cluster at a time. A scale-out file system is always active on the node where its virtual IP address is online. A virtual IP address is associated with a scale-out file system when the file system is created.

You can find what virtual IP address is associated with a scale-out file system by using the `NFS> share show` command.

Veritas Access supports access to scale-out file systems using NFS-Ganesha and S3. NFS shares that are created on scale-out file systems must be mounted on the NFS clients using the virtual IP address that is associated with the scale-out file system, similarly S3 buckets created on a scale-out file system must be accessed using the same virtual IP address. You can use the CLISH `objectaccess> bucket show` command to get the virtual IP address. S3 buckets created on a scale-out file system must be accessed using virtual-hosted-style URL (rather than the path-style URL) and the S3 client's DNS must be updated to this virtual IP address for the corresponding virtual-hosted-style URL. If a bucket "bucket1" is created by the S3 client, then its virtual-hosted-style URL would be "bucket1.s3.cluster_name:8143," where the *cluster_name* is the Veritas Access cluster name and 8143 is the port on which the Veritas Access S3 server is running.

Scale-out file system specifications:

- Twenty percent of a scale-out file system's size is devoted to the metadata container.
- The maximum size of a metadata container is 10 TB.

- The minimum size of a scale-out file system is 10 GB.
- The maximum size of a scale-out file system is 3 PB.
- To create a scale-out file system above 522 TB, you need to provide the file system size in multiples of 128 GB.
- You can grow a scale-out file system up to 522 TB.
- You can shrink the scale-out file system only if its size is less than 522 TB.

See "Using the NFS-Ganesha server" on page 60.

A scale-out file system is structured as a layered file system that includes a set of storage containers. The data that is stored in the cloud (Amazon S3, Veritas Access S3, or any S3-compatible cloud storage provider) can be one of the storage containers. One storage container stores the metadata and the other containers store the actual data. This data can be on-premises or can be in the cloud. This modular structure allows the scale-out file system to be more resilient in cases where high capacity or fault tolerance is needed. A scale-out file system accomplishes this without compromising on file system performance.

You can configure Amazon S3, Amazon Glacier, or any S3-compatible cloud storage as a cloud container. Data can be moved between the on-premises container and the cloud container.

With a scale-out file system, you can move file data to the cloud using a tiering mechanism. File metadata including any attributes set on the file resides on-premises. The cloud as a tier feature is best used for moving infrequently accessed data to the cloud.

When Amazon S3 or any S3-compatible cloud storage provider is used as the cloud tier, the data present on S3 can be accessed any time (unlike in Amazon Glacier). An EIO error is returned if you try to write, or truncate the files moved to the S3 tier. If you want to modify the data, move the data to on-premises using tier move or using policies.

Amazon Glacier is an offline cloud tier, which means that data moved to Amazon Glacier cannot be accessed immediately. An EIO error is returned if you try to read, write, or truncate the files moved to the Amazon Glacier tier. If you want to read or modify the data, move the data to on-premises using tier move or using policies. The data is available after some time based on the Amazon Glacier retrieval option you selected.

Note: You cannot use the CIFS protocol with a scale-out file system.

See "Configuring the cloud as a tier feature for scale-out file systems" on page 92.

See "Moving files between tiers in a scale-out file system" on page 93.

Considerations for creating a file system

The following sections describe the considerations and best practices for creating file systems.

Best practices for creating file systems

The following are the best practices for creating file systems:

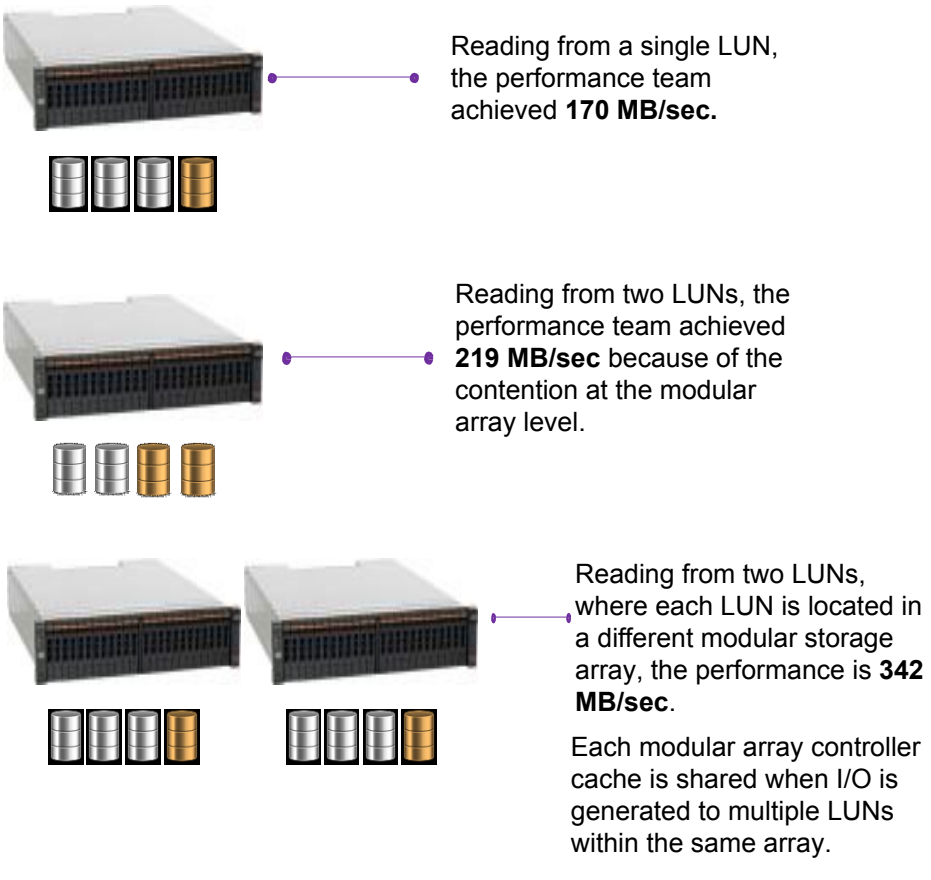
- Ensure all the disks (LUNs) in each storage pool have an identical hardware configuration.
  Best performance results from a striped file system that spans similar disks. The more closely you match the disks by speed, capacity, and interface type, the better the performance you can expect. When striping across several disks of varying speeds, performance is no faster than that of the slowest disk.
- Create striped file systems rather than simple file systems when creating your file systems.
  See "About striping file systems" on page 115.
- In a given storage pool, create all the file systems with the same number of columns.
- Ensure that the number of disks in each storage pool is an exact multiple of the number of columns used by the file systems created in that storage pool.
- Consider how many disks you need to add to your storage pool to grow your striped file systems.
  A 5-TB file system using five columns cannot be grown in a storage pool containing 8*1-TB disks, despite having 3 TB of disk space available. Instead create the file system with either four or eight columns, or else add 2*1-TB disks to the pool. See further examples in the table.

Use case	Action	Result
storage pool with eight disks of the same size (1 TB each)	Create a 5 TB striped file system with five columns.	You cannot grow the file system greater than 5 TB, even though there are three unused disks.
storage pool with eight disks of the same size (1 TB each)	Create a 5 TB striped file system with eight columns.	You can grow the file system to 8 TB.

storage pool with eight disks of the same size (1 TB each)	Create a 4 TB striped file system with four columns.	You can grow the file system to 8 TB.
storage pool with eight disks of the same size (1 TB each)	Create a 3 TB striped file system with three columns.	You cannot grow the file system to 8 TB.
storage pool with eight disks of the different sizes (3 are 500 GB each, and 5 are 2 TB each)	Create an 8 TB striped file system with eight columns.	You cannot create this 8-TB file system.

- Consider the I/O bandwidth requirement when determining how many columns you require in your striped file system.
  Based on the disks you have chosen, I/O throughput is limited and potentially restricted. Figure 17-1 describes the LUN throughput restrictions.
- Consider populating each storage pool with the same number of disks from each HBA. Alternatively, consider how much of the total I/O bandwidth that the disks in the storage pool can use.
  If you have more than one card or bus to which you can connect disks, distribute the disks as evenly as possible among them. That is, each card or bus must have the same number of disks attached to it. You can achieve the best I/O performance when you use more than one card or bus and interleave the stripes across them.
- Use a stripe unit size larger than 64 KB. Performance tests show 512 KB as the optimal size for sequential I/O, which is the default value for the stripe unit. A greater stripe unit is unlikely to provide any additional benefit.
- Do not change the operating system default maximum I/O size of 512 KB.

Figure 17-1 LUN throughput - details on the LUN throughput restrictions



About striping file systems

You can obtain huge performance benefits by striping (RAID-0) using software-defined storage (SDS). You achieve performance benefits regardless of the choice of LUN configuration in your storage hardware. Striping is useful if you need large amounts of data that is written to or read from physical disks, and consistent performance is important. SDS striping is a good practice for all Veritas Access use cases and workloads.

Veritas strongly recommends that you create striped file systems when creating your file system for the following reasons:

- Maximize the I/O performance.
- Proportion the I/O bandwidth available from the storage layer.

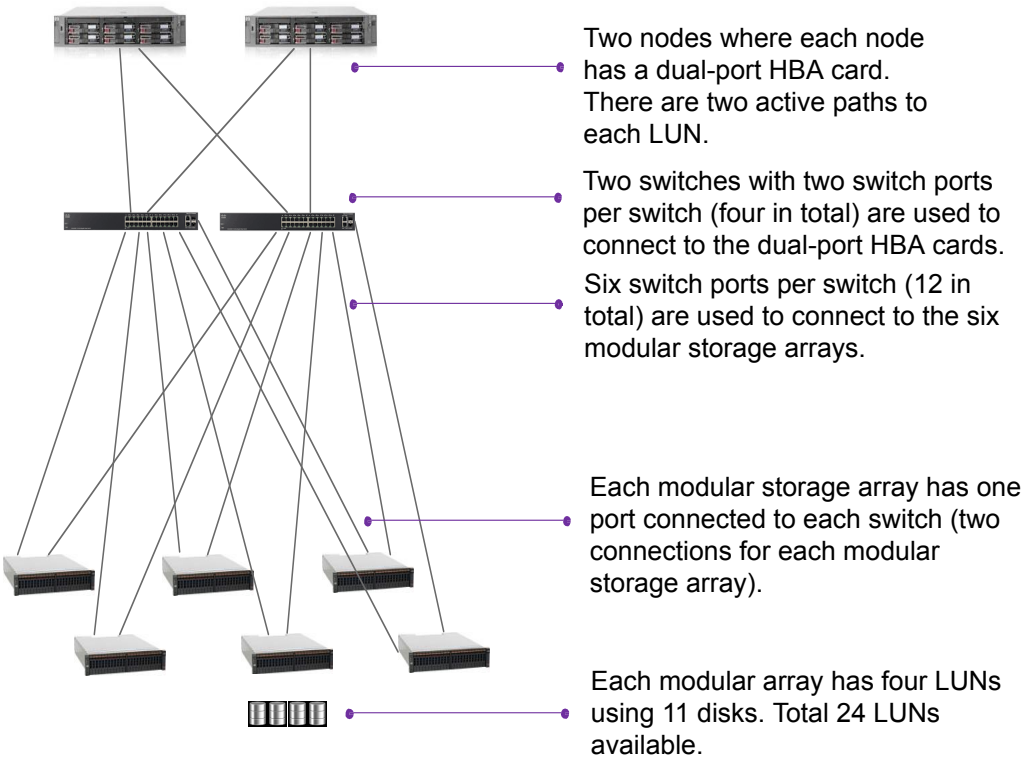
- Balance the I/O load evenly across multi-user applications running on multiple nodes in the cluster.

However there are also pitfalls to avoid.

The following information is essential before selecting the disks to include in your striped file system:

- Understanding of your hardware environment
- Storage capabilities and limitations (bottlenecks)
- Choice of LUNs (each LUN, or disk, equates to a column in a SDS-striped volume)

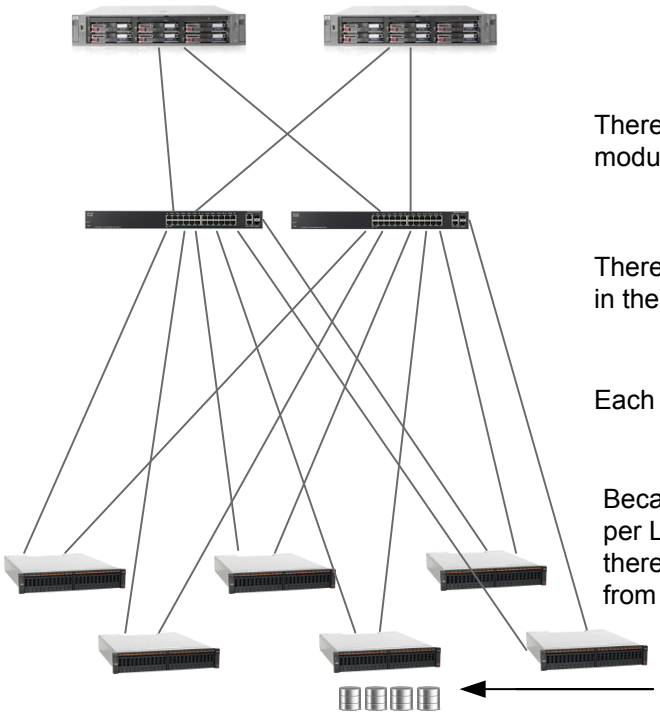
Figure 17-2 An example hardware configuration



An extreme example might be if one column (equal to one LUN) is composed of only hard disk drives (HDDs) in the storage array. All of the other columns in the same striped volume are composed of only SSDs in the storage array. The overall I/O performance bottlenecks on the single slower HDD LUN.

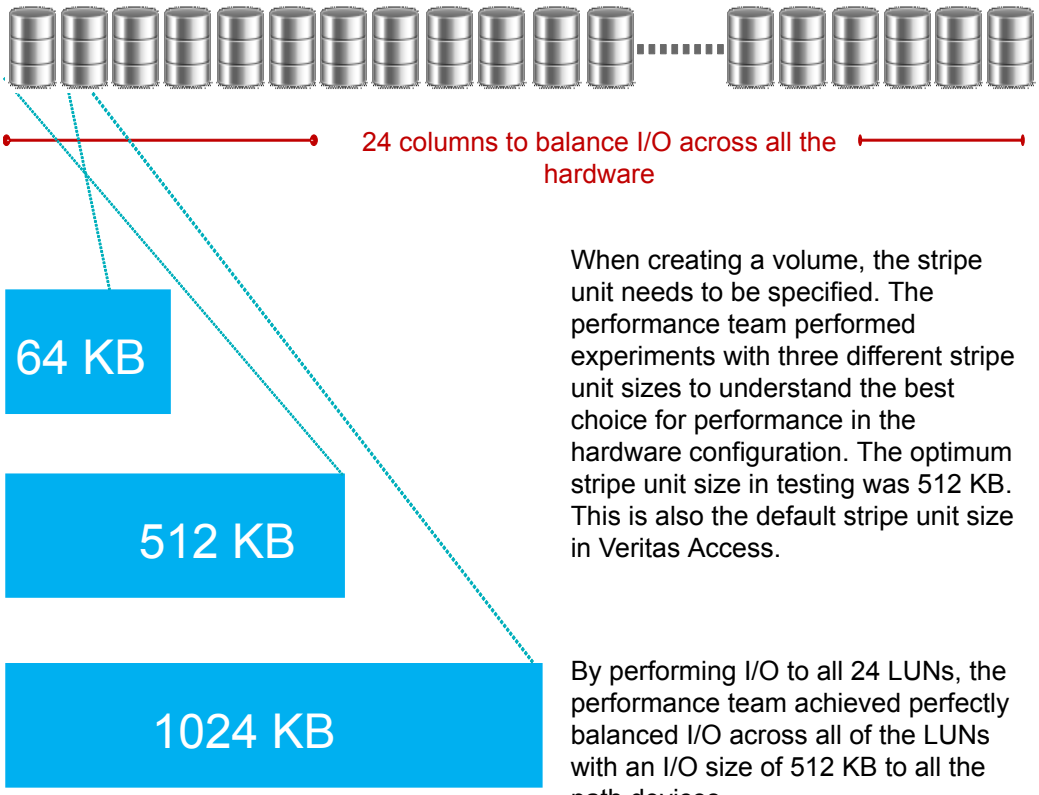
Understanding the LUN configuration and ensuring that all of the LUNs have an identical configuration is therefore essential for maximizing performance and achieving balanced I/O across all the LUNs.

Figure 17-3 LUN configuration



All 24 LUNs have an identical hardware configuration.

Figure 17-4 Volume configuration



The performance team created a volume with 24 columns to use the entire storage bandwidth in one file system. Veritas does not advise changing the operating system default maximum I/O size of 512 KB. The optimum stripe-unit size is 512 KB.

About FastResync

The FastResync feature performs quick and efficient resynchronization of stale mirrors (a mirror that is not synchronized). FastResync optimizes mirror resynchronization by keeping track of updates to stored data that have been missed by a mirror.

When FastResync has been enabled, it does not alter how you administer mirrors. The only visible effect is that repair operations conclude more quickly.

About creating a tuned file system for a specific workload

Veritas Access provides an easy way to create a well-tuned file system for a given type of workload.

You can use the newly created file system for the following common client applications:

- Virtual machine workloads
- Media server workloads
  Streaming media represents a new wave of rich Internet content. Recent advancements in video creation, compression, caching, streaming, and other content delivery technology have brought audio and video together to the Internet as rich media. You can use Veritas Access to store your rich media, videos, movies, audio, music, and picture files.

See the `Storage> fs` man page for more information.

```
Storage> fs create pretuned media_fs 100g pool2 workload=mediaserver layout=striped 8
```

The `workload=mediaserver` option creates a file system called `media_fs` that is 100g in size in `pool2` striped across eight disks.

Note: You can select only one workload for a specified file system. You specify the workload when you create the file system, and you cannot change the workload after you have created the file system.

Virtual machine workloads

A virtual machine disk file, also known as a VMDK file, is a file held in the Veritas Access file system that represents a virtual disk for a virtual machine. A VMDK file is the same size as the virtual disk, so VMDK file sizes are typically very large. As writes are performed to the virtual disk within the virtual machine, the VMDK file is populated with extents in the Veritas Access file system. Because the VMDK files are large, they can become heavily fragmented, which gradually impedes performance when reading and writing to the virtual disk. If you create a file system specific to a virtual machine workload, Veritas Access internally tunes the file system to allocate a fixed extent size of 1MB for VMDK files. The 1MB block size significantly reduces both file system and VMDK file fragmentation while improving the virtual machine workload performance.

Media server workloads and tunable for setting write_throttle

Media server workloads involve heavy sequential reads and writes. Striping across multiple disks yields better I/O latency.

See "Best practices for creating file systems" on page 113.

For media server workloads, Veritas Access provides a tunable that can help restrict the amount of write I/O throughput. The tunable helps prevent the streaming of information (sequential reads) from being affected by other processes performing write I/O on the same NAS server. An example use case is as follows. You want to stream a movie, this is reading a file (sequential reads). You do not want the movie experience to pause due to buffering. Another user might be uploading new content to the same file system (the upload is writing data to a different file). The uploading (writing) can cause the streaming (reading) to pause due to buffering. Veritas Access throttles the writing processes so that they do not consume too much of the system memory.

Each file system must tune the value write_thottle independently of other file systems. The default value is 0, which implies there is no write_throttle. The throttle is per file, so when writing to multiple files at the same time, the write_thottle threshold applies to each file independently.

Setting a non-zero value for a file system prevents the number of dirty memory pages that are associated with the file from increasing beyond the threshold. If you set a write_thottle value of 256, then writes to a file pause to flush the file to disk once 256 dirty memory pages have built up for the file. After the number of dirty pages for a file reaches the write_throttle threshold, further dirtying of pages is paused, and the file system starts flushing the file's pages to disk, even if free memory is available. Each memory page is 4KB of file data, so 256 pages is 1MB of file data. Setting a value for write_thottle means a writing thread pauses upon reaching the threshold (on the NAS server) while the file's dirty pages are flushed to disk, before resuming further writes to that file. Once flushed, the dirty pages become clean pages, which means the memory (the pages) can then be reused for perhaps pre-fetching data from disk for streaming reads. Setting a value for write_throttle helps prevent write I/O from consuming too much of the system memory.

Setting write_throttle requires some experimentation, which is why Veritas Access does not set a non-zero value by default. The valid range for write_throttle is 0 to 2048 pages. A good starting value for experimentation is 256.

About scale-out fsck

The scale-out fsck operation does the following:

- Checks the consistency of the metadata container, the data container, and the database, and repairs any inconsistencies.
- Checks if the metadata container and the data container are marked for full fsck. If yes, scale-out fsck performs a full fsck of the corresponding file systems. Based on the actions taken by fsck on the individual file systems, the scale-out fsck operation repairs the inconsistencies in other parts of the scale-out file system.
  See “About scale-out file systems” on page 111.
- Goes through all the file handles present in the database, and checks if the corresponding metadata container and the data container file handles are consistent with each other.
  In some cases, full fsck might delete files from the data container. To maintain consistency, the corresponding files from the metadata container and the data container are removed, and the corresponding key is removed from the database.

```
Storage> fs fsck fs1
fsck of largefs fs1 is successful
```

Section 10

Provisioning and managing Veritas Access shares

Chapter 18

Creating shares for applications

This chapter includes the following topics:

About file sharing protocols

Veritas Access provides support for multiple file sharing protocols. Veritas Access offers unified access, which provides the option to share a file system or a directory in a file system with more than one protocol. For unified access, only certain protocols combinations are supported.

See "About concurrent access" on page 124.

Table 18-1 Protocols

Protocol	Definition
Amazon S3	The object server lets you store and retrieve the data that is stored in Veritas Access using the Amazon Simple Storage Service (Amazon S3) protocol. See "About the object server" on page 83.

Table 18-1 Protocols *(continued)*

Protocol	Definition
CIFS	CIFS is active on all nodes within the Veritas Access cluster. The specific shares are read/write on the node they reside on, but can failover to any other node in the cluster. Veritas Access supports CIFS home directory shares. See "About configuring Veritas Access for CIFS" on page 64.
FTP	Allows clients to access files on Veritas Access servers. See "About FTP" on page 79.
NFS	All the nodes in the cluster can serve the same NFS share at the same time in read-write mode. This creates very high aggregated throughput rates, because you can use the sum of the bandwidth of all the nodes. Cache-coherency is maintained throughout the cluster. Veritas Access supports both the NFS kernel-based server and the NFS-Ganesha server in a mutually exclusive way. See "About using NFS server with Veritas Access" on page 59.
Oracle Direct NFS	Optimized NFS client that provides faster access to NFS storage located on NAS storage devices. See "About using Veritas Access with Oracle Direct NFS" on page 74.

About concurrent access

Veritas Access provides support for multi-protocol file sharing where the same file system can be exported to both Windows and UNIX users using the Common Internet File System (CIFS), Network File System (NFS), and Simple Storage Service (S3) protocols. The result is an efficient use of storage by sharing a single data set across multiple application platforms.

Note: When a share is exported over both NFS and CIFS protocols, the applications running on the NFS and CIFS clients may attempt to concurrently read or write the same file. This may lead to unexpected results, such as reading stale data, since the locking models used by these protocols are different. For this reason, Veritas Access warns you when the share export is requested over NFS or CIFS and the same share has already been exported for write access over CIFS or NFS.

The following sections describe concurrent access with multiple protocols.

See "Sharing directories using CIFS and NFS protocols" on page 125.

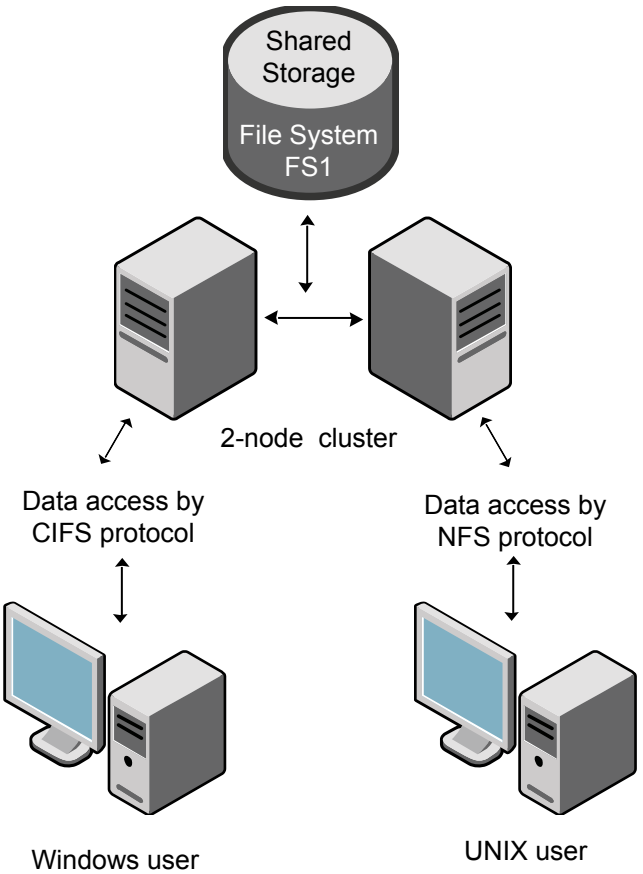
See “About file sharing protocols” on page 123.

Sharing directories using CIFS and NFS protocols

Veritas Access provides support for multi-protocol file sharing where the same directory or file system can be exported to both Windows and UNIX users using the CIFS and NFS protocols. The result is an efficient use of storage by sharing a single data set across multi-application platforms.

Figure 18-1 shows how the directory sharing for the two protocols works.

Figure 18-1 Exporting and/or sharing CIFS and NFS directories



It is recommended that you disable the `oplocks` option when the following occurs:

- A file system is exported over both the CIFS and NFS protocols.
- Either the CIFS and NFS protocol is set with read and write permission.

To export a directory to Windows and UNIX users

1 To export a directory to Windows and UNIX users with read-only and read-write permission respectively, enter the `CIFS` mode and enter the following commands:

```
CIFS> show
                     Name  Value
                     ----  -----
             netbios name  Pei60
                ntlm auth  yes
   allow trusted domains  no
                homedirfs
                 aio size  0
           idmap backend  rid:10000-1000000
                workgroup  PEI-DOMAIN
                 security  ads
                   Domain  PEI-DOMAIN.COM
              Domain user  Administrator
       Domain Controller  10.200.107.251
         Clustering Mode  normal
CIFS> share add fs1 share1 ro
Exporting CIFS filesystem : share1...
CIFS> share show
ShareName FileSystem ShareOptions
share1 fs1 owner=root,group=root,ro
```

Exit CIFS mode:

```
CIFS> exit
```

2 Enter the `NFS` mode and enter the following commands:

```
NFS> share add rw fs1
ACCESS nfs WARNING V-288-0 Filesystem (fs1)
is already shared over CIFS with 'ro' permission.
Do you want to proceed (y/n): y
Exporting *:/vx/fs1 with options rw
..Success.
NFS> share show
/vx/fs1 * (rw)
```

Chapter 19

Creating and maintaining NFS shares

This chapter includes the following topics:

About NFS file sharing

The Network File System (NFS) protocol enables exported directories (including all files under the directory that reside on the exported directory's file system) hosted by an NFS server to be accessed by multiple UNIX and Linux client systems.

Using NFS, a local system can mount and use a disk partition or a file system from a remote system (an NFS server), as if it were local. The Veritas Access NFS server exports a directory, with selected permissions and options, and makes it available to NFS clients.

The selected permissions and options can also be updated, by adding or removing permissions, to restrict or expand the permitted use.

The Veritas Access NFS service is clustered. The NFS clients continuously retry during a failover transition. Even if the TCP connection is broken for a short time,

the failover is transparent to NFS clients, and NFS clients regain access transparently as soon as the failover is complete.

See the `nfs(1)` manual pages for all the supported operations.

See "About using NFS server with Veritas Access" on page 59.

Displaying file systems and snapshots that can be exported

To display a file system and snapshots that can be exported

- To display online file systems and the snapshots that can be exported, enter the following:

  ```
  NFS> show fs
  ```

  For example:

  ```
  NFS> show fs
  FS/Snapshot
  ===========
  fs1
  ```

Exporting an NFS share

You can export an NFS share with the specified NFS options that can then be accessed by one or more client systems.

If you add a directory that has already been exported with a different NFS option (rw, ro, async, or secure, for example), Veritas Access provides a warning message saying that the directory has already been exported. Veritas Access updates (overwrite) the old NFS options with the new NFS options.

Directory options appear in parentheses.

If a client was not specified when the `NFS> share add` command was used, then `*` is displayed as the system to be exported to, indicating that all clients can access the directory.

Directories that have been exported to multiple clients appear as separate entries. Directories that are exported to `<world>` and other specific clients also appear as separate entries.

For example:

Consider the following set of exported directories where only the client (1.1.1.1) has **read-write** access to directory (fs2), while all other clients have **read** access only.

```
/vx/fs2         * (ro)

/vx/fs2  1.1.1.1 (rw)
```

Exporting the same directory to multiple clients with different permissions is not supported with NFS-Ganesha.

When sharing a directory, Veritas Access does not check whether the client exists or not. If you add a share for an unknown client, then an entry appears in the `NFS> show` command output.

The `NFS> show fs` command displays the list of exportable file systems. If a directory does not exist, the directory is automatically created and exported when you try to export it.

Valid NFS options include the following:

`rw`	Grants read and write permission to the directory (including all files under the directory that reside on the exported directory's file system). Hosts mounting this directory will be able to make changes to the directory.
`ro (Default)`	Grants read-only permission to the directory. Hosts mounting this directory will not be able to change it.
`sync (Default)`	Grants synchronous write access to the directory. Forces the server to perform a disk write before the request is considered complete.
`async`	Grants asynchronous write access to the directory. Allows the server to write data to the disk when appropriate.
`secure (Default)`	Grants secure access to the directory. Requires that clients originate from a secure port. A secure port is between 1-1024.
`insecure`	Grants insecure access to the directory. Permits client requests to originate from unprivileged ports (those above 1024).
`secure_locks (Default)`	Requires authorization of all locking requests. This option is not supported with NFS-Ganesha.

`insecure_locks`	Some NFS clients do not send credentials with lock requests, and therefore work incorrectly with `secure_locks`, in which case you can only lock world-readable files. If you have such clients, either replace them with better ones, or use the `insecure_locks` option. This option is not supported with NFS-Ganesha.
`root_squash (Default)`	Prevents the root user on an NFS client from having root privileges on an NFS mount. This effectively "squashes" the power of the remote root user to the lowest local user, preventing remote root users from acting as though they were the root user on the local system.
`no_root_squash`	Disables the `root_squash` option. Allows root users on the NFS client to have root privileges on the NFS server.
`wdelay (Default)`	Causes the NFS server to delay writing to the disk if another write request is imminent. This can improve performance by reducing the number of times the disk must be accessed by separate write commands, reducing write overhead. **Note:** The `wdelay` option is deprecated, and is supported for backward-compatibility only. This option is not supported with NFS-Ganesha.
`no_wdelay`	Disables the `wdelay` option. The `no_wdelay` option has no effect if the `async` option is also set. **Note:** The `no_wdelay` option is deprecated, and is supported for backward-compatibility only. Using the `no_wdelay` option is always effective. This option is not supported with NFS-Ganesha.
`subtree_check`	Verifies that the requested file is in an exported subdirectory. If this option is turned off, the only verification is that the file is in an exported file system. This option is not supported with NFS-Ganesha.
`no_subtree_check (Default)`	Sometimes subtree checking can produce problems when a requested file is renamed while the client has the file open. If many such situations are anticipated, it might be better to set `no_subtree_check`. One such situation might be the export of the home directory. Most other situations are best handled with `subtree_check`. This option is not supported with NFS-Ganesha.

`fsid (Default)`	Allows the Veritas Access administrator to associate a specific number as fsid with the share. This option is not supported with NFS-Ganesha.
`nordirplus`	Allows you to disable a readdirplus remote procedure call (RPC).
`sec`	Specifies the Kerberos security options for exporting an NFS share. The value can be `krb5`, `krb5i` , `krb5p`, or `sys`. The `sys` option does not provide Kerberos authentication. The other options use Kerberos V5 to authenticate users to the NFS server.

For example, you could issue the following commands:

```
NFS> share add rw,async /vx/fs2

NFS> share add rw,sync,secure,root_squash /vx/fs3 10.10.10.10

NFS> share add rw,sync,no_root_squash /vx/fs3 2001:21::/120
Exporting /vx/fs3 with options rw,sync,no_root_squash
Success.
```

Note: With `root_squash`, the root user can access the share, but with 'nobody' permissions.

To export a directory/file system

1 To see your exportable online file systems and snapshots, enter the following:

```
NFS> show fs
```

For example:

```
NFS> show fs
FS/Snapshot
===========
fs2
fs3
```

2 To see your NFS shares and their options, enter the following:

```
NFS> share show
```

For example:

```
NFS> share show
/vx/fs2          * (sync)
/vx/fs3          * (secure,ro,no_root_squash)

NFS> share show
/vx/fs3         2001:21::/120 (rw,sync,no_root_squash)
```

3 To export a directory, enter the following command:

```
NFS> share add nfsoptions export_dir [client]
```

nfsoptions	Comma-separated list of export options from the set.
export_dir	Specifies the name of the directory you want to export. The directory name should start with `/vx`, and only `a-zA-Z0-9_/@+=.:-` characters are allowed for `export_dir`.

client

Clients may be specified in the following ways:

- Single host - specify a host either by an abbreviated name that is recognized by the resolver (DNS is the resolver), the fully qualified domain name, or an IP address.
- Netgroups - specify netgroups as *@group*. Only the host part of each netgroup member is considered for checking membership.
- IP networks - specify an IP address and netmask pair (address/netmask) to simultaneously export directories to all hosts on an IP sub-network. Specify the netmask as a contiguous mask length. You can specify either an IPv4 address or an IPv6 address.

If the client is not given, then the specified directory can be mounted or accessed by any client. To re-export new options to an existing share, the new options will be updated after the command is run.

Example using NFS options:

```
NFS> share add async /vx/fs1
Exporting *:/vx/fs1 with options async
..Success.
```

Displaying exported directories

You can display the exported directories and the NFS options that are specified when the directory was exported. For NFS-Ganesha (GNFS), the output also shows the virtual IP address that must be used on the client to mount the GNFS shares for the shares that are exported from scale-out file systems.

To display exported directories

To display exported directories, enter the following:

```
NFS> share show
```

For example, for the kernel NFS server:

```
NFS> share show
/vx/fs2          * (sync)
/vx/fs3          * (secure,ro,no_root_squash)
```

The command output displays the following columns:

Left-hand column Displays the directory that was exported.

For example:

```
/vx/fs2
```

Right-hand column Displays the system that the directory is exported to, and the NFS options with which the directory was exported.

For example:

```
* (secure,ro,no_root_squash)
```

For example, for the NFS-Ganesha server:

```
NFS> share show
/vx/fs1          * (rw,async) 10.209.106.180
```

The command output displays the following columns:

Left-hand column Displays the directory that was exported.

For example:

```
/vx/fs1
```

Middle column Displays the system that the directory is exported to, and the NFS options with which the directory was exported.

For example:

```
* (rw,async)
```

Right-hand column Displays the virtual IP address that must be used on the client to mount the GNFS shares of scale-out file systems.

For example:

Use the address `10.209.106.180` to mount the `/vx/fs1` file system.

About managing NFS shares using netgroups

A netgroup defines a network-wide group of hosts and users. You use netgroups for restricting access to shared NFS file systems and to restrict remote login and shell access.

Each line in the netgroup file consists of a netgroup name followed by a list of members, where a member is either another netgroup name, or a comma-separated list of host, user, or a domain. Host, user, and domain are character strings for the corresponding components. Any of these three fields can be empty, which indicates a wildcard, or may consist of the string "-" to indicate that there is no valid value for the field. The domain field must either be the local domain name or empty for the netgroup entry to be used. This field does not limit the netgroup or provide any security. The domain field refers to the domain in which the host is valid, not the domain containing the trusted host.

When exporting a directory by NFS with the specified options, clients may be specified using netgroups. Netgroups are identified using @group. Only the host part of each netgroup member is considered when checking for membership.

```
NFS> share add rw,async /vx/fs1/share @client_group
```

Unexporting a directory or deleting NFS options

You can unexport the share of the exported directory.

Note: You will receive an error message if you try to remove a directory that does not exist.

To unexport a directory or delete NFS options

1 To see your existing exported resources, enter the following command:

```
NFS> share show
```

Only the directories that are displayed can be unexported.

For example:

```
NFS> share show
/vx/fs2          * (sync)
/vx/fs3          * (secure,ro,no_root_squash)
```

2 To delete a directory from the export path, enter the following command:

```
NFS> share delete export_dir [client]
```

For example:

```
NFS> share delete /vx/fs3
Removing export path *:/vx/fs3
..Success.
```

export_dir

Specifies the name of the directory you want to delete.

The directory name should start with `/vx`, and only `a-zA-Z0-9_/@+=.:-` characters are allowed in `export_dir`.

You cannot include single or double quotes that do not enclose characters.

```
NFS> share delete "*:/vx/example"
```

client

Clients may be specified in the following ways:

- Single host - specify a host either by an abbreviated name that is recognized by the resolver (DNS is the resolver), the fully qualified domain name, or an IP address.
- Netgroups - specify netgroups as @group. Only the host part of each netgroup member is considered for checking membership.
- IP networks - specify an IP address and netmask pair (address/netmask) to simultaneously export directories to all hosts on an IP sub-network. Specify the netmask as a contiguous mask length.

If *client* is included, the directory is removed from the export path that was directed at the *client*.

If a directory is being exported to a specific client, the `NFS> share delete` command must specify the client to remove that export path.

If the client is not specified, then the specified directory can be mounted or accessed by any client.

Chapter 20

Creating and maintaining CIFS shares

This chapter includes the following topics:

About managing CIFS shares

You can export the Veritas Access file systems to clients as CIFS shares. When a share is created, it is given a name. The share name is different from the file system name. Clients use the share name when they import the share.

Note: You cannot export a scale-out file system as a CIFS share.

You create and export a share with one command. The same command binds the share to a file system, and you can also use it to specify share properties.

In addition to exporting file systems as CIFS shares, you can use Veritas Access to store user home directories. Each of these home directories is called a home directory share. Shares that are used to export ordinary file systems (that is, file systems that are not used for home directories), are called ordinary shares to distinguish them from home directory shares.

About the CIFS export options

The following are the CIFS export options.

Table 20-1 CIFS export options

CIFS export option	Definition
rw	There is a share option which specifies if the files in the share will be read-only or if both read and write access will be possible, subject to the authentication and authorization checks when a specific access is attempted. This share option can be given one of these values, either `rw` or `ro`. Grants read and write permission to the exported share.
ro (Default)	Grants read-only permission to the exported share. Files cannot be created or modified.
guest	Another configuration option specifies if a user trying to establish a CIFS connection with the share must always provide the user name and password, or if they can connect without it. In this case, only restricted access to the share will be allowed. The same kind of access is allowed to `anonymous` or `guest` user accounts. This share option can have one of the following values, either guest or noguest. Veritas Access allows restricted access to the share when no user name or password is provided.
noguest (Default)	Veritas Access always requires the user name and password for all of the connections to this share.
full_acl	All Windows Access Control Lists (ACLs) are supported except in the case when you attempt using the Windows Explorer folder **Properties > Security GUI** to inherit down to a non-empty directory hierarchy while denying all access to yourself.
no_full_acl (Default)	Some advanced Windows Access Control Lists (ACLs) functionality does not work. For example, if you try to create ACL rules on files saved in a CIFS share using Windows explorer while allowing some set of file access for `user1` and denying file access for `user2`, this is not possible when CIFS shares are exported using `no_full_acl`.
hide_unreadable	Prevents clients from seeing the existence of files and directories that are not readable to them. The default is: `hide_unreadable` is set to off.

Table 20-1 CIFS export options *(continued)*

CIFS export option	Definition
veto_sys_files	To hide some system files (lost+found, quotas, quotas.grp) from displaying when using a CIFS normal share, you can use the `veto_sys_files` CIFS export option. For example, when adding a CIFS normal share, the default is to display the system files. To hide the system files, you must use the `veto_sys_files` CIFS export option.
fs_mode	When a file system or directory is exported by CIFS, its mode is set to an `fs_mode` value. It is the UNIX access control set on a file system, and CIFS options like `rw`/`ro` do not take precedence over it. This value is reset to `0755` when the CIFS share is deleted. The default is: `fs_mode = 1777`.
dir_mask	When a directory is created under a file system or directory exported by CIFS, the necessary permissions are calculated by mapping DOS modes to UNIX permissions. The resulting UNIX mode is then bit-wise 'AND'ed with this parameter. Any bit not set here is removed from the modes set on a directory when it is created. The default is: `dir_mask = 0775`.
create_mask	When a file is created under a file system or directory exported by CIFS, the necessary permissions are calculated by mapping DOS modes to UNIX permissions. The resulting UNIX mode is then bit-wise 'AND'ed with this parameter. Any bit not set here is removed from the modes set on a file when it is created. The default is: `create_mask = 0775`.
oplocks (Default)	Veritas Access supports the CIFS opportunistic locks. You can enable or disable them for a specific share. The opportunistic locks improve performance for some workloads, and there is a share configuration option which can be given one of the following values, either oplocks or nooplocks. Veritas Access supports opportunistic locks on the files in this share.
nooplocks	No opportunistic locks will be used for this share. Disable the oplocks when: ■ 1) A file system is exported over both CIFS and NFS protocols. ■ 2) Either CIFS or NFS protocol has read and write access.

Table 20-1 CIFS export options *(continued)*

CIFS export option	Definition
owner	There are more share configuration options that can be used to specify the user and group who own the share. If you do not specify these options for a share, Veritas Access uses the current values as default values for these options. You may want to change the default values to allow a specific user or group to be the share owner. Irrespective of who are owner and group of the exported share, any CIFS clients can create folders and files in the share. However, there are some operations that require owner privileges; for example, changing the owner itself, and changing permissions of the top-level folder (that is, the root directory in UNIX terms). To enable these operations, you can set the owner option to a specific user name, and this user can perform the privileged operations.
group	By default, the current group is the primary group owner of the root directory of the exported share. This lets CIFS clients create folders and files in the share. However, there are some operations that require group privileges; for example, changing the group itself, and changing permissions of the top-level folder (that is, the root directory in UNIX terms). To enable these operations, you can set the `group` option to a specific group name, and this group can perform the privileged operations.
ip	Veritas Access lets you specify a virtual IP address. If you set `ip=virtualip`, the share is located on the specified virtual IP address. This address must be part of the Veritas Access cluster, and is used by the system to serve the share internally. **Note:** ip is not a valid CIFS option when using the ctdb clustering mode.
max_connections	Specify the maximum limit for concurrent CIFS connections for a CIFS share. The default value is `0`, indicating that there are no limited connections.
shadow_copy	Indicates that this is a `shadow_copy` capable CIFS share.

Table 20-1 CIFS export options *(continued)*

CIFS export option	Definition
enable_encryption	If `enable_encryption` is set, then all the traffic to a share must be encrypted once the connection has been made to the share. The server will return an `access denied` message to all unencrypted requests on such a share. As SMB3 is the max protocol, only SMB3 clients supporting encryption will be able to connect to the share.
disable_encryption	If `disable_encryption` is set, then encryption cannot be negotiated by the client. SMB1, SMB2, and SMB3 clients can connect to the share.
enable_durable_handles	Enables support for durable handles for CIFS shares. Enabling this option disables use of POSIX/fcntl locks. Exporting the same CIFS share using NFS may result in data corruption. For support for durable handles on CIFS shares, you must specify this option.

Chapter 21

Using Veritas Access with OpenStack

This chapter includes the following topics:

About the Veritas Access integration with OpenStack

OpenStack is a cloud operating system that controls large pools of computer, storage, and networking resources in a data center. OpenStack provides a dashboard that lets you provision resources using a web interface.

Veritas Access is integrated with the following OpenStack components:

- Cinder - is a block storage service for OpenStack. Cinder provides the infrastructure for managing volumes in OpenStack. Cinder volumes provide persistent storage to guest virtual machines (known as instances) that manage OpenStack compute software. Cinder allows the ability for OpenStack instances to use the storage hosted by Veritas Access.
  See "About the Veritas Access integration with OpenStack Cinder" on page 144.
- Manila - lets you share Veritas Access file systems with virtual machines on OpenStack.
  See "About the Veritas Access integration with OpenStack Manila" on page 151.

About the Veritas Access integration with OpenStack

OpenStack is a cloud operating system that controls large pools of computer, storage, and networking resources in a data center. OpenStack provides a dashboard that lets you provision resources using a web interface.

Veritas Access is integrated with the following OpenStack components:

- Cinder - is a block storage service for OpenStack. Cinder provides the infrastructure for managing volumes in OpenStack. Cinder volumes provide persistent storage to guest virtual machines (known as instances) that manage OpenStack compute software. Cinder allows the ability for OpenStack instances to use the storage hosted by Veritas Access.
  See "About the Veritas Access integration with OpenStack Cinder" on page 144.
- Manila - lets you share Veritas Access file systems with virtual machines on OpenStack.
  See "About the Veritas Access integration with OpenStack Manila" on page 151.

About the Veritas Access integration with OpenStack Cinder

Cinder is a block storage service for OpenStack. Cinder provides the infrastructure for managing volumes in OpenStack. Cinder volumes provide persistent storage to guest virtual machines (known as instances) that manage OpenStack compute software.

Veritas Access is integrated with OpenStack Cinder, which provides the ability for OpenStack instances to use the storage hosted by Veritas Access.

Table 21-1 Mapping of OpenStack Cinder operations to Veritas Access

Operation in OpenStack Cinder	Operation in Veritas Access
Create and delete volumes	Create and delete files.
Attach and detach the volumes to virtual machines	This operation occurs on the OpenStack controller node. This operation is not applicable in Veritas Access.
Create and delete snapshots of the volumes	Create and delete file system snapshots.
Create a volume from a snapshot	This operation occurs on the OpenStack controller node. This operation is not applicable in Veritas Access.

Table 21-1 Mapping of OpenStack Cinder operations to Veritas Access *(continued)*

Operation in OpenStack Cinder	Operation in Veritas Access
Copy images to volumes	This operation occurs on the OpenStack controller node. This operation is not applicable in Veritas Access.
Copy volumes to images	This operation occurs on the OpenStack controller node. This operation is not applicable in Veritas Access.
Extend volumes	Extending files.

Note: To perform these operations, you need to use the OpenStack Cinder commands, not the Veritas Access commands.

The Veritas NFS OpenStack Cinder driver is a Python script that is checked in to the OpenStack source code in the public domain. To use the Veritas Access integration with OpenStack Cinder, you need to make some configuration changes on the OpenStack controller node.

For the supported OpenStack versions for running the OpenStack Cinder driver, see the *Veritas Access Installation Guide*.

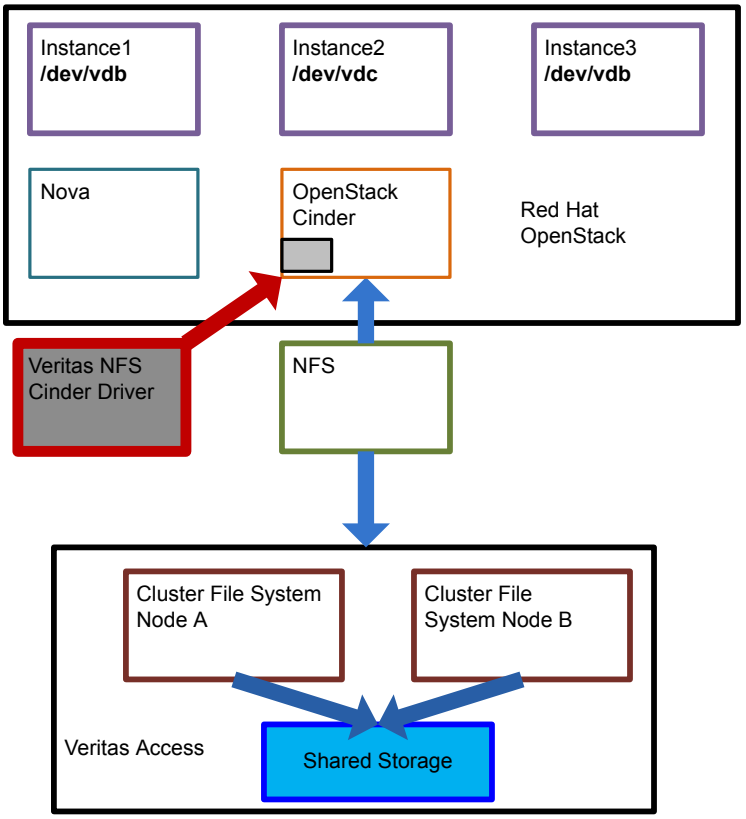
See "Configuring OpenStack Cinder" on page 147.

About the Veritas Access integration with OpenStack Cinder architecture

Figure 21-1 describes the Veritas Access integration with OpenStack Cinder architecture.

OpenStack instances are the individual virtual machines running on physical compute nodes. The compute service, Nova, manages the OpenStack instances.

Figure 21-1 Veritas Access integration with OpenStack Cinder architecture



Configuring OpenStack Cinder

To create a new volume backend named ACCESS_HDD in OpenStack Cinder

1 Add the following configuration block in the `/etc/cinder/cinder.conf` file on your OpenStack controller node.

```
enabled_backends=access-1
[access-1]
 volume_driver=cinder.volume.drivers.veritas_cnfs.VeritasCNFSDriver
 volume_backend_name=ACCESS_HDD
 nfs_shares_config=/etc/cinder/access_share_hdd
 nfs_mount_point_base=/cinder/cnfs/cnfs_sata_hdd
 nfs_sparsed_volumes=True
 nfs_disk_util=df
 nfs_mount_options=nfsvers=3
```

Add the lines from the configuration block at the bottom of the file.

`volume_driver`	Name of the Veritas Access Cinder driver.
`volume_backend_name`	For this example, ACCESS_HDD is used. This name can be different for each NFS share. If several backends have the same name, the OpenStack Cinder scheduler decides in which backend to create the volume.
`nfs_shares_config`	This file has the share details in the form of *`vip:/exported_dir`*.
`nfs_mount_point_base`	Mount point where the share will be mounted on OpenStack Cinder. If the directory does not exist, create it. Make sure that the `Cinder` user has write permission on this directory.
`nfs_sparsed_volumes`	Preallocate or sparse files.
`nfs_disk_util`	Free space calculation.
`nfs_mount_options`	These are the mount options OpenStack Cinder uses to NFS mount.

This same configuration information for adding to the `/etc/cinder/cinder.conf` file can be obtained by running the `OPENSTACK CINDER>` **`configure`** ***`export_dir`*** command.

2 Append the following in the `/etc/cinder/access_share_hdd` file on your OpenStack controller node:

```
vip:/vx/fs1
```

Use one of the virtual IPs for *vip*:

- 192.1.1.190
- 192.1.1.191
- 192.1.1.192
- 192.1.1.193
- 192.1.1.199

You can obtain Veritas Access virtual IPs using the `OPENSTACK> cinder` **`configure export-dir`** option.

3 Create the `/etc/cinder/access_share_hdd` file at the root prompt, and update it with the NFS share details.

```
# cnfs_sata_hdd(keystone_admin)]# cat /etc/cinder/access_share_hdd
192.1.1.190:/vx/fs1
```

4 The Veritas Access package includes the Veritas Access OpenStack Cinder driver, which is a Python script. The OpenStack Cinder driver is located at `/opt/VRTSnas/scripts/OpenStack/veritas_cnfs.py` on the Veritas Access node. Copy the `veritas_cnfs.py` file to `/usr/lib/python2.6/site-packages/cinder/volume/drivers/veritas_cnfs.py` if you are using the Python 2.6 release.

If you are using the OpenStack Kilo version of RDO, the file is located at:

```
/usr/lib/python2.7/site-packages/cinder/volume/drivers/veritas_cnfs.py
```

5 Make sure that the NFS mount point on the OpenStack controller node has the right permission for the cinder user. The cinder user should have write permission on the NFS mount point. Set the permission using the following command.

```
# setfacl -m u:cinder:rwx /cinder/cnfs/cnfs_sata_hdd
```

6 Restart the OpenStack Cinder driver.

```
# cnfs_sata_hdd(keystone_admin)]# /etc/init.d/openstack-cinder-volume
restart
Stopping openstack-cinder-volume: [ OK ]
Starting openstack-cinder-volume: [ OK ]
```

Restarting the OpenStack Cinder driver picks up the latest configuration file changes.

After restarting the OpenStack Cinder driver, `/vx/fs1` is NFS-mounted as per the instructions provided in the `/etc/cinder/access_share_hdd` file.

```
# cnfs_sata_hdd(keystone_admin)]# mount |grep /vx/fs1
192.1.1.190:/vx/fs1 on
cnfs_sata_hdd/e6c0baa5fb02d5c6f05f964423feca1f type nfs
(rw,nfsvers=3,addr=10.182.98.20)
```

You can obtain OpenStack Cinder log files by navigating to:

```
/var/log/cinder/volume.log
```

7 If you are using OpenStack RDO, use these steps to restart the OpenStack Cinder driver.

Login to the OpenStack controller node.

For example:

```
source /root/keystonerc_admin
```

Restart the services using the following command:

```
(keystone_admin)]# openstack-service restart openstack-cinder-volume
```

For more information, refer to the *OpenStack Administration Guide*.

8 On the OpenStack controller node, create a volume type named `isa_vol_type`.

This volume type is used to link to the volume backend.

```
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]#
cinder type-create isa_vol_type
+--------------------------------------+------------------+
|   ID                                 | Name             |
+--------------------------------------+------------------|
| d854a6ad-63bd-42fa-8458-a1a4fadd04b7 | isa_vol_type     |
+--------------------------------------+------------------+
```

9 Link the volume type with the ACCESS_HDD back end.

```
[root@c1059-r720xd-111046cnfs_sata_hdd(keystone_admin)]# cinder type-key
isa_vol_type set volume_backend_name=ACCESS_HDD
```

10 Create a volume of size 1gb.

```
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder create --volume-type
isa_vol_type --display-name isa_vol1 1
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2014-02-08T01:47:25.726803      |
| display_description |                 None                 |
|     display_name    |                isa_vol1              |
|          id         |                disk ID 1             |
|       metadata      |                  {}                  |
|         size        |                  1                   |
|     snapshot_id     |                 None                 |
|     source_volid    |                 None                 |
|        status       |               creating               |
|     volume_type     |           isa_vol_type               |
+---------------------+--------------------------------------+
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder list
+---------------+----------+-------------+-----+--------------+--------+------------+
|      ID       |  Status  | Display Name| Size| Volume Type  |Bootable| Attached to|
+---------------+----------+-------------+-----+--------------+--------+------------+
| disk ID 1     | available| isa_vol1    |  1  | isa_vol_type |   false|            |
+---------------------------------------+-----+--------------+--------+------------+
```

11 Extend the volume to 2gb.

```
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder extend isa_vol1 2

[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder list
+------------+-----------+--------------+------+--------------+---------+------------+
|     ID     |  Status   | Display Name | Size |  Volume Type | Bootable| Attached to|
+------------------------+--------------+------+--------------+----------------------+
| disk ID 1  |  available| isa_vol1     |  2   | isa_vol_type |  false  |            |
+------------+-----------+--------------+------+--------------+---------+------------+
```

12 Create a snapshot.

```
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder snapshot-create
--display-name isa_vol1-snap isa_vol1
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|      created_at     |      2014-02-08T01:51:17.362501      |
| display_description |                 None                 |
|     display_name    |              isa_vol1-snap           |
|          id         |               disk ID 1              |
|       metadata      |                  {}                  |
|         size        |                  2                   |
|        status       |               creating               |
|      volume_id      | 52145a91-77e5-4a68-b5e0-df66353c0591 |

[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder snapshot-list
+-----------+--------------------------------------+-----------+-----------------+------+
|       ID  |  Volume ID                           |  Status   | Display Name    | Size |
+-----------+--------------------------------------+-----------+-----------------+------+
| disk ID 1 |  52145a91-77e5-4a68-b5e0-df66353c0591| available | isa_vol1-snap   |  2   |
+-----------+--------------------------------------+-----------+-----------------+------+
```

13 Create a volume from a snapshot.

```
[root@c1059-r720xd-111046 cnfs_sata_hdd(keystone_admin)]# cinder
create --snapshot-id e9dda50f-1075-407a-9cb1-3ab0697d274a --display-name
isa-vol2 2
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2014-02-08T01:57:11.558339      |
```

About the Veritas Access integration with OpenStack Manila

OpenStack Cinder had the limitation of not being able to share a block device simultaneously between virtual machines. OpenStack Manila solves this problem.

OpenStack Manila provides a shared file system as a service. Using OpenStack Manila, you can share a single file system between multiple virtual machines.

Veritas Access is integrated with OpenStack Manila through a OpenStack Manila driver that lets you share Veritas Access file systems with virtual machines on OpenStack.

For the supported OpenStack versions for running the OpenStack Manila driver, see the *Veritas Access Installation Guide*.

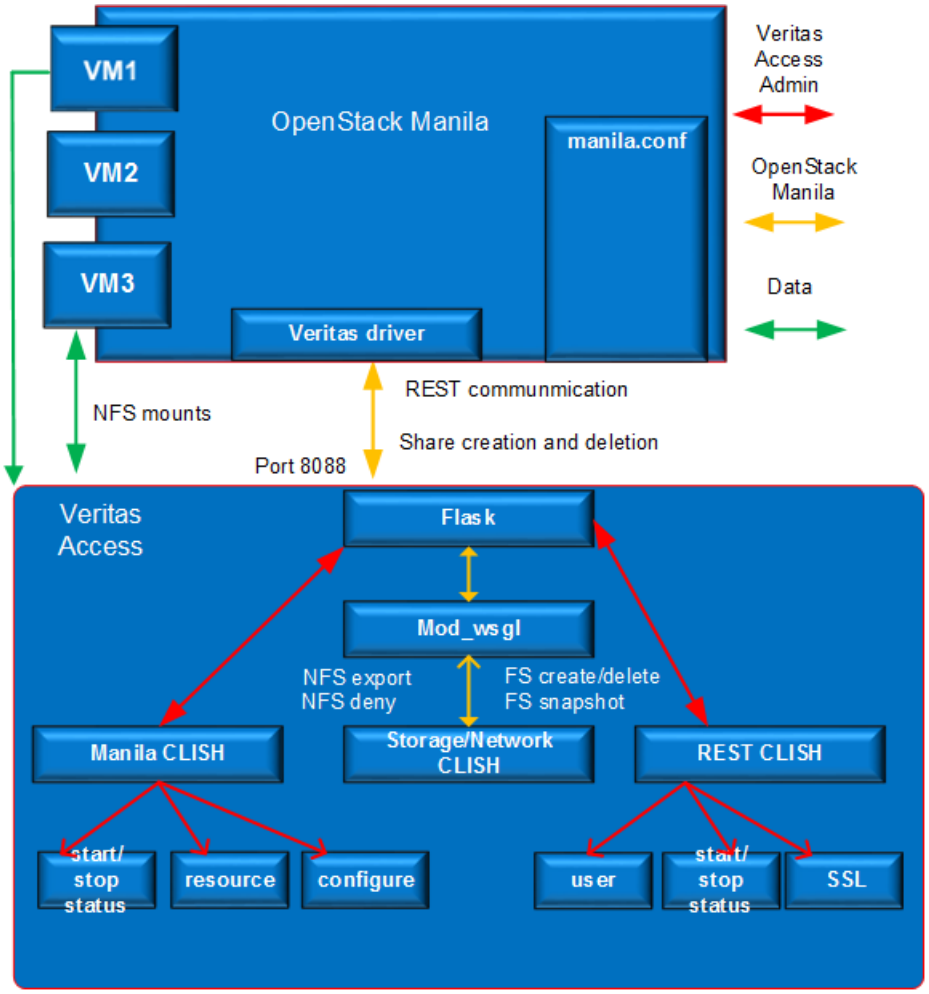
The OpenStack Manila driver communicates with Veritas Access using Representational State Transfer (REST) APIs, which provide access to resources (data entities) using HTTP or HTTPS, one at a time. By default, the REST server is configured for HTTP access only. Flask is the web application that is used to authenticate the Manila user and to run the OpenStack Manila driver.

The OpenStack Manila driver can create and manage simple file systems. For the backend to create simple file systems, use `isa_fstype=simple` in the `manila.conf` file.

Veritas Access REST APIs perform some of the following operations. This is not a complete list of the REST API operations.

- Create a REST user named Manila.
  The Manila REST user is created during Veritas Access installation. There is no Veritas Access CLI command for creating the Manila REST user.
  The OpenStack Manila driver authenticates with Veritas Access using Manila user credentials.
  Veritas Access supports a single OpenStack tenant.
- Provide a mechanism for adding existing storage pools to the OpenStack Manila user.
  The OpenStack Manila driver can create or delete file systems only from the storage pools that have been added through the appropriate REST operation. This prevents the OpenStack Manila driver from accidentally deleting the file system from any storage pools on Veritas Access.
  You configure the added storage pools in the `manila.conf` file on the OpenStack controller node.
  A single OpenStack Manila backend supports a single storage pool. If a second storage pool is required, another storage backend must be configured giving the details of the second storage pool. The second storage pool must be added to the Manila user using the appropriate command.

Figure 21-2 OpenStack Manila architecture



Creating an OpenStack Manila share type

An OpenStack Manila share type is an administrator-defined type of service that is used by the Manila scheduler to make scheduling decisions. OpenStack tenants can list share types and then use them to create new shares.

To create an OpenStack Manila share type

- On the OpenStack controller node, create a share type for `isa-backend1` and `isa_backend2`.

```
manila@C4110-R720xd-111045:~/OpenStack$ manila type-create isa-backend1
False
```

To associate the share type to a share backend

- On the OpenStack controller node, associate the share type to a share backend.

```
manila@C4110-R720xd-111045:~/OpenStack$ manila type-key isa-backend1 set
driver_handles_share_servers=false share_backend_name=isa-share1
manila@C4110-R720xd-111045:~/OpenStack$ manila type-key isa-backend2
set driver_handles_share_servers=false share_backend_name=isa-share2
```

Creating an OpenStack Manila file share

An OpenStack Manila file share is equivalent to a file system in Veritas Access. You can create an OpenStack Manila file share on the OpenStack controller node.

Note: Due to a REST limitation, you cannot create or delete more than five file systems in parallel using the OpenStack Manila GUI.

To create an OpenStack Manila file share on the OpenStack controller node

1 On the OpenStack controller node, if you wanted to create two OpenStack Manila file shares called `prod_fs` and `finance_fs` of size 1 GB accessible over NFS, enter the following:

One of the file shares resides on `isa_backend1`, and one of the file shares resides on `isa-backend2`.

```
manila@C4110-R720xd-111045:~/OpenStack$ manila create --name prod_fs
--share-type isa-backend1 NFS 1

manila@C4110-R720xd-111045:~/OpenStack$ manila create --name finance_fs
--share-type isa-backend2 NFS 1
```

Use the `manila list` command to see how the file shares look on the OpenStack controller node.

You can see how the file systems look on Veritas Access as part of the share creation process.

```
Storage> fs list
FS                  STATUS SIZE  LAYOUT MIRRORS   COLUMNS USE%    NFS  SHARED CIFS SHARED FTP  SHARED SECONDARY TIER POOL LIST
=================== ====== ====  ====== ========= ======= ======= ==== ====== ==== ====== ==== ====== ============== =========
CB94D066-61977D6A online 1.00G simple -         -       6%      no   no     no   no                                 pool2
C491EFE6-5A0484C0 online 1.00G simple -         -       6%      no   no     no   no                                 pool1
```

2 Give `prod_fs` read-write access to 10.182.111.84.

```
manila@C4110-R720xd-111045:~/OpenStack$ manila access-allow --access-level rw
ecba1f14-86b0-4460-a286-a7e938162fb4 ip 10.182.111.84

+--------------+--------------------------------------+
| Property     |  Value                               |
+--------------+--------------------------------------+
| share_id     | ecba1f14-86b0-4460-a286-a7e938162fb4 |
| deleted      | False                                |
| created_at   | 2015-04-28T17:59:45.514849           |
| updated_at   | None                                 |
| access_type  | ip                                   |
| access_to    | 10.182.111.84                        |
| access_level | rw                                   |
| state        | new                                  |
| deleted_at   | None                                 |
| id           | 8a1c2d0b-a3fc-4405-a8eb-939adb8799db |
+--------------+--------------------------------------+
```

In the `manila access-allow` command, you can get the ID (ecba1f14-86b0-4460-a286-a7e938162fb4) from the output of the `manila list` command.

3 Give `finance_fs` read-write access to 10.182.111.81.

```
manila@C4110-R720xd-111045:~/OpenStack$ manila access-allow --access-level rw
f8da8ff6-15e6-4e0c-814b-d6ba8d08543c ip 10.182.111.81
+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| share_id     | f8da8ff6-15e6-4e0c-814b-d6ba8d08543c |
| deleted      | False                                |
| created_at   | 2015-04-28T18:01:49.557300           |
| updated_at   | None                                 |
| access_type  | ip                                   |
| access_to    | 10.182.111.81                        |
| access_level | rw                                   |
| state        | new                                  |
| deleted_at   | None                                 |
| id           | ddcfc2d2-7e71-443a-bd94-81ad05458e32 |
+--------------+--------------------------------------+

manila@C4110-R720xd-111045:~/OpenStack$ manila access-list
ecba1f14-86b0-4460-a286-a7e938162fb4root@finance:~$
```

Creating a new share backend on the OpenStack controller node

A backend is an instance of the OpenStack Manila share service, which is defined in a section of the `manila.conf` file. Each backend has exactly one driver.

To create a new share backend `isa-share1` in OpenStack Manila, make the following changes on the OpenStack controller node, and restart the OpenStack Manila driver.

To create a new share backend on the OpenStack controller node

1 On the OpenStack controller node, add the following configuration entries in the OpenStack Manila `/etc/manila/manila.conf` file.

- In the DEFAULT section, add the following:

```
#####
enabled_share_backends=isa-share1
#####
```

If the entry `generic1` is already there, add the `isa-share1` entry after a comma. For example:

```
enabled_share_backends = generic1,isa-share1
```

- At the end of all sections in the `/etc/manila/manila.conf` file, add the following configuration entries:

```
#####
[isa-share1]
share_driver= manila.share.drivers.veritas_isa.VeritasShareDriver
driver_handles_share_servers = False
share_backend_name = isa-share1
isa_server_ip = 10.182.96.179
isa_port = 8088
isa_ssl = False
isa_fstype = simple
isa_user = manila
isa_pwd = password
isa_pool = pool1
#####
```

The following table describes the options.

`share_backend_name`	Name of the share backend. This name can be different for each share backend.

`share_driver`	OpenStack Manila driver name.
`isa_server_ip`	Console IP address of the Veritas Access cluster.
`isa_port`	8088 The port on Veritas Access to which the Manila driver is connected.
`isa_ssl`	SSL certificate on the REST server.
`isa_fstype`	Type of file system to be created on the specified pool. It can be `simple`.
`isa_user`	REST user name.
`isa_pwd`	REST password.
`isa_pool`	Existing storage pool on Veritas Access from which the file systems are to be created.

You use the `OPENSTACK> manila configure` command to display the configuration options that need to be performed on the OpenStack controller node.

2 Restart the OpenStack Manila services.

The restart is on the OpenStack controller node, not on Veritas Access.

Creating an OpenStack Manila share snapshot

You can create an OpenStack Manila share snapshot, which is equivalent to creating a snapshot (checkpoint) in Veritas Access. Creating an OpenStack Manila share snapshot creates a checkpoint of the specific file system on Veritas Access. The checkpoint that is created is non-removable.

Deleting a snapshot deletes the checkpoint of that file system.

To create an OpenStack Manila share snapshot

- On the OpenStack controller node, if you want to create `fin_snap` and `prod_snap` snapshots, enter the following:

```
manila@C4110-R720xd-111045:~/OpenStack$ manila snapshot-create --name fin_snap
d3ab5cdc-4300-4f85-b4a5-e2a55d835031

manila@C4110-R720xd-111045:~/OpenStack$ manila snapshot-create --name prod_snap
2269b813-0031-419e-a2d3-0073cdb2776e
```

Use the `manila snapshot-list` command to display the snapshots you created.

Section 11

Managing Veritas Access storage services

Chapter 22

Deduplicating data

This chapter includes the following topics:

About data deduplication

Data deduplication is the process by which redundant data is eliminated to improve storage utilization. Using data deduplication, you can reduce the amount of storage required for storing user and application data. It is most effective in use-cases where many copies of very similar or even identical copies of data are stored. The deduplication feature in Veritas Access provides storage optimization for primary storage (storage of active data).

Each file in the configured file system is broken into user-configurable chunks for evaluating duplicates. The smaller the chunk size, the higher the percentage of sharing due to better chances of matches.

The first deduplication of a file system is always a full deduplication of the entire file system. This is an end-to-end deduplication process that identifies and eliminates duplicate data. Any subsequent attempt to run deduplication on that file system results in incremental deduplication.

Note: Deduplication with a small chunk size increases the deduplication time and load on the system.

Veritas Access deduplication is periodic, that is, as per the user-configured frequency, redundant data in the file system is detected and eliminated.

Use cases for deduplication

The following are potential use cases for Veritas Access file system deduplication:

- Microsoft Exchange mailboxes
- File systems hosting user home directories
- Virtual Machine Disk Format (VMDK) or virtual image stores.

Relationship between physical and logical data on a file system

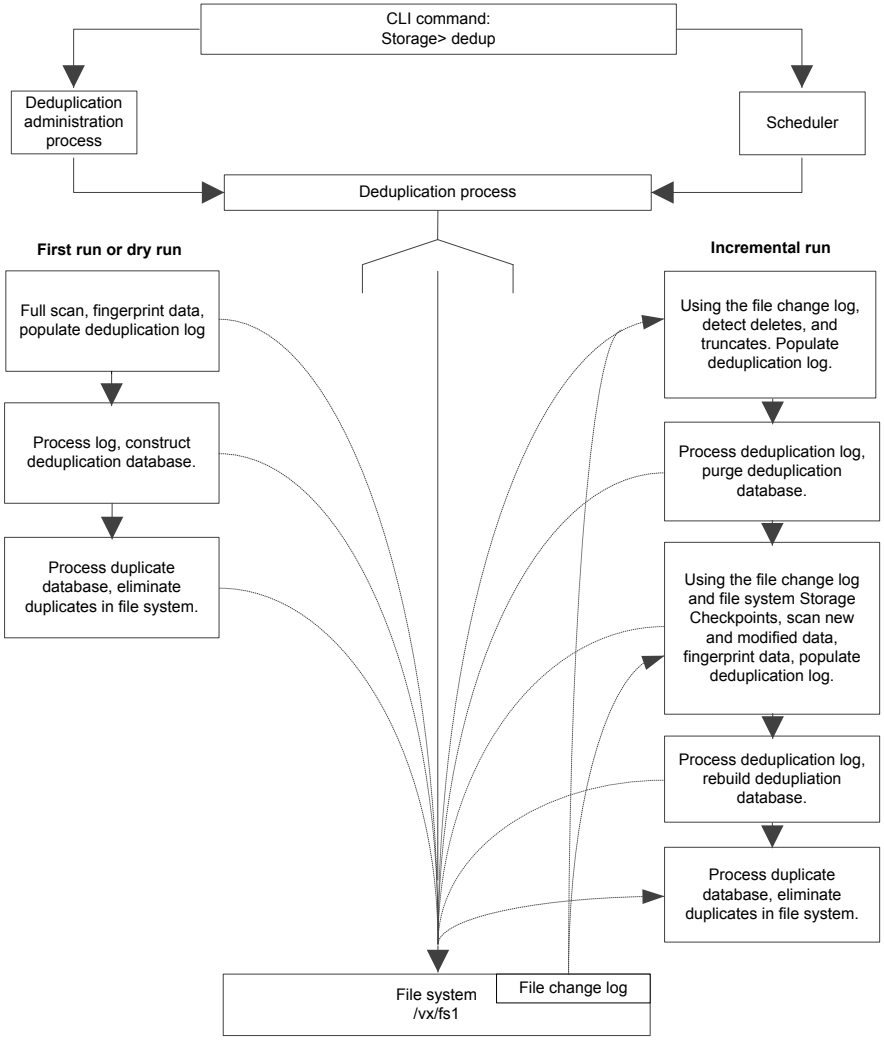
Table 22-1 shows an estimated file system data size that can be supported for a Veritas Access deduplicated file system.

Table 22-1 Relationship between physical and logical data on a file system for two billion unique fingerprints with various deduplication ratios

Fingerprint block size	Deduplication ratio	Unique signature per TB	Physical file system data size	Effective logical file system data size
4 K	50%	128 M	16 TB	32 TB
4 K	65%	90 M	23 TB	65 TB
4 K	80%	51 M	40 TB	200 TB
8 K	50%	64 M	32 TB	64 TB
8 K	65%	45 M	46 TB	132 TB
8 K	80%	25 M	80 TB	400 TB
16 K	50%	32 M	64 TB	128 TB
16 K	65%	22 M	93 TB	266 TB
16 K	80 %	13 M	158 TB	800 TB

Overview of the deduplication workflow

Figure 22-1 Overview of the deduplication workflow



The `Storage> dedup` commands perform administrative functions for the Veritas Access deduplication feature. The deduplication commands allow you to enable, disable, start, stop, and remove deduplication on a file system. You can also reset several deduplication configuration parameters and display the current deduplication status for your file system.

Note: Some configuration parameters can be set as local (specific to a file system) and or global (applicable to all deduplication-enabled file systems). Local parameters override the value of a global parameter.

Best practices for using the Veritas Access deduplication feature

The following are best practices when using the Veritas Access deduplication feature:

- Deduplication is most effective when the file system block size and the deduplication block size are the same for file systems with block sizes of 4K and above. This also allows the deduplication process to estimate space savings more accurately.
- The smaller the file system block size and the deduplication block size, the higher is the time required for performing deduplication. Smaller block sizes such as 1 KB and 2 KB, increase the number of data fingerprints that the deduplication database has to store.
  Though the file system block size is data-dependent, the recommended block size for optimal deduplication is 4 KB for file systems less than 1 TB. For file systems 1 TB and above, it is 8 KB.
- For VMware NFS datastores that store Virtual Machine Disk Format (VMDK) images, a 4 KB block size is optimal.
- Compressed media files for images, music, and video, like JPEG, mp3, .MOV, and databases do not deduplicate or compress effectively.
- Home directory file systems are good candidates for deduplication.
- Deduplication is a CPU and I/O intensive process. It is a best practice to schedule deduplication when the load on your file systems is expected to be low.
- Evaluation of changes in the file system is done by the file system's File Change Log (FCL). Setting the frequency on a too infrequent basis may cause the FCL to rollover, thereby missing changes and deduplication opportunities to the file system.
- After enabling deduplication on file systems with existing data, the first deduplication run does a full deduplication. This can be time-consuming, and may take 12 to 15 hours per TB, so plan accordingly.
- The deduplication database takes up 1% to 7% of logical file system data. In addition, during deduplication processing, an additional but temporary storage space is required. Though 15% free space is enforced, it is recommended to

have 30% free space when the deduplication block size is less than 4096 (4 KB).

- If you plan to use the deduplication scheduler, you must have a Network Time Protocol (NTP) server enabled and configured.

Chapter 23

Compressing files

This chapter includes the following topics:

About compressing files

Compressing files reduces the space used, while retaining the accessibility of the files and being transparent to applications. Compressed files look and behave almost exactly like uncompressed files: the compressed files have the same name, and can be read and written as with uncompressed files. Reads cause data to be uncompressed in memory, only; the on-disk copy of the file remains compressed. In contrast, after a write, the new data is uncompressed on disk.

Only user data is compressible. You cannot compress Veritas File System (VxFS) metadata.

After you compress a file, the inode number does not change, and file descriptors opened before the compressions are still valid after the compression.

Compression is a property of a file. Thus, if you compress all files in a directory, for example, any files that you later copy into that directory do not automatically get compressed. You can compress the new files at any time by compressing the files in the directory again.

You compress files with the `Storage> compress` command.

See the `storage_compress`(1) manual page.

To compress files, you must have VxFS file systems with disk layout Version 8 or later.

Note: When you back up compressed files to tape, the backup program stores the data in an uncompressed format. The files are uncompressed in memory and subsequently written to the tape. This results in increased CPU and memory usage when you back up compressed files.

About the compressed file format

A compressed file is a file with compressed extents. A `compress` call compresses all extents of a file. However, writes to the file cause the affected extents to get uncompressed; the result can be files with both compressed and uncompressed extents.

About the file compression attributes

When you compress a file with the `Storage> compress` command, `compress` attaches the following information to the inode:

- Compression algorithm
- Compression strength, which is a number from 1 to 9
- Compression block size

This information is referred to as the file compression attributes. The purpose of the attributes are to collect the parameters used to create the compressed file. The information can then be read by a backup program.

The file compression attributes guarantee that a particular compressed file can only use one type and strength of compression. Recompressing a file using different attributes fails. To change the file compression attributes, you must explicitly uncompress first, and then recompress with the new options, even in the case where all extents are already uncompressed.

The file compression attributes do not indicate if all extents are compressed. Some extents might be incompressible, and other extents or even all extents might be uncompressed due to writes, but the file compression attributes remain. Only an explicit file uncompression can remove the attributes.

About the file compression block size

The file compression algorithm compresses data in the specified block size, which defaults to 1MB. Each compression block has its own extent descriptor in the inode. If the file or the last extent is smaller than the compression block size, then that smaller size gets compressed. The maximum block size is 1MB.

Extents with data that cannot be compressed are still marked as compressed extents. Even though such extents cannot be compressed, marking these extents as compressed allows successive compression runs to skip these extents to save time. Shared extents cannot be compressed and do not get marked as compressed. Since the file compression algorithm looks at fixed-size blocks, the algorithm finds these incompressible extents in units of the file compression block size.

Best practices for using compression

Best practices for using compression:

- Schedule compression during non-peak hours.

Use cases for compressing files

The following list contains common use case categories:

- If files are old and not accessed frequently. For example:
  - Compress database archive logs which are older than 8 days.
  - Compress jpeg files which are not accessed in 30 days.

Chapter 24

Configuring SmartTier

This chapter includes the following topics:

About Veritas Access SmartTier

The Veritas Access SmartTier feature makes it possible to allocate two tiers of storage to a file system.

The following features are part of the Veritas Access SmartTier solution:

- Relocate files between primary and secondary tiers automatically as files age and become less business critical.
- Prune files on secondary tiers automatically as files age and are no longer needed.
- Promote files from a secondary storage tier to a primary storage tier based on I/O temperature.
- Retain original file access paths to eliminate operational disruption, for applications, backup procedures, and other custom scripts.
- Let you manually move folders, files and other data between storage tiers.
- Enforce the policies that automatically scan the file system and relocate files that match the appropriate tiering policy.

In Veritas Access, there are two predefined tiers for storage:

- Current active tier 1 (primary) storage.
- Tier 2 (secondary) storage for aged or older data.

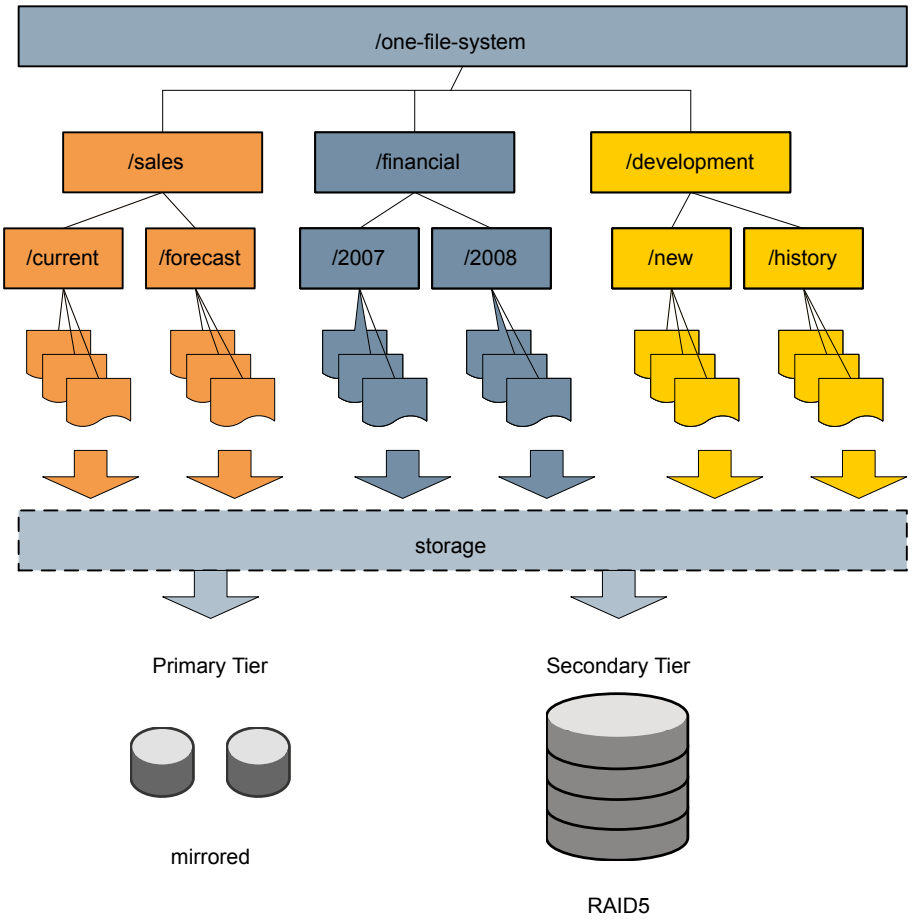
To configure Veritas Access SmartTier, add tier 2 (secondary) storage to the configuration. Specify where the archival storage resides (storage pool) and the total size.

Files can be moved from the active storage after they have aged for a specified number of days, depending on the policy selected. The number of days for files to age (not accessed) before relocation can be changed at any time.

Note: An aged file is a file that exists without being accessed.

Figure 24-1 depicts the features of Veritas Access and how it maintains application transparency.

Figure 24-1 SmartTier features



If you are familiar with Veritas Volume Manager (VxVM), every Veritas Access file system is a multi-volume file system (one file system resides on two volumes). The SmartTier tiers are predefined to simplify the interface. When an administrator wants to add storage tiering, a second volume is added to the volume set, and the existing file system is encapsulated around all of the volumes in the file system.

How Veritas Access uses SmartTier

Veritas Access provides two types of tiers:

- Primary tier

- Secondary tier

Each newly created file system has only one primary tier initially. This tier cannot be removed.

For example, the following operations are applied to the primary tier:

```
Storage> fs addmirror
```

```
Storage> fs growto
```

```
Storage> fs shrinkto
```

The `Storage> tier` commands manage file system tiers.

All `Storage> tier` commands take a file system name as an argument and perform operations on the combined construct of that file system.

The Veritas Access file system default is to have a single storage tier. An additional storage tier can be added to enable storage tiering. A file system can only support a maximum of two storage tiers.

`Storage> tier` commands can be used to perform the following:

- Adding/removing/modifying the secondary tier
- Setting policies
- Scheduling policies
- Locating tier locations of files
- Listing the files that are located on the primary or the secondary tier
- Moving files from the secondary tier to the primary tier
- Allowing metadata information on the file system to be written on the secondary tier
- Restricting metadata information to the primary tier only

About tiering policies

Each tier can be assigned a policy.

The tiering policies include:

- Specify on which tier (primary or secondary) the new files get created.
- Relocate files from the primary tier to the secondary tier based on any number of days of inactivity of a file.
- Relocate files from the secondary tier to the primary tier based on the access temperature of the file.

- Prune files on the secondary tier based on the number of days of inactivity of a file.

About configuring the policy of each tiered file system

You can configure the policy of each tiered file system.

Table 24-1 Tier policy commands

Command	Definition
tier policy list	Displays the policy for each tiered file system. You can have one policy for each tiered file system.
tier policy modify	Modifies the policy of a tiered file system. The new files are created on the primary tier. If a file has not been accessed for more than seven days, the files are moved from the primary tier to the secondary tier. If the access temperature is more than five for of the files in the secondary tier, these files are moved from the secondary tier to the primary tier. The access temperature is calculated over a three-day period.
tier policy prune	Specifies the prune policy of a tiered file system. Once files have aged on the secondary tier, the prune policy can be set up to delete those aged files automatically. The sub-commands under this command are: ■ `tier policy prune list` ■ `tier policy prune modify` ■ `tier policy prune remove`
tier policy run	Runs the policy of a tiered file system.
tier policy remove	Removes the policy of a tiered file system.

Best practices for setting relocation policies

Consider the following relocation policy. The following clauses for relocating the files are:

- Clause 1: If the files on the primary tier are not accessed for 10 days, relocate the files to the secondary tier.
- Clause 2: If the Access Temperature of the files on the secondary tier is more than 100 in the last 15 days, then relocate the files to the primary tier.

```
Storage> tier policy list
FS                        Create on   Days  MinAccess Temp  PERIOD
========================= =========   ====  ==============  ======
non_pgr                   primary      10        100         15
```

Setting such policies where the "PERIOD" is more than the "Days" may result in files moving between the tiers after running the `Storage> tier policy run` command. For example, if a file `a.txt` was being used a lot between 1-5 days, and the number of inputs/outputs rises to 3000. After the fifth day, the file was not used for 10 days and the `Storage> tier policy run` command was issued. The file `a.txt` has the Access Temp as 3000/15, which is equal to 200. As the file has not been used in the last ten days, the file is moved to the secondary tier. If the `Storage> tier policy run` command is run again, the file moves to the primary tier, as the Min Access Temperature is more than 100.

A best practice is to keep the period for calculating the Minimum Access Temperature to lower than the number of days for checking the access age.

Chapter 25

Configuring SmartIO

This chapter includes the following topics:

About SmartIO for solid-state drives

Solid-state drives (SSDs) are devices that do not have spinning disks. Today's solid-state technologies, such as DRAM and NAND flash, provide faster data access, are more efficient, and have a smaller footprint than traditional spinning disks. The data center uses solid-state technologies in many form factors: in-server, all flash arrays, all flash appliances, and mixed with traditional HDD arrays. Each form factor offers a different value proposition. SSDs also have many connectivity types: PCIe, FC, SATA, and SAS.

Due to the current cost per gigabyte of SSD devices, the best value of SSDs is not as high capacity storage devices. The benefit of adopting SSDs is to improve performance and reduce the cost per I/O per second (IOPS). Data efficiency and placement is critical to maximizing the returns on any data center's investment in solid state.

The SmartIO feature of Veritas Access enables data efficiency on your SSDs through I/O caching. Using SmartIO to improve efficiency, you can optimize the cost per IOPS. SmartIO does not require in-depth knowledge of the hardware technologies underneath. SmartIO uses advanced, customizable heuristics to determine what

data to cache and how that data gets removed from the cache. The heuristics take advantage of Veritas Access' knowledge of the characteristics of the workload.

SmartIO uses a cache area on the target device or devices. The cache area is the storage space that SmartIO uses to store the cached data and the metadata about the cached data. To start using SmartIO, you can create a cache area with a single command, while the application is online.

When the application issues an I/O request, SmartIO checks to see if the I/O can be serviced from the cache. As applications access data from the underlying volumes or file systems, certain data is moved to the cache based on the internal heuristics. Subsequent I/Os are processed from the cache.

SmartIO supports read and write caching for the VxFS file systems that are mounted on VxVM volumes, in several caching modes and configurations.

See "About SmartIO read caching for applications running on Veritas Access file systems" on page 176.

See "About SmartIO writeback caching for applications running on Veritas Access file systems " on page 177.

About configuring SmartIO

The `SmartIO` commands control the SmartIO caching functionality of the Veritas Access software.

About SmartIO read caching for applications running on Veritas Access file systems

Veritas Access supports read caching on solid-state drives (SSDs) for applications running on Veritas Access file systems. In this scenario, application reads are satisfied from the cache whenever possible. As the application accesses the file system, the file system loads data from the disk into the cache. Application writes go to the disk in the usual way. With each write, the file system synchronizes the cache to ensure that applications never see stale data. If a cache device fails, a file that is cached in read mode is completely present on the disk. Therefore, the cache failure does not affect the application I/Os for the file and the application I/Os continue without interruption.

By default, the cache area is enabled for caching. All file systems on the system are cached unless you explicitly disable caching for that file system. You do not need to explicitly enable caching on a file system.

About SmartIO writeback caching for applications running on Veritas Access file systems

Veritas Access supports writeback caching on solid-state drives (SSDs) for applications running on Veritas Access file systems. In this scenario, application reads and writes are satisfied from the cache whenever possible.

SmartIO provides write caching in the writeback mode. In writeback mode, an application write returns success after the data is written to the SmartIO cache, which is usually on an SSD. At a later time, SmartIO flushes the cache, which writes the dirty data to the disk. Writeback caching expects to improve the latencies of synchronous user data writes. Write order fidelity is not guaranteed while flushing the dirty data to the disk.

Writeback caching is superset of read caching. When writeback caching is enabled, read caching is implicitly enabled. Reads are satisfied from the cache if possible, and the file system transparently loads file data into the cache. Both read and writeback caching may be enabled for the same file at the same time.

The writeback caching mode gives good performance for writes, but also means that the disk copy may not always be up to date. If a cache device fails, a file that is cached in writeback mode may not be completely present on the disk. SmartIO has a mechanism to flush the data from the cache device when the device comes back online. Veritas Access provides additional protection from data loss with cache reflection. Cache reflection is enabled by default.

Writeback caching requires a cluster with exactly two nodes. Writeback caching cannot be enabled if the cluster has more than two nodes or if the cluster has a single node.

When writeback caching is enabled, SmartIO mirrors the writeback data at the file system level to the other node's SSD cache. This behavior, called cache reflection, prevents loss of writeback data if a node fails. If a node fails, the other node flushes the mirrored dirty data of the lost node as part of reconfiguration. Cache reflection ensures that writeback data is not lost even if a node fails with pending dirty data.

After writeback caching is enabled on the mount point, the qualified synchronous writes in that file system are cached. SmartIO determines if a write qualifies for writeback caching, using criteria such as the following:

- The write request must be PAGESIZE aligned (multiple of 4 KB).
- The write request is not greater than 2 MB.
- The file on which the writes are happening is not mapped.
- Writeback caching is not explicitly disabled by the administrator.

- Writeback caching is not qualified if the cluster has more than two nodes.

You can also customize which data is cached, by adding advisory information to assist the SmartIO feature in making those determinations.

Chapter 26

Configuring replication

This chapter includes the following topics:

About Veritas Access file-level replication

The Veritas Access Replication solution provides high performance, scalable (one-to-many) data replication and is ideal for use as a content distribution solution, and for use to create hot standby copies of important data sets.

Veritas Access Replication lets you asynchronously replicate a file system from one node in a source cluster to another node in a destination cluster at regularly timed intervals. This allows for content sharing, replication, and distribution.

The Veritas Access Replication functionality allows episodic replication with a minimum timed interval update of 15 minutes and no set maximum. Unlike many replication solutions, Veritas Access Replication also allows the destination file system to be online for reads while replication is active.

Major features of Veritas Access Replication include:

- Online access (read-only) to replicated data.
- Immediate read/write access to destination replicated data in the unlikely event that the source file system goes offline for a sustained period of time.
- Load balancing across replication links.
- Transport failover of replication service from one node to another.
- Unlimited simultaneous replication operations.

The Veritas Access Replication feature is designed to copy file systems only between Veritas Access clusters.

Note: The Veritas Access Replication feature does not support user modifications to the target file system if replication is configured.

You can manage bandwidth during replication by actively manage bandwidth using the bandwidth limit throttle.

Figure 26-1 describes the workflow for configuring replication between two Veritas Access clusters.

Figure 26-1 Replication workflow

How Veritas Access Replication works

Veritas Access Replication is an incremental file-level replication service that runs on top of the Cluster File System that is used by Veritas Access which is, in turn, based on the Veritas File System (VxFS). Veritas Access Replication uses two file system specific features: File Change Log (FCL) and Storage Checkpoint services, to retrieve file changes between replication periods.

For a given period, the FCL records every change that is made to the file system. By scanning the FCL, Veritas Access Replication quickly identifies the file(s) that have changed and generates the modified file list. This list prevents the expensive file system scanning that is normally associated with file-based replication, and which typically results in sub-optimal performance.

Next, Veritas Access Replication uses VxFS Storage Checkpoint's metadata comparison feature to retrieve the modified extent list of each changed file. It does not need to access the file data. Veritas Access Replication replicates files data,

metadata, access control lists (ACLs), extended attributes, and modification timestamp changes. It does not replicate any changes made to access time and change time.

Chapter 27

Using snapshots

This chapter includes the following topics:

About snapshots

A snapshot is a virtual image of the entire file system. You can create snapshots of a parent file system on demand. Physically, it contains only data that corresponds to the changes that are made in the parent, and so consumes significantly less space than a detachable full mirror.

Note: You cannot create a snapshot of a scale-out file system.

See "About scale-out file systems" on page 111.

Snapshots are used to recover from data corruption. If files, or an entire file system, are deleted or become corrupted, you can replace them from the latest uncorrupted snapshot. You can mount a snapshot and export it as if it were a complete file system. Users can then recover their own deleted or corrupted files. You can limit the space snapshots consume by setting a quota on them. If the total space that snapshots consume exceeds the quota, Veritas Access rejects attempts to create additional ones.

You can create a snapshot by either using the `snapshot create` command or by creating a schedule to create the snapshot at a specified time.

About snapshot schedules

The `Storage> snapshot schedule` commands let you automatically create or remove snapshots for a file system at a specified time. The schedule indicates the time for the snapshot operation as values for minutes, hour, day-of-the-month, month, and day-of-the-week. The schedule stores these values in the crontab along with the name of the file system.

For example, `snapshot schedule create schedule1 fs1 30 2 * * *` automatically creates a snapshot every day at 2:30 AM, and does not create snapshots every two and a half hours. If you wanted to create a snapshot every two and a half hours with at most 50 snapshots per schedule name, then run `snapshot schedule create schedule1 fs1 50 */30 */2 * * *`, where the value `*/2` implies that the schedule runs every two hours. You can also specify a step value for the other parameters, such as day-of-month or month and day-of-week as well, and you can use a range along with a step value. Specifying a range in addition to the `numeric_value` implies the number of times the crontab skips for a given parameter.

Automated snapshots are named with the schedule name and a time stamp corresponding to their time of creation. For example, if a snapshot is created using the name `schedule1` on February 27, 2016 at 11:00 AM, the name is: `schedule1_Feb_27_2016_11_00_01_IST`.

Note: If the master node is being rebooted, snapshot schedules will be missed if scheduled during the reboot of the master node.

Chapter 28

Using instant rollbacks

This chapter includes the following topics:

About instant rollbacks

Instant rollbacks are volume-level snapshots. All rollback commands take a file system name as an argument and perform operations on the underlying volume of that file system.

Note: If you plan to add a tier to the file system, add the tier first and then create the rollback. If you add the tier after a rollback exists, the rollback hierarchy would have inconsistencies because the rollback is not aware of the tier.

Both space-optimized and full-sized rollbacks are supported by Veritas Access. Space-optimized rollbacks use a storage cache, and do not need a complete copy of the original volume's storage space. However, space-optimized rollbacks are not suitable for write-intensive volumes, because the copy-on-write mechanism may degrade the performance of the volume. Full-sized rollbacks use more storage, but that has little impact on write performance after synchronization is completed.

Both space-optimized rollbacks and full-sized rollbacks can be used instantly after operations such as create, restore, or refresh.

Note: When instant rollbacks exist for a volume, you cannot disable the FastResync option for a file system.

When creating instant rollbacks for volumes bigger than 1T, there may be error messages such as the following:

```
ACCESS instant_snapshot ERROR V-288-1487 Volume prepare for full-fs1-1
failed.
```

An error message may occur because the default amount of memory allocated for a Data Change Object (DCO) may not be large enough for such big volumes. You can use the `vxtune` command to change the value. The default value is 6M, which is the memory required for a 1T volume.

To change it to 15M, use the following command:

```
vxtune volpagemod_max_memsz `expr 15 \* 1024 \* 1024`
```

Chapter 29

Configuring Veritas Access with the NetBackup client

This chapter includes the following topics:

- Configuring backup operations using NetBackup or other third-party backup applications
- Performing a backup or restore of a Veritas Access file system over a NetBackup SAN client
- Performing a backup or restore of a snapshot
- Installing or uninstalling the NetBackup client

About Veritas Access as a NetBackup client

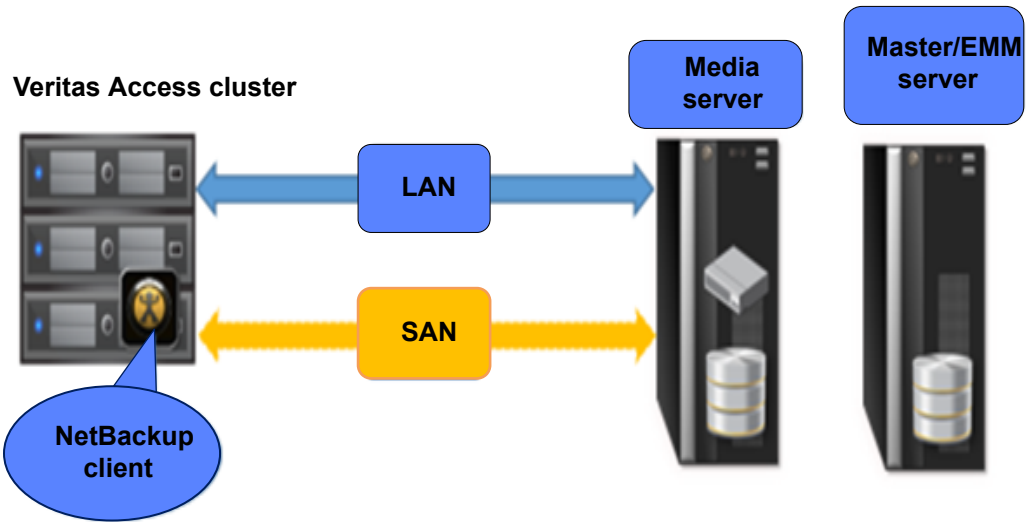
Veritas Access is integrated with Veritas NetBackup so that a NetBackup administrator can back up your Veritas Access file systems to NetBackup master or media servers and retain the data as per your company policy. Once data is backed up, a storage administrator can delete unwanted data from Veritas Access. The NetBackup master and media servers that run on separate computers from Veritas Access are licensed separately from Veritas Access.

You configure NetBackup domain information using any one of the following Veritas Access interfaces:

- CLISH
  The Veritas Access CLISH has a dedicated `Backup>` menu. From the `Backup>` menu, register the NetBackup client with the NetBackup domain. Information is saved in the `bp.conf` file on Veritas Access.
- GUI
  Settings > NetBackup Configuration
  See the online Help for how to configure NetBackup using the GUI.
- RESTful APIs
  See the *Veritas Access RESTful API Guide*.

Consolidating storage reduces the administrative overhead of backing up and restoring many separate file systems. Critical file data can be backed up and restored through the NetBackup client on Veritas Access.

Figure 29-1 Configuration of Veritas Access with NetBackup



See "Prerequisites for configuring the NetBackup client" on page 188.

See "Workflow for configuring Veritas Access for NetBackup" on page 190.

Prerequisites for configuring the NetBackup client

Before configuring the NetBackup client for Veritas Access, you must have completed the following:

- You must have a NetBackup master server external to your Veritas Access cluster. The NetBackup administrator configures the NetBackup master server. See the NetBackup product documentation for more information.
- Add the valid licenses on the NetBackup master server.
- Make sure that the Veritas Access server and the NetBackup server can resolve the host name.

About the NetBackup Snapshot Client

A snapshot is a point-in-time, read-only, disk-based copy of a client volume. After the snapshot is created, NetBackup backs up data from the snapshot, not directly from the client's primary or original volume. Users and client operations can access the primary data without interruption while data on the snapshot volume is backed up. The contents of the snapshot volume are cataloged as if the backup was

produced directly from the primary volume. After the backup is complete, the snapshot-based backup image on storage media is indistinguishable from a traditional, non-snapshot backup.

About NetBackup snapshot methods

NetBackup can create different types of snapshots. Each snapshot type that you configure in NetBackup is called a snapshot method. Snapshot methods enable NetBackup to create snapshots within the storage stack (such as the file system, volume manager, or disk array) where the data resides. If the data resides in a logical volume, NetBackup can use a volume snapshot method to create the snapshot. If the data resides in a file system, NetBackup can use a file system method, depending on the client operating system and the file system type.

You select the snapshot method in the backup policy as explained in the *Veritas NetBackup Snapshot Client Administrator's Guide*.

Note: When using Veritas Access with NetBackup, select the `VxFS_Checkpoint` snapshot method.

About NetBackup instant recovery

This feature makes backups available for quick recovery from disk. Instant recovery combines snapshot technology—the image is created with minimal interruption of user access to data—with the ability to do rapid snapshot-based restores. The snapshot is retained on disk as a full backup image.

Enabling or disabling the NetBackup SAN client

You can enable or disable the NetBackup SAN client on Veritas Access. The NetBackup SAN client should only be enabled on Veritas Access if the required licenses are installed on the NetBackup Master Server. If you do not have the required license for the NetBackup SAN client, then you must disable the SAN client on Veritas Access. Otherwise, the Veritas Access backup service fails to start.

To enable or disable the NetBackup SAN client

- To enable or disable the NetBackup SAN client, enter the following:

```
Backup> netbackup sanclient enable | disable
```

`enable`	Enables the NetBackup SAN client.
`disable`	Disables the NetBackup SAN client.

```
Backup> netbackup sanclient enable
Success.
```

Workflow for configuring Veritas Access for NetBackup

To back up your data with NetBackup, you must register the installed and configured NetBackup master server with Veritas Access.

To configure NetBackup for Veritas Access, perform the following tasks in the order shown:

Make sure that the prerequisites are met.	See "Prerequisites for configuring the NetBackup client" on page 188.
Register the NetBackup master server.	See "Registering a NetBackup master server, an EMM server, or adding an optional media server " on page 191.
Display the current status of the NetBackup client.	See "Displaying the status of NetBackup services" on page 196.
Reset the values for the NetBackup master server or the NetBackup EMM server.	See "Registering a NetBackup master server, an EMM server, or adding an optional media server " on page 191.
Display the current status of the NetBackup client.	See "Displaying the status of NetBackup services" on page 196.
Reset the NetBackup virtual name.	See "Configuring the virtual name of NetBackup" on page 195.
Register the NetBackup master server with the NetBackup client.	See "Registering a NetBackup master server, an EMM server, or adding an optional media server " on page 191.

Configure the virtual name that the NetBackup master server uses for the NetBackup client.	See "Configuring the virtual name of NetBackup" on page 195.
Display the current status of the NetBackup client.	See "Displaying the status of NetBackup services" on page 196.
Verify that Veritas Access is configured with the NetBackup client.	See "Displaying the status of NetBackup services" on page 196.
Configure the `/etc/hosts` file to ping the NetBackup master or media server.	
Specify the files to back up or restore.	See "Performing a backup or restore of a Veritas Access file system over a NetBackup SAN client" on page 199.
Specify the snapshot to back up or restore	See "Performing a backup or restore of a snapshot" on page 200.
Uninstalling or installing the NetBackup client.	See "Installing or uninstalling the NetBackup client" on page 200.
Configuring Veritas Access for NetBackup cloud storage	

Registering a NetBackup master server, an EMM server, or adding an optional media server

You register the NetBackup master server or the EMM server so that it can communicate with Veritas Access. If necessary, you can reset the values of the NetBackup master server or the EMM server to their default configurations. You can optionally add a media server.

The NetBackup EMM server is generally the master server. The NetBackup master server can be the NetBackup media server, but it is not mandatory that the NetBackup master server be the NetBackup media server. In production environments, the NetBackup media server is separate from the NetBackup master server.

See the `backup_netbackup(1)` man page for detailed examples.

To register the NetBackup master server or the NetBackup EMM server with Veritas Access

1. Register the NetBackup master server with Veritas Access.

   For a NetBackup master server:

   ```
   Backup> netbackup master-server set server
   ```

2. Register the NetBackup EMM server with Veritas Access.

   ```
   Backup> netbackup emm-server set server
   ```

To reset the value for the NetBackup master server or the NetBackup EMM server

1. Reset the value for the NetBackup master server.

   For a NetBackup master server:

   ```
   Backup> netbackup master-server reset
   ```

   For a NetBackup EMM server:

2. Reset the value for the NetBackup EMM server.

   ```
   Backup> netbackup emm-server reset
   ```

To add an optional NetBackup media server

- Add an optional NetBackup media server.

  If the NetBackup master server is also acting as a NetBackup media server, then add the NetBackup media server using the NetBackup master server hostname.

  For example:

  ```
  Backup> netbackup media-sever add FQDN of master server
  ```

To delete an already configured NetBackup media server

- Delete an already configured NetBackup media server.

  ```
  Backup> netbackup media-server delete server
  ```

Displaying the excluded files from backup

To display the entries in the excluded list from backup

- Display the entries in the excluded list from backup.

```
Backup> netbackup exclude_list show [policy] [schedule]
```

policy	Lists the excluded entries by specifying a NetBackup policy.
schedule	If a NetBackup policy schedule is specified, then the excluded list entries for the specified NetBackup policy and NetBackup policy schedule are displayed.

```
Backup> netbackup exclude_list show

Pattern          Policy         Schedule
-------          ------         --------
hosts            -              -
iscsid.conf      -              -
iscsid.conf      NBU_access12   -
/vx/fs100/as*    policy         -
/vx/fs100/*mp3   policy         -
/vx/fs200/bs*    policy2        sched
```

The hyphens in the command output indicate that no values have been entered.

Displaying the included and excluded files for backups

You can specify a policy pattern that lets you specify which files to include or exclude from NetBackup backups. For example, you can specify that only `.gif` files are backed up, and `.iso` files are excluded. You can then display those files.

See the `backup_netbackup(1)` man page for detailed examples.

To display files included or excluded for backups

- Display the files that are included or excluded for backups.

  For included files:

  ```
  Backup> netbackup include_list show [policy] [schedule]
  ```

  For excluded files:

  ```
  Backup> netbackup exclude_list show [policy] [schedule]
  ```

Adding or deleting patterns to the list of files in backups

You can add or delete specified patterns to or from the files that you want to include or exclude from NetBackup backups. For example, you can create a backup policy with different patterns such that only `.gif` files are backed up and `.iso` files are excluded.

See the `backup_netbackup(1)` man page for detailed examples.

To add or delete the given pattern to the list of files included for backup

- Add the specified pattern to the files that are included for backup.

  For adding specified patterns to included files:

  ```
  Backup> netbackup include_list add pattern [policy] [schedule]
  ```

  For deleting specified patterns from included files:

  ```
  Backup> netbackup include_list delete pattern [policy] [schedule]
  ```

To add or delete a given pattern to the list of files excluded from backup

- Add a given pattern to the list of files that are excluded from backup.

  For adding a given pattern to excluded files:

  ```
  Backup> netbackup exclude_list add pattern [policy] [schedule]
  ```

  For deleting the given pattern from excluded files:

  ```
  Backup> netbackup exclude_list delete pattern [policy] [schedule]
  ```

Configuring or resetting the virtual IP address used by NetBackup

You can configure or reset the virtual IP address of NetBackup. This address is a highly-available virtual IP address in the cluster.

Note: Configure the virtual IP address using the `Backup> virtual-ip set` command so that it is different from all of the virtual IP addresses, including the console server IP address and the physical IP addresses that are used to install Veritas Access. Use the `Network> ip addr show` command to display the currently assigned virtual IP addresses on Veritas Access.

See the `backup_virtual-ip(1)` man page for detailed examples.

To configure or reset the virtual IP address used by NetBackup

1. Configure the virtual IP address of NetBackup on Veritas Access.

   ```
   Backup> virtual-ip set ipaddr [device]
   ```

2. Reset the virtual IP address of NetBackup on Veritas Access.

   ```
   Backup> virtual-ip reset
   ```

See "Configuring the virtual name of NetBackup" on page 195.

Configuring the virtual name of NetBackup

You can either configure the virtual name for the NetBackup client, or you can reset the value to its default or unconfigured state.

See the `backup_virtual-name(1)` man page for detailed examples.

To set or reset the NetBackup virtual name

- Set or reset the NetBackup virtual name.

  For setting the virtual name:

  ```
  Backup> virtual-name set name
  ```

  For resetting the virtual name:

  ```
  Backup> virtual-name reset
  ```

  Make sure that *name* can be resolved through DNS, and its IP address can be resolved back to *name* through the DNS reverse lookup. Also, make sure that *name* resolves to an IP address that is configured by using the `Backup> virtual-ip` command.

  See "Configuring or resetting the virtual IP address used by NetBackup" on page 195.

Displaying the status of NetBackup services

Use the backup commands to display the status of the NetBackup services.

See the following man pages for detailed examples:

- `backup_show(1)`
- `backup_status(1)`
- `backup_start(1)`
- `backup_stop(1)`

To display NetBackup services

- Display the current NetBackup services.

  ```
  Backup> show
  ```

  Example:

  ```
  Backup> show
  Virtual Name:              nbuclient.veritas.com
  Virtual IP:                10.10.10.10/24
  NetBackup Master Server:   nbumaster.veritas.com
  NetBackup EMM Server:      nbumaster.veritas.com
  NetBackup Media Server(s): nbumaster.veritas.com
  Backup Device:             pubeth1
  ```

  If the settings were configured while the backup and the restore services were online, they may not be in use by Veritas Access. To display all of the configured settings, first run the `Backup> stop` command, then run the `Backup> start` command.

To display the status of backup services

- Display the status of backup services.

  ```
  Backup> status
  ```

  Example:

  ```
  Backup> status
  Virtual IP state:            up
  Backup service online node:  node_01
  NetBackup client state:      online
  NetBackup SAN client state:  online
  No backup or restore jobs running.
  ```

  If the NetBackup server is started and online, then `Backup> status` displays any on-going backup or restore jobs.

To start or stop backup services

1 Start the backup services.

```
Backup> start [nodename]
```

You can also change the status of a virtual IP address to online after it has been configured using the `Backup> virtual-ip` command. This command applies to any currently active node in the cluster that handles backup and restore jobs.

See "Configuring or resetting the virtual IP address used by NetBackup" on page 195.

2 Stop the backup services.

```
Backup> stop
```

You can also change the status of a virtual IP address to offline after it has been configured using the `Backup> virtual-ip` command.

See "Configuring or resetting the virtual IP address used by NetBackup" on page 195.

The `Backup> stop` command does nothing if any backup jobs are online that involve Veritas Access file systems.

Configuring backup operations using NetBackup or other third-party backup applications

You can backup Veritas Access using NetBackup client capability, or backup applications from other third-party companies that use the standard NFS mount to backup over the network.

For information on NetBackup, refer to the NetBackup product documentation set.

The Backup commands configure the local NetBackup installation of Veritas Access to use an external NetBackup master server, Enterprise Media Manager (EMM) server, or media server. When NetBackup is installed on Veritas Access, it acts as a NetBackup client to perform IP-based backups of Veritas Access file systems.

Note: A new public IP address, not an IP address that is currently used, is required for configuring the NetBackup client. Use the `Backup> virtual-ip` and `Backup> virtual-name` commands to configure the NetBackup client.

Performing a backup or restore of a Veritas Access file system over a NetBackup SAN client

You can perform a backup or restore of a Veritas Access file system over a NetBackup SAN client. A NetBackup SAN client is a NetBackup client for which Fibre Transport services are activated.

Backup and restore operations are done on the NetBackup master server by a NetBackup administrator using the NetBackup Administration Console. If the NetBackup master server can connect to the NetBackup client on Veritas Access, the NetBackup master server starts the backup or restore operations.

Before performing a backup or restoration of a Veritas Access file system over a NetBackup SAN client, do the following:

- Verify that the virtual IP address is online.
- Verify that the NetBackup client state is online.
- Configure the Fibre Transport media server.
  See the *Veritas NetBackup SAN client and Fibre Transport Guide* for more information on configuring the NetBackup Fibre Transport media server.

See the `backup(1)` man page for detailed examples.

To perform a backup of a file system over a NetBackup SAN client

1. Check the status of the NetBackup client.

   ```
   Backup> status
   ```

2. Enable the SAN client from the CLI.

   ```
   Backup> netbackup sanclient enable
   ```

3. Verify if the SAN client has been enabled or not from the CLI.

   ```
   Backup> status
   ```

4. Using the NetBackup Administration Console, start a backup operation.

To perform a restore of a file system over a NetBackup SAN client

1. Check the status of the NetBackup client.

   ```
   Backup> status
   ```

2. Using the NetBackup Administration Console, start a restore operation.
3. Check the status of the NetBackup client.

   ```
   Backup> status
   ```

Performing a backup or restore of a snapshot

Using the NetBackup Administration Console, a NetBackup administrator can perform a backup or restore of a snapshot.

Veritas Access as a NetBackup client supports the VxFS_Checkpoint snapshot method. See the *Veritas NetBackup Snapshot Client Administrator's Guide* for more information on configuring snapshot policies.

To perform a backup of a snapshot

- Using the NetBackup Administration Console, start a snapshot backup.

  The snapshot triggered by the NetBackup job can be seen from the CLI.

  ```
  Storage> snapshot list
  ```

To perform a restore of a snapshot

1. Using the NetBackup Administration Console, navigate to **Backup, Archive, and Restore**.
2. Click the **Restore Files** tab.
3. Click the **Restore** option.
4. Specify the directory location for performing the restore operation.

   /vx/*name_of_file_system*/Restores

Installing or uninstalling the NetBackup client

The NetBackup master server version should be higher or equal to the NetBackup client version. To upgrade the NetBackup client, uninstall the currently installed version of the NetBackup client and then install the specified version of the

NetBackup client. The uninstall command runs on all the nodes of the Veritas Access cluster.

Veritas Access supports two major versions of the NetBackup client, version 7.6 and 7.7. By default, Veritas Access comes with the 7.7 version of the NetBackup client.

See the `backup(1)` man page for detailed examples.

To install the NetBackup client

1. Display the currently installed version of the NetBackup client.

```
Backup> show
```

2. Install the specified NetBackup client.

```
Backup> install version [URL]
```

You must specify the version of the NetBackup client that you want to install. If you do not specify a URL, the `Backup> install` command has the information on the file system for the location it needs. Specify the major release version (7.7 or 7.6) as the second parameter. You can specify the NetBackup package minor release or patches (7.7.1 for a 7.7 major release) as the third parameter to install.

If the base NetBackup client version is 7.6.

```
Backup> install 7.6 scp://support@192.168.2.10:/home/NetBackup_7.6_
CLIENTS2.tar.gz
```

If the base NetBackup client version is 7.7.

```
Backup> install 7.7 scp://support@192.168.2.10:/home/NetBackup_7.7_
CLIENTS2.tar.gz
```

If there are minor releases or patches from the NetBackup client.

```
Backup> install 7.6 scp://support@192.168.2.10:/home/NetBackup_7.6.0.1_
CLIENTS2.tar.gz
```

```
Backup> install 7.7 scp://support@192.168.2.10:/home/NetBackup_7.7.1_
CLIENTS2.tar.gz
```

For example, consider that the NetBackup client binaries are placed on the following host:

```
192.168.2.10
```

```
Backup> install 7.6 scp://support@192.168.2.10:/home/NetBackup_7.6.0.1_
CLIENTS2.tar.gz
```

Where 192.168.2.10 is the host IP address on which the NetBackup client packages are placed.

```
NetBackup_7.6.0.1_CLIENTS2.tar.gz
```

This is the NetBackup client package.

3 Double check if the Red Hat compatible NetBackup client is available in this package.

```
support:
system user and specify password when prompted.
```

4 Verify that the specified NetBackup client is installed.

```
Backup> show
```

To uninstall the NetBackup client

1 Display the currently installed version of the NetBackup client.

```
Backup> show
```

2 Uninstall the existing version of the NetBackup client.

```
Backup> uninstall
```

3 Display the current running version of the NetBackup client.

```
Backup> show
```

4 Verify that the NetBackup client is not installed.

```
Backup> show
```

Section 12

Reference

Appendix A

Veritas Access documentation

This appendix includes the following topics:

Using the Veritas Access product documentation

The latest version of the Veritas Access product documentation is available on the Veritas Services and Operations Readiness Tools (SORT) website.

https://sort.veritas.com/documents

You need to specify the product and the platform and apply other filters for finding the appropriate document.

Make sure that you are using the current version of documentation. The document version appears on page 2 of each guide. The publication date appears on the title page of each document. The documents are updated periodically for errors or corrections.

See the *Veritas Access Release Notes* for information on documentation changes in this release.

The following documents are available on the SORT site:

- *Veritas Access Administrator's Guide*
- *Veritas Access Amazon Web Services Cloud Storage Tiering Solutions Guide*
- *Veritas Access on Amazon Web Services Cloud - Deployment Guide*
- *Veritas Access Command Reference Guide*

- *Veritas Access Getting Started Guide*
- *Veritas Access Installation Guide*
- *Veritas Access NetBackup Solutions Guide*
- *Veritas Access Quick Start Guide*
- *Veritas Access Release Notes*
- *Veritas Access RESTful API Guide*
- *Veritas Access Third-Party License Agreements*
- *Veritas Access Troubleshooting Guide*

About accessing the online man pages

You access the online man pages by typing `man name_of_command` at the command line.

The example shows the result of entering the `Network> man ldap` command.

```
Network> man ldap
NAME
       ldap - configure LDAP client for authentication

SYNOPSIS
       ldap enable
       ldap disable
       ldap show [users|groups|netgroups]
       ldap set {server|port|basedn|binddn|ssl|rootbinddn|users-basedn|
            groups-basedn|netgroups-basedn|password-hash} value
       ldap get {server|port|basedn|binddn|ssl|rootbinddn|
            users-basedn|groups-basedn|netgroups-basedn|password-hash}
```

You can also type a question mark (?) at the prompt for a list of all the commands that are available for the command mode that you are in. For example, if you are within the `admin` mode, if you type a question mark (?), you will see a list of the available commands for the `admin` mode.

```
ACCESS> admin ?
Entering admin mode...
ACCESS.Admin>

exit         --return to the previous menus
logout       --logout of the current CLI session
```

```
man           --display on-line reference manuals
passwd        --change the administrator password
show          --show the administrator details
supportuser   --enable or disable the support user
user          --add or delete an administrator
```

To exit the command mode, enter the following: `exit`.

For example:

```
ACCESS.Admin> exit
ACCESS>
```

To exit the system console, enter the following: `logout`.

For example:

```
ACCESS> logout
```

Appendix B

Veritas Access tuning

This appendix includes the following topics:

File system mount-time memory usage

Mounting a file system on a computer system allocates system memory that is not freed until the file system is unmounted. The amount of memory allocated at mount time is directly proportional to the size of the file system being mounted. The amount of memory that is allocated at mount-time is therefore important information to help determine the system memory requirements for a Veritas Access environment. The mount-time memory requirement is different if you expect to mount a total of 1 PB of storage or 2 PBs of storage. The number of files currently in the file system does not affect the amount of memory allocated at mount-time. The amount of memory allocated at mount-time is also inversely proportional to the file system block size.

The information required to determine the amount of memory allocated at mount time is the total size of all the file systems that are mounted on the same computer system at the same time and the block size of each file system.

The amount of memory allocated at mount time can therefore be estimated by obtaining the total size of all the file systems that are mounted on a system according to the file system block size. So four totals in all, one for each file system block size of 1 KB, 2 KB, 4 KB, and 8 KB.

Table B-1 File system mount-time memory usage

File system block size	Total size of mounted file systems	Memory allocation at mount time
1 KB	‘a’ TBs	‘w’MBs allocated per TB
2 KB	‘b’ TBs	‘x’MBs allocated per TB

Table B-1 File system mount-time memory usage *(continued)*

File system block size	Total size of mounted file systems	Memory allocation at mount time
4 KB	‘c’ TBs	‘y’MBs allocated per TB
8 KB	‘d’ TBs	‘z’MBs allocated per TB

The mount-time memory requirement is therefore:

((a*w) + (b*x) + (c*y) + (d*z))

A file system using a 1 KB block size (the smallest file system block size) allocates approximately eight times more memory at mount time than a file system of the same size using a 8 KB block size (the largest file system block size). For this reason, the Veritas Access file system defaults to a block size of 8 KB if a block size is not specified when creating a file system.

Some customers might like to create small file systems using a 1 KB file system block size and subsequently grow the file system size significantly, as the file system block size cannot be changed after the file system is created. This procedure can result in very large file systems using a 1 KB block size that can result in an unexpectedly large allocation of system memory at mount time.

A Clustered File System (CFS) primary mount requires slightly more memory allocated at mount-time than a CFS secondary. The performance team recommends that the memory utilization of a CFS primary be used as the guideline for calculating the file system mount-time memory requirement.

Table B-2 Memory footprint of 16 file systems with 32 TB size each - CFS primary mount

	32 TB each file system			
Block size/file system	CFS primary mount			
	Memory used (MB)			
	1 KB	2 KB	4 KB	8 KB
1	329	164	82	41
2	659	328	165	82
3	988	491	248	125
4	1326	657	337	166
5	1649	821	414	210

Table B-2 Memory footprint of 16 file systems with 32 TB size each - CFS primary mount *(continued)*

	32 TB each file system			
6	1977	985	498	249
7	2306	1150	581	291
8	2635	1329	665	333
9	2964	1483	747	375
10	3293	1646	829	418
11	3624	1810	913	459
12	3953	1975	995	534
13	4281	2140	1077	546
14	4614	2307	1161	589
15	4942	2471	1243	629
16	5272	2636	1325	671

Table B-3 Memory footprint of 16 file systems with 32 TB size each - CFS secondary mount

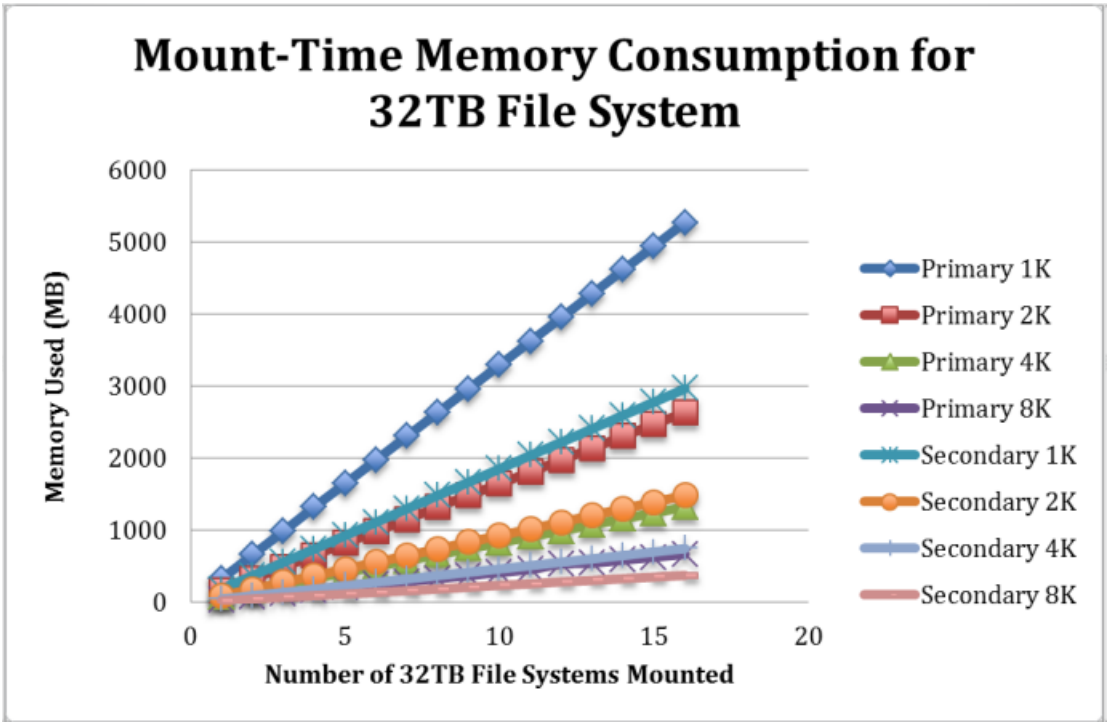
	32 TB each file system			
Block size/file system	CFS secondary mount			
	Memory used (MB)			
	1 KB	2 KB	4 KB	8 KB
1	187	93	47	21
2	372	186	94	48
3	558	279	139	71
4	742	371	186	94
5	929	465	233	117
6	1113	557	280	140
7	1300	650	326	164

Table B-3 Memory footprint of 16 file systems with 32 TB size each - CFS secondary mount *(continued)*

	32 TB each file system			
8	1485	743	373	187
9	1670	837	419	213
10	1854	928	465	237
11	2040	1020	512	259
12	2224	1114	558	286
13	2410	1208	606	306
14	2596	1301	652	330
15	2780	1393	701	353
16	2966	1485	747	376

Figure B-1 provides the guideline for the system memory utilization at mount time.

Figure B-1 Mount-time memory consumption for 32 TB file systems



Index

D

E

F

I

L

M

N

O

P

Q

R

S

T

U

V

W