Data ONTAP™ 7.0
Data Protection
Online Backup and Recovery Guide

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089 USA
Telephone: +1 (408) 822-6000
Fax: +1 (408) 822-4501
Support telephone: +1 (888) 4-NETAPP
Documentation comments: doccomments@netapp.com
Information Web: http://www.netapp.com

Part number 210-00681
October 2004

Copyright and trademark information

Copyright information

Trademark information

Table of Contents

Preface

About this guide

This guide describes how to protect, back up, restore, and copy data between Network Appliance™ filers that run Data ONTAP™ 7.0 software and that are Serving Data by Design™. This guide covers all filer models.

Audience

This guide is for system administrators who are familiar with operating systems that run on the filer's clients, such as UNIX®, Windows 95™, Windows NT®, and Windows® 2000. It also assumes that you are familiar with how to configure the filer and how the NFS, CIFS, and HTTP protocols are used for file sharing or transfers. This guide doesn't cover basic system or network administration topics, such as IP addressing, routing, and network topology; it emphasizes the characteristics of the NetApp® filer.

Terminology

Network Appliance's storage products (filers, FAS appliances, and NearStore systems) are all storage systems—also sometimes called filers or storage appliances.

This guide uses the term "type" to mean pressing one or more keys on the keyboard. It uses the term "enter" to mean pressing one or more keys and then pressing the Enter key.

FilerView as an alternative to commands

Tasks you perform as a filer administrator can be performed by entering commands at the console, in configuration files, or through a Telnet session or Remote Shell connection.

Another method of performing man common filer tasks is to use the FilerView®, graphical management interface for viewing and managing a filer from a Web browser. FilerView is easy to use, and it includes Help that explains FilerView features and how to use them.

For more information about accessing a filer with FilerView, and about FilerView Help, see the *Storage Management Guide*.

Command conventions

You can enter filer commands either on the system console or from any client computer that can access the filer through a Telnet session.

In examples that illustrate commands executed on a UNIX workstation, this guide uses the command syntax of SunOS 4.1.x. The command syntax and output might differ, depending on your version of UNIX.

Keyboard conventions

When describing key combinations, this guide uses the hyphen (-) to separate individual keys. For example, “Ctrl-D” means pressing the “Control” and “D” keys simultaneously. Also, this guide uses the term “Enter” to refer to the key that generates a carriage return, although the key is named “Return” on some keyboards.

Typographic conventions

The following table describes typographic conventions used in this guide.

Convention	**Type of information**
Italic font	Words or characters that require special attention. Placeholders for information you must supply. For example, if the guide says th enter the `arp -d` *hostname* command, you enter the characters “arp -d” followed by the actual name of the host. Book titles in cross-references.
`Monospaced` font	Command and daemon names. Information displayed on the system console or other computer monitors. The contents of files.
`Bold monospaced` font	Words or characters you type. What you type is always shown in lowercase letters, unless you must type it in uppercase letters.

Special messages

This guide contains special messages that are described as follows:

Note

A note contains important information that helps you install or operate the system efficiently.

Caution

A caution contains instructions that you must follow to avoid damage to the equipment, a system crash, or loss of data.

WARNING

A warning contains instructions that you must follow to avoid personal injury.

Introduction to Data Protection 1

About this chapter

This chapter introduces the data backup, restore, and protection capabilities that are available in this release of Data ONTAP.

Topics in this chapter

This chapter discusses the following topics:

Data protection options

About data protection

Data protection means backing up data and being able to recover it. You protect the data by making copies of it so that it is available for restoration even if the original is no longer available.

Why data needs protection

Reasons that businesses need data backup and protection systems include the following:

- To protect data from accidentally deleted files, application crashes, data corruption, and viruses
- To archive data for future use
- To recover from a disaster

Methods of protecting data

Depending on your data protection and backup needs, Data ONTAP offers a variety of features and methods to ensure against accidental, malicious, or disaster-induced loss of filer data. This guide describes how to use Data ONTAP online features like SnapMirror® technology to protect data. See the *Tape Backup and Recovery Guide* for information about using tape to protect data.

Data Protection Feature	**Description**
Snapshot™	Backup within a volume. This feature allows you to manually or automatically create, schedule, and maintain multiple backups (also called snapshots) of data on a volume. Snapshots use only a minimal amount of additional volume space on your filer and do not have a performance cost. Snapshots are also used to create clones of flexible volumes and LUNs. If a user accidentally modifies or deletes crucial data on a volume with Snapshot enabled, that data can be easily and quickly restored from one of the last several snapshots taken. See Chapter 2, “Snapshot Management” on page 17. You can also create clones of flexible volumes and LUNs using snapshots. See the *Storage Management Guide* for more details.
SnapRestore® (license required)	Fast, space efficient restoration of large volumes of data backed up to snapshots. The SnapRestore feature performs on request snapshot recovery from snapshots on an entire volume. See Chapter 3, “Data Recovery Using SnapRestore” on page 57.

Data Protection Feature	Description
SnapMirror® (license required)	Volume-to-volume and qtree-to-qtree replication. This feature enables you to periodically make snapshots of data on one volume or qtree, replicate that data to a partner volume or qtree, usually on another filer, and archive one or more iterations of that data as snapshots. Replication on the partner volume or qtree ensures quick availability and restoration of data, from the point of the last snapshot, should the filer of the original volume or qtree be disabled. If you conduct tape backup and archival operations, you can carry them out on the data already backed to the SnapMirror partner filer, thus freeing the original filer of this time-consuming, performance-degrading chore. See Chapter 4, "Data Protection Using SnapMirror" on page 73.
SnapVault™ (license required)	Centralized backup of multiple qtrees on multiple filers using snapshot technology. This feature enables you to back up qtrees on multiple volumes and filers to a single Network Appliance SnapVault secondary storage system specialized for quick backup and restore of its filer sources. You can also install the Open Systems SnapVault agent on non-filer Windows NT, Windows 2000, Solaris, Linux, AIX, or HP-UX systems. This agent allows SnapVault to back up and restore data to these systems also. If you conduct tape backup and archival operations, you can carry them out on the data already backed up to the SnapVault secondary storage system, thus freeing your filers of this time-consuming, performance-degrading chore. See Chapter 5, "Data Protection Using SnapVault" on page 195.

Data Protection Feature	Description
Tape backup `dump` and `restore` commands	Tape backup and restore. The `dump` and `restore` commands allow you to back up snapshots to tape. The `dump` command takes a snapshot of the volume and then copies that data to tape. Because the snapshot, not the active file system, is backed up to tape, Data ONTAP can continue its normal functions while the tape backup takes place. See the *Tape Backup and Recovery Guide* for more information.
`vol copy`	Fast block-copy of data from one volume to another. The `vol copy` command enables you to quickly block-copy stored filer data from one volume to another. See Chapter 6, “Volume Copy” on page 279.
SyncMirror™ (cluster configuration required)	Continuous mirroring of data to two separate filer volumes. This feature allows you to mirror filer data real-time to matching volumes physically connected to the same filer head. In case of unrecoverable disk error on one volume, the filer automatically switches access to the mirrored volume. Filer cluster configuration is required for this feature. See Chapter 7, “SyncMirror Management” on page 301.
nvfail option to the `vol options` command	Protection against data corruption by failures of non-volatile RAM (NVRAM). See Chapter 8, “Database Protection Using NVFAIL” on page 341.
virus scan support	Support for third-party virus-scanning software for files accessed by CIFS clients. See Chapter 9, “Virus Protection for CIFS” on page 347.

Data Protection Feature	**Description**
MetroCluster	SyncMirror functionality enhanced to provide continuous volume mirroring over 500-meter to 30-kilometer distances. See Appendix A, "Disaster Protection Using MetroCluster," on page 375.

Online backup and recovery

About online backup and recovery

Data ONTAP creates online data backups to enable online data recovery. Online backup refers to backup data that is stored on disks rather than on tape. Data stored on disk is thus available for quick restoring in the event that disaster recovery operations are necessary.

Online backup and recovery solutions include: SnapShot, SnapMirror, SnapRestore, SnapVault, SyncMirror and the `vol copy` command.

- The SnapShot feature enables you to schedule weekly, daily, or hourly online backups. Snapshot technology makes online point-in-time copies in the same volume as the original data and enables users to recover their own deleted or modified files without assistance from a system administrator.
- The SnapMirror feature allows you to schedule regular automatic copies of file system snapshots of a volume or qtree onto another volume or qtree (on the same filer or a different filer).
- The SnapRestore feature restores an entire volume to the state recorded in a previously created snapshot with maximum speed and disk space efficiency.
- The SnapVault feature protects the data in one or more qtrees in a series of snapshots stored on a separate filer. SnapVault maintains an online, asynchronous, permanently read-only replica of the qtree data. SnapVault backup and snapshot creation runs on an automated schedule.

  Note

  SnapVault, in addition to providing backup to NetApp filers, also provides direct backup of non-filer storage systems, such as servers running Windows NT, Windows 2000, Solaris, or HP-UX.

- SyncMirror provides continuous real-time mirroring of data between two partner volumes on a shared or partner filer system.
- The MetroCluster feature provides SyncMirror continuous mirroring over extended distances (500 meters to 30 kilometers).
- You can also use the `vol copy` command to snapshot and copy file systems to separate volumes or filers manually or by means of a script. You can use the `ndmpcopy` command to copy any subtree to any location on any filer.

You can use these online data backup and recovery systems to supplement tape backup and recovery.

Advantages of online backup and recovery

Online backup and recovery protection gives you the following advantages over tape archives alone:

- Speedy backups and restores greatly reduce backup time requirements.
- Backups can be made more frequently because they are faster.
- It is easy to recover a particular file, directory, or volume from an online backup.
- Disaster recovery is quicker with online mirroring and restores.
- Data availability is higher because of the high speed of data recovery.
- More data can be backed up in less time.

Disadvantages of online backup and recovery

Online snapshot data protection has the following disadvantages over tape archives:

- Online data protection is physically vulnerable. Filers and disk shelves are vulnerable to physical catastrophes.
- Online data protection consumes resources, such as disk space, that could be used for day-to-day activities.

Online backup and restore methods

To find detailed information about Data ONTAP data protection and online backup and recovery methods, consult the following table.

Method	**Chapter**
Using snapshots to make a read-only image of a file system on the same disk	Chapter 2, "Snapshot Management" on page 17
Using SnapRestore to restore data to a corrupted volume from a previous snapshot	Chapter 3, "Data Recovery Using SnapRestore" on page 57
Using SnapMirror to maintain a replica of one volume in another volume	Chapter 4, "Data Protection Using SnapMirror" on page 73

Method	**Chapter**
Using SnapVault to keep copies of volumes on the server, from which individual qtrees are available at any time to the client	Chapter 5, “Data Protection Using SnapVault” on page 195
Using the `vol copy` command to copy data from one volume to another	Chapter 6, “Volume Copy” on page 279
Using SyncMirror to maintain two identical copies of a volume at all times	Chapter 7, “SyncMirror Management” on page 301

Database protection with the NVFAIL feature

About database protection

If NVRAM problems occur that compromise database validity, the NVFAIL feature can warn you and automatically rename the database so that it does not restart automatically. You can then make sure that the database is valid before restarting it.

Where to find more information

See "Database Protection Using NVFAIL" on page 341 for more information about how Data ONTAP provides database protection using the `nvfail` option of the `vol options` command.

When to use NVFAIL

You use this feature only when you have databases on your filer.

Protecting against a data loss disaster

What a data loss disaster is

A disaster is a situation in which service from one physical site (for example, a building or a corporate campus) on the network is lost for an extended period of time. The following are examples of disasters:

- Fire
- Earthquake
- Prolonged power outages at a site
- Prolonged loss of connectivity from clients to the filers at a site

What a disaster affects

When a disaster occurs, it can affect all of your computing infrastructure including Network Appliance filers, application servers, networking connectivity, and client connectivity. When you create a disaster plan, take into consideration all of your computing infrastructure.

Determining if a disaster occurred

It is critical that you follow some predefined procedure to confirm that a disaster really has occurred. The procedure should determine the status of the supposed disaster site.

- Use external interfaces to the filer, such as the following:
    - Ping
    - Remote shell
    - FilerView
- Use network management tools to test connectivity to the disaster site.
- Physically inspect the disaster site, if possible.

You should declare a disaster only after determining that a disaster has occurred and that service cannot be restored.

Tools for protecting against data loss disasters

The following Network Appliance features and products are best for enabling the administrator to back up or replicate data stored on a local filer to an off-site network location. This ensures data can be restored if data loss is caused by disaster at a primary data storage site.

Feature	Description
SnapVault: Inter-site snapshot backup and restorability	A SnapVault secondary storage device can be located offsite, any distance from the primary storage units that it is backing up. **Data recoverability:** In event of a data-loss disaster at a primary storage site, data that is backed up to SnapVault secondary storage can be restored to primary storage units that have not suffered physical damage or that have been replaced either at the disaster site or at an alternate location. **Currency of restore:** Data can be restored from the time that the last SnapVault snapshot was created on the secondary storage system. **Connection requirements:** DSL connections or faster are recommended between the primary and secondary storage systems. Modem connections are possible. Routers, switches, and DNS servers should be pre-configured to direct users to alternate primary storage sites if the primary storage system that they first attempt to access becomes unavailable. **Advantage:** Centralized, inexpensive off-site backup **For more information:** See Chapter 5, "Data Protection Using SnapVault" on page 195.

Feature	Description
SnapMirror: Inter-site snapshot backup, availability, and restorability	A SnapMirror destination storage device can be located offsite, any distance from the source filer whose volumes it is mirroring. **Data availability:** In event of a data-loss disaster at a source filer site, SnapMirror data at the destination site can be made quickly available at the destination site. **Data recoverability:** SnapMirror data can be restored to source storage units that have not suffered physical damage or that have been replaced either at the disaster site or at an alternate location. **Currency of restore:** Data can be restored from the time of the last SnapMirror snapshot transfer from source to destination. **Connection requirements:** DSL connections or faster are recommended between the source and destination filers. Modem connections are possible. Routers, switches, and DNS servers should be pre-configured to direct users to the destination storage site if the source filer they are attempting to access becomes unavailable. **Advantage:** Combined off-site protection and availability **For more information:** See Chapter 4, "Data Protection Using SnapMirror" on page 73.

Feature	Description
Metro-Cluster: Inter-site real-time backup, availability, and restorability	Synchronously mirrored MetroCluster filers can be located at different sites, up to ten miles from one another. **Data availability:** In event of a data-loss disaster at one filer site, data that has been mirrored to the partner filer site can be made quickly available. **Data recoverability:** This data can also be mirrored to source storage units that have not suffered physical damage or that have been replaced either at the disaster site or at an alternate location. **Currency of restore:** Data can be restored from the time of the last NVRAM checkpoint. **Connection requirements:** Data ONTAP cluster connections supplemented with switches and DSL or faster connections are required. Routers, switches, and DNS servers should be pre-configured to direct users to the MetroCluster partner if the clustered filer that they first attempt to access becomes unavailable. **Advantage:** Combined real-time off-site protection and availability **For more information:** See Appendix A, "Disaster Protection Using MetroCluster," on page 375.

Data protection in a SAN environment

Protecting volumes containing LUNs

If your filer volumes contain logical units of storage (LUNs) created to accommodate integration into a NetApp storage area network (SAN) environment, you must carry out modified procedures to implement data protection using the features described in this document.

See the chapter on data protection in the *Block Access Management Guide for iSCSI* or the *Block Access Management Guide for iSCSI* for description of data backup and restore on filer volumes containing LUNs.

Snapshot Management 2

About this chapter

This chapter discusses what Snapshot™ copies are and how to manage them.

Topics in this chapter

This chapter discusses the following topics:

Understanding snapshots

What a snapshot is

A snapshot is a frozen, read-only image of a traditional volume, a flexible volume, or an aggregate that reflects the state of the file system at the time the snapshot was created. Snapshots are your first line of defense for backing up and restoring data. See the *Storage Management Guide* for information about traditional volumes, flexible volumes, or aggregates.

When snapshots are created

Data ONTAP maintains a configurable snapshot schedule that creates and deletes snapshots automatically for each volume. See “Creating snapshot schedules” on page 25. Snapshots can also be created and deleted manually.

Maximum number of snapshots

You can store up to 255 snapshots at one time on each filer volume.

Maximum space snapshots can occupy

You can specify the percentage of disk space that snapshots can occupy. The default setting is 20% of the total (both used and unused) space on the disk. For a full explanation of how snapshots consume disk space, see “Understanding snapshot disk consumption” on page 35.

How snapshots handle file permissions

Snapshot files carry the same permissions and inode numbers as the original files, keeping the integrity of the security system intact. Inodes are data structures that hold information (including permissions information) about files on the filer. There is an inode for each file and a file is uniquely identified by the file system on which it resides and its inode number on that system.

Note

The inode number for a file in a snapshot is the same as the inode number for the corresponding file in the active file system. As a result, some programs on UNIX clients consider the two files to be the same. For example, if you use the GNU `diff` program to compare the two files, it does not find any differences between them. In some cases, if you try to restore a file from a snapshot, you might see the following error message:

`cp:.snapshot/xxx and xxx are identical`.

To make sure that the two files have different inode numbers before the copy or comparison, copy one of the files to different name.

What you can do with snapshots

Snapshots enable system administrators to

- ◆ Create instantaneous backups from an active filer
- ◆ Create a clone of a flexible volume
- ◆ Create a clone of a LUN

Note

See the *Storage Management Guide* for information about cloning a flexible volume. See either the *Block Access Management Guide for iSCSI* or the *Block Access Management Guide for FCP* for information about cloning a LUN.

Snapshots enable end-users to

- ◆ Recover older versions or sets of files that were accidentally changed or deleted
- ◆ Restore their own files without needing a system administrator to restore files from tape

Accessing snapshots from clients

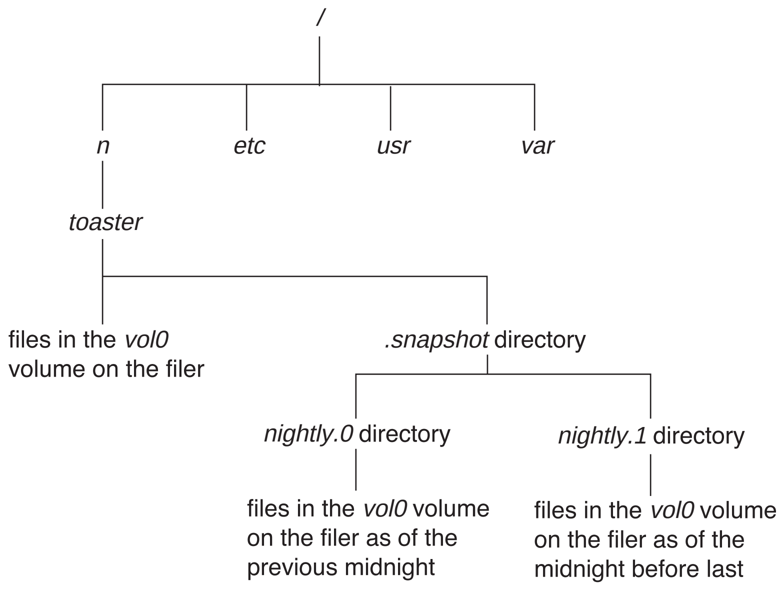
About user access to snapshots

By default, every filer volume contains a directory named .snapshot, through which users can access old versions of files in that directory. How users gain access to snapshots depends on the file-sharing protocol used—NFS or CIFS—and is described in the following sections. Access to snapshots can be turned off.

Snapshot files carry the same permissions as the original file. A user who has permission to read a file in the volume can read that file in a snapshot. A user without read permission to the volume cannot read that file in a snapshot. Write permission does not apply because snapshots are read-only. Snapshots can be accessed by any user with the appropriate permissions.

NFS user access to snapshots

The following illustration shows the directory structure on an NFS client with the vol0 volume of a filer named toaster mounted on the /n/toaster directory.


/
n
etc
usr
var
toaster
files in the vol0 volume on the filer
.snapshot directory
nightly.0 directory
nightly.1 directory
files in the vol0 volume on the filer as of the previous midnight
files in the vol0 volume on the filer as of the midnight before last

Explanation: In this example, the user can obtain access to snapshots by way of the /n/toaster/.snapshot directory. Notice that the .snapshot directory is shown only at the mount point, although it actually exists in every directory in the tree.

The user, however, can only see the .snapshot directory at the mount point. That is, the .snapshot directory is accessible by name in each directory, but is only seen in the output of the `ls` command at the mount point.

For example, at the mount point of a filer file system, a directory listing looks like this:

```
filerA> ls -a
.     ..     .snapshot     dir1     dir2
```

The same command entered in a directory below the mount point does not show the .snapshot directory; for example:

```
filerA> cd dir1
filerA> ls -a
.     ..    file1     file2
```

If you enter the `ls` command with the directory name .snapshot, you can see a directory for each of the snapshots for the dir1 directory:

```
filerA> ls .snapshot
hourly.0     hourly.4     nightly.0     nightly.4
hourly.1     hourly.5     nightly.1     nightly.5
hourly.2     hourly.6     nightly.2     weekly.0
hourly.3     hourly.7     nightly.3     weekly.1
```

If the .snapshot directory entry appeared in every directory, it would cause many commands to work improperly. For instance, all recursive commands for deleting files would fail because everything below the .snapshot directory is read-only. Recursive copies would copy everything in the snapshots as well as files in the active file system, and a `find` command would generate a list much longer than expected.

CIFS user access to snapshots

By default, CIFS users cannot see the .snapshot directory. Use the `cifs.show_snapshot` option to see the .snapshot directory. See the na_options(1) manual (man) page for details.

To CIFS users, the .snapshot directory appears only at the root of a share. For example, if a user’s home directory is a share named bill that corresponds to the /vol/vol0/home/bill directory, only the /vol/vol0/home/bill/.snapshot directory is visible. When this user displays the contents of the home directory, the .snapshot directory is displayed as ~snapshot if the operating system supports long file names and as ~SNAPSHT if the operating system supports only short file names.

Note

The .snapshot directory can be made visible in a directory listing or Windows Explorer display if the client operating system is configured to show hidden files.

In each directory within the share, a snapshot directory exists but is not visible to clients. For example, if the client operating system supports long file names, the applications on that operating system can use the snapshot at each level of the share by using .snapshot, ~snapshot, or ~SNAPSHT as the directory name. The user cannot, however, display the directory name in any listing.

Accessing snapshots from CIFS clients

To access snapshots on Windows NT 4 or other Windows clients (Windows 95 or later), complete the following step.

Step	Action
1	From the Start > Run menu, enter the following command: **`\\filername\share\.snapshot`** (or **`~snapshot`** or **`~SNAPSHT`**). *filername* is the name of the filer you are using. *share* is the name of the share you want to access.

Example: `\\filerA\home\.snapshot`

Snapshots can also be accessed lower in the share by providing a path to a lower directory. Snapshots can be accessed through DOS on any system by changing to the ~SNAPSHT directory.

Disabling and enabling client access to snapshots

Sometimes you might want to disable and enable client access to snapshots in a specific volume, for example, for security reasons or to prevent access to corrupted or virus-infected files.

To disable or enable client access to snapshots in a volume, complete the following step.

Step	Action	
	If you want to...	**Then...**
1	Disable client access to snapshots and make the .snapshot directory invisible to clients	Enter the following command: **`vol options`** ***`volume_name`*** **`nosnapdir on`** *volume_name* is the name of the volume for which you want to disable client snapshot access.
	Enable client access to snapshots and make the .snapshot directory visible to clients	Enter the following command: **`vol options`** ***`volume_name`*** **`nosnapdir off`** *volume_name* is the name of the volume for which you want to enable client snapshot access.

Restoring files from snapshots

Why you restore files

You might need to restore a file from a snapshot if the file was accidentally erased or corrupted.

Note

If you have purchased the SnapRestore option, you can automatically restore files or volumes from snapshots with one command. To restore a volume, see "Deciding when to use SnapRestore" on page 59. To restore a single file, see "Reverting a file to a selected snapshot" on page 66.

How to restore a file

To restore a file from a snapshot, complete the following steps.

Step	Action
1	If the original file still exists and you do not want it overwritten by the snapshot file, then use your UNIX or Windows client to rename the original file or move it to a different directory.
2	Locate the snapshot containing the version of the file you want to restore.
3	Copy the file from the .snapshot directory to the directory in which the file originally existed.

Snapshot restoration using Shadow Copy Client tools

You can access and restore Data ONTAP Snapshot files using the Windows Shadow Copy Client. The Shadow Copy Client provides a Previous Versions tab in the Properties menu from which you can view and restore Data ONTAP Snapshot images.

The Shadow Copy Client software for Windows 2003 is called the Previous Versions Client. Downloads available from Microsoft allow you to use Shadow Copy client tools on most older versions of Windows. Consult the Microsoft documentation for more information about Shadow Copy Client or Previous Versions Client software.

Creating snapshot schedules

About creating snapshot schedules

Data ONTAP provides a default snapshot schedule for each volume. You can configure the schedule to fit your needs. This schedule creates snapshots automatically and deletes old snapshots after a predetermined amount of time.

The default snapshot schedule

When you install Data ONTAP on a filer, it creates a default snapshot schedule. The default snapshot schedule automatically creates one nightly snapshot Monday through Saturday at midnight, and four hourly snapshots at 8 a.m., noon, 4 p.m., and 8 p.m. Data ONTAP retains the two most recent nightly snapshots and the six most recent hourly snapshots, and deletes the oldest nightly and hourly snapshots when new snapshots are created. You can see an example of the output for the default schedule in "Default snap schedule command results" on page 29.

User-specified snapshot schedules

There are three types of schedules that you can set up to run automatically using the `snap sched` command. The following table describes the three types.

Type	**Description**
Weekly	Data ONTAP creates these snapshots every Sunday at midnight. Weekly snapshots are called weekly.n, where *n* is an integer. The most recent weekly snapshot is weekly.0, and weekly.1 is the next most recent weekly snapshot.
Nightly	Data ONTAP creates these snapshots every night at midnight, except when a weekly snapshot is scheduled to occur at the same time. Nightly snapshots are called nightly.n, where *n* is an integer. The most recent nightly snapshot is nightly.0, and nightly.1 is the next most recent nightly snapshot.

Type	Description
Hourly	Data ONTAP creates these snapshots on the hour or at specified hours, except at midnight if a nightly or weekly snapshot is scheduled to occur at the same time. Hourly snapshots are called hourly.n, where *n* is an integer. The most recent hourly snapshot is hourly.0, and hourly.1 is the next most recent hourly snapshot.

Note

When Data ONTAP creates a weekly, nightly, or hourly snapshot, the value of *n* is adjusted for all the weekly, nightly, or hourly snapshots. The higher the value of *n*, the older the snapshot.

Displaying the snapshot schedule at the command line

To display the snapshot schedule for a volume on a filer, complete the following step.

Step	Action
1	Enter the following command: **`snap sched [`*`volume_name`*`]`** **Note** If you do not specify a volume name, `snap sched` displays the snapshot schedule for each volume on the filer.

Example: The following is an example of the `snap sched` command output:

```
filerA>snap sched vol1
Volume vol1: 2 6 8@8,12,16,20
```

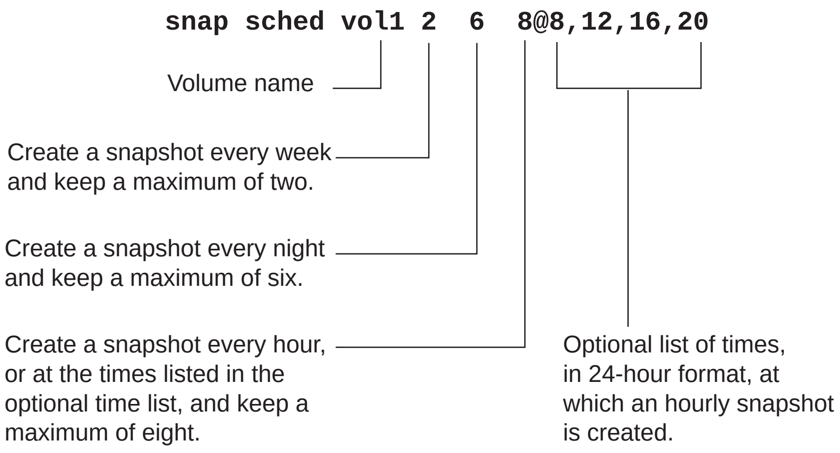
Displaying the snapshot schedule using FilerView

To display the snapshot schedule for a volume using FilerView, complete the following steps.

Step	Action
1	In FilerView, click Volumes in the list on the left.
2	In the list under Volumes, click Snapshots.
3	In the list under Snapshots, click Configure.

What the snap sched command arguments mean

The following illustration explains the arguments in a sample `snap sched` command output.


snap sched vol1 2 6 8@8,12,16,20
Volume name
Create a snapshot every week and keep a maximum of two.
Create a snapshot every night and keep a maximum of six.
Create a snapshot every hour, or at the times listed in the optional time list, and keep a maximum of eight.
Optional list of times, in 24-hour format, at which an hourly snapshot is created.

Snapshot schedule results: This schedule keeps the two most recent weekly snapshots, the six most recent nightly snapshots, and the eight most recent hourly snapshots, created at 8 a.m., noon, 4 p.m., and 8 p.m. every day. Whenever the snapshot schedule creates a new snapshot of a particular type, it deletes the oldest one and renames the existing ones. On the hour, for example, the filer deletes hourly.7, renames hourly.0 to hourly.1, and so on.

Note

If you omit the `@` argument specifying the hours for the hourly snapshots, Data ONTAP creates a snapshot every hour. Nightly and weekly snapshots are always created at midnight.

Strategies for creating a snapshot schedule

Following are some strategies for scheduling and retaining snapshots:

- If users rarely lose files or typically notice lost files right away, use the default snapshot schedule. This schedule creates no weekly snapshot; it creates a snapshot every night and keeps two; and it creates hourly snapshots at 8 a.m., noon, 4 p.m., and 8 p.m, and keeps six. Following is the default snapshot command:

  ```
  snap sched volume_name 0 2 6@8,12,16,20
  ```

- If users commonly lose files or do not typically notice lost files right away, Network Appliance recommends that you delete the snapshots less often than you would if you used the default schedule.

  Following is the recommended snapshot schedule for this situation. It keeps two weekly snapshots, six nightly snapshots, and eight hourly snapshots:

  ```
  snap sched vol1 2 6 8@8,12,16,20
  ```

  On many systems, only 5% or 10% of the data changes each week, so the snapshot schedule of six nightly and two weekly snapshots consumes 10% to 20% of disk space. Considering the benefits of snapshots, it is worthwhile to reserve this amount of disk space for snapshots. For more information on how snapshots consume disk space, see “Understanding snapshot disk consumption” on page 35.

- You can create different snapshot schedules for different volumes on a filer. On a very active volume, schedule snapshots every hour and keep them for just a few hours, or turn off snapshots. For example, the following schedule creates a snapshot every hour and keeps the last three:

  ```
  snap sched vol2 0 0 3
  ```

  This schedule does not consume much disk space, and it lets users recover files in recent snapshots as long as they notice their mistake within a couple of hours.

- When you create a new volume on a filer, the new volume inherits the snapshot schedule from the root volume. After you use the volume for a while, check how much disk space the snapshots consume and how often users need to recover lost files, and then adjust the schedule as necessary.

Changing the snapshot schedule

To change the automatic snapshot schedule for a specific volume, complete the following step.

Step	Action
1	Enter the following command: **`snap sched`** ***`volume_name weekly nightly hourly@n,n,....`*** *volume_name* is the name of the specific volume for the snapshot. *weekly* is the number of weekly snapshots to keep. *nightly* is the number of nightly snapshots to keep. *hourly* is the number of hourly snapshots to keep. *n,n,...* specifies the hours at which to create the hourly snapshots. **Note** A zero in any of the three schedules (weekly, nightly, hourly) disables snapshots for that interval.

Default snap schedule command results

This is the default automatic snapshot schedule:

```
snap sched volx 0 2 6 @8,12,16,20
```

The following example lists the snapshots created using the default schedule (where January 11 is a Sunday):

```
ls -lu .snapshot
total 64
drwxrwsrwx  1 root 4096 Jan 14 12:00 hourly.0
drwxrwsrwx  1 root 4096 Jan 14 08:00 hourly.1
drwxrwsrwx  1 root 4096 Jan 13 20:00 hourly.2
drwxrwsrwx  1 root 4096 Jan 13 16:00 hourly.3
drwxrwsrwx  1 root 4096 Jan 13 12:00 hourly.4
drwxrwsrwx  1 root 4096 Jan 13 08:00 hourly.5
drwxrwsrwx  1 root 4096 Jan 14 00:00 nightly.0
drwxrwsrwx  1 root 4096 Jan 13 00:00 nightly.1
```

Note

Daily snapshots are created at midnight of each day except Sunday, and weekly snapshots are created at midnight on Sunday. Only one snapshot is created at a time. If a weekly snapshot is being created, for instance, no daily or hourly snapshot will be created even if one is scheduled.

Disabling and enabling automatic snapshots

You can disable automatic snapshots for a period of time without changing the automatic snapshot schedule.

To temporarily disable or enable automatic snapshots, complete the following step.

Step	Action	
1	**If you want to...**	**Then...**
	Disable automatic snapshots	Enter the following command: **`vol options volume_name nosnap on`** *volume_name* is the name of the volume for which you want to disable automatic snapshots.
	Enable automatic snapshots	Enter the following command: **`vol options volume_name nosnap off`** *volume_name* is the name of the volume for which you want to enable automatic snapshots.

Note

You can also disable automatic snapshots by changing the snapshot schedule so that no snapshots are scheduled. To do this, enter the following command:

```
snap sched volume_name 0 0 0
```

Creating snapshots

Automatic or manual snapshots

The `snap sched` command automatically creates snapshots at preset intervals in a snapshot schedule. You can also create snapshots manually at any time.

Creating snapshots manually

To create a snapshot manually, complete the following step.

Step	Action
1	Enter the following command: **`snap create`** ***`volume_name snapshot_name`*** *volume_name* is the name of the volume on which you want to create the snapshot. *snapshot_name* is the name you want to give the snapshot.

Note

The `snap create` command does not accept a snapshot name containing a slash (/); therefore, it is not possible to enter a specific path for the snapshot file.

Creating snapshots manually using FilerView

To create snapshots manually using FilerView, complete the following steps.

Step	Action
1	In FilerView, click Volumes in the list on the left.
2	In the list under Volumes, click Snapshots.
3	In the list under Snapshots, click Add.

Finding the snapshot you need

Why you need to access a particular snapshot

You might need to access a particular snapshot (a version of a file at a particular point in time) because a file was changed, corrupted, or erased and the problem was not noticed until after one or more snapshots of it were created. When you start looking for the *version* of the file you need, you will look for it by means of the *access time* of the snapshot.

File version versus file access time

The *version* of a file refers to the last time the file was modified before a snapshot was created. The *access time* of a file refers to the snapshot creation time for a file, whether or not any modifications were made to that file.

Finding all file versions in snapshots

The best way to find all versions of a particular file preserved in snapshots is to use the `ls` command from an NFS client, or to use the Find function in Windows.

To find all the versions of a particular file in the snapshots, complete the following step.

Step	Action	
1	**If you are using...**	**Then...**

	An NFS client	Enter the following command:
		`ls -l` ***`filename`***
		`.snapshot/*/`***`file_name`***
		Result: A list is displayed of all versions of the requested file.

	A CIFS client	**1.** Choose Find from the Windows Start Menu and select Files and Folders. **Result:** A search window opens prompting you for a directory and file name. **2.** In the search window, enter the file name to search for in the ~snapshot directory.

NFS client example: The following example shows how to find all versions of the myfile.txt file:

```
ls -l myfile.txt .snapshot/*/myfile.txt
-rw-r--r--  1 smith 0 Jan 14 09:40  myfile.txt
-rw-r--r--  1 smith 0 Jan 13 18:39  .snapshot/nightly.0/myfile.txt
-rw-r--r--  1 smith 0 Jan 12 19:17  .snapshot/nightly.1/myfile.txt
```

The version of myfile.txt in the active file system was last modified on January 14, but the old versions available in the snapshots were modified on January 13 and January 12. Although users can use standard UNIX commands to read the saved versions of myfile.txt, they cannot modify or delete these older versions because everything in the .snapshot directory is read-only.

CIFS client example: If a user maps the home share to drive F: and wants to find all versions of myfile.txt in snapshots, the user can choose Find from the Windows Start menu to search for myfile.txt in the f:\~snapshot folder.

How to determine access times from an NFS client

When Data ONTAP creates a snapshot, the access time of each file in the snapshot is updated to the snapshot creation time.

To determine when snapshots were created, complete the following step.

Step	Action
1	From an NFS client, enter the following command: **`ls -lu`** ***`filename`*** **`.snapshot/*/`*****`file_name`***

Example: Following is an example of the `ls -lu` command:

```
ls -lu myfile.txt .snapshot/*/myfile.txt
-rw-r--r--  1 smith 0 Jan 14 09:40  myfile.txt
-rw-r--r--  1 smith 0 Jan 14 00:00  .snapshot/nightly.0/myfile.txt
-rw-r--r--  1 smith 0 Jan 13 00:00  .snapshot/nightly.1/myfile.txt
```

Note

On a UNIX client, if you use `ls -l` instead of `ls -lu` to list the snapshot creation times, the times are not necessarily all different. The times listed by `ls -l` reflect the modification times of the directory at the time of each snapshot, and are not related to the times at which the snapshots are created.

How to determine access times from a CIFS client

You can determine the access time of a file from a CIFS client by checking its properties.

Understanding snapshot disk consumption

Importance of understanding snapshot disk consumption

You use information about the amount of disk space that snapshots consume and are likely to consume to determine

- How often to create a snapshot
- How long to keep a snapshot
- How much disk space you need for the snapshot reserve

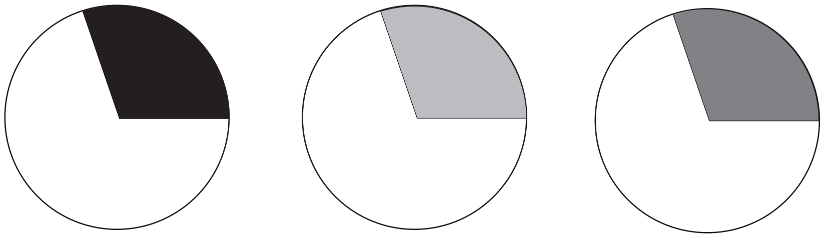
How snapshots consume disk space

Data ONTAP preserves pointers to all the disk blocks currently in use at the time the snapshot is created. When a file is changed, the snapshot still points to the disk blocks where the file existed before it was modified, and changes are written to new disk blocks.

Disk consumption is minimized by preserving individual disk blocks rather than whole files. Snapshots begin to consume extra space only when files in the active file system are changed or deleted. When this happens, the original file blocks are still preserved as part of one or more snapshots, but in the active file system the changed blocks are rewritten to different locations on disk or removed as active file blocks entirely. The result is that, in addition to the disk space used by blocks in the modified active file system, disk space used by the original blocks is still reserved in snapshots to reflect what the active file system was before the change.

Example of how snapshots consume disk space

The following illustration shows how snapshots consume disk space before and after you delete a file named myfile.txt.



Before any snapshot is taken, disk space is consumed by the active file system only.

After a snapshot is taken, the active file system and snapshot point to the same disk blocks. The snapshot does not use extra disk space.

After *myfile.txt* is deleted from the active file system, the snapshot still includes the file and references its disk blocks. That's why deleting active file system data does not always free disk space.

■ Space used by the active file system

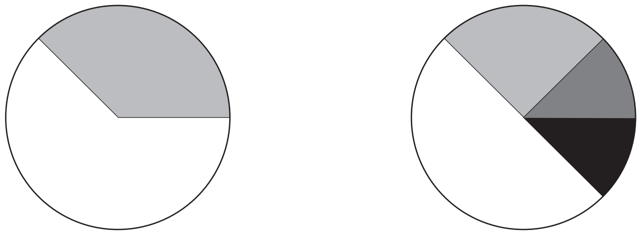
▩ Space used by the snapshot only

▨ Space shared by the snapshot and the active file system

□ Unused disk space

Example of how changing file content consumes disk space

Changing the contents of the myfile.txt file creates a situation similar to the one illustrated above. New data written to myfile.txt cannot be stored in the same disk blocks as the current contents because the snapshot is using those disk blocks to store the old version of myfile.txt. Instead, the new data is written to new disk blocks. As the following illustration shows, there are now two separate copies of myfile.txt on disk—a new copy in the active file system and an old one in the snapshot.



After a snapshot is taken, the active file system and snapshot point to the same disk blocks and the snapshot does not use any extra space.

After a change is made to the file, the active file system and snapshot no longer point to the same disk blocks and the snapshot now uses extra space.

■ Space used by the active file system
▩ Space used by the snapshot only
▨ Space shared by the snapshot and the active file system
□ Unused disk space

Disk consumption by multiple snapshots

Multiple snapshots can reference the same file without consuming extra disk space. Suppose a snapshot contains a 1-MB file that has not changed since Data ONTAP created the snapshot. If you delete that file from the active file system, the snapshot then consumes 1 MB of disk space.

The same version of that 1-MB file might be referenced by several snapshots: hourly.0, hourly.1, and hourly.2. If those snapshots all contain the 1-MB file that has not changed since Data ONTAP created them, only 1 MB of disk space is consumed by the snapshots even though all three snapshots contain the file. This is because they are all pointing to the same disk blocks.

For detailed information

The following sections discuss how to reserve disk space for snapshots and how to monitor snapshot disk space usage:

- "Monitoring snapshot disk consumption" on page 38
- "Displaying snapshot disk consumption statistics" on page 39
- "Understanding the snapshot reserve" on page 43
- "Changing the snapshot reserve" on page 46

Monitoring snapshot disk consumption

Displaying disk consumption

The `df` command displays the amount of free space on a disk. It treats snapshots as a partition different from the active file system.

To display information about snapshot disk consumption, complete the following step.

Step	Action
1	Enter the following command: **`df`**

Example: Following is a partial sample `df` command output:

```
filerA> df

Filesystem            kbytes    used    avail   capacity
/vol/vol0             3000000  2000000  1000000  65%
/vol/vol0/.snapshot   1000000  500000   500000   50%
```

Explanation of sample output: In the sample output, the kbytes column shows that the vol0 volume contains 3,000,000 KB (3 GB) of disk space for the active file system and 1,000,000 KB (1 GB) of disk space reserved for snapshots, for a total for 4,000,000 KB (4 GB) of disk space. In this example, 66% of the active disk space is used and 33% is available. Note that the capacity percentage rounds down to 65%. The 1,000,000 KB (1 GB) of disk space for snapshots represents 25% of the volume capacity, of which 500,000 KB (0.5 GB) is used and 500,000 KB (0.5 GB) is available, so that the space for snapshots is at 50% capacity.

Note that the 50% figure is not 50% of disk space, but 50% of allotted snapshot space. If allotted snapshot space is exceeded, this number will be over 100%.

It is important to understand that the /vol/vol0/.snapshot line counts data that exists only in a snapshot. The snapshot calculation does not include snapshot data that is shared with the active file system.

Displaying snapshot disk consumption statistics

How to display disk consumption for snapshots in a volume

The `snap list` command shows the amount of disk space consumed by snapshots in a specified volume. This command enables you to see how much disk space each snapshot uses and helps you determine an appropriate snapshot reserve. The `snap list` command also shows whether a snapshot is currently being used for backup and whether it is needed for a restartable dump or another backup application.

Listing snapshot statistics for a volume

To display the snapshot statistics for a volume, complete the following step.

Step	Action
1	Enter the following command: **`snap list`** ***`volume_name`*** *volume_name* is the name of the volume for which you want statistics.

Sample snap list command output

Following is an example of the `snap list` command output. If you do not specify a volume name in the command, the output contains statistics about each volume in the system.

```
filerA> snap list vol0
Volume vol0
%/used      %/total    date          name
----------  ----------  ------------  --------
0% ( 0%)    0% ( 0%)    Jan 19 08:01  hourly.0
1% ( 1%)    1% ( 1%)    Jan 19 00:01  nightly.0
2% ( 2%)    2% ( 2%)    Jan 18 20:01  hourly.1
3% ( 2%)    2% ( 2%)    Jan 18 16:01  hourly.2
3% ( 2%)    3% ( 2%)    Jan 18 12:01  hourly.3
5% ( 3%)    4% ( 3%)    Jan 18 00:01  nightly.1
7% ( 4%)    6% ( 4%)    Jan 17 00:00  nightly.2
8% ( 4%)    7% ( 4%)    Jan 16 00:01  nightly.3
10%( 5%)    9% ( 4%)    Jan 15 00:01  nightly.4
```

How the snap list output is calculated

The %/used column: The %/used column shows space consumed by snapshots as a percentage of disk space being used in the volume. The first number is cumulative for all snapshots listed so far, and the second number is for the specified snapshot alone.

- The first number is equal to

$$100\% \times \frac{\text{cumulative snapshot space}}{\text{cumulative snapshot space + file system space}}$$

- The second number is equal to

$$100\% \times \frac{\text{this snapshot}}{\text{this snapshot + file system space}}$$

The %/total column: The %/total column shows space consumed by snapshots as a percentage of total disk space (both space used and space available) in the volume.

- The first number is equal to

$$100\% \times \frac{\text{cumulative snapshot space}}{\text{total disk space in this volume}}$$

  Cumulative snapshot space is the total space used by this snapshot and all other more recent snapshots (the ones preceding this snapshot in the `snap list` output).

- The second number is equal to

$$100\% \times \frac{\text{this snapshot}}{\text{total disk space in this volume}}$$

Summary of the snap list output

The %/used number is the most useful for planning the snapshot reserve because it is more likely to remain constant as the file system fills.

The information in "Sample snap list command output" on page 39 shows a volume that keeps five nightly snapshots and four hourly snapshots.

The sample output shows that the overhead for snapshots is only 10%, so the default snapshot reserve of 20% seems to be a waste of disk space. Assuming that this pattern of change holds, a reserve of 12% to 15% provides a safe margin to ensure that deleting files frees disk space when the active file system is full. For information about the snapshot reserve, see “Understanding the snapshot reserve” on page 43.

The values in parentheses, which show the space used by an individual snapshot, are useful in identifying a particular snapshot to delete when the file system is full. However, deleting a particular snapshot does not necessarily release the amount of disk space indicated, because other snapshots might be referring to the same blocks.

About using cumulative snapshot values

If you do not want the amount of disk space consumed by all snapshots to exceed a certain percentage of the used disk space, use the cumulative values in the `snap list` output to determine which snapshots to delete. In the preceding example, if you do not want more than 5% of used disk space to be spent by snapshots, delete all snapshots listed below nightly.1 in the `snap list` output; that is, nightly.2, nightly.3, and nightly.4. After deleting the snapshots, nightly.1 and all the other more recent snapshots consume 5% of the used disk space.

Displaying snapshot use and dependencies

The `snap list` command displays the notation “busy” after the name of a snapshot if the snapshot is being actively used by an application, as shown the example below for the snapshot hourly.0. Busy snapshots cannot be deleted.

```
snap list vol0

Volume vol0
 %/used    %/total    date          name
--------  ------     ------------  --------
0% ( 0%)  0% ( 0%)   Jan 19 08:01  hourly.0 (busy)
1% ( 1%)  1% ( 1%)   Jan 19 00:01  nightly.0(snapmirror)
2% ( 2%)  2% ( 2%)   Jan 18 20:01  hourly.1 (busy,snapmirror)
3% ( 2%)  2% ( 2%)   Jan 18 16:01  hourly.2
3% ( 2%)  3% ( 2%)   Jan 18 12:01  hourly.3
5% ( 3%)  4% ( 3%)   Jan 18 00:01  nightly.1 (backup[9])
7% ( 4%)  6% ( 4%)   Jan 17 00:00  nightly.2
8% ( 4%)  7% ( 4%)   Jan 16 00:01  nightly.3
10%( 5%)  9% ( 4%)   Jan 15 00:01  nightly.4
```

The `snap list` command also displays the name of an application next to a snapshot name if the application needs the snapshot currently or at a later time. For example, “backup” is displayed next to the snapshot name to show that the snapshot is the result of a `dump` command transfer that was interrupted but is restartable. The number following “backup” is the backup ID assigned by the `backup status` command. The notation “snapmirror” next to the snapshot name means that SnapMirror is retaining the snapshot to maintain a source-destination relationship.

snap list performance after a snap restore file operation

If you restore files with the `snap restore` command, and then issue the `snap list` command, the `snap list` command can take up to several minutes to complete. This condition persists until the snapshot from which you restored the file is purged from the system after reaching the end of its normal snapshot retention cycle.

For more information see “Reverting a file to a selected snapshot” on page 66.

Understanding the snapshot reserve

What the snapshot reserve is

The snapshot reserve specifies a set percentage of disk space for snapshots. By default, the snapshot reserve is 20% of disk space. The snapshot reserve can be used only by snapshots, not by the active file system. This means that if the active file system runs out of disk space, any disk space still remaining in the snapshot reserve is not available for active file system use.

Note

Although the active file system cannot consume disk space reserved for snapshots, snapshots can exceed the snapshot reserve and consume disk space normally available to the active file system.

Snapshot reserve management tasks

Snapshot reserve management involves the following tasks:

- Ensuring that enough disk space is set aside for snapshots so that they do not consume active file system space
- Keeping disk space consumed by snapshots below the snapshot reserve
- Ensuring that the snapshot reserve is not so large that it wastes space that could be used by the active file system

Use of deleted active file disk space

When enough disk space is available for snapshots in the snapshot reserve, deleting files in the active file system frees disk space for new files, while the snapshots that reference those files consume only the space in the snapshot reserve. If Data ONTAP created a snapshot when the disks were full, deleting files from the active file system would not create any free space because everything in the active file system would also be referenced by the newly created snapshot. Data ONTAP would have to delete the snapshot before it could create any new files.

Example: The following example shows how disk space being freed by deleting files in the active file system ends up in the snapshot:

If Data ONTAP creates a snapshot when the active file system is full and there is still space remaining in the snapshot reserve, the output from the `df` command—which displays statistics about the amount of disk space on a volume—is as follows:

```
Filesystem            kbytes    used     avail     capacity
/vol/vol0/            3000000   3000000  0         100%
/vol/vol0/.snapshot   1000000   500000   500000    0%
```

If you delete 100,000 KB (0.1 GB) of files, the disk space used by these files is no longer part of the active file system, so the space is reassigned to the snapshots instead.

Explanation: Data ONTAP reassigns 100,000 KB (0.1 GB) of space from the active file system to the snapshot reserve. Because there was reserve space for snapshots, deleting files from the active file system freed space for new files. If you enter the `df` command again, the output is as follows:

```
Filesystem            kbytes    used     avail     capacity
/vol/vol0/            3000000   2900000  100000    97%
/vol/vol0/.snapshot   1000000   600000   400000    60%
```

Snapshots can exceed reserve

There is no way to prevent snapshots from consuming disk space greater than the amount reserved for them; therefore, it is important to reserve enough disk space for snapshots so that the active file system always has space available to create new files or modify existing ones.

Example: Consider what would happen in the example if all files in the active file system were deleted. Before the deletion, the `df` output was as follows:

```
Filesystem            kbytes    used     avail     capacity
/vol/vol0/            3000000   3000000  0         100%
/vol/vol0/.snapshot   1000000   500000   500000    50%
```

After the deletion, the `df` command generates the following output:

```
Filesystem            kbytes    used     avail     capacity
/vol/vol0/            3000000   2500000  500000    83%
/vol/vol0/.snapshot   1000000   3500000  0         350%
```

Explanation: The entire 3,000,000 KB (3 GB) in the active file system is still being used by snapshots, along with the 500,000 KB (0.5 GB) that were being used by snapshots before, making a total of 3,500,000 KB (3.5 GB) of snapshot data. This is 2,500,000 KB (2.5 GB) more than the space reserved for snapshots; therefore, 2.5 GB of space that would be available to the active file system is now unavailable to it. The post-deletion output of the `df` command lists this unavailable space as “used” even though no files are stored in the active file system.

Recovering disk space for file system use

Whenever snapshots consume more than 100% of the snapshot reserve, the system is in danger of becoming full. In this case, you can create files only after you delete enough snapshots.

Example: If 500,000 KB (0.5 GB) of data is added to the active file system in the preceding example, a `df` command generates the following output:

```
Filesystem            kbytes    used      avail     capacity
/vol/vol0             3000000   3000000   0         100%
/vol/vol0/.snapshot   1000000   3500000   0         350%
```

Explanation: As soon as Data ONTAP creates a new snapshot, every disk block in the file system is referenced by some snapshot. Therefore, no matter how many files you delete from the active file system, there is still no room to add any more. The only way to recover from this situation is to delete enough snapshots to free more disk space.

See “Displaying snapshot disk consumption statistics” on page 39 for information about how to use the `snap list` command to determine which snapshots to delete.

Changing the snapshot reserve

Changing the snapshot reserve

After you understand how snapshots consume disk space and how the snapshot reserve works, you can change the snapshot reserve when needed.

To change the percentage of disk space used for the snapshot reserve, complete the following step.

Step	Action
1	Enter the following command: **`snap reserve`** ***`volume_name percent`*** *volume_name* is the name of the volume on the filer. *percent* is the percentage of disk space you want to reserve for snapshots.

Example: `snap reserve vol0 25`

Saving disk space by using file folding

What file folding means and how it saves disk space

File folding describes the process of checking the data in the most recent snapshot, and, if it is identical to the snapshot currently being created, just referencing the previous snapshot instead of taking up disk space writing the same data in the new snapshot. Disk space is saved by sharing unchanged file blocks between the active version of the file and the version of the file in the latest snapshot, if any.

The filer must compare block contents when folding a file, so there is a tradeoff between performance and space utilization.

If the folding process reaches a maximum limit on memory usage, it is suspended. When memory usage falls below the limit, the processes that were halted are restarted.

How to turn file folding on

To turn file folding on, complete the following step.

Step	Action
1	Enter the following command: **`options cifs.snapshot_file_folding.enable on`**

How to turn file folding off

To turn file folding off, complete the following step.

Step	Action
1	Enter the following command: **`options cifs.snapshot_file_folding.enable off`** This option is off by default.

Note
This option is not available to NFS users in this release.

Displaying data change rate between snapshots

What is displayed

You can display the rate of change stored between two snapshots as well as the rate of change between a snapshot and the active file system. Data ONTAP displays the rates of change in two tables. The first table displays rates of change between successive snapshots. The second table displays a summary of the rate of change between the oldest snapshot and the active file system. See the na_snap(1) man page for details.

Displaying rates of change on a volume

To display data change rates on a volume, complete the following step.

Step	Action
1	Enter the following command: **`snap delta`** ***`vol_name`*** *vol_name* is the name of the volume containing the snapshots. **Note** You can display change rates for all volumes by omitting the volume name

Example: The following command lists the rates of change for the vol0 volume.

```
filer> snap delta vol0
Volume vol0 working...
From Snapshot To                KB changed Time       Rate (KB/hour)
------------- ----------------- ---------- ---------- --------------
hourly.0      Active File System 149812       0d 03:43 40223.985
hourly.1      hourly.0          326232       0d 08:00 40779.000
hourly.2      hourly.1          2336         1d 12:00 64.888
hourly.3      hourly.2          1536         0d 04:00 384.000
hourly.4      hourly.3          1420         0d 04:00 355.000
nightly.0     hourly.4          1568         0d 12:00 130.666
hourly.5      nightly.0         1400         0d 04:00 350.000
nightly.1     hourly.5          10800      201d 21:00 2.229
Summary...
From Snapshot To                KB changed Time       Rate (KB/hour)
------------- ----------------- ---------- ---------- --------------
nightly.1     Active File System 495104     204d 20:43 100.697
```

Displaying rates of change between snapshots

To display data change rates between two snapshots, complete the following step.

Step	Action
1	Enter the following command: **`snap delta`** ***`vol_name snap1 snap2`*** *vol_name* is the name of the volume containing the snapshots. *snap1* and *snap2* are the names of the two snapshots.

Example: The following command lists the rate of change between nightly.0 and hourly.1 of the vol0 volume.

```
filer> snap delta vol0 nightly.0 hourly.1
Volume vol0 working...
From Snapshot To                 KB changed Time       Rate (KB/hour)
------------- ------------------ ---------- ---------- --------------
hourly.2      hourly.1           2336         1d 12:00 64.888
hourly.3      hourly.2           1536         0d 04:00 384.000
hourly.4      hourly.3           1420         0d 04:00 355.000
nightly.0     hourly.4           1568         0d 12:00 130.666
Summary...
From Snapshot To                 KB changed Time       Rate (KB/hour)
------------- ------------------ ---------- ---------- --------------
nightly.0     hourly.1           6860         2d 08:00 122.500
```

Displaying space reclaimed from deleted snapshots

About reclaimed space

You can display the amount of space you can reclaim by deleting one or more snapshots in a volume. The amount of space displayed is an approximation because writing to the volume, creating snapshots, or deleting snapshots cause the reclaimed amount to change.

Displaying amount of reclaimed space

To display the amount of space you can reclaim by deleting snapshots, complete the following step.

Step	Action
1	Enter the following command: **`snap reclaimable`** ***`vol_name snap1`*** **`[`*****`snap2`*** **`...]`** *vol_name* is the volume which contains the snapshots you might delete. *snap1* [*snap2* ...] are the names of snapshots you might delete. The names are separated by a space. **Note** It might take a while for Data ONTAP to display the amount of freed space. You can press Ctrl-C to interrupt the command.

Example: The following command displays the amount of space reclaimed by deleting the hourly.4, hourly.5, and nightly.0 snapshots in the vol1 volume:

```
filer> snap reclaimable vol1 hourly.4 hourly.5 nightly.0
Processing (Press Ctrl-C to exit) ...
snap reclaimable: Approximately 240 kbytes would be freed.
```

Deleting snapshots

How to determine which snapshots to delete on the basis of size

You can use the `snap list` command output to determine which snapshots to delete to free the most disk space.

In “Sample snap list command output” on page 39, the cumulative disk space used by snapshots gradually increases from top to bottom.

For example, in the %/used column, the cumulative space used by hourly.1 is 2% and the cumulative space used by hourly.2 is 3%. This is not always the case.

Before trying to conserve space by deleting a large snapshot file, examine the cumulative values in the `snap list` output. If two adjacent snapshot files show little difference in their cumulative values, most of the data referenced by these snapshots is the same. In this case, deleting only one of the snapshots does not free much disk space.

In many cases, you can use the default snapshot schedule and the default snapshot reserve because these settings are appropriate for most environments. When you create a new volume on a filer, the new volume inherits the snapshot schedule from the root volume. After you use the volume for several days, check how much disk space the snapshots are consuming in the volume. If the amount seems high, decrease the amount of time that snapshots are kept or increase the snapshot reserve.

As you use snapshots, continue to watch the statistics change over time. The statistics help you gain a better understanding of how snapshots use disk space.

Caution

As a general rule, avoid deleting snapshots that are not the product of the `snap sched` command (for example, snapshots generated by SnapMirror or SnapVault commands). Deleting these snapshots could halt the SnapMirror or SnapVault processes. An exception would be snapshots left over from old SnapMirror relationships that you no longer want to maintain. See “Releasing partners from a SnapMirror relationship” on page 159.

Deleting a snapshot manually

The `snap sched` command deletes obsolete regularly scheduled snapshots automatically; however, you might want to delete a snapshot before the preset interval to increase disk space or because it is a manual snapshot that is no longer needed but is not going to be automatically deleted.

To delete a snapshot manually from a specific volume, complete the following step.

Step	Action
1	Enter the following command: **`snap delete`** ***`volume_name snapshot_name`*** *volume_name* is the name of the volume that contains the snapshot to delete. *snapshot_name* is the specific snapshot to delete. **Note** To delete all snapshots on a volume, use the `-a` parameter: **`snap delete -a`** *volume_name*

Deleting a snapshot manually if the snapshot is locked

If a snapshot is locked, the `snap delete` operation fails until you execute a `snapmirror release` (see "How to release a source from a SnapMirror relationship" on page 159) or `snapvault release` (see "Releasing SnapVault relationships" on page 273) command to unlock the snapshot. Snapshots are locked because SnapMirror or SnapVault is maintaining them for the next update. Deleting a locked snapshot would prevent SnapMirror or SnapVault from correctly replicating a file or volume as specified in the schedule you set up.

Example of how to delete a locked SnapMirror snapshot

The following example shows how you would delete a SnapMirror snapshot that was locked because SnapMirror needed it for an update.

```
filerA> snap delete vol0 oldsnap
Can’t delete oldsnap: snapshot is in use by snapmirror.
Use ’snapmirror destinations -s’ to find out why.
filerA> snapmirror destinations -s vol0
Path        Destination
/vol/vol0   filerB:vol0
filerA> snapmirror release vol0 filerB:vol0
filerA> snap delete vol0 oldsnap
```

Example of how to delete a locked SnapVault snapshot

The following example shows how you would delete a SnapVault snapshot that was locked because SnapVault needed it for an update.

```
filerA> snap delete vol0 oldsnap
Can’t delete oldsnap: snapshot is in use by snapvault.
Use ’snapvault status -l’ to find out why.

filerA> snapvault status -l
SnapVault client is ON.
Source:                     filerA:/vol/vol0/qt3
Destination                 filerB:/vol/sv_vol/qt3
....

filerA> snapvault release /vol/vol0/qt3 filerB:/vol/sv_vol/qt3

filerA> snap delete vol0 oldsnap
```

Renaming snapshots

Why to rename a snapshot

You might want to rename a snapshot generated by the `snap sched` command if it contains data that you want to save. You can prevent it from being overwritten or deleted by the `snap sched` command process, which deletes regularly scheduled snapshots automatically, by using the `snap rename` command.

How to rename a snapshot

To rename a snapshot, complete the following step.

Step	Action
1	Enter the following command: **`snap rename`** ***`volume_name from_name to_name`*** *volume_name* is the name of the volume that contains the snapshot to rename. *from_name* is the current name of the snapshot to rename. *to_name* is the new name you want to give to the snapshot.

Example: `snap rename vol0 hourly.2 MyDataSave`

Data Recovery Using SnapRestore 3

About this chapter

This chapter describes how to restore data to a file or volume using the Data ONTAP SnapRestore feature. Read this chapter if you want to recover data that is no longer available or if you are testing a volume or file and want to restore that volume or file to pre-test conditions.

Note

SnapRestore is a licensed feature. You must purchase and install the license code before you can use it.

Topics in this chapter

This chapter discusses the following topics:

Understanding SnapRestore

What SnapRestore does

SnapRestore enables you to revert a local volume or file on a filer quickly to the state it was in when a particular snapshot was taken. In most cases, reverting a file or volume is much faster than restoring files from tape or copying files from a snapshot to the active file system.

How SnapRestore works

After you select a snapshot for reversion, the filer reverts the specified file or the volume to the data and timestamps that it contained when the selected snapshot was taken. Data that was written after the selected snapshot was taken is lost. If the volume you select for reversion is a root volume, the filer reboots the system.

What SnapRestore reverts

SnapRestore reverts only the file contents. It does not revert attributes of a volume. For example, the snapshot schedule, volume option settings, RAID group size, and maximum number of files per volume remain unchanged after the reversion.

When to use SnapRestore

You use SnapRestore to recover from data corruption. If a primary storage system application corrupts data files in a volume, you can revert the volume or specified files in the volume to a snapshot taken before the data corruption.

Why use SnapRestore rather than copying from a snapshot

SnapRestore carries out snapshot restoration more quickly, using less disk space, than an administrator can achieve by manually copying volumes, qtrees, directories, or large files to be restored from the filer’s snapshot system to the active file system. A large volume directory restore can be carried out in a few seconds using the SnapRestore feature.

SnapRestore can restore large volumes or files even if space limitations would prevent restoring by copying from a snapshot.

Deciding when to use SnapRestore

Considerations before using SnapRestore

You must take into account the following considerations before deciding whether to use SnapRestore to revert a file or volume.

- If the volume that you need to restore is a root volume, it is easier to copy the files from a snapshot or restore the files from tape than to use SnapRestore because you can avoid rebooting. If you need to restore a corrupted file on a root volume, however, a reboot is not necessary.
- If you revert the whole root volume, the filer reboots with configuration files that were in effect when the snapshot was taken.
- If the amount of data to be recovered is large, SnapRestore is the preferred method, because it takes a long time to copy large amounts of data from a snapshot or to restore from tape.
- If a file to be recovered needs more space than the amount of free space in the active file system, you cannot restore the file by copying from the snapshot to the active file system. For example, if a 10-GB file is corrupted and only 5 GB of free space exists in the active file system, you cannot copy the file from a snapshot to recover the file. However, SnapRestore can quickly recover the file in these conditions. You do not have to spend time making the additional space available in the active file system.

Caution

SnapRestore lets you revert to a snapshot from a previous release of Data ONTAP. However, doing so can cause problems because of potential version incompatibilities and can prevent the filer from booting completely.

Prerequisites

You must meet these prerequisites before using SnapRestore:

- SnapRestore must be licensed on your filer.
- There must be at least one snapshot on your filer that you can select for the reversion.
- The volume to be reverted must be online.
- The volume to be reverted must not be a mirror used for data replication.

General cautions

Be sure that you understand the following facts before using SnapRestore:

- SnapRestore overwrites all data in the file or volume. After you use SnapRestore to revert to a selected snapshot, you cannot undo the reversion.
- If you revert to a snapshot taken before a SnapMirror snapshot, Data ONTAP can no longer perform an incremental update of the mirror; it must re-create the baseline version of the mirror.
- Snapshot deletions are irrevocable. If you delete a snapshot, you cannot recover the snapshot by using SnapRestore.
- After you revert a volume to a selected snapshot, you lose all the snapshots that were taken after the selected snapshot.
- Between the time you enter the `snap restore` command and the time when reversion is completed, Data ONTAP stops deleting and creating snapshots.
- If you are reverting a file from a snapshot, you can delete other snapshots, except for the snapshot you are reverting from.

Caution about reverting the root volume

Because the /etc directory of the root volume contains configuration information about the filer, reverting the root volume might change the filer configuration. In addition, restoring the root volume restores options settings for the entire filer to the settings that were in effect when the snapshot was taken. Reverting a root volume requires rebooting the system.

Preserving configuration files

To avoid reverting the configuration files, complete the following step.

Step	Action
1	Store all data that might need to be reverted in a volume other than the root volume. This ensures that you never need to revert the root volume.

Reverting a root volume

If you need to revert a root volume, complete the following step.

Step	Action
1	If the data you want to revert resides in the root volume, back up the /etc directory to another volume or another filer before using SnapRestore. After you revert the root volume, restore the /etc directory and reboot the filer. If you back up the /etc directory to another volume, you can use the following command to make the filer reboot with that volume as the root volume: `vol options` *`volume`* `root` In this way, when the filer reboots during the reversion, it can use the correct settings in the /etc directory.

Installing the SnapRestore license

How to install the SnapRestore license code

To install the SnapRestore license code, complete the following step.

Step	Action
1	On the server, enter the following command: **`license add`** ***`xxxxxxx`*** *xxxxxxx* is the license code you purchased. This setting persists across reboots.

Reverting a volume to a selected snapshot

How to revert a volume

To revert a volume, complete the following steps.

Note

To cancel volume reversion, press Ctrl-C at any time before you press **y** in Step 8.

Step	Action	
1	Notify network users that you are going to revert a volume so they know that the current data in the volume will be replaced by that of the selected snapshot. **Note** NFS users should unmount the files and directories in the volume before the reversion. If they do not unmount the files and directories, they might get a “stale file handle” error message after the volume reversion.	
2	**If...**	**Then...**
	You know the name of the snapshot for reverting each volume you want to revert	Go to Step 6.
	You want to choose a snapshot from the list of snapshots available for reversion	Go to Step 3.

Step	Action
3	Enter the following command: **`snap restore -t vol`** ***`volume_name`*** `-t vol` specifies the volume name to revert. *volume_name* is the name of the volume to be reverted. Enter the name only, not the complete path. You can enter only one volume name. **Note** Use the -f option to avoid warning messages and prompts to confirm your decision to revert the volume. See the na_snap(1) man page for more information.
4	Press **y** to confirm that you want to revert the volume. **Result:** Data ONTAP displays a list of snapshots.
5	Enter the name of the snapshot for reverting the volume, then go to Step 8. **Result:** Data ONTAP displays the name of the volume to be reverted and the name of the snapshot to be used for the reversion.
6	Enter the following command: **`snap restore -t vol -s`** ***`snapshot_name volume_name`*** `-t vol` specifies the volume name to revert. `-s` *snapshot_name* specifies the name of the snapshot from which to revert the data. You can enter only one snapshot name. *volume_name* is the name of the volume to be reverted. Enter the name only, not the complete path. You can enter only one volume name.
7	Press **y** to confirm that you want to revert the volume. **Result:** Data ONTAP displays the name of the volume and the name of the snapshot for the reversion and, if you have not used the `-f` option, prompts you to decide whether to proceed with the reversion.

Step	Action	
8	**If...**	**Then...**
	You want to continue with the reversion	Press **y**. **Result:** The filer reverts the volume from the selected snapshot. If you are reverting the root volume, the filer reboots the system.
	You do not want to proceed with the reversion	Press **`n`** or **`Ctrl-C`**. **Result:** The volume is not reverted and you are returned to a filer prompt.

Example: `filer> snap restore -t vol -s nightly.0 /vol/vol1`

```
filer> WARNING! This will restore a volume from a snapshot into the
active filesystem.  If the volume already exists in the active
filesystem, it will be overwritten with the contents from the
snapshot.
Are you sure you want to do this? y

You have selected file /vol/vol1, snapshot nightly.0
Proceed with restore? y
```

Result: Data ONTAP restores the volume called vol1 at /vol/vol1.

After a volume is reverted with SnapRestore, all user-visible information (data and attributes) for that volume in the active file system is identical to that contained in the snapshot.

Reverting a file to a selected snapshot

About snap restore file reversion

Using `snap restore` to revert a single file to a selected snapshot is practical when the file is so large that you cannot copy the previous file version from the snapshot to the active file system.

Some notes concerning `snap restore` file reversion:

- ◆ You cannot use SnapRestore for single file reversion on files with NT streams, or on directories.
- ◆ If you restore single files with the `snap restore` command, and then issue the `snap list` command, the `snap list` command might take up to several minutes to complete. You can minimize the amount of time required to complete by using the `snap list -n` command. See the manual (man) pages for more details.

How to revert a file

To revert a single file (rather than a volume), complete the following steps.

Note
To cancel file reversion, press Ctrl-C at any time before you press **y** in Step 10.

Step	Action
1	Notify network users that you are going to revert a file so that they know that the current data in the file will be replaced by that of the selected snapshot. **Note** NFS users who try to access a reverted file without first reopening it might get a “stale file handle” error message after the volume reversion.

Step	Action	
2	**If...**	**Then...**
	You know the name of the snapshot for reverting the file you want to revert	Go to Step 6.
	You want to choose a snapshot from the list of snapshots available for reversion	Go to Step 3.
3	Enter the following command: **`snap restore -t file -r`** ***`restore_as_new_path path_and_file_name`*** `-t file` specifies that you are entering the name of a file to revert. `-r` *restore_as_new_path* restores the file to a location different from (but in the same volume as) the location in the snapshot. For example, if you specify /vol/vol0/vol3/myfile as the argument to `-r`, SnapRestore reverts the file called myfile to the location /vol/vol0/vol3 instead of to the path in vol3 indicated by *path_and_file_name*. *path_and_file_name* is the complete path to the name of the file to be reverted. You can enter only one path name. A file can be restored only to the volume where it was originally. The directory structure to which a file is to be restored must be the same as specified in the path. If this directory structure no longer exists, you must recreate it before restoring the file. **Note** Use the `-f` option to avoid warning messages and prompts to confirm your decision to revert the volume. See the na_snap(1) man page for more information. **Result:** Data ONTAP displays a warning message and prompts you to confirm your decision to revert the file.	
4	Press **y** to confirm that you want to revert the file. **Result:** Data ONTAP displays a list of snapshots.	

Step	Action
5	Enter the name of the snapshot for reverting the file, then go to Step 8. **Result:** Data ONTAP displays the name of the file to revert and the name of the snapshot to be used for the reversion.
6	Enter the following command: **`snap restore -t file -s`** ***`snapshot_name`*** **`-r`** ***`restore_as_path path_and_file_name`*** `-t file` specifies that you are entering the name of a file to revert. `-s` *snapshot_name* specifies the name of the snapshot from which to revert the data. `-r` *restore_as_path* restores the file to a location different from the location in the snapshot. For example, if you specify /vol/vol0/vol3/myfile as the argument to `-r,` SnapRestore reverts the file called myfile to the location /vol/vol0/vol3 instead of to the file structure indicated by the path in *path_and_file_name*. *path_and_file_name* is the complete path to the name of the file to be reverted. You can enter only one path name. A file can be restored only to the volume where it was originally. The directory structure to which a file is to be restored must be the same as specified in the path. If this directory structure no longer exists, you must recreate it before restoring the file. Unless you enter `-r` and a path name, only the file at the end of the *path_and_file_name* is reverted. You can enter only one path name. **Result:** If you have not used the `-f` option, Data ONTAP displays a warning message and prompts you to confirm your decision to revert the file.
7	Press **`y`** to confirm that you want to revert the file. **Result:** Data ONTAP displays the name of the file and the name of the snapshot for the reversion and, if you have not used the `-f` option, prompts you to decide whether to proceed with the reversion.

Step	Action	
8	**If...**	**Then...**
	You want to continue with the reversion	Press **y**. **Result:** The filer reverts the file from the selected snapshot.
	You do not want to proceed with the reversion	Press **`n`** or **`Ctrl-C`**. **Result:** The file is not reverted and you are returned to a filer prompt.

Example:
```
filer> snap restore -t file /vol/vol1/users/jim/myfile
-s nightly.0

filer> WARNING! This will restore a file from a snapshot into the
active filesystem.  If the file already exists in the active
filesystem, it will be overwritten with the contents from the
snapshot.
Are you sure you want to do this? y

You have selected file /vol/vol1/users/jim/myfile, snapshot
nightly.0
Proceed with restore? y
```

Result: Data ONTAP restores the file called myfile to the existing volume and directory structure /vol/vol1/users/jim.

Example:
```
filer> snap restore -t file -s nightly.0
-r /vol/vol2/archive/eng/myfile /vol/vol2/users/jim/myfile

filer> WARNING! This will restore a file from a snapshot into the
active filesystem.  If the file already exists in the active
filesystem, it will be overwritten with the contents from the
snapshot.
Are you sure you want to do this? y

You have selected file /vol/vol1/users/jim/myfile, snapshot
nightly.0
Proceed with restore? y
```

Result: Data ONTAP restores the file called myfile to a new location at /vol/vol2/archive/eng.

After a file has been reverted with SnapRestore, all user-visible information (data and file attributes) for that file in the active file system is identical to that contained in the snapshot.

Obtaining correct incremental backups after reversion

Reversion effects on backup and restore

All files in a reverted volume have timestamps that are the same as those when the snapshot was created. After a reversion, incremental backup and restore operations on the file or volume cannot rely on the timestamps to determine what data needs to be backed up or restored.

Ensuring correct incremental backups after reversion

To ensure correct incremental backups, complete the following steps.

Step	Action
1	Perform a base-level backup of the volume after you restore it.
2	If you need to restore data from tape, use only backups created after the volume was restored.

Data Protection Using SnapMirror 4

About this chapter

This chapter discusses how to use the optional SnapMirror feature of Data ONTAP to copy data from specified volumes or qtrees to other volumes or qtrees and how to access and use the online copied data.

Note

You must have a SnapMirror license to use SnapMirror.

Topics in this chapter

This chapter discusses the following topics:

- “Using SnapMirror to copy a volume to local tape” on page 176
- “How SnapMirror works with the dump command” on page 188
- “Fixing changes to SnapMirror elements” on page 189
- “Creating SnapLock destination volumes” on page 191

SnapMirror overview

SnapMirror modes

The Data ONTAP SnapMirror feature enables an administrator to mirror snapshot images either asynchronously or synchronously.

- When mirroring asynchronously, SnapMirror replicates snapshot images from a source volume or qtree to a partner destination volume or qtree, thus replicating source object data on destination objects at regular intervals.
- When mirroring synchronously, SnapMirror replicates snapshot images from a source volume to a partner destination volume at the same time it is written to the source volume.

  Additionally, you can configure a synchronous SnapMirror replication to lag behind the source volume by a user-defined number of write operations or milliseconds. This option is useful if you are balancing the need for synchronous mirroring with the performance benefit of asynchronous mirroring.

  See "Synchronous SnapMirror" on page 83 for more information.

Reasons for accessing destination volume information

You can access the information on the destination volume or qtree to

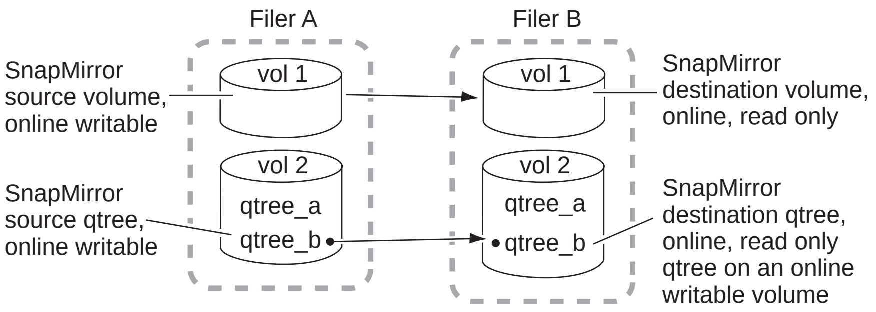
- Provide users quick access to mirrored data in the event of a disaster that makes the source volume or qtree unavailable.
- Update the source to recover from disaster, data corruption (qtrees only), or user error
- Archive the data to tape
- Balance resource loads
- Back up or distribute the data to remote sites

The components of SnapMirror

The basic SnapMirror deployment consists of the following components.

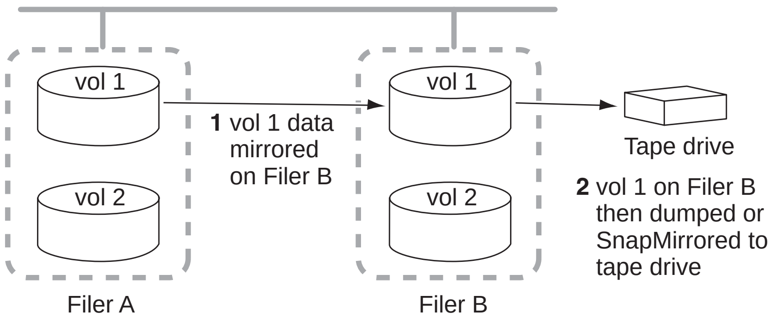
Source volumes or qtrees: SnapMirror source volumes and qtrees are writable data objects whose data is to be replicated. The source volumes and qtrees are the objects normally visible, accessible, and writable by the filer clients.

Destination volumes or qtrees: The SnapMirror destination volumes and qtrees are read-only objects, usually on a separate filer, to which the source volumes and qtrees are replicated. The destination volumes and qtrees are normally accessed by users only when a disaster takes down the source volumes or qtrees and the administrator uses SnapMirror commands to make the replicated data at the destination accessible and writable.


Filer A
Filer B
SnapMirror source volume, online writable
vol 1
vol 1
SnapMirror destination volume, online, read only
vol 2
vol 2
SnapMirror source qtree, online writable
qtree_a
qtree_b
qtree_a
qtree_b
SnapMirror destination qtree, online, read only qtree on an online writable volume

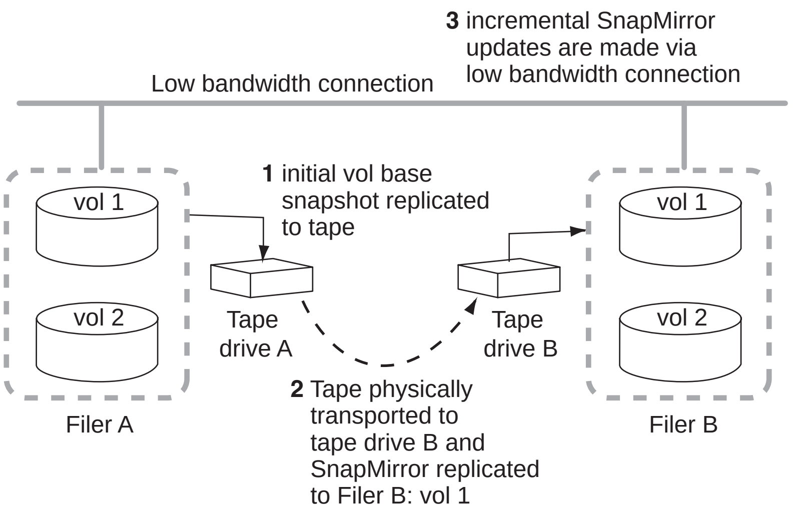
Source to destination to tape variation

A common variation to the basic SnapMirror backup deployment adds a tape backup of the destination volume. By running a tape backup off the SnapMirror destination volume, you do not subject the heavily user-accessed source volume to the performance degradation, system unavailability, and complexity of a direct tape backup.


vol 1
vol 1
1 vol 1 data mirrored on Filer B
Tape drive
2 vol 1 on Filer B then dumped or SnapMirrored to tape drive
vol 2
vol 2
Filer A
Filer B

Source to tape to destination variation

A SnapMirror deployment that supports SnapMirror replication over low-bandwidth connections accommodates an initial mirroring between a source and destination volume via physically transported tape. Once the large-sized base snapshot replication has been carried out, smaller-sized, incremental snapshot updates can be carried out over the low-bandwidth connection.

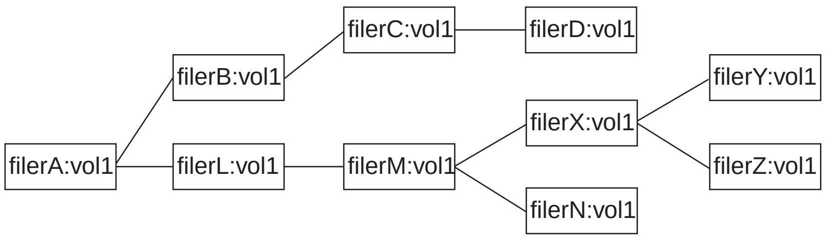

3 incremental SnapMirror updates are made via low bandwidth connection
Low bandwidth connection
1 initial vol base snapshot replicated to tape
vol 1
vol 2
vol 1
vol 2
Tape drive A
Tape drive B
2 Tape physically transported to tape drive B and SnapMirror replicated to Filer B: vol 1
Filer A
Filer B

Cascading destinations variation

A variation on the basic SnapMirror deployment and function involves a writable source volume replicated to multiple read-only destinations. The function of this deployment is to make a uniform set of data available on a read-only basis to users from various locations throughout a network and to allow for updating that data uniformly at regular intervals.

Note

The cascade deployment is supported for SnapMirror volume replication only. It is not supported for SnapMirror qtree replication.



How SnapMirror works

The SnapMirror feature uses the information in the /etc/snapmirror.conf file and the information you enter via the `snapmirror.access` option or the /etc/snapmirror.allow file to establish a relationship between a specified *source* volume or qtree that you want to back up, and the *destination* volume or qtree where the backup is kept.

The SnapMirror feature does the following:

1. Creates a snapshot of the data on the source volume or qtree
2. Copies it to the destination, a read-only volume or qtree
3. Updates the destination to reflect incremental changes on the source, on the schedule you specify

The result of this process is an online, read-only volume or qtree that contains the same data as the source at the time of the most recent update.

What to use SnapMirror for

You might want to copy or use the data stored on a SnapMirror destination if you are in any of the situations described in “When to copy volumes” on page 281. The additional advantages of SnapMirror make it useful in other data retrieval situations, as described in the following table.

Situation	How to use SnapMirror
Disaster recovery: You want to provide immediate access to data after a disaster has made a qtree, volume, or filer unavailable.	You can make the destination writable so clients can use the same data that was on the source volume the last time data was copied.
Data restoration: You want to restore lost data on a qtree or volume source from its mirrored qtree or volume SnapMirror partner.	You can temporarily reverse the roles for the source and destination qtrees or volumes and copy the mirrored information back to its source.
Application testing: You want to use an application on a database, but you want to test it on a copy of the database in case the application damages the data.	You can make a copy of the database to be used in the application testing to ensure that the data on the source cannot be lost.

Situation	How to use SnapMirror
Load balancing: A large number of users need read-only access to a qtree or volume.	You can copy the data in a qtree or volume to multiple volumes or filers to distribute the load.
Off-loading tape backups: You need to reserve all processing and networking resources on a filer for serving NFS and CIFS requests.	After copying data on the source filer, you can back up the data in the destination to tape. This means that the source filer does not have to allocate resources for performing backups.
Remote access to data: Users who need read access to a volume are distributed over a large geographical area.	You can copy the source volume to other filers that are geographically closer to the users. Users accessing a local filer can read the data using less resource time than if they connected to a distant filer.

Differences between the vol copy command and SnapMirror

The `vol copy` command and the SnapMirror feature carry out similar functions. They both enable the administrator to copy snapshots of data from a source filer volume to a destination filer volume, as long as the volumes are of the same type. When using the `vol copy` and `snapmirror` commands, the source and destination volumes must both be traditional volumes or flexible volumes. However, these commands differ in several important ways.

- The `vol copy` feature does not require an additional license.
- The SnapMirror feature supports automated and scheduled updates of snapshot data mirrored between the source and destination filer volumes.
- The SnapMirror feature supports incremental snapshot updates between source and destination filer volumes.
- The SnapMirror feature supports qtree-level replication between the source and destination systems.

Differences between SnapMirror volume replication and qtree replication

You can configure SnapMirror replication for either whole volumes on a filer or individual qtrees on a volume.

SnapMirror volume replication: SnapMirror volume replication has the following characteristics:

- SnapMirror volume replication can only occur with volumes of the same type, that is, both traditional volumes or both flexible volumes.
- SnapMirror volume replication copies a snapshot of a source volume and all its qtrees to a destination.
- A destination volume set up for SnapMirror volume replication must first be set to restricted, read-only status.
- SnapMirror volume replication is a block-for-block replication; it transfers the file system verbatim. Therefore, older releases of Data ONTAP cannot understand file system transfers from a later release of Data ONTAP.

Note

See the caution in "Prerequisites to running SnapMirror" on page 89 for information about source and destination upgrading order.

SnapMirror qtree replication: SnapMirror qtree replication has the following characteristics:

- SnapMirror qtree replication occurs between qtrees regardless of the type of volume (traditional or flexible) in which the qtree resides.
- SnapMirror qtree replication is asynchronous only.
- SnapMirror qtree replication copies only the contents of an individual qtree to a destination.
- If you need to mirror only the data stored on an individual qtree, then SnapMirror replication of that individual qtree uses less disk space on the filer.
- A destination qtree is read-only, but the volume on which it is located must be online and writable.
- SnapMirror qtree replication can be set up for a maximum of 255 qtrees on any one volume.
- SnapMirror qtree replication is a logically replication; all of the files and directories in the source file system are created in the destination file system. Therefore, replication can occur from any release to any release.

Note

If the source file system contains a file type that cannot be represented on the destination file system, the replication will fail. For example, Data ONTAP 7.0 supports files up to 16 TB in size, whereas, previous Data ONTAP versions support files up to 4 TB. If the source filer is running Data ONTAP 7.0, the qtree you want to replicate contains a file greater than 4 TB, and the destination filer is running an earlier version of Data ONTAP, the replication will fail.

Maximum number of simultaneous replications on a filer

The maximum number of simultaneous replications that each filer model can support is shown in the following table.

Model	Number of simultaneous transfers allowed
F85	4
F87	4
F720	4
F740	4
F760	4
F810	8
F820	8
F825	8
F840	16
F880	16
FAS250	4
FAS270	8
FAS920	8
FAS940	16
FAS960	16
FAS980	16

Model	Number of simultaneous transfers allowed
R100	128
R150	128
R200	128

How transfers beyond the limit are handled: If more than the allowed number of SnapMirror volume or qtree replications are scheduled to run at the same time, each additional transfer will generate an error message stating that resource limits have been reached. Each transfer beyond the limit will re-attempt to run once per minute until it succeeds, SnapMirror is turned off, or the update is terminated.

Synchronous SnapMirror

What synchronous SnapMirror is

Synchronous SnapMirror is a SnapMirror feature in which the data on one filer is replicated on another filer at, or near, the same time it is written to the first filer. Synchronous SnapMirror synchronously replicates data between single filers or clustered filers situated at remote sites using either an IP or a Fibre Channel connection.

Note

You can use Synchronous SnapMirror only on volumes, not qtrees.

How SnapMirror replicates data synchronously

Before Data ONTAP saves data to disk, it collects written data in NVRAM. Then, at a point in time called a consistency point, it sends the data to disk. When the Synchronous SnapMirror feature is enabled, the source filer forwards data to the destination filer as it is written in NVRAM. Then, at the consistency point, the source filer sends its data to disk and tells the destination filer to also send its data to disk. Finally, the source filer waits for the destination filer to acknowledge that it sent data to disk before continuing with the next write.

How network problems are handled

If there are problems with your network, your synchronous replication might go into an asynchronous mode. Ordinarily, the source and destination filers periodically communicate with each other. In the event of a network outage, synchronous SnapMirror goes into an asynchronous mode if the periodic communication is disrupted. When in asynchronous mode, the source filer tries to communicate with the destination filer once every minute until communication is reestablished. Once reestablished, the source filer asynchronously replicates data to the destination every minute until a synchronous replication can be reestablished.

Considerations before running synchronous SnapMirror

You should consider the following criteria before using the synchronous SnapMirror feature:

Not all filers are supported: Synchronous SnapMirror is not supported on the following filers:

- F87
- F700 Series
- F810
- F820

Not all data requires synchronous transfers: You decide which data you want synchronously replicated. For example, you might synchronously replicate database data and asynchronously replicate home directories. If the home directories contain important log data, you might synchronously replicate it, but adjust the synchonicity of the transfers to maintain performance levels. Synchronous SnapMirror allows a user-defined lag time of a number of operations or milliseconds before transferring data. See "snapmirror.conf file entry syntax" on page 107 and the na_snapmirror.conf(5) man page for details.

The source filer and destination filer should be adequately configured for the replication traffic: Ideally, the source filer and destination filer should have the same configuration (type of filer and disk geometry).

The type of filer and disk geometry of the destination filer impacts the perceived performance of the source filer. Therefore, the destination filer should have the bandwidth for the increased traffic and for message logging. Log files are kept on the root volume; therefore, you should ensure that the root volume spans enough disks to handle the increased traffic. NetApp recommends that your root volume spans four to six disks.

Note

Disk geometry is less of an issue using flexible volumes because the root volume can be expanded on the destination. See the *Storage Management Guide* for more information about flexible volumes.

The transport should be optimized for best performance: Synchronous SnapMirror can support traffic over Fibre Channel and IP transports. SnapMirror also allows multipathing, giving you the ability either to balance the load between two paths or to reserve the second path for failover. For optimizing performance, you can use the best route available between the source and destination filer and you can restrict the route to traffic between the two filers. See "Using SnapMirror over multiple paths" on page 114 for details.

SnapMirror commands and configuration files

About SnapMirror commands

This section lists the Data ONTAP commands and configuration files associated with the SnapMirror features that are described in this chapter.

Commands to set up SnapMirror

You can use the following commands and configuration files to set up SnapMirror replication.

Command or File	Function
`license add`	Enables the SnapMirror license on the source and destination filers. See "Entering license codes" on page 116.
`options snapmirror.access` or /etc/snapmirror allow file	Specifies the host names of filers that are allowed to copy data directly from the source filer. See "Specifying destination filers on the source filer" on page 101.
/etc/snapmirror.conf file	Specifies the destination volumes and qtrees to be replicated to and the schedule on which the destinations are updated. See "Defining source and destination through snapmirror.conf" on page 104.

`snapmirror on`
 	
or
 	
`options snapmirror.enable`
	Enables SnapMirror on the source and destination filers. See "Enabling SnapMirror" on page 116.

`vol create`
 	
and
 	
`vol restrict`
	In combination, create the restricted read-only volumes that are required as destinations for SnapMirror volume replications.
	
	See "Initializing a SnapMirror destination" on page 118.

Command or File	**Function**
`snapmirror initialize`	Begins the initial complete (baseline) SnapMirror snapshot replication from a source volume or qtree to a destination volume or qtree. See "Initializing a SnapMirror destination" on page 118 for more information.

SnapMirror management commands

You can use the following commands to perform SnapMirror management tasks:

Command or file	**Function**
`snapmirror update`	Performs a manual (unscheduled) update of the destination. See "Updating a destination manually" on page 126.
`snapmirror status`	Displays the status of SnapMirror data transfers. See "Checking SnapMirror data transfer status" on page 132.
/etc/log/snapmirror	Displays the SnapMirror data transfer history. See "Checking SnapMirror data transfer logs" on page 141.
`snapmirror abort`	Stops a transfer that is in progress. See "Aborting a SnapMirror transfer" on page 146.
/etc/snapmirror.conf	Enables you to specify or modify scheduled updates for one or more volumes or qtrees. See "Turning off or changing scheduled updates for volumes or qtrees" on page 151.

SnapMirror shutdown commands

You can use the following commands to temporarily or permanently shut down SnapMirror processes on a filer, volume or qtree:

Command	Function

`snapmirror off`
or
`options snapmirror.enable off`
	Turns off SnapMirror functionality for a specified filer. See "Turning off SnapMirror updates for the filer" on page 154.

`snapmirror break`	Converts the destination to a writable volume or qtree, breaking the SnapMirror relationship between the source and destination. See "Converting a destination to a writable volume or qtree" on page 156.
`snapmirror release`	Releases for deletion SnapMirror snapshots on former source volumes or qtrees. See "Releasing partners from a SnapMirror relationship" on page 159.

SnapMirror advanced function commands

You can use the following commands to perform advanced functions with SnapMirror:

Command or File	Function
`snapmirror quiesce`	Stabilizes the contents of a destination just prior to a snapshot by allowing ongoing data transfers to finish and temporarily preventing new transfers from beginning. This action ensures a manual snapshot of a stable database. See "Stabilizing (quiescing) destinations before taking a snapshot" on page 148.
`snapmirror resume`	Resumes normal data transfer to a destination after it has been quiesced. See "Resuming transfers after quiescing a destination" on page 150.

Command or File	Function
`snapmirror resync`	Returns a former destination volume or qtree to the SnapMirror relationship after the `snapmirror break` command and resynchronizes its contents with the source without repeating the initial transfer. See “Resynchronizing SnapMirror” on page 161.

/etc/snapmirror.conf file
`snapmirror initialize`
`snapmirror destinations`
	These commands set up a cascading series of SnapMirror destinations. They make a uniform set of data available on a read-only basis to users from various locations throughout a network and they update that data uniformly at regular intervals. See “Copying from one destination to another in a series (cascading)” on page 170.

`snapmirror store`
and
`snapmirror use`
	These commands copy a volume to local tape and continue the backup on subsequent tapes if necessary. See “Using SnapMirror to copy a volume to local tape” on page 176.

`snapmirror retrieve`
and
`snapmirror use`
	These commands initialize or restore a volume from local tape. See “Initializing a SnapMirror destination via local tape” on page 181.

Considerations when planning and running SnapMirror

Prerequisites to running SnapMirror

The following prerequisites must be met before you can run SnapMirror:

- You must purchase and enable the SnapMirror license.

  If the SnapMirror source and destination are on different filers, you must purchase a license and enter the SnapMirror license code on each filer. For information on enabling the SnapMirror license, see "Entering license codes" on page 116.

- For SnapMirror volume replication, you must create a restricted volume to be used as the destination volume.

  SnapMirror does not automatically create a volume. See the section on organizing data using volumes and qtrees in the *Storage Management Guide* for information about how to create volumes.

- For SnapMirror volume replication, the destination volume must run under a version of Data ONTAP equal to or later than that of the SnapMirror source volume.

  Caution
  If you upgrade your filers to a newer version of Data ONTAP, upgrade the filers of SnapMirror destination volumes before you upgrade the filers of SnapMirror source volumes.

- For SnapMirror qtree replication, you must not create a qtree to be used as a destination qtree; the `snapmirror initialize` command creates the destination qtree automatically.
- For SnapMirror qtree replication, the destination qtree must run under Data ONTAP version 6.2 or later.
- The name and IP address of the source filer must be in the /etc/hosts file of the destination filer or must be resolvable by way of DNS or `yp`.

Restrictions

When planning SnapMirror configuration, consider the following restrictions:

- The source volume must be online.

  See the *Storage Management Guide* for information about how to put a volume online.

- For SnapMirror volume replication, the capacity of the destination volume must be greater than or equal to the capacity of the source volume. See "Verifying the size of each volume" on page 284 for information about using the `vol status -b` command to check the capacity of the source volume and destination volume. See the *Storage Management Guide* for information about how to add disks to a volume. See “Initializing a SnapMirror destination” on page 118 for information about disk sizes.

- To support SnapMirror qtree replication, the destination volume must contain 5% more free space than the source qtree consumes.
- The SnapMirror destination volume cannot be the root volume of a filer. The SnapMirror source volume, however, can be the root volume.
- A destination qtree can be on the root volume, but the /etc qtree cannot be a destination qtree.
- Destination qtrees names cannot
  - Consist of “*” or “-” or “/etc”
  - Contain the character “.” or the character combination “->” (this restriction applies to source qtrees too)
  - Contain space or tab characters
  - Be longer than 64 characters
  - Be specified as “/vol/*volname*/” (with no qtree name)
  - Be specified as “*volname*/*qtree_name* (with no /vol/)”
- There must be a functional network to transfer data between two different filers.
- Each filer model supports a specified number of `snapmirror update` or `vol copy` operations at a time. (See “Maximum number of simultaneous replications on a filer” on page 81 for the number of copy operations each filer can support.)

Cautions

Be advised of the following cautions:

- Do not delete snapshots that SnapMirror creates in the source volume and then copies to the destination. The most recent SnapMirror snapshot is referred to as the newest common snapshot (NCS). Incremental changes to the destination depend on the newest common snapshot. If SnapMirror cannot find the required snapshot on the source, it cannot perform incremental changes to the destination.
- Do not use the `snapmirror release` or `snapmirror break` command on the destination volume or qtree unless you no longer need to copy incremental changes from the source. The destination must be actively functioning as a destination to receive incremental updates.

- Do not restrict or take the destination volume offline while SnapMirror is configured to transfer. Taking the destination offline prevents SnapMirror from performing updates to the destination.

Recommendations

Follow these recommendations to copy data efficiently:

- For smoothest performance, schedule SnapMirror snapshot and updates times (in the etc/snapmirror.conf file) to be different from the time of any regular system snapshots (set through the `snap_sched` command) on the source filer.

  Note

  Avoid scheduling a snapshot at the same time SnapMirror is transferring the data in the snapshot. SnapMirror locks snapshots before they are transferred; therefore, if SnapMirror is transferring data in a snapshot when an automatic snapshot is scheduled, the automatic snapshot is not created.

- For optimum SnapMirror volume replication performance, make sure that the SnapMirror source volume and destination volume contain disks of the same size, organized in the same RAID configuration.

  If the SnapMirror source and destination are qtrees, volume size and configuration do not make any difference.
- If you plan to do SnapMirror transfers through a firewall, you might need to know the port number used by SnapMirror. SnapMirror listens for connections on port 10566.

Ensuring accessible destinations when using CIFS

Directories on all SnapMirror source volumes that support CIFS clients must be in Unicode format before being replicated to a destination; otherwise, the directories in the read-only destination volume will not be in Unicode format and attempts through CIFS to access directories and open files on the destination volume can receive “access denied” errors.

You can ensure that both source volume and destination volume directories are in Unicode format using the following methods:

- On the filer console for the source volume, enter the following two commands:
  - `vol options` *volname* `convert_ucode on` to convert any existing directories on the source volume to Unicode format.
  - `vol options` *volname* `create_ucode on` to automatically create any new directories on the source volume in Unicode format.

- Another option is to make sure that all directories on the source volume that will be accessed by CIFS clients are accessed by a CIFS client before initial replication to a destination. Such access on a writable source volume automatically converts that directory to Unicode format.

Adjusting the TCP window size

A TCP window is the amount of data that a source can send on a connection before it requires acknowledgement from the destination that the data was received. The default window size for SnapMirror operations is 1,994,752 bytes. This is a large window and, in some networks, particularly WANs, can result in the termination of the SnapMirror transfer or very low throughput. To change the TCP window size to fit your network better and avoid these problems, complete the following steps.

Step	Action
1	Calculate a TCP window size that works well in your network using the following fomula: WindowSize = (RoundTrip Delay) (DesiredRate) **Example:** If your average round trip delay is 100 milliseconds and you want a rate of 10 Mbps, you should set the TCP window size to 125,000 bytes as the following equation shows: 0.1 * 10,000,000/8 = 125,000
2	Adjust the TCP window size by entering the following command: **`options snapmirror.window_size`** ***`rate`*** *rate* is the desired TCP window size.

Setting up a basic SnapMirror operation

Completing a basic SnapMirror setup

This section describes the use of the SnapMirror feature for basic data replication between the SnapMirror source and destination, and basic recovery operations.

Note

If your source volumes contain directories that are accessed by CIFS clients, ensure that those directories are in Unicode format before carrying out the SnapMirror replication of that volume. See “Ensuring accessible destinations when using CIFS” on page 91.

To complete a basic SnapMirror setup, complete the following steps.

Step	Action
1	Make sure you have purchased a SnapMirror license for both the source and destination filers.
2	In both the source filer and the destination filer consoles, use the `license add` command to enable the SnapMirror license on the source and destination filers. **`license add`** ***`snapmirror_license_code`*** For more information see “Entering license codes” on page 116.
3	On the source filer console, use the `options snapmirror.access` command to specify the host names of filers that are allowed to copy data directly from the source filer. For example: **`options snapmirror.access host=d_filerA`** For more information see “How to specify destination filers on the source filer” on page 101.

Step	Action
4	Through the Data ONTAP AdminHost, create or edit the /etc/snapmirror.conf file on the destination filer to specify the volumes and qtrees to be copied and the schedule (*minute hour day_of_month day_of_week* or `sync`) on which the destination is updated. For example, the following entry specifies snapshot mirroring from volume 0 of s_filerA to volume 1 of d_filerA at a maximum of 2,000 kilobits per second 15 minutes past every hour, Monday through Friday. **`s_filerA:vol0 d_filerA:vol1 kbs=2000,restart=always 15 * * 1,2,3,4,5`** If you want to synchronously mirror volume 0 to volume 1, you use a command similar to the following: **`s_filerA:vol0 d_filerA:vol1 - sync`** For more information on schedule entries in the destination filer's /etc/snapmirror.conf file, see "Defining source and destination through snapmirror.conf" on page 104 and the na_snapmirror.conf(5) man page.
5	In both the source filer and destination filer console, use the `snapmirror on` command to enable SnapMirror on the source and destination filers. For more information, see "Enabling SnapMirror" on page 116.

Step	Action
6	Prepare the destination filer appropriately, depending on whether you are setting up SnapMirror volume or qtree replication. ◆ If you are setting up SnapMirror volume replication, in the destination filer console, use the `vol create` command to create a volume to be the destination for the volume on the source, then use the `vol restrict` command to mark the volume as restricted. ◆ If you are setting up SnapMirror qtree replication, make sure that the volume on the destination filer where you want to replicate a qtree with SnapMirror is online and not restricted. Do not manually create a destination qtree. For more information, see the information about creating volumes in the volume management chapter of the *Storage Management Guide*; and see the information on restricting volumes in “How to initialize a SnapMirror destination” on page 119.

Step	Action
7	In the destination filer console, use the `snapmirror initialize` command to create an initial complete (baseline) copy of the source on the destination and start the mirroring process. **SnapMirror volume replication example:** Invoking the following command line transfers a complete copy of the source volume (vol0 on filerA) to the destination volume (vol2 on filerB). The destination volume must be configured as restricted and read-only. **`snapmirror initialize -S filerA:vol0 filerB:vol2`** **SnapMirror qtree replication example:** Invoking the following command line creates a destination qtree (qtree4 on vol1on filerB) and transfers a complete copy of the qtree source (qtree4 on vol1 on filerA) to that destination qtree. The volume in which the destination qtree is created must be online and writable. **`snapmirror initialize -S filerA:/vol/vol1/qtree4`** **`filerB:/vol/vol1/qtree4`** After you invoke the `snapmirror initialize` command, the scheduled snapshot mirroring that you specified in Step 4 will automatically update the destination volume or qtree at the specified times. See "Initializing a SnapMirror destination" on page 118 for more information.
8	In the event that the SnapMirror source volume or qtree is disabled, you can use the `snapmirror break` command to make the destination volume or qtree writable and able to provide continuous file and data service to the clients who are no longer able to access the disabled source filer. For more information, see "Converting a destination to a writable volume or qtree" on page 156.

Converting asynchronous SnapMirror to synchronous SnapMirror

If you have an asynchronous SnapMirror relationship and you want to convert it to a synchronous SnapMirror relationship, complete the following step.

Step	Action
1	On the administration host, edit the snapmirror.conf file on the destination filer to change the schedule to `sync`. **Result:** Data ONTAP uses the new schedule to change the asynchronous SnapMirror to a synchronous SnapMirror.

Configuring for Fibre Channel use

If you are using Fibre Channel connections between the source filer and the destination filer, the filers must use 2352 (X1024) adapters. After configuring the Fibre Channel adapters, you can perform asynchronous and synchronous SnapMirror replication across the Fibre Channel connection. To configure the filers for Fibre Channel use, complete the following steps.

Step	Action
1	Install the Fibre Channel adapters in the source and destination filer. See the hardware and service guide for your filer for installation instructions and the *System Configuration Guide* at the NetApp on the Web (NOW) site at http://now.netapp.com/ to ensure you install the adapter in the correct slot.
2	Connect the filers to Fibre Channel switches. See the hardware and service guide for your filer for more information.

Step	Action

3	On the console of each switch, configure the switch for a default configuration with no zoning and add long-distance Extended Fabric keys. See your switch documentation for configuration information.
	
	Note
	Ensure that the Fibre Channel switches are in fabric mode and support the Simple Name Service (SNS) protocol with support for symbolic names.
	
	Example: The following Brocade switch commands might be used to configure a Brocade switch.
	
	`switch_con> switchdisable`
	
	`switch_con> configdefault`
	
	`switch_con> fastboot`

4	Reboot both filers.

5	Configure the Fibre Channel adapter on each filer by entering the following command at the console for each filer:
	
	`ifconfig ql`*`Xa ipaddr`*` netmask 255.255.255.0 up`
	
	X is the filer slot number.
	
	a is either port a or port b (labeled on the adapter).
	
	ipaddr is the IP address of the adapter.
	
	Note
	Ensure that IP addresses for the ports on the same fabric have the same net number.
	
	Example: The following commands configure port a of two Fibre Channel adapters. The net address is 10.10.10.0.
	
	On filer A:
	
	`ifconfig ql3a 10.10.10.24 netmask 255.255.255.0 up`
	
	On filer B:
	
	`ifconfig ql3a 10.10.20.24 netmask 255.255.255.0 up`

Step	Action

6	Ensure that the two filers can communcate by using the `ping` command.
	
	Example: The IP address on the destination filer is 10.10.20.24.
	
	`source> ping 10.10.20.24`
	`10.10.20.24 is alive`

7	Follow the steps in “Completing a basic SnapMirror setup” on page 93 to complete the SnapMirror setup.

How to troubleshoot reset messages: After configuring Fibre Channel adapters, you might see adapter reset messages similar to the following:

```
Sun Sep  7 20:30:00 PDT [filer1: cf.nm.nicReset:warning]:
Interconnect nic 2 is being reset
Sun Sep  7 20:30:03 PDT [filer1: nic_mgr:info]: Saving reg dump for
Qlogic VI Fibre Channel Adapter(2) to file /etc/ispfcvi_regdump.5
```

If you see an adapter reset message like this, check whether you have any of the problems listed in the following table and use the recommended solutions as needed.

Problem	Solution
The cable from the Fibre Channel adapter to the Fibre Channel switch is disconnected.	Connect the cable to the switch.
The Inter Switch Link (ISL) connecting the two Fibre Channel switches is disconnected.	Connect the ISL.
The snapmirror.conf file specifies an IP address or host name that is not configured on the Fibre Channel fabric.	Edit the snapmirror.conf file to use the correct IP address or host name.

Problem	Solution
The host name used in the snapmirror.conf file does not match the IP address in the /etc/hosts file on the filer.	Ensure that the IP addresses used for the hosts on the Fibre Channel fabric are all in the same subnet.

Specifying destination filers on the source filer

How to specify destination filers on the source filer

There are two methods of specifying destination filers on the source filer:

- You can specify the destination filers that are allowed access to the source filer by using the `snapmirror.access` option. This option specifies which SnapMirror destination filers can initiate transfers, and which network interfaces they can use.This is the preferred method for controlling SnapMirror access on a SnapMirror source filer.
- You can generate a snapmirror.allow file in the /etc/ directory on the source filer. The /etc/snapmirror.allow file specifies the host names of filers that are allowed to copy data directly from the source filer.

  If the `snapmirror.access` option is set to `legacy` (the default setting), the snapmirror.allow file defines the access permissions.

Specifying destinations using the snapmirror.access option

To specify the SnapMirror destinations allowed access to the SnapMirror source using the `snapmirror.access` option, complete the following step.

Step	Action
1	On the source filer, enter the following command: **`options snapmirror.access`** ***`access_specification`*** The syntax for specifying which filers are allowed access to the server is the same for SNMP, Telnet, and rsh, and is described in the na_protocolaccess(8) man page. For more information about the `options` command, see the na_options(1) man page. This option persists across reboots.

Example: If you want SnapMirror to copy data locally on filerA and to another filer named filerB, enter the following at the prompt on filerA:

```
filerA> options snapmirror.access host=filerA,filerB
```

Specifying destinations using the snapmirror.allow file

To specify the SnapMirror destinations allowed access to the SnapMirror source using the /etc/snapmirror.allow file, complete the following steps.

Step	Action
1	If the snapmirror.allow file does not already exist, use a text editor to create a file called snapmirror.allow in the /etc directory on the root volume of the source filer.
2	Add the name of each filer to which you are replicating data, on its own line, to the /etc/snapmirror.allow file. You do not have to add the name of the local filer.
3	Save edits to the file.

Example: If you want SnapMirror to copy data locally on filerA and to other filers named filerB and filerC, create a /etc/snapmirror.allow file on filerA with the following lines added:

```
filerB
filerC
```

Entries in the /etc/snapmirror.allow file are case-sensitive. You can use the `hostname` command on the destination filer to find the correct format for each entry in the /etc/snapmirror.allow file.

Resolving host names to their IP addresses

By default, the SnapMirror feature checks host names in the /etc/snapmirror.allow file against the host name sent from the destination filer. Alternatively, you can set SnapMirror to resolve the host names in the /etc/snapmirror.allow file to their IP addresses and compare them with the IP address of the destination filer.

The `snapmirror.checkip.enable` option controls how the host names are checked. When the option is off, which is the default, the entries in the /etc/snapmirror.allow file must match the host name of the destination filer reported by the `hostname` command. When the option is on, the source filer resolves the names in the snapmirror.allow file to IP addresses and then checks for a match with the IP address of the requesting destination filer. In this mode, literal IP addresses (for example, 123.45.67.89) and fully qualified names (for example, filerA.acme.com) can be valid entries in the /etc/snapmirror.allow file.

The /etc/snapmirror.allow file entry must map to the IP address of the originating network interface on the destination filer. For example, if the request comes from the IP address of a Gigabit Ethernet interface e10 named filerA-e10, then the /etc/snapmirror.allow file must contain “filerA-e10” or “filerA-e10.acme.com” so the name resolves to the correct IP address.

A local SnapMirror relationship, between two volumes on the same filer, does not require an entry in the /etc/snapmirror.allow file.

To configure SnapMirror to resolve host names to their IP addresses, complete the following step.

Step	Action
1	On the source filer, enter the following command: `options snapmirror.checkip.enable on`

Defining source and destination through snapmirror.conf

What the snapmirror.conf file does

The /etc/snapmirror.conf file defines the relationship between the source and the destination, the schedule used by the destination to copy data, and the arguments that control SnapMirror when copying data. This file resides on the destination filer.

How to distribute a snapmirror.conf file

You can create a single /etc/snapmirror.conf file for your site and copy it to all the filers that use SnapMirror. The /etc/snapmirror.conf file can contain entries pertaining to other filers. For example, the /etc/snapmirror.conf file on filerB can contain an entry for copying a volume from filerC to filerD. When filerB reads the /etc/snapmirror.conf file, it ignores the entries for other filers. However, each time the file is read, a warning message is displayed on the system console for each line that is ignored.

Limitation on entries for each filer

There is no limit on the total number of entries in the /etc/snapmirror.conf file; however, there is a limit of 600 entries for each filer in the /etc/snapmirror.conf file. There is a 150-entry limit for each filer for Data ONTAP 6.2 and a 210-entry limit for each filer for Data ONTAP 6.3 through Data ONTAP 6.4.2. Entries beyond the entry limit for each filer are ignored and a warning message is displayed on the system console.

If you have a cluster, the limit on the number of entries applies to the filer pair combination. For example, if your cluster is running Data ONTAP 6.4.2, the 210 entry limit is shared by both filers in the cluster. If one filer is using 120 entries, the other filers has 90 entries available for its use.

Note

This limitation is different from the maximum number of of simultaneous replications you can have on a filer. For that information, see “Maximum number of simultaneous replications on a filer” on page 81.

When changes to the snapmirror.conf file take effect

If SnapMirror is enabled, changes to the /etc/snapmirror.conf file take effect within two minutes. If SnapMirror is not enabled, changes to the /etc/snapmirror.conf file take effect immediately after you enter the `snapmirror on` command to enable SnapMirror.

Creating and editing the snapmirror.conf file

To create or edit the /etc/snapmirror.conf file, complete the following steps.

Step	Action
1	If it does not already exist, use a text editor to create a file called snapmirror.conf in the /etc directory on the root volume of the destination filer.
2	If you want to add comments to the /etc/snapmirror.conf file, precede the comment with a pound sign (#). **Example:** `# Replicating from filerA`

Step	Action
3	For each destination volume or qtree that is to be located on this filer, type an entry specifying the source, destination, characteristics and schedule of the data transfer on one line using the following syntax: ***source_filer*:{*source_volume* | /vol/*volume_name*/*qtree_name*}** ***dest_filer*:{*dest_volume* | /vol/*volume_name*/*qtree_name*}** ***arguments schedule*** For a full description of the snapmirror.conf file entry syntax see "snapmirror.conf file entry syntax" on page 107. **Note** When you use a qtree as a source, you do not create another qtree to serve as the destination. SnapMirror automatically creates one for you, using the name you specify. However, you must specify the name of the qtree destination in the volume where you want it to be placed, either in the /etc/snapmirror.conf file or in the `snapmirror initialize` command. **Note** Do not specify more than 254 destination qtrees for any one volume in your /etc/snapmirror.conf file entries. If you want to combine all non-qtree data in a volume into a qtree to use as a SnapMirror source, use this syntax: ***source_filer*:/vol/*source_volume*/-** ***dest_filer*:/vol/*dest_volume*/*qtree_name*** The dash (-) character indicates all non-qtree data in the specified volume. **Note** The data in /vol/*source_volume*/- qtree can only be a SnapMirror source, never a destination. That is, after you create the /- qtree, you can copy data from it to another qtree you name, but you cannot copy data to it from another qtree.
4	Save edits to the file.

snapmirror.conf file entry syntax

The syntax for entries in the snapmirror.conf file is as follows:

```
src_filer:/vol/src_vol/[src_qtree]
dest_filer:/vol/dest_vol[/dest_qtree] [arguments] [sched]
```

The parameters in the entries are:

Parameter	Description
src_filer	The name of the filer from which you are copying data. SnapMirror uses the /etc/hosts file or the database used by DNS and NIS for name resolution. When SnapMirror searches for the source filer name, it should find the IP address for the source filer on the network over which you want the transfer to occur. **Example:** Assume you created a private network connecting a source filer and a destination filer and you named the interface on the source filer filerA-e0. The interface name filerA-e0 is what you enter in the *source_filer* field.
src_vol /*src_qtree*	The name of the volume or qtree that you are copying. Use the volume name alone for volumes. Use the full path name for qtrees. **Example:** If the name of the volume is vol1, enter vol1 in the *src_vol*/*src_qtree* field. If the name of the qtree is qtree3, and it is contained in volume vol3, enter the full path, /vol/vol3/qtree3, in the *src_vol*/*src_qtree* field.
dest_filer	The host name of the filer to which the data is copied. The name you use must be the exact host name of the destination filer. The *dest_filer* field is case-sensitive. You can use the `hostname` command on the destination filer to determine what you enter for this field. **Example:** If the name of the destination filer is filerA, enter filerA in the *dest_filer* field.

Parameter	Description
dest_vol [/*dest_qtree*]	The name of the destination volume or qtree to which you are copying data. Use the volume name alone for volumes. Use the full path name for qtrees. **Example:** If the name of the volume is vol1, enter vol1 in the *dest_vol*[/*dest_qtree*] field. If the name of the qtree is qtree4, and it is in vol2, enter the full path, /vol/vol2/qtree4, in the *dest_vol*[/*dest_qtree*] field.
kbs=*kbs*	Maximum transfer speed, in kilobytes per second, that Data ONTAP can use to transfer data. The `kbs` and `restart` arguments are expressed as a comma-separated list of *name*=*value* pairs, with no spaces, for example: `kbs=2000,restart=always`
restart= {never | always | default}	Restart mode that SnapMirror uses to continue an incremental transfer from a checkpoint if it is interrupted. There are three options: ◆ Never—Transfers are always restarted from the beginning of a transfer and never from where they were before an interruption. This mode is useful if you must have the latest data on the destination. ◆ Always—Transfers are always restarted if possible from where they were before an interruption. This mode is useful for copying large volumes. ◆ Default—Transfers are restarted if they do not conflict with a scheduled transfer. This is the recommended option. SnapMirror always restarts from where the transfer was before an interruption. However, it does not restart if the restart occurs after a scheduled transfer, because the scheduled transfer gives more up-to-date data than the restarted transfer. The `kbs` and `restart` arguments are expressed as a comma-separated list of *name*=*value* pairs, with no spaces, for example: `kbs=2000,restart=always`

Parameter	**Description**
cksum= {none | crc32c}	Selects the checksum algorithm that is used to protect SnapMirror transmitted data.
outstanding= {*x*ops | *x*ms | *x*s}	Determines the synchronicity level of the synchronous SnapMirror. The ops suffix allows *x* number of outstanding write operations before forcing the client to wait for an acknowledgement. The ms and s suffixes allow *x* number of milliseconds and seconds, respectively.
visibility_ interval={*x*s | *x*m | *x*h}	Determines the amount of time before an automatic snapshot is created on the source volume that is synchronously mirrored. With synchronous SnapMirror, changes to the source volume do not show immediately on the destination volume, even though the changes have been mirrored. The changes are shown only after the source filer takes an automatic snapshot of the source volume. This happens every three minutes by default. You can change the interval for automatic snapshots, but performance can degrade if you set smaller intervals because more snapshots are taken more often. The smallest interval you can set is 30 seconds. The s, m, and h suffixes specify seconds, minutes, and hours, respectively.
wsize=*size*	Determines the TCP window size used by a connection. Different from the `snapmirror.window_size` option which determines the TCP window size for allSnapMirror replications. Use the formula shown in "Adjusting the TCP window size" on page 92 to calculate a TCP window size that works well for a particular connection. TCP window size is in bytes.

Parameter	Description

schedule	Determines the schedule the destination uses for updating data. The schedule is mandatory.
	
	The schedule consists of four space-separated fields in order:
	
	minute hour dayofmonth dayofweek
	
	minute can be a value from 0 to 59.
	
	hour can be a value from 0 (midnight) to 23 (11 p.m.).
	
	dayofmonth can be a value from 1 to 31.
	
	dayofweek can be a value from 0 (Sunday) to 6 (Saturday).
	◆ Multiple values, separated by commas, can be entered for any field.
	◆ All possible values for a field can be applied with an asterisk (*). If you specify an asterisk in each field of the schedule, SnapMirror updates the destination every minute.
	◆ A single dash (-) in any field means "never" and prevents this schedule entry from executing. (This is useful if you want the server to appear in the /etc/snapmirror.conf file so that `snapmirror update` can find it, but you do not want the SnapMirror scheduler to run automatically.)
	◆ A range of values for any field can be indicated with a low value and a high value separated by a dash. For example, you can indicate that you want an update every hour from 8:00 a.m. to 5:00 p.m by entering this value in the hour field:
	`8-17`
	◆ A range of values followed by a slash and a number indicates the frequency of the update. For example, you can indicate that you want an update every five minutes by entering this value in the minutes field:
	`0-59/5`
	◆ Typing sync instead of the four space-separated fields specifies synchronous replication. See the na_snapmirror.conf(5) man page for more information.

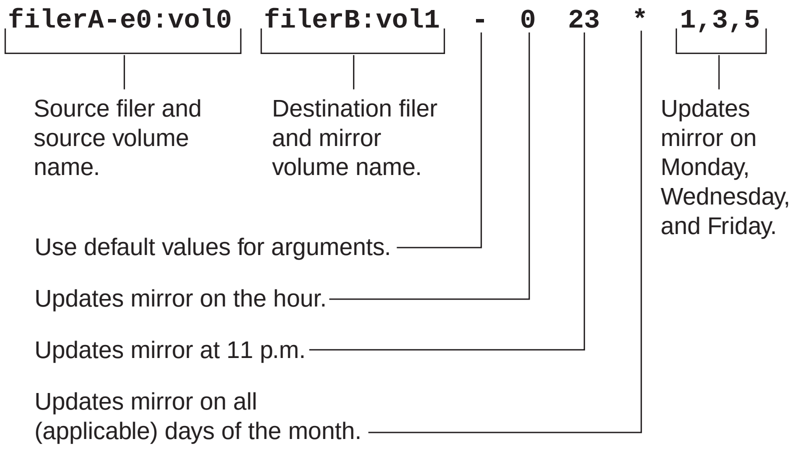
Example of snapmirror.conf schedule entries

Suppose you create a private network between filerA and filerB. In the /etc/hosts file on filerA, you give the host name for the interface as filerA-e0, and you ensure that filerA-e0 is also in the /etc/hosts file on filerB, the destination filer. You want to copy vol0 of filerA to vol1 of filerB over the private network every Monday, Wednesday, and Friday at 11 p.m. You also want to use the default for the arguments field.

To copy the data over the private network every Monday, Wednesday, and Friday at 11 p.m., you would enter the following in the /etc/snapmirror.conf file:

```
filerA-e0:vol0 filerB:vol1 - 0 23 * 1,3,5
```

The following figure illustrates what the entry in each field in the example means.



filerA-e0:vol0 filerB:vol1 - 0 23 * 1,3,5
Source filer and source volume name.
Destination filer and mirror volume name.
Updates mirror on Monday, Wednesday, and Friday.
Use default values for arguments.
Updates mirror on the hour.
Updates mirror at 11 p.m.
Updates mirror on all (applicable) days of the month.

Additional snapmirror.conf examples

Example setting maximum update speed: The following line in an /etc/snapmirror.conf file sets the speed to 2000 kilobytes per second.

```
filerA:vol0 filerA:vol1 kbs=2000 15 * * 1,2,3,4,5
```

Note

The specified transfer speed might not be achievable because transfer speed is limited by factors such as network bandwidth.

Example specifying always restart: The following line in an /etc/snapmirror.conf file sets the restart value to `always`.

```
filerA:vol0 filerA:vol1 kbs=2000,restart=always 15 * * 1,2,3,4,5
```

Example specifying default values for maximum speed and restart:

If you set the value of only one argument (`kbs` or `restart`), the other will use the default value. If you want to use the default argument for both values, enter a dash (-).

The following line in an /etc/snapmirror.conf file sets both arguments to the default value.

```
filerA:vol0 filerA:vol1 - 15 * * 1,2,3,4,5
```

Example specifying 15-minute interval updates during specific hours: If you want to schedule an update every afternoon at 1:00, 1:15, 1:30, 1:45, 5:00, 5:15, 5:30, 5:45, 7:00, 7:15, 7:30, and 7:45, you enter the following in the schedule field:

```
filerA:vol0 filerA:vol1 0,15,30,45 13,17,19 * *
```

Note

An update is started when the current time matches a value in all four fields. Be careful that a value in the day of the month field does not exclude a value in the day of the week field. For example, the following schedule would update the destination every afternoon at 1:00, 1:30, 3:00, 3:30, 5:00 and 5:30 but only when the first day of the month falls on a Monday.

```
filerA:vol0 filerA:vol1 0,30 13,15,17 1 1
```

Note

The schedule represents the goal of the SnapMirror feature. Factors that may prevent SnapMirror from updating every minute include resource limitations or network stability. If an update is in progress when another is scheduled to occur, SnapMirror will start another transfer as soon as the first is complete. However, if three updates pass while the current transfer is in progress, SnapMirror does only one more update; it does not go back and run updates that have been made obsolete by those scheduled later.

What restarts and retries are

In SnapMirror, a *retry* is an automatic attempt to start the transfer process after an interruption, whether or not any data was successfully transferred. A *restart* is the resumption of a previous transfer process from a restart checkpoint.

If a transfer fails, because of network failure, for example, SnapMirror automatically *retries* the transfer. SnapMirror makes a restart checkpoint every five minutes during a transfer. If a restart checkpoint exists and conditions are right, SnapMirror *restarts* the previous transfer where it left off. If not, SnapMirror creates a new snapshot and starts a new transfer.

After a reboot, SnapMirror does not automatically retry a transfer that was interrupted, but the next scheduled or manual transfer restarts it at the restart checkpoint, if the checkpoint is still valid.

A restart can happen if all of the following conditions are present:

◆ A restart checkpoint exists.

◆ All snapshots being transferred still exist.

◆ The value for restart mode in the /etc/snapmirror.conf file is set to *always* or is not set, and the next scheduled update has not arrived.

An initial transfer can be restarted but will not be retried automatically. To restart an initial transfer, you need to re-enter the `snapmirror initialize` command. Scheduled incremental updates automatically retry the transfer. A manual update issued from the command line is not retried automatically.

Using SnapMirror over multiple paths

Two modes of multiple paths

You might need more than one physical path for a mirror. SnapMirror supports up to two paths for a particular SnapMirror relationship. The paths can be Ethernet, Fibre Channel, or a combination of Ethernet and Fibre Channel. The two paths can be used in one of two modes:

- Multiplexing mode—SnapMirror uses both paths at the same time, essentially load balancing the transfers. If one path fails, the transfers occur on the remaining path. After the failed path is repaired, the transfers resume using both paths.
- Failover mode—SnapMirror uses the first specified path as the desired path and use the second specified path only after the first path fails.

Setting up SnapMirror using multiple paths

To implement multiple paths between the source filer and destination filer, complete the following steps.

Step	Action	

1	Ensure that you have two valid paths using the `ping` command from the source filer to each of the IP addresses on the destination filer.
	
	Example: The IP addresses on the destination filer are 10.10.10.23 and 10.10.10.24.
	
	`source>` **`ping 10.10.10.24`**
	`10.10.10.24 is alive`
	`source>` **`ping 10.10.10.23`**
	`10.10.10.23 is alive`	

2	**If you are...**	**Then...**
	Setting up SnapMirror for the first time and want to use multiple paths	Follow the steps shown in “Setting up a basic SnapMirror operation” on page 93, then follow Step 3 and Step 4.
	Converting a single path SnapMirror	Follow Step 3 and Step 4.

Step	Action
3	On the administration host, edit the snapmirror.conf file on the destination filer to add a connection name line that defines the mode of the connection and what the two connections are. The format of the line follows: `name = mode (src_filer1, dest_filer1) (src_filer2, dest_filer2)` *mode* is either multi or failover. See the na_snapmirror.conf(5) man page for details.
4	In the same snapmirror.conf file, edit the schedule entry to reflect the new connection name as the source filer.

Note

Multiple paths are supported by SnapMirror running asynchronously and synchronously. The following are examples of implementing multiple paths using synchronous SnapMirror.

Example 1: You want to synchronously mirror volume vol1 on a filer called NYC to volume vol1 on a filer called Newark. To satisfy a business requirement that there be two paths between each synchronously mirrored volume, you have four network interface cards, two in each filer. You named the two interfaces on the NYC filer NYC-pri and NYC-sec and the two on the Newark filer Newark-pri and Newark-sec. To implement multiple paths in failover mode, you edit the snapmirror.conf file on Newark to include the following two lines:

```
NYC-Newark = failover (NYC-pri, Newark-pri) (NYC-sec, Newark-sec)
NYC-Newark:vol1 Newark:vol1 - sync
```

Example 2: If NYC-pri and Newark-pri are Fibre Channel adapters and you want to replicate data using both connections, you follow the procedure to configure Fibre Channel adapters for SnapMirror. See “Configuring for Fibre Channel use” on page 97. Then you edit the snapmirror.conf file on Newark to include the following two lines to implement multiple paths in multiplexing mode:

```
NYC-Newark = multi (NYC-pri, Newark-pri) (NYC-sec, Newark-sec)
NYC-Newark:vol1 Newark:vol1 - sync
```

Enabling SnapMirror

SnapMirror enabling procedures

Before any SnapMirror replication process can begin, the SnapMirror feature must be licensed and enabled on the filer.

Entering license codes

Enter the SnapMirror license code on the filer that is to be the source and on the filer that is to be the destination.

To enter the SnapMirror license code, complete the following step.

Step	Action
1	On the server, enter the following command: **`license add xxxxxxx`** *xxxxxxx* is the license code you purchased. This setting persists across reboots.

Turning SnapMirror on

To turn SnapMirror on, complete the following steps.

Step	Action
1	Enter the following command on both the source filer and destination filer to enable SnapMirror: **`options snapmirror.enable on`** Alternatively, you can still use the older command instead to turn SnapMirror on: **`snapmirror on`** This setting persists across reboots.

Step	Action	
2	**If you are using the snapmirror.access option to specify allowed destinations...**	**If you are using the /etc/snapmirror.conf file to specify allowed destinations...**
	On the source, enter the following command as a single line: **`options snapmirror.access host=[`*`dest_filer1,dest_filer2...`*`]`** The default value for the `snapmirror.access` option is `legacy`, which lets the /etc/snapmirror.allow file define the access permissions. This option persists across reboots.	Add the name of each destination filer, on its own line, to the /etc/snapmirror.allow file.

Initializing a SnapMirror destination

When to run a complete SnapMirror transfer

You must use the `snapmirror initialize` command to perform a complete (baseline) transfer of information whenever you start up a SnapMirror source-destination relationship for the first time. This process is known as initializing a destination.

How quotas apply to destination qtrees

Qtree quotas apply to qtrees in a SnapMirror relationship. If a destination qtree is limited to 100 GB, transfers from a source greater than 100 GB will fail until the source drops to less than 100 GB or the qtree quota is increased on the destination.

User quotas also apply to SnapMirror destination qtrees; however, a SnapMirror qtree update will not fail if user quotas are exceeded.

Guidelines when creating a qtree SnapMirror relationship

The following are guidelines for qtrees in a SnapMirror relationship:

- ◆ Establish a qtree SnapMirror relationship between volumes that have the same `vol lang` settings.
- ◆ Once you establish a qtree SnapMirror relationship, do not change the language assigned to the destination volume.
- ◆ Avoid whitespace (space or tab characters) in names of source and destination qtrees.
- ◆ Do not rename volumes or qtrees after establishing a qtree SnapMirror relationship.

Initializing a SnapMirror destination volume from tape

You can also initialize a destination volume from tape using the `snapmirror store` command on the source volume and the `snapmirror retrieve` command on the destination volume. See “Using SnapMirror to copy a volume to local tape” on page 176 and “Initializing a SnapMirror destination via local tape” on page 181 for more information. The `snapmirror store` and `snapmirror retrieve` functions are valid only for volumes, not for qtrees in a SnapMirror relationship.

How to initialize a SnapMirror destination

To initialize a SnapMirror destination, complete the following steps.

Note

If your source volumes contain directories that are accessed by CIFS clients, ensure that those directories are in Unicode format before carrying out the initial SnapMirror replication of that volume. See "Ensuring accessible destinations when using CIFS" on page 91.

Step	Action
1	If the destination is a volume, is online, and has not been initialized before, enter the following command from the destination filer: **`vol restrict`** ***`dest_volume`*** *dest_volume* is the destination volume. **Note** Do not use the `vol restrict` command on a qtree. If you are initializing a qtree, go to step 2.

Step	Action
2	From the destination filer, enter the following command: **`snapmirror initialize [`*`options`*`] [`*`dest_filer:`*`]`** **`{`*`dest_volume`*` | `*`qtree_path`*`}`** *options* can be one or more of the following: ◆ `-k` *n* sets the maximum transfer speed to *n* kilobytes per second. This option has the same effect as the `kbs` argument in the /etc/snapmirror.conf file. ◆ `-S` [*source_filer*:]{*source_volume* | *source_qtree_path*} specifies the source filer and volume or qtree to copy. *source_volume* is the volume you want to copy. The source specified must match an entry for *source_volume* in the /etc/snapmirror.conf file, if one exists. If an entry exists but does not match, the operation displays an error message and terminates. If there is no entry for the specified source, the command runs. *source_qtree_path* is the path to the qtree you want to copy. If the `-S` option is not set, the source must be specified in the /etc/snapmirror.conf file. If it is not specified, the operation displays an error message and terminates. **Note** The *source_qtree_path* can be a qtree in a SnapMirror destination volume. ◆ `-c` *snapshot_name* creates a snapshot (with the name *snapshot_name*) of a qtree on the destination after the next update (so that it does not compete with any ongoing updates). SnapMirror does not lock or delete this snapshot. *snapshot_name* cannot be minutely.x, hourly.x, nightly.x, or weekly.x, because these names are reserved for scheduled snapshots. This option is valid only for a qtree. ◆ `-s` *snapshot_name* specifies an existing source qtree snapshot to be transferred. This prevents the normal action of the source creating a snapshot to transfer. SnapMirror does not lock or delete this snapshot. This option is valid only for a qtree SnapMirror replication.

Step	Action
	dest_filer is the name of the destination filer. The destination can reside on the same filer as the source or on another filer. *dest_volume* or *qtree_path* specifies the destination volume or qtree. If it is associated with a local source specified in the /etc/snapmirror.conf file, SnapMirror uses that source. If the destination volume or qtree specified is not in a scheduled relationship, then the `-S` option must be used to provide a source. The `snapmirror initialize` command creates the destination qtree, but you must specify the destination qtree name at the end of the path as though it already existed. If the destination qtree exists before the command runs, the command fails.

Example 1: Using the following command, SnapMirror transfers a complete copy of the source volume (vol0 on filerA) to the destination volume (vol2 on filerB):

```
filerB> snapmirror initialize -S filerA:vol0 filerB:vol2
```

Example 2: Using the following command, SnapMirror transfers a complete copy of the qtree source (qtree4 on vol1 on filerA) to the destination qtree (qtree4bak on vol1on filerB):

```
filerB> snapmirror initialize -S filerA:/vol/vol1/qtree4
filerB:/vol/vol1/qtree4bak
```

Initializing a destination for non-qtree data

Non-qtree data is any data on a filer that is not contained in its qtrees. Non-qtree data can include:

- Configuration and logging directories on the filer (for example, /etc or /logs) that are not normally visible to filer clients
- Directories and files on a filer volume that has no qtree configured

To use SnapMirror to replicate non-qtree data from a source to a destination, complete the following step.

Step	Action

1	From the destination filer, enter the following command:
	
	`snapmirror initialize -S`
	`source_filer`**`:/vol/`*****`source_volume`*****`/-`**
	`dest_filer`**`:/vol/`*****`dest_volume/qtree_name`***
	
	The dash (-) character indicates all non-qtree data in the specified volume.
	
	The `snapmirror initialize` command transfers the non-qtree data in the volume you specify, but you must specify the destination qtree name at the end of the path as though it already existed. If the destination qtree exists before the command runs, the command fails.

Example: Using the following command, SnapMirror transfers to the destination qtree (non_qtree_data_in_vol3 on vol4 on filerB) a complete copy of all the data in filerA, volume 3, that is not a part of a qtree:

```
filerB> snapmirror initialize -S filerA:/vol/vol3/-
filerB:/vol/vol4/non_qtree_data_in_vol3
```

After the transfer, the non-qtree data can only be a SnapMirror source, never a destination. Although you can copy data from the non-qtree data to another qtree, you cannot copy data to the non-qtree data from another qtree.

If you run the `snapmirror quiesce` or the `snapmirror break` command on the destination volume (/vol/vol4/non_qtree_data_in_vol3), you can resynchronize the destination volume to the source volume, as in this example:

```
filerB> snapmirror resync -S /vol/vol3/-
/vol/vol4/non_qtree_data_in_vol3
```

You cannot resynchronize the two qtrees in the opposite direction.

To summarize, you cannot make /vol/volname/- a SnapMirror destination. This non-qtree data can only be a SnapMirror source.

How snapmirror initialize copies volumes

When the `snapmirror initalize` command copies a volume, it creates a snapshot of all the data on the source and transfers it to the destination. The destination is a volume you have already created and marked restricted. After SnapMirror finishes transferring the data, it brings the destination online, but in a read-only state. This version of the destination is the baseline for the first incremental update.

Note

While the initial data transfer is taking place, the destination is marked "invalid" in the output of a `vol status` command. The volume becomes valid and goes online once the initial transfer is complete.

How snapmirror initialize copies qtrees

To use SnapMirror to copy a qtree, you do not create a destination qtree because the `snapmirror initialize` command creates it. The volume where you want the destination qtree to be must be online. After the destination qtree is initialized, it is no longer writable; however, the rest of the volume where that qtree resides is still writable.

The destination snapshot created by qtree initialization is marked "busy" in the output of the `snap list` command until the next transfer is complete.

What happens after SnapMirror makes the initial copy to the destination

After you initialize a SnapMirror volume replication, the files and snapshots in the source volume are available on the destination.

After you initialize a SnapMirror qtree replication, the files on the source qtree are available on its destination qtree.

You can export the destination for NFS mounting or add a share corresponding to the destination for CIFS sharing.

How to check on the initialization of a volume

To check that a destination volume has been initialized, you can use the `snapmirror status` command. For more information, see "Checking SnapMirror data transfer status" on page 132. If you specify no options or arguments, the `snapmirror status` command shows the status of the volumes in the filer, as shown in the example below. You also can use the `vol status` or the `qtree` command to check whether the volume or qtree is a SnapMirror destination.

```
filerA> snapmirror status
Snapmirror is on.
```

```
Source        Destination     State         Lag       Status
filerA:vol0   filerA:vol0bak  Snapmirrored  00:56:58  Idle
filerA:vol1   filerB:vol6     Source        23:69:26  Transferring
(126 MB done)
```

Checking on the initialization of a qtree

To check that a destination qtree has been created and initialized, complete the following step.

Step	Action

1	Enter the following command:
	`qtree`
	If you specify no options or arguments, the `qtree` command shows the status of all the qtrees in the filer, as in the example below.
	```
	filerA> qtree
	Volume         Tree   Style Oplocks   Status
	--------       -----  ----- -------   ------
	vol0                  unix  enabled   normal
	qtree24               unix  enabled   normal
	filerB_vol0           unix  disabled  normal
	filerB_vol0    qt1    mixed enabled   snapmirrored
	filerB_vol0    qt2    unix  disabled  normal
	filerB_vol0    qt3    ntfs  enabled   snapmirrored
	```

Note

When the `qtree` command is run from a MultiStore™ vfiler™ unit, any qtree that is either a SnapMirror destination or part of a volume that is a SnapMirror destination has read_only in the Status column of the output.

How snapmirror initialize matches source and destination volume size

When you use the `snapmirror initialize` command to initialize a volume replication, SnapMirror sets the `vol options fs_size_fixed` option to `on`. This option forces the file system on the destination volume to remain the same size as the file system on the source volume.

What you can do if initialization fails or is interrupted

When an initialization fails because of a reboot or other interruption, you can restart it by re-entering the `snapmirror initialize` command if all the following conditions are present:

- ◆ The output of the `snapmirror status` command shows that the process has a restart checkpoint.
- ◆ The snapshots being transferred still exist.
- ◆ The disk geometry has not changed.
- ◆ The value for restart mode in the /etc/snapmirror.conf file is set to *always* or is set to the default and the next scheduled update has not begun.

SnapMirror does not automatically retry to initialize a destination.

Updating a destination manually

Why to run a manual incremental SnapMirror update

Ordinarily, SnapMirror updates the destination automatically according to the update schedule you specify in the /etc/snapmirror.conf file. However, you can also initiate updates manually with the `snapmirror update` command. You might need to run an unscheduled update to prevent data loss resulting from a scheduled or threatened power outage or from a destination volume being taken offline for maintenance, repair, upgrade, or data migration. You can also include the `snapmirror update` command in an external script if you want to drive updates using that script.

How to perform a manual SnapMirror update

To perform an unscheduled SnapMirror incremental update, independent of the schedule in the /etc/snapmirror.conf file, complete the following step.

Step	Action
1	From the destination filer, enter the following command: **`snapmirror update [`*`options`*`] [`*`dest_filer`*`:] {`*`dest_volume`*` | /vol/`*`dest_volume/qtree_path`*`}`** *options* can be one or more of the following: ◆ `-k` *n* sets the maximum transfer speed to *n* kilobytes per second. This option has the same effect as the `kbs` argument in the /etc/snapmirror.conf file. ◆ `-s` *snapshot_name* specifies an existing (qtree only) source snapshot to be transferred, rather than a snapshot taken by the source. SnapMirror does not lock or delete this snaphot. *snapshot_name* cannot be minutely.x, hourly.x, nightly.x, weekly.x, snapshot_for_backup.x or snapshot_for_volcopy.x. You must rename such snapshots on the source and then copy them.

Step	Action
	◆ `-c` *snapshot_name* creates a snapshot named *snapshot_name* of a qtree on the destination after the next update (so that it does not compete with any ongoing updates). SnapMirror does not lock or delete this snapshot. *snapshot_name* cannot be minutely.x, hourly.x, nightly.x, or weekly.x, because these names are reserved for scheduled snapshots. ◆ `-S` [*source_filer:*]*source_volume* | *qtree_path* specifies the source filer and volume for the update. *source_volume* is the volume you want to copy. The source specified by the `-S` option must match an entry for *source_volume* in the /etc/snapmirror.conf file. If one exists but does not match, the operation displays an error message and terminates. If there is no entry for the specified source volume, the command runs. If the `-S` option is not set, the source must be specified in the /etc/snapmirror.conf file. If it is not specified, the operation displays an error message and terminates. *dest_filer* specifies the name of the destination filer. *dest_volume* specifies the destination volume. If it is a scheduled destination of a local source volume as specified in the /etc/snapmirror.conf file, that source volume is assumed to be the source. If the destination volume specified is not in a scheduled relationship, then the `-S` option must be used to provide a source.

Example 1: Using the following command, SnapMirror updates the destination (vol2 on filerB) from the source specified in the /etc/snapmirror.conf file:

```
filerB> snapmirror update filerB:vol2
```

Example 2: Using the following command, SnapMirror updates the qtree destination on filerB:/vol/vol2/usersbak from the source qtree on filerA:/vol/vol1/users:

```
filerB> snapmirror update -S filerA:/vol/vol1/users
filerB:/vol/vol2/usersbak
```

Creating extra backup snapshots for SnapMirror qtrees

To establish an extra backup snapshot on both sides of a SnapMirror qtree relationship, you can create a manual snapshot using the `snap create` command and then use the `-c` and `-s` options of the `snapmirror update` command together. These extra backup snapshots can serve as the newest common snapshot in case a base snapshot was accidentally deleted. You can also use them to resynchronize a SnapMirror qtree relationship to an earlier resynchronization point.

Example:

```
filerB> snap create vol2 my_snap

filerA> snapmirror update -S filerB:/vol/vol2/qtree1 -s my_snap -c
my_dest_snap vol/vol4/qtreeSafe
```

Result: You create a snapshot on filerB (the source) called my_snap. Then on filerA (the destination), you update the data in filerA:/vol/ vol4/qtreeSafe from the data in my_snap and store the updated data in the snapshot you create called my_dest_snap. SnapMirror does not automatically delete my_snap and my_dest_snap because they are user-originated snapshots.

What happens after SnapMirror makes incremental updates to the destination

Changes on the source are reflected on the destination after SnapMirror completes the transfer. Changes being copied are not visible during the transfer and are not visible if the transfer is interrupted. After the destination update is finished, you see the changes if you open the file.

SnapMirror automatically deletes old snapshots that are no longer necessary for updating data.

Listing SnapMirror snapshots

About SnapMirror listing

You can use the Data ONTAP `snap list` command to list all snapshots, including the SnapMirror-specific snapshots, that are stored on your filer.

To list all the snapshots on your system, complete the following step.

Step	Action
1	In the console of either your source or destination filer, enter the following command: **`snap list`** ***`vol_name`*** **Result:** A listing of all snapshots stored on your system appears. SnapMirror snapshots are distinguished from system snapshots by a more elaborate naming convention and the label “snapmirror” in parentheses.

Naming conventions for snapshots used by SnapMirror

When you run the `snap list` command, SnapMirror snapshots are distinguished from the regular filer system snapshots by their naming conventions.

For volume replication, SnapMirror creates a snapshot of the whole source volume that is copied to the destination volume. For qtree replication, SnapMirror creates snapshots of one or more source qtrees on the source volume that are copied to a qtree on the destination volume.

A SnapMirror volume snapshot name is in the following format:

dest_filer(sysid)_name.number

dest_filer is the host name of the destination filer.

sysid is the destination system ID number.

name is the name of the destination volume.

number is the number of successful transfers for the snapshot, starting at 1. Data ONTAP increments this number for each transfer.

A SnapMirror qtree snapshot name is in the following format:

dest_filer(sysid)_name-src|dst.number

dest_filer is the host name of the destination filer.

sysid is the destination system ID number.

name is the name of the destination volume or qtree path.

src|dst is the source or destination name.

number is an arbitrary start point number for the snapshot. Data ONTAP increments this number for each transfer.

In the output of the `snap list` command, SnapMirror snapshots are followed by the SnapMirror name in parentheses.

Volume example:

```
filerA(0016791363)_vol0.9 (snapmirror)
```

Qtree example:

```
filerA(0016789302)_vol1_qtree3-dst.15 (snapmirror)
```

Caution

Do not delete manually created snapshots marked "snapmirror" in the output of the `snap list` command.

Using snap list to show SnapMirror updates on the destination volume

The following example describes SnapMirror snapshots that are created on a source volume and copied to a destination volume. In this example, data is copied from vol1 of filerA (the source) to vol2 of filerB (the destination).

To create a baseline version of a destination volume, filerA creates a snapshot named filerB(0016782130)_vol2.1 on filerA. All snapshots in vol1 of filerA, including filerB(0016782130)_vol2.1, are transferred to vol2 of filerB. When replicating a qtree, SnapMirror transfers only the qtree's data in the snapshot for the qtree.

If the administrator were to run the `snap list` command on the destination filerB after the filerB(0016782130)_vol2.1 snapshot was transferred from filerA to filerB, a listing similar to the following example would be generated.

```
filerB> snap list vol2
working.....

%/used    %/total  date          name
-------- --------  ------------  --------
0% ( 0%) 0% ( 0%)  Nov 17 10:50  filerB(0016782130)_vol2.1
(snapmirror)
1% ( 0%) 0% ( 0%)  Nov 17 10:00  hourly.0
```

```
1% ( 0%) 0% ( 0%)  Nov 17 00:00  nightly.0
1% ( 0%) 0% ( 0%)  Nov 15 16:00  hourly.1
1% ( 0%) 1% ( 0%)  Nov 15 15:00  hourly.2
2% ( 0%) 1% ( 0%)  Nov 15 14:00  hourly.3
2% ( 0%) 1% ( 0%)  Nov 15 13:00  hourly.4
2% ( 0%) 1% ( 0%)  Nov 15 12:00  hourly.5
```

When it is time to update the destination, another snapshot is created on filerA.

The `snap list` command on filerA would generate the following display after the filerB(0016782130)_vol2.2 snapshot was created on filerA:

```
filerA> snap list vol1
working....
%/used   %/total     date          name
-------- ----------  ------------  --------
0% ( 0%) 0% ( 0%)    Nov 17 10:52  filerB(0016782130)_vol2.2
(snapmirror)
0% ( 0%) 0% ( 0%)    Nov 17 10:51  filerB(0016782130)_vol2.1
(snapmirror)
1% ( 0%) 0% ( 0%)    Nov 17 10:00  hourly.0
1% ( 0%) 0% ( 0%)    Nov 17 00:00  nightly.0
1% ( 0%) 0% ( 0%)    Nov 15 16:00  hourly.1
1% ( 0%) 1% ( 0%)    Nov 15 15:00  hourly.2
```

After the filerB(0016782130)_vol2.2 snapshot is transferred from filerA to filerB, both snapshots exist on filerB. On filerA, however, filerB(0016782130)_vol2.1 is no longer needed and is deleted; only filerB(0016782130)_vol2.2 is retained to be used for the next transfer.

You can see a list of each SnapMirror snapshot on the server, and the qtrees it contains, and the client sources of those qtrees and their timestamps by using the `snap list -q` command.

You can use the `snap list -o` command to display the names, timestamps, and sources (if they are copies) of the qtrees in a specified volume or at a path name.

Checking SnapMirror data transfer status

Why to check data transfer status

You check data transfer status, using the `snapmirror status` command, to determine the status of all existing SnapMirror relationships on the filer.

What checking status shows you

If you check the status and you enabled SnapMirror, messages of the following form are displayed:

```
filerA> snapmirror status
Snapmirror is on.
Source                   Destination             State          Lag       Status
filerA:vol0              filerA:vol1             Snapmirrored   02:25:11  Transferring
(60 MB done)
filerB:/vol/vol1/qtree   filerB:/vol/vol3/qtree  Quiesced       00:01:15  Idle
```

You see a status report for any SnapMirror source that contains the base snapshot, and for any destination in a current SnapMirror relationship or listed in the /etc/snapmirror.conf file. Destinations that were broken through the `snapmirror break` command but still contain the base snapshot are listed.

If you check the status of data transfer and you did not enable SnapMirror, the following message is displayed:

```
filerA> snapmirror status
Snapmirror is off.
```

The status of SnapMirror relationships, if any, are still displayed, as shown in the example above.

Checking the status

To check the status of data copying and check on how current the information at a destination is, complete the following step.

Step	Action
1	Enter the following command: **`snapmirror status [`*`options`*`] [[`*`filer:`*`] [`*`path`*`] ...]`** *options* can be one of the following: ◆ `-l` displays the long format of the output, which contains more detailed information.See Example 2, below. ◆ `-q` displays which volumes or qtrees are quiesced or quiescing. For more information, see Example 3, below, and "Stabilizing (quiescing) destinations before taking a snapshot" on page 148. *filer* is the name of the source filer. *path* is the name of the source volume or the path to and name of the source qtree. **Result:** If no arguments or options are given, SnapMirror displays a message showing whether a transfer is in progress, how much of the data transfer has been completed, the state of the destination, and the amount of time since the last snapshot was created and transferred successfully, as shown in Example 1.

Example 1: With no options, the information displayed by the `snapmirror status` command might look like the following. See the table "What the categories in the command output mean" on page 136 for more information.

```
filerB> snapmirror status
Snapmirror is on.
Source                 Destination            State          Lag        Status
filerA:vol0            filerB:vol2            Broken-off     29:09:58   Idle
filerC:vol0            filerB:vol3            Snapmirrored   00:09:53   Idle with
restart checkpoint (23 MB done)
filerC:vol4            filerB:vol5            Snapmirrored   00:04:58   Transferring
(36 MB done)
filerA:/vol/vol1/qt5   filerB:/vol/vol4/qt5   Quiesced       00:05:12   Idle
filerC:/vol/vol2/qt1   filerB:/vol/vol1/qt2   Snapmirrored   00:02:33   Quiescing
```

Example 2: With the `-l` option, the configuration described in Example 1 would look like this.

```
filerB> snapmirror status -l

Snapmirror is on.
Source:                 filerA:vol0
Destination:            filerB:vol2
Status:                 Idle
Progress                -
State:                  Broken-off
Lag:                    29:09:58
Mirror Timestamp:       Sat Jul 15 00:50:02 GMT 2000
Base Snapshot           tpubs-f720(0016791363)_vol2.1249
Current Transfer Type:  -
Current Transfer Error: -
Contents:               -
Last Transfer Type:     Update
Last Transfer Size:     122000 KB
Last Transfer Duration: 00:01:02
Last Transfer From:     filerA:vol0

Source:                 filerC:vol0
Destination:            filerB:vol3
Status:                 Idle with restart checkpoint
Progress                23552 KB done
State:                  Snapmirrored
Lag:                    00:09:53
Mirror Timestamp:       Sun Jul 16 05:50:07 GMT 2000
Base Snapshot:          filer2(0016778780)_vol3.985
Current Transfer Type:  -
Current Transfer Error: Abort by user
Contents:               Replica
Last Transfer Type:     Update
Last Transfer Size:     432000 KB
Last Transfer Duration: 00:01:23
Last Transfer From:     filerC:vol0

Source:                 filerC:vol4
Destination:            filerB:vol5
Status:                 Transferring
Progress                36864 KB done
State:                  Snapmirrored
Lag:                    00:04:58
Mirror Timestamp:       Sun Jul 16 05:55:02 GMT 2000
Base Snapshot:          filerB(0016778780)_vol5.57843
Current Transfer Type:  Scheduled
Current Transfer Error: -
Contents:               Replica
Last Transfer Type:     Scheduled
Last Transfer Size:     345000 KB
Last Transfer Duration: 00:03:23
Last Transfer From:     filerB:vol4

Source:                 filerC:/vol/vol1/qt5
Destination:            filerB:/vol/vol4/qt5
Status:                 Idle
Progress                -
State:                  Quiesced
Lag:                    0:05:12
Mirror Timestamp:       Sun Jul 16 05:56:12 GMT 2000
Base Snapshot:          filerB(0016778780)_vol_vol4_qt5.54
Current Transfer Type:  -
Current Transfer Error: -
Contents:               Replica
Last Transfer Type:     Scheduled
Last Transfer Size:     45000 KB
Last Transfer Duration: 0:00:12
Last Transfer From:     filerC:/vol/vol1/qt5

Source:                 filerC:/vol/vol2/qt1
Destination:            filerB:/vol/vol4/qt2
Status:                 Quiescing
Progress                -
State:                  Snapmirrored
Lag:                    0:02:33
Mirror Timestamp:       Sun Jul 16 05:58:20 GMT 2000
Base Snapshot:          filerB(0016778780)_vol_vol4_qt2.122
Current Transfer Type:  -
Current Transfer Error: -
Contents:               Transitioning
Last Transfer Type:     Scheduled
Last Transfer Size:     80 KB
Last Transfer Duration: 0:00:08
Last Transfer From:     filerC:/vol/vol2/qt1
```

Example 3: With the `-q` option, the output might look like this.

```
filerC> snapmirror status -q
Snapmirror is on.
vol3 is quiesced
vol2 has quiesced/quiescing qtrees:
         /vol/vol2/qt1 is Quiescing
         /vol/vol2/qt2 is Quiesced
```

What the categories in the command output mean

Information messages that the `snapmiror status` command can display are as follows.

snapmirror status Source entries:

Source entry	Description
filer:vol	The source filer and source volume
filer:qtree_path	The source filer and qtree path
-	Either the SnapMirror destination is an imported volume without an entry in the /etc/snapmirror.conf file, or a Data ONTAP upgrade is in progress.
filer:tape_device	The source tape device; transfer from this tape device is still in progress
base snapshot	The name of the base snapshot from which a completed transfer was made, if the source is a tape device and there is no entry in the /etc/snapmirror.conf file for the destination

snapmirror status Destination entries:

Destination entry	Description
filer:vol	The destination filer and volume
filer:qtree_path	The destination filer and qtree path
filer:tape_device	The destination tape device; transfer to this tape device is in progress
tape_destination	Displayed after a transfer to tape is finished. The `snapmirror destinations` command also displays this information. For more information, see "Listing SnapMirror destinations for a volume in a cascading series" on page 172.

State entry	**Description**
Uninitialized	The destination is listed in the /etc/snapmirror.conf file, but the volume or qtree has not been initialized or the destination is being initialized.
Snapmirrored	The volume or qtree is in a SnapMirror relationship.
Broken-off	The destination was in a SnapMirror relationship, but a `snapmirror break` command made the volume or qtree writable. This state is reported as long as the base snapshot is still present in the volume. If the snapshot is deleted, the state is listed as “uninitialized” if the destination is in the /etc/snapmirror.conf file or is no longer listed if it is not. A successful `snapmirror resync` command restores the snapmirrored status.
Quiesced	SnapMirror is in a consistent internal state and no SnapMirror activity is occurring. In this state, you can create snapshots with confidence that all destinations are consistent. The `snapmirror quiesce` command brings the destination into this state. The `snapmirror resume` command restarts all SnapMirror activities.
Unknown	The destination volume or the volume that contains the destination qtree is in an unknown state. It might be offline or restricted.
Source	When the `snapmirror status` command is run on the source filer and the destination is on another filer, the state of the destination is unknown, so the source status is reported.

Lag entry	**Description**
hh:mm:ss	Indicates the difference between the current time and the timestamp of the snapshot last successfully transferred to the destination.
-	The destination is not initialized.

snapmirror status Status entries:

Status entry	Description
Idle	No data is being transferred.
Idle with restart checkpoint (*n X*B done)	No data is being transferred. The last transfer attempt was aborted, but the transfer saved a restart checkpoint and thus can be restarted at the next attempt. Transfer sizes are reported in KB up to 10,240, MB up to 10,240, GB up to 10,240, then TB.
Transferring	Transfer has been initiated but has not yet started, or is just finishing.
Transferring (*n X*B done)	Data transfer is in progress. Transfer sizes are reported in KB up to 10,240, then MB up to 10,240, then GB up to 10,240, then TB.
Pending	The destination was not updated because of a transfer failure; the transfer will be retried automatically.
Pending with restart checkpoint (*n X*B done)	The destination was not updated because of a transfer failure. The transfer will be retried automatically from the restart checkpoint. Transfer sizes are reported in KB up to 10,240, MB up to 10,240, GB up to 10,240, then TB.
Aborting	A transfer is being aborted and cleaned up.
Quiescing	The specified volume or qtree is waiting for all existing transfers to complete. The destination is being brought into a stable state.
Resyncing	The specified volume or qtree is being matched with data in the common snapshot.
Waiting	SnapMirror is waiting for a new tape to be put in the tape device.

snapmirror status additional entries:

Additional -l option entries	Description
Progress	Shows the amount of data (in KB) transferred by the current transfer. Shows the restart check point if the status is Idle or Pending.
Mirror Timestamp	The timestamp of the last snapshot successfully transferred from the source to the destination. **Note** A resynchronization may change the base snapshot to a snapshot with an older timestamp.
Base Snapshot	The name of the base snapshot for the destination. For volumes in a SnapMirror relationship, this field is the same on the source side and the destination side. For qtrees in a SnapMirror relationship, the destination side lists the name of the exported snapshot for that qtree on the destination. **Note** A resynchronization may change the name of the base snapshot.
Current Transfer Type	Indicates the kind of transfer now in progress: scheduled, retry, resync, update, initialize, store, or retrieve. This field applies only to the destination side.
Current Transfer Error	Displays an error message if the latest transfer attempt failed.
Contents	Indicates whether the contents of the destination volume or qtree in the active file system are up-to-date replicas or in transition. The field applies only to the destination side. ◆ Under SnapMirror volume replication, the contents are always a replica. ◆ Under SnapMirror qtree replication, the contents are usually a replica, but sometimes are transitioning.

Additional -l option entries	**Description**
Last Transfer Type	Indicates the kind of transfer previously performed: scheduled, retry, resync, update, initialize, store, or retrieve. This field applies only to the destination side.
Last Transfer Size	Shows the amount of data (in KB) transferred in the last successful transfer.
Last Transfer Duration	Shows the elapsed time for the last successful transfer to complete. If the transfer failed and restarted, this includes time waiting to restart the transfer. If a transfer aborted and was retried from the beginning, it includes only the time required for the final successful attempt.
Last Transfer From	This field applies only to the destination side and shows the name of the source filer and volume or qtree. This field is useful if you have changed the source in the /etc/snapmirror.conf file but the data is actually from the old source. For `snapmirror retrieve` (from tape) operations, this field lists the tape device used in the retrieve operation.

Checking SnapMirror data transfer logs

Why you check data transfer logs

You can use the `options snapmirror.log.enable` command to check SnapMirror data transfer logs. You can find out whether transfers are occurring as planned, how long the transfers take, and how well the system setup works. You find this information in the SnapMirror log file.

What checking data transfer history shows you

The SnapMirror log file shows you

- The start time and the end time of the SnapMirror logging process
- The start time, end time, and size of each transfer
- Any abnormal termination and restart of a transfer
- Other SnapMirror-related activities

What you can do with the information recorded

You can use the raw information provided to do the following:

- Calculate the average transfer size
- Calculate the average transfer time
- Look at the number of successful transfers and the failure rate
- Tune the schedule
- Create a notifier for aborted transfers
- Monitor performance on a per-volume level
- Be assured that things are working as planned

How to find out whether SnapMirror logging is turned on or off

SnapMirror logging is turned on by default. However, if you need to find out whether SnapMirror logging is turned on or off, complete the following step.

Step	Action
1	Enter the following command on the filer for which you want the information: **`options snapmirror.log.enable`** **Result:** SnapMirror reports whether logging is enabled or not.

Example:

```
filerA> options snapmirror.log.enable
snapmirror.log.enable      on
```

How to turn SnapMirror logging on

To turn SnapMirror logging on, complete the following step.

Step	Action
1	Enter the following command on the filer for which you want the log: **`options snapmirror.log.enable on`** **Result:** SnapMirror enables the logging of transfers for the filer. This setting is persistent across reboots.

Where the log files are kept

SnapMirror keeps the current log on the root volume of the executing filer, as /etc/log/snapmirror.0. A new log file is generated every week as /etc./log/snapmirror.0. Older log files are renamed /etc./log/snapmirror.[1-5] and the oldest log file is deleted. The log files can be read with a text editor.

Format of the log files

The log file is in the following format:

```
type timestamp source_filer:source_path dest_filer:dest_path
event_info
```

type can be one of the following: `src`, `dst`, `log`, `cmd`.

type specifies whether the record is for the source side (`src`) or destination side (`dst`) of the transfer. Certain events apply to only one side. The type `log` indicates a record about the logging system itself, for example, `Start_Logging` and `End_Logging`. The type `cmd` indicates a record of user commands, for example, `Release_command` and `Resync_command`.

`timestamp` is expressed in `ctime` format, for example:

```
Fri Jul 27 20:41:09 GMT.
```

`event_info` includes the following event names:

```
Request ( IP address | transfer type )
Start
Restart (@ num KB)
End (num KB done)
Abort (error_msg)
Defer (reason)
Rollback_start
Rollback_end
Rollback_failed
Start_Logging
End_Logging
Wait_tape
New_tape
Snapmirror_on
Snapmirror_off
Quiesce_start
Quiesce_end
Quiesce_failed
Resume_command
Break_command
Release_command
Abort_command
Resync_command
Migrate_command
```

The `Request` event on the source side includes the IP address of the filer that made the transfer request; the `Request` event on the destination side includes the type of transfer. At the end of each successful transfer, the `End` event also reports the total size of the transfer in KB. Error messages are included with the `Abort` and `Defer` events.

Examples of log files

The following is an example of a log file from the source side.

```
log Fri Jul 27 20:00:01 GMT - - Start_Logging
cmd Fri Jul 27 20:00:20 GMT - - Snapmirror_on
src Fri Jul 27 20:41:09 GMT filer1:vol1 filer2:vol1 Request (10.56.17.133)
src Fri Jul 27 20:41:32 GMT filer1:vol1 filer2:vol1 Abort (Destination not allowed)
src Fri Jul 27 20:45:31 GMT filer1:vol0 filer1:vol1 Request (10.56.17.132)
src Fri Jul 27 20:45:35 GMT filer1:vol0 filer1:vol1 Start
src Fri Jul 27 20:51:40 GMT filer1:vol0 filer1:vol1 End (26200 KB)
src Fri Jul 27 22:41:09 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Request
(10.56.17.133)
src Fri Jul 27 22:41:12 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Start
src Fri Jul 27 22:41:13 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Abort (Non
unicode directory found in source qtree.)
src Fri Jul 27 22:45:53 GMT filer1:/vol/vol1/qtb filer2:/vol/vol1/qsmb Request
(10.56.17.133)
src Fri Jul 27 22:45:56 GMT filer1:/vol/vol1/qtb filer2:/vol/vol1/qsmb Start
src Fri Jul 27 22:45:59 GMT filer1:/vol/vol1/qtb filer2:/vol/vol1/qsmb End (3800 KB)
cmd Fri Jul 27 22:50:29 GMT filer1:/vol/vol1/qtb filer2:/vol/vol1/qsmb Release_command
```

The following is an example of a log file from the destination side.

```
dst Fri Jul 27 22:50:18 GMT filer1:vol0 filer1:vol1 Request (Initialization)
dst Fri Jul 27 22:50:20 GMT filer1:vol0 filer1:vol1 Abort (Destination is not
restricted)
dst Fri Jul 27 22:57:17 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Request
(Initialize)
dst Fri Jul 27 22:57:24 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Start
dst Fri Jul 27 22:57:36 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB End (55670 KB)
dst Fri Jul 27 23:10:03 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Request
(Scheduled)
dst Fri Jul 27 23:10:07 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Start
dst Fri Jul 27 23:10:18 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB End (12900 KB)
cmd Sat Jul 28 00:05:29 GMT - filer2:/vol/vol1/qtB Quiesce_start
cmd Sat Jul 28 00:05:29 GMT - filer2:/vol/vol1/qtB Quiesce_end
cmd Sat Jul 28 00:05:40 GMT - filer2:/vol/vol1/qtB Break_command
cmd Sat Jul 28 00:41:05 GMT filer1:/vol/vol1/qtA filer2:/vol/vol1/qtB Resync_command
log Sat Jul 28 00:41:10 GMT - - End_Logging
```

The following is an example of a log file from a retrieve (from tape) request.

```
dst Fri Jun 22 03:07:34 GMT milton:rst0l milton:bigtwo Request
(retrieve)
dst Fri Jun 22 03:07:34 GMT milton:rst0l milton:bigtwo Start
dst Fri Jun 22 05:03:45 GMT milton:rst0l milton:bigtwo Wait_tape
dst Fri Jun 22 15:16:44 GMT milton:rst0l milton:bigtwo New_tape
dst Fri Jun 22 17:13:24 GMT milton:rst0l milton:bigtwo Wait_tape
dst Fri Jun 22 17:56:43 GMT milton:rst0l milton:bigtwo New_tape
dst Fri Jun 22 18:10:37 GMT milton:rst0l milton:bigtwo End
(98602256 KB)
```

How to turn off the SnapMirror log

To turn off the SnapMirror log process, complete the following step.

Step	Action
1	Enter the following command on the filer for which you want to disable the log: **`options snapmirror.log.enable off`** **Result:** SnapMirror disables the logging process.

Aborting a SnapMirror transfer

What aborting a transfer does

You can use the `snapmirror abort` command to abort a volume or qtree copy operation before the source finishes transferring data to the destination. The transfer can be the result of a scheduled update, a manual update, or an initialization. If you abort a copy operation, data transfer stops and SnapMirror is put in a restartable mode. If you abort a transfer that has been aborted before, you cannot restart that transfer again.

If you use the `-h` option (hard abort) with the `snapmirror abort` command, you cannot restart the transfer.

How to tell whether you can restart an aborted transfer

To find out whether you can restart an aborted copy operation, check the output of the `snapmirror status` command. If the status message is "Idle with restart checkpoint (*n X*B done)," the transfer can be restarted.

Example:

```
Source         Destination      State         Lag        Status
filerA:vol0    filerA:vol1      Broken-off    29:09:58   Idle
with restart checkpoint (135 MB done)
```

Note

You can abort only transfers that are displayed by the `snapmirror status` command.

Aborting a transfer

To abort a SnapMirror data transfer, complete the following step.

Step	Action
1	From either the source or the destination filer, enter the following command: **`snapmirror abort [-h] {[`*`dest_filer`*`:]`*`dest_volume`*` | [`*`dest_filer`*`:]/vol/`*`volume_name/qtree_name`*` ...}`** `-h` specifies a hard abort; the transfer cannot be restarted. SnapMirror stops the transfer and clears the restartable transfer log. This option applies only to the SnapMirror destination. *dest_filer* is the name of the destination filer. *dest_volume* is the destination volume. /`vol`/*volume_name/qtree_name* is the path name of a destination qtree. If no destination filer is specified, the local host's name is used for the filer name. You can enter more than one destination volume. You can obtain the destination filer and volume from the `snapmirror status` output. If no destination volume or qtree is specified, the command returns an error message; it does not abort all transfers. To abort all transfers, use the `snapmirror off` command. If you enter an invalid SnapMirror destination (one that is not shown in the output of the `snapmirror status` command), the command fails and displays an error message.

Example:
```
filerA> snapmirror abort vol1 filerB:vol2
filerC:/vol3/qtree3

snapmirror abort: Aborting transfer to vol1 filerB:vol2
filerC:/vol3/qtree3
```

Result: SnapMirror aborts the transfer to vol1 on filerA, where the command was entered, and aborts the transfer to vol2 on filerB and the transfer to qtree 3 in vol3 on filerC.

Stabilizing (quiescing) destinations before taking a snapshot

Why to block data transfers to a destination temporarily

You may need to temporarily stop transfers to a destination. For example, if you want to create a snapshot of a SnapMirror destination volume or qtree that contains a database, you may need to ensure that its contents are stable during the snapshot. You can use the `snapmirror quiesce` command to block transfers to the destination after the destination reaches a stable state.

What the quiesce command does

The `snapmirror quiesce` command waits for all existing transfers to both volumes and qtrees to complete and blocks any further updates. If a qtree is not in a stable state (is in transition), the `snapmirror quiesce` command forces it into a stable state.

You can quiesce *only* volumes and qtrees that are online and that are SnapMirror destinations.You cannot quiesce a restricted or offline volume or a qtree in a restricted or offline volume.

The `snapmirror quiesce` command stops a volume or qtree from acting as a SnapMirror destination, but does not prevent it from acting as a SnapMirror source.

Note

The quiesced state persists across reboots.

How to quiesce data transfers

To stabilize and halt updates to data on a SnapMirror destination volume or qtree, complete the following step.

Step	Action
1	Enter the following command on the filer on which you want to block transfers: **`snapmirror quiesce {`*`dest_volume`*` | /vol/`*`volume_name/qtree_name`*`}`** *dest_volume* is the name of the destination volume. *qtree_name* is the name of a qtree in *volume_name.*

Example 1: `filerA> snapmirror quiesce vol1`

```
snapmirror quiesce: in progress.
snapmirror quiesce: vol1: successfully quiesced
```

Result: SnapMirror stops any further data transfers to vol1.

Example 2: `filerA> snapmirror quiesce vol2`

```
snapmirror quiesce: in progress.
This can be a long-running operation. Use Control-C to interrupt.
......................................
snapmirror quiesce: vol2: successfully quiesced
```

Result: SnapMirror waits for a transfer to finish and stops any further data transfers to vol2.

Example 3: `filerA> snapmirror quiesce /vol/vol1/qtree1`

Result: SnapMirror stops data transfers to qtree1 in vol1.

If you use the `snapmirror break` command on a destination that is quiesced, the quiesce condition is automatically cleared when the destination becomes writable.

Resuming transfers after quiescing a destination

You can use the `snapmirror resume` command to restore the capability for data transfer to a volume or qtree you have quiesced.

To resume SnapMirror transfers, complete the following step.

Step	Action

1	Enter the following command for the filer on which you want to resume transfers:
	`snapmirror resume {`*`dest_volume`*` | /vol/`*`volume_name/qtree_name`*`}`
	dest_volume is the name of the destination volume.
	qtree_name is the name of a qtree in *volume_name*.
	Example:
	`filerA> snapmirror resume vol2`
	`snapmirror resume: vol2: Successfully resumed`
	Result: SnapMirror resumes normal data transfer capability for vol2.

Turning off or changing scheduled updates for volumes or qtrees

Why you turn off scheduled updates

You can edit the destination’s etc/snapmirror.conf file to change or turn off scheduled updates for a particular volume or qtree if you decide that there is no need to update the destination. For example, you might want to change the time or frequency of scheduled updates if the pattern of use or the configuration of the filers has changed. Or, if you want to use the volume for a different purpose, you can change the destination to a writable volume.

Note

Editing entries in the destination’s etc/snapmirror.conf file to turn off scheduled updates does not change the destination to a writable volume. If you want to change the destination to a writable volume or qtree, you use the `snapmirror break` command to turn the destination into a writable volume or qtree and the `snapmirror release` command to allow SnapMirror to delete the snapshots it no longer needs on the source. See “Converting a destination to a writable volume or qtree” on page 156 and “How to release a source from a SnapMirror relationship” on page 159.

When you can turn off or change scheduled updates

You can edit the destination’s etc/snapmirror.conf file to turn off or change scheduled updates at any time, even when data transfer is underway. The destination remains the same as before the transfer. The snapshot taken in the source for the data transfer remains, but it can be deleted and replaced by a new snapshot the next time the destination is updated.

Changing scheduled updates for one volume or qtree

To change scheduled updates for one volume or qtree, complete the following step.

Step	Action

1	In the destination’s /etc/snapmirror.conf file, edit the destination volume or schedule information to specify the configuration you want.
	
	Example:
	
	Original update schedule:
	`filerA:vol0 filerA:vol1 - 0 23 * 1,3,5`
	`filerA:vol1 filerB:vol6 - 0 23 * 1,3,5`
	
	Changed update schedule:
	`filerA:vol0 filerA:vol2 - 0 23 * 1,3,5`
	`filerA:vol1 filerB:vol6 - 0 23 * 2,4,6`

Turning off scheduled updates for one volume

To turn off scheduled updates for one volume or qtree, complete the following step.

Step	Action

1	Either delete the entry in the /etc/snapmirror.conf file or change the entry by
	
	◆ Commenting out the entry by preceding it with a pound sign (#)
	
	Example:
	`filerA:vol0 filerA:vol1 - 0 23 * 1,3,5`
	`filerA:vol1 filerB:vol6 - 0 23 * 1,3,5`
	`#filerB:vol1 filerC:vol2 - 0 23 * 1,3,5`
	
	◆ Putting a dash (-) in one of the schedule fields (minute/hour/dayofmonth/dayofweek)
	
	Note
	Deleting or commenting out a destination or putting a dash in one of the schedule fields of a destination in the /etc/snapmirror.conf file does not prevent you from performing manual updates to that destination.
	
	Example:
	`filerA:vol0 filerA:vol1 - 0 23 * 1,3,5`
	`filerA:vol1 filerB:vol6 - 0 23 * 1,3,5`
	`filerB:vol1 filerC:vol2 - - 23 * 1,3,5`

Turning off SnapMirror updates for the filer

Why you turn off updates

You can use the `snapmirror off` command to turn off updates, both scheduled and manual, for the entire filer at any time, even when copying is underway. Any active transfer is aborted when you turn off SnapMirror for the filer. The destination remains unchanged after you turn off updates.

This process affects all SnapMirror transfers for the filer, whether the filer is the source or the destination of the SnapMirror relationship.

To turn off scheduled and manual updates for the entire filer, complete the following steps.

Step	Action
1	Enter the following command on both the source filer and destination filer to disable SnapMirror: **`options snapmirror.enable off`** Alternatively, you can still use the older command to turn SnapMirror off: **`snapmirror off`** **Result:** If SnapMirror is currently transferring data from one volume or qtree to another, the transfer aborts immediately. The destination remains the same as before the transfer. The snapshot taken in the source volume for the data transfer remains. SnapMirror stops monitoring the /etc/snapmirror.conf file for changes. Entering the `snapmirror off` command on the destination filer alone does not affect SnapMirror on the source filer. Other filers can continue to copy data from the source filer. **Note** Both the `snapmirror off` command and the `snapmirror.enable off` option are persistent across reboots.

Step	Action
2	If the `snapmirror on` command is in the /etc/rc file, remove the command (to keep the current setting after reboot). Otherwise the setting in the /etc/rc file overrides the command you entered.

Converting a destination to a writable volume or qtree

Why to convert a destination to a writable volume

You can use the `snapmirror break` command to convert a SnapMirror destination, with read-only status, to a writable volume or qtree. You might want to convert a destination to a writable volume or qtree to perform one of the following tasks:

- Data migration—You want to move your user’s working data from one volume or qtree (your current source volume or qtree) to another volume or qtree (your current destination volume or qtree) and make that data accessible and writable in its new location.
- Disaster recovery—In case your source volume or qtree is suddenly unavailable, you want your current destination volume or qtree to serve as the substitute for your users’ retrieval and input source.
- Application testing—You want to make your current destination volume or qtree writable to test a new application on a mirrored replication of your current data rather than risk corruption of original data on the source volume or qtree.

Converting the destination to a writable volume or qtree lets you use data on the destination for these situations or in any other situation in which the original source is unavailable.

Preserving quota restrictions

Quotas are always disabled on a SnapMirror volume destination, regardless of whether quotas are enabled on the source volume. If you try to enable quotas on a volume destination, SnapMirror displays an error message. Quotas are not disabled on SnapMirror destination qtrees.

If the source volume or qtree and the destination reside on different filers, and you want the same quota restrictions to be applied after you make the destination writable, the destination filer must have an /etc/quotas file that includes all the entries from the /etc/quotas file used by the source filer.

- If you use SnapMirror replication for data migration, you can copy the /etc/quotas entries from the source filer to the /etc/quotas file of the destination filer before you use the `snapmirror break` command to make the destination writable.
- If you use SnapMirror replication for backup and potential disaster recovery, you must keep a copy on the destination filer of all /etc/quotas entries used

by the source filer at all times. That way, you can apply the quota entries to the destination volume or qtree if the source filer becomes unavailable.

Converting to a writable volume or qtree

To convert a destination to a writable volume or qtree, complete the following steps.

Step	Action

1	On the destination filer use the `snapmirror break` command to make the destination volume or qtree writable.
	◆ To make a destination volume writable, enter the following command on the destination filer.
	`snapmirror break` ***`volume_name`***
	◆ To make a destination qtree writable, enter the following commands on the destination filer.
	`snapmirror quiesce /vol/`***`volume_name/qtree_name`***
	`snapmirror break /vol/`***`volume_name/qtree_name`***

2	If you want to enable quotas on the former destination volume, carry out the following steps. **1.** Edit the /etc/quotas file on the former destination filer so that, after the conversion, the former destination includes the same quota restrictions as the source volume. If the original source volume uses per-volume quotas, replace the original source volume name with the former destination name in the quota entries. **2.** Enter the following command to enable quotas on the former destination: **`quota on`** ***`volume_name`***

Step	Action
3	Consider the following optional measures: ◆ If you want to stop a SnapMirror source from trying to update a broken-off destination, you can delete or comment out the entry in the /etc/snapmirror. conf file. Otherwise SnapMirror continues to try to update the destination. ◆ You might also want to use the `options fs_size_fixed off` command to turn off the option that restricts the size of the file system on a destination volume. **Note** If you set `options fs_size_fixed off`, the ability of the destination and source volumes to resync is not guaranteed. For more information, see “How snapmirror initialize matches source and destination volume size” on page 124.

After using snapmirror break

After using the `snapmirror break` command to temporarily break a SnapMirror relationship between a source and destination, you can use other SnapMirror commands to either make the break permanent or restore or redefine the SnapMirror relationship.

◆ Use the `snapmirror release` command to make the break permanent. See “Releasing partners from a SnapMirror relationship” on page 159.

◆ Use the `snapmirror resync` command to restore or redefine the SnapMirror relationship. See “Resynchronizing SnapMirror” on page 161.

Releasing partners from a SnapMirror relationship

When to release partners from a SnapMirror relationship

If you want to permanently end a SnapMirror relationship between a source and destination volume or qtree, you invoke different commands on the source and destination filers.

- On the source filer, you use the `snapmirror release` command. Releasing a source from a destination volume or qtree allows the source to delete its base snapshot for the SnapMirror relationship. See "How to release a source from a SnapMirror relationship" below.
- On the destination filer, you use the `snapmirror break` command, and carry out some additional operations. See "How to scrub a destination of a broken SnapMirror relationship" on page 160.

How to release a source from a SnapMirror relationship

To release a source from a SnapMirror relationship, complete the following step.

Step	Action
1	On the source filer, enter the following command: **`snapmirror release {`*`source_volume`*`	`*`qtree_path`*`}`** **`[`*`dest_filer`*`:]{`*`dest_volume`*`	`*`qtree_path`*`}`** *source_volume* or *qtree_path* is the name of the source volume or path to the qtree that you want to release from the destination. *dest_filer* is the name of the filer where the destination is located. *dest_volume* or *qtree_path* is the name of the volume or path to the qtree that is the destination. If you do not enter the name of the destination filer, SnapMirror uses the name of the filer on which you entered the command. **Result:** SnapMirror frees all resources on the source filer that had been dedicated to the SnapMirror relationship.

Volume example: `filerA> snapmirror release vol0 filerB:vol2`

Qtree example: `filerA> snapmirror release vol/vol1/qtree2 filerB:/vol/vol2/qtree5`

How to scrub a destination of a broken SnapMirror relationship

In order to permanently break a SnapMirror relationship on a destination volume or qtree, you need to carry out extra steps in addition to invoking `snapmirror break` on the destination filer and `snapmirror release` on the source filer. Unless these extra steps are taken, the snapshots associated with the broken relationship remain stored on the destination filer, and a `snapmirror status` command will continue to list the former destination object as a current destination object. To scrub a destination of a SnapMirror relationship, complete the following steps.

Step	Action
1	If you have not already done so, ◆ invoke the `snapmirror break` command on the destination filer to break the SnapMirror relationship between the source and destination objects. For details see "Converting to a writable volume or qtree" on page 157. ◆ invoke the `snapmirror release` command on the source filer to release the source object from this SnapMirror relationship. For details see "How to release a source from a SnapMirror relationship" on page 159.
2	On the destination filer, invoke the `snapmirror status -l` command to determine which snapshot basename is associated with the SnapMirror relationship that you just broke. ◆ For broken SnapMirror volume relationships enter: `snapmirror status -l` *`dest_vol`* ◆ For broken SnapMirror qtree relationships enter: `snapmirror status -l /vol/`*`dest_vol`*`/`*`dest_qtree`* In the detailed output that is displayed, note the snapshot basename associated with the SnapMirror relationship that you just broke.
3	On the destination filer, delete the snapshot set that you displayed in Step 2. Enter: `snap delete` *`dest_vol snapshot_basename`*
4	Through the Adminhost client, edit the /etc/snapmirror.conf file on the destination filer. Locate and delete the entry that specifies the SnapMirror relationship you want to end. For details on editing the /etc/snapmirror.conf file, see "snapmirror.conf file entry syntax" on page 107.

Resynchronizing SnapMirror

Why you resynchronize a source and destination volume

You can use the `snapmirror resync` command to restore or redefine a SnapMirror source or destination relationship that was broken with the `snapmirror break` command. You might want to resynchronize a source and a destination volume or qtree when

- ◆ You are changing the current source to a different volume or qtree
- ◆ You make a destination volume writable for application testing and then want to make it a SnapMirror destination again
- ◆ You need to recover from a disaster that disabled the source
- ◆ You want to reverse the functions of the source and the destination

What the snapmirror resync command does

After the `snapmirror break` command, you apply the `snapmirror resync` command to either the original SnapMirror destination or the original source.

- ◆ Applied to the original destination— the `snapmirror resync` command will put a volume or qtree back into a SnapMirror relationship and resynchronize its contents with the source without repeating the initial transfer.
- ◆ Applied to the source volume— the `snapmirror resync` command can turn the source volume into a copy of the original destination volume. In this way, the roles of source and destination can be reversed.

How the snapmirror resync command helps minimize data loss

The `snapmirror resync` command enables you to re-establish a broken SnapMirror relationship without a lengthy baseline transfer.

This command offers the choice of either source or destination to serve as the source in the restarted SnapMirror relationship. It finds the newest common snapshot (NCS) shared by the two volumes or qtrees, and removes all newer information on the filer on which the command is run.

Note

The `snapmirror resync` command requires that the two volumes or qtrees have at least one snapshot in common. You can resynchronize a volume or qtree to any other volume or qtree as long as both have at least one snapshot in common.

Resynchronization will cause the loss of all data written to the destination after the base snapshot was made. The `snapmirror resync` command informs you what data could be lost during the resynchronization and requests permission to proceed. If you want to save the data on the destination, you can stop the resynchronization, manually copy the desired data elsewhere, and reissue the `snapmirror resync` command to the destination.

Resynchronizing a SnapMirror relationship

To resynchronize a destination to a source, complete the following steps.

Step	Action
1	From the destination filer, enter the following command: **`snapmirror resync [`*`options`*`]`** **`[`*`dest_filer`*`:]{`*`dest_volume`*` | /vol/`*`qtree_path`*`}`** *options* can be any of the following: ◆ `-n` does not execute the resynchronization, but displays what would be done if the `snapmirror resync` command were run. You can use this option to find whether you have a snapshot on the source and on the destination that can be used as the newest common snapshot (base snapshot) so that you can resync a specific SnapMirror relationship. ◆ `-f` forces the operation to proceed without prompting you for confirmation. ◆ `-k` *n* sets the maximum transfer speed to *n* kilobytes per second. This option has the same effect as the `kbs` argument in the /etc/snapmirror.conf file. ◆ `-S` [*source_filer*:]{*source_volume* | *qtree_path*}specifies the filer and volume or qtree you want to use as the source for resynchronization. The source specified by the `-S` option must match a source entry in the /etc/snapmirror.conf file. If entries exist but the source does not match, the operation displays an error message and terminates. If there is no entry for the specified source, the command runs. If the `-S` option is not set, the source must be specified in the /etc/snapmirror.conf file. If it is not specified, the operation displays an error message and terminates.

Step	Action	
	dest_filer is the name of the destination filer. *dest_volume* or */vol/qtree_path* is the destination volume or qtree. If it is a scheduled destination as specified in the /etc/snapmirror.conf file, that source volume or qtree is assumed to be the source. If the destination volume or qtree specified is not in a scheduled relationship, then the `-S` option must be used to provide a source. **Result:** SnapMirror identifies the newest common snapshot, which will be used as the base for resynchronization, and generates a list of snapshots on the destination volume that ◆ are newer than the base snapshot and will be deleted ◆ are older than the base snapshot and have already been deleted from the source **Note** For qtree resynchronization, only the common snapshot is displayed. SnapMirror then prompts you to choose whether to continue.	
2	**If...**	**Then...**
	You want to ◆ Reestablish the SnapMirror pair ◆ Delete the listed snapshots on the destination volume (if you are resynchronizing volumes)	Type `y` at the prompt. **Result:** SnapMirror ◆ Deletes the listed snapshots on the destination volume (if you are resynchronizing volumes). ◆ Makes the destination read-only. ◆ Initiates an update of the destination.

Step	Action	
	If...	**Then...**
	You do not want to lose the data in a snapshot that was created after the common snapshot on the destination, but you want to resynchronize the two volumes or qtrees after the data is saved	◆ Type `n` at the prompt. ◆ Manually copy the data you want to save to the source or other volume. ◆ Return to Step 1 to rerun the `snapmirror resync` command.
	You do not want to reestablish the SnapMirror relationship	Type `n` at the prompt. **Result:** SnapMirror terminates the command.

Volume example: `filerB> snapmirror resync filerB:vol2`

```
The resync base snapshot will be vol2(0001234567)_d.4
These newer snapshots will be deleted from the destination:
hourly.0
hourly.1
These older snapshots have already been deleted from the source and
will be deleted from the destination:
vol2(0001234567)_d.3
Are you sure you want to resync the volume?
```

Qtree example: `filerB> snapmirror resync -S filerA:/vol/vol2/qtreeBob filerB:/vol/vol3/qtreeBak`

```
The resync base snapshot will be vol2(0001234567)_d.4
Data could be lost as a result of this operation.
Are you sure you want to resync the volume?
```

If no common snapshot on the source and destination exists

If SnapMirror cannot find a common snapshot on the source and destination to use as the basis for resynchronization, resynchronization is not possible. SnapMirror generates an error message that states the problem and terminates the command. You must re-initialize the destination to establish the SnapMirror relationship.

Database application testing: a special use for snapmirror resync

Testing software applications that run on a database can sometimes change or even corrupt the database. To ensure that you do not lose data while testing such software applications, you can copy the data to another volume for testing purposes, break the destination volume to return it to writable state, and run the test application on it. Upon completion of the test, you can resynchronize the source and the destination volume to restore the data and repeat the testing as often as you need.

In the following procedure, you can use a combination of the `snapmirror break` and `snapmirror resync` commands to

- make a destination volume writable for testing.
- restore the newly writable volume to its original state if further testing is required.

To set up a destination for testing software applications, complete the following steps.

Step	Action	
1	Create or choose a volume or qtree to be used as a destination for the volume or qtree containing the database. (This example uses a qtree called Testqtree.)	
2	Follow the steps described in “Enabling SnapMirror” on page 116.	
3	On the destination filer, enter the following command to make the destination writable: **`snapmirror break Testqtree`**	
4	Run the application on the data in the former destination (Testqtree).	
5	Check the data in the former destination (Testqtree).	
6	**If the data...**	**Then...**
	Has been altered in some way that is not useful and you want to import a fresh copy of the data for further testing	From the destination filer, enter the following command: **`snapmirror resync Testqtree`** **Result:** SnapMirror makes the former destination volume into a SnapMirror destination again and updates the destination with the latest data.

Step	Action	
	Has not been altered deleteriously, or you wish to stop testing	You have finished.
7	Repeat Steps 4, 5, and 6 until you are satisfied with the testing.	

Disaster recovery: a special use for snapmirror resync

When disaster disables the source of a SnapMirror relationship, you can use the `snapmirror resync` command as part of a strategy to update the repaired source and reestablish the original configuration of the filers.

Example summary: In the following example, the original source (the one disabled by the disaster) is FilerA:vol/volA and the original destination is FilerB:/vol/volB. You use a combination of `snapmirror break` and `snapmirror resync` or `snapmirror initialize` commands to

- temporarily make FilerB:volB the source and FilerA:volA the destination to restore mirrored data back to FilerA:volA and to update FilerA:volA.
- restore FilerA:/vol/volA and FilerB:volB to their original roles as SnapMirror source and SnapMirror destination volume.

What data is preserved: In this example, all data from the last scheduled SnapMirror snapshot before the source was disabled and all the data written to FilerB:vol/volB after it was made writable is preserved. Any data written to FilerA:vol/volA between the last SnapMirror snapshot and the time that FilerA:vol/volA was disabled is not preserved.

Step	Action
1	After the source volume (in this case, FilerA:volA) is disabled, the administrator uses the `snapmirror break` command to make the destination volume, FilerB:volB, writable. **`snapmirror break FilerB:volB`**
2	The administrator redirects the clients of source FilerA to source FilerB. The former clients of FilerA are now accessing and writing to FilerB.

Step	Action
3	The administrator temporarily makes the original source volume into a read-only destination volume. ◆ If FilerA:volA is recoverable, and its data is intact, then the administrator on FilerA uses the `snapmirror resync` command to resynchronize FilerA with FilberB. **`snapmirror resync -S FilerB:VolB FilerA:volA`** ◆ If FilerA:volA is unrecoverable, the administrator makes a new volA on FilerA, and from FilerA, initializes FilerA:volA from FilerB. **`snapmirror initialize -S FilerB:volB FilerA:volA`** This command also makes FilerA:volA a read-only destination.
4	The administrator then redirects the clients from FilerB to FilerA. The clients cannot access or write to FilerA:volA, but they are no longer writing new data to FilerB:volB.
5	The administrator updates FilerA:volA from FilerB to transfer the latest data from Filer B. **`snapmirror update -S FilerB:volB FilerA:volA`**
6	Now, the administrator uses the `snapmirror break` command to make FilerA:volA writable. In FilerA the administrator enters: **`snapmirror break volA`**
7	From FilerB, the administrator uses the `snapmirror resync` command to make FilerB, the original destination, the destination again. **`snapmirror resync volB`**

Migrating data between volumes by using SnapMirror

What SnapMirror does when migrating data

SnapMirror can migrate data between volumes and redirect NFS clients to the new volume without rebooting the filer or remounting to volume on NFS clients. The migration must be run on two volumes which are currently the source volume and destination volume in a SnapMirror relationship. SnapMirror does the following once you start the migration process:

- Performs a SnapMirror incremental transfer to the destination volume.
- Stops NFS and CIFS services to the source volume.
- Migrates NFS file handles to the destination volume.
- Makes the source volume restricted.
- Makes the destination volume read-write.

What SnapMirror does not do when migrating data

SnapMirror does not do the following:

- Transfer IP addresses, license keys, or quota information. You must remount on the NFS clients unless one of the following is true:
    - You transfer the IP address of the source filer to the destination filer independently after the migration.
    - The source and destination volumes reside on the same filer, in which case, the IP address to access either volume is the same.
- Migrate CIFS clients. You must reestablish CIFS client sessions after migrating data to the destination volume.

Migrating data

To migrate data to another volume, complete the following step.

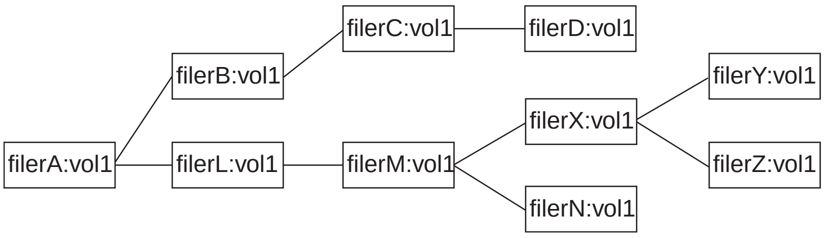
Step	Action
1	Enter the following command: **`snapmirror migrate [`*`srcfiler`*`:]`*`srcvolume`*`[`*`dstfiler`*`:]`*`dstvolume`*** *srcfiler* is the source filer. *srcvolume* is the source volume. *dstfiler* is the destination filer. *dstvolume* is the destination volume.

Copying from one destination to another in a series (cascading)

When to copy from a destination

It can be useful to copy from a SnapMirror destination when the data you want to copy is on a destination that is closer to you than the same data on a source. It is also useful to copy from a destination when you need to copy data from one site to many sites. Instead of propagating the data from one central master site to each of the destination sites, which would require expensive network connections or excessive CPU time, you can propagate the data from one destination to another destination and from that one to the next, in a series.

In the example below, the data on filerA is copied on nine different filers, but only two of the filers copy the data directly from the source. The other seven filers copy the data from a destination site.



Note

The cascading procedure is supported for SnapMirror volume replication only. Cascading is not supported for SnapMirror qtree replication.

How to copy from a destination

You can copy from a destination volume the same way you copy from a writable source volume. For information on how to set up a destination, see "Enabling SnapMirror" on page 116.

Sample cascade setup

To support a series of cascading volume destinations as shown in the diagram above, the /etc/snapmirror.conf file in each of the cascaded filers would look like this:

```
filerA:vol1 filerB:vol1 - 15 * * 1,2,3,4,5
filerA:vol1 filerL:vol1 - 15 * * 1,2,3,4,5
filerB:vol1 filerC:vol1 - 25 * * 1,2,3,4,5
filerC:vol1 filerD:vol1 - 35 * * 1,2,3,4,5
```

```
filerL:vol1 filerM:vol1 - 25 * * 1,2,3,4,5
filerM:vol1 filerX:vol1 - 35 * * 1,2,3,4,5
filerM:vol1:filerN:vol1 - 35 * * 1,2,3,4,5
filerX:vol1 filerY:vol1 - 45 * * 1,2,3,4,5
filerX:vol1 filerZ:vol1 - 45 * * 1,2,3,4,5
```

Note

When specifying the destination update schedule in the snapmirror.conf file, stagger the update times instead of starting multiple destination updates at the same time. If SnapMirror does not have enough resources to perform all scheduled destination updates, it postpones some updates. As a result, SnapMirror might need to perform subsequent updates at times that are different from those you specify in the snapmirror.conf file.

How SnapMirror handles snapshots for cascading destinations

SnapMirror retains the snapshots on the original source volume needed for transfers to destinations further down the line. Snapshots that are still needed by a destination are labeled “snapmirror” in the output of the `snap list` command. SnapMirror deletes the snapshots it no longer needs.

If you remove a destination from the cascade, you use the `snapmirror release` command from the immediate source to tell SnapMirror to delete the snapshots associated with that destination.

Cascading from a synchronous SnapMirror

You can cascade volume destinations from a synchronous Snapmirror destination, but the cascading series is slightly different from that for asynchronous SnapMirror. For a synchronous SnapMirror, the first replication is the only synchronous SnapMirror replication in the cascade, and the synchronous replication can be to one destination filer only. Subsequent SnapMirror replications cascading from that destination filer must be asynchronous and can be to multiple destination filers.

Listing SnapMirror destinations for a volume in a cascading series

To display the SnapMirror destinations for a source volume, complete the following step.

Step	Action
1	From the filer with the volume serving as the source, enter the following command: **`snapmirror destinations [-s] [`*`volume_name`*`]`** The `-s` option generates a list of the names of the snapshots retained for each destination. *volume_name* is the name of the source volume for which you want to see the destinations.

Sample snapmirror destinations command output

Following is an example of the `snapmirror destinations` command output for the configuration in the diagram above.

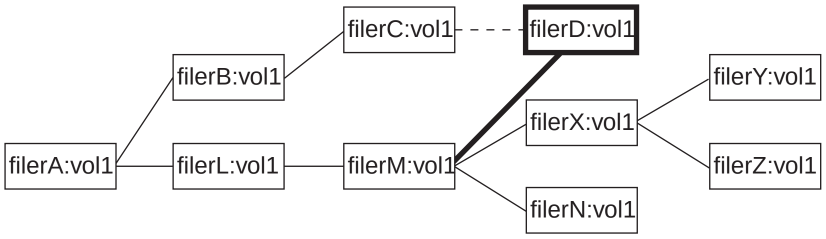
```
filerA> snapmirror destinations vol1

Path        Destination
/vol/vol1   filerB:vol1->filerC:vol1->filerD:vol8
/vol/vol1   filerL:vol1->filerM:vol1->filerX:vol1->filerY:vol1
/vol/vol1   filerL:vol1->filerM:vol1->filerX:vol1->filerZ:vol1
/vol/vol1   filerL:vol1->filerM:vol1->filerN:vol1
```

If you do not specify a volume name in the command, the output includes information about each destination volume on the filer.

Restructuring a cascade

You might restructure a cascade to balance loading of your filers; to use a filer or volume for a different purpose; or to perform upgrades, maintenance, or repairs. For example, in the cascade structure that follows, you might want to make filerD:vol1 a destination of filerM:vol1 instead of a destination of filerC:vol1.



To restructure the relationship of the destinations in a cascade, complete the following steps.

Step	Action
1	On the destination filer, change the /etc/snapmirror.conf file to indicate the new source for the destination. **Example:** `filerM:vol1 filerD:vol1 - 35 * * 1,2,3,4,5`

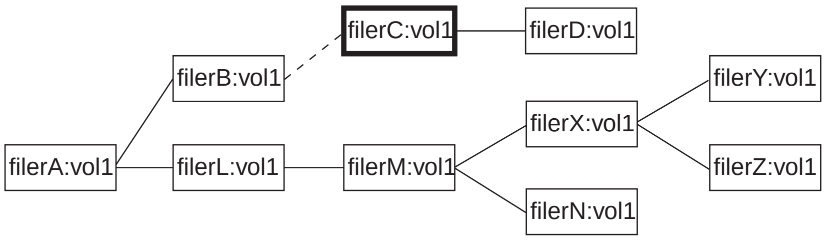
Step	Action	
2	**If...**	**Then**
	The newest snapshot on the destination exists on the source	Use the following command to update the destination from the new source: `snapmirror update -S` ***`source_volume dest_filer`*****`:`*****`dest_volume`*** **Example:** `snapmirror update -S filerM:vol1 filerD:vol1`
3	The newest snapshot on the destination does not exist on the source	Perform one of the following tasks: **1.** Update the new source from the original source using the `snapmirror update` command. Wait for the destination to update. **2.** Make the destination writable using the `snapmirror break` command. Then resynchronize the destination with the new source using the `snapmirror resync` command.

4	Release the former source using the following command:
	`snapmirror release` ***`source_volume`***
	`[[`***`dest_filer`*****`:]`*****`dest_volume`*****`]`**
	Example:
	`filerC> snapmirror release filerC:vol1 filerD:vol1`	

Example of disconnecting a destination from a cascading series:

For the example shown in the diagram that follows, suppose that from filerB you enter the following command:

```
snapmirror release vol1 filerC:vol1
```



These results follow:

- filerA:vol1 continues to be the source for the destination filerB:vol1.
- filerC:vol1 no longer copies from filerB:vol1. SnapMirror retains snapshots for filerC and below.
- If filerC requests an update from filerB, the destination is reestablished if it is still not writable and the base snapshot still exists on the source.
- filerD:vol1 still copies filerC:vol1.
- All the destinations that depend on filerL:vol1 continue functioning as before.

You can check that the destination was released by running the `snapmirror destinations` command on filerA, as follows:

```
filerA> snapmirror destinations -s filerA:vol1
Volume Snapshot                      Destination
vol1   filerB(0015269532)_vol1.37   filerB:vol1
vol1   filerL(0015269532)_vol1.42   filerL:vol1->filerM:vol1-
>filerXvol1->filerY:vol1
vol1   filerL(0015269532)_vol1.42   filerL:vol1->filerM:vol1-
>filerXvol1->filerZ:vol1
vol1   filerL(0015269532)_vol1.42   filerL:vol1->filerM:vol1-
>filerN:vol1
```

Note

If you want to permanently release a destination, you need to either delete the entry in the /etc/snapmirror.conf file or comment out the entry by preceding it with a pound sign (#). Otherwise, SnapMirror continues to try to update the destination.

Using SnapMirror to copy a volume to local tape

Why use SnapMirror to local tape

You might want use SnapMirror to copy a volume from a source filer to local tape for the following reasons:

- Network transfer time of the baseline transfer between a SnapMirror source and a SnapMirror destination is prohibitive. For details see the following section, "SnapMirror source-to-tape-to-destination scenario" on page 177.
- You are backing up SnapVault secondary storage data to tape for offline storage or to protect against the possible loss of the SnapVault secondary. For details see "SnapVault-to-tape backup scenario" on page 183.

Note

The `snapmirror store` command does not support SnapMirror qtree replication to tape. If you specify a qtree path as the source or destination, SnapMirror returns an error message.

Considerations before using as a backup method

SnapMirror replication to copy data to local tape is meant as an initialization process for SnapMirror relationships and has limitations when used as a backup menthod. Consider the following limitations before using SnapMirror to tape as a backup method.

Disk geometry: When SnapMirror replicates data to tape, it optimizes writing of the file system based on the disk geometry of either the source or destination file system. The disk geometry of the source file system is used by default, but you can specify the disk geometry of the destination file system using the `snapmirror store -g` command. See the na_snapmirror(1) man page for more information.

If you retrieve a backup tape into a file system that does not match the disk geometry of the filer used when writing the data on to tape, the retrieve can be extremely slow.

File system version: Data written to tape is a block by block copy of the file system associated with a particular version of Data ONTAP; therefore, when you retrieve data from tape, the destination of the retrieval must use the same or later version of Data ONTAP used when storing data to tape. If you attempt to retrieve data to a destination that uses an older version of Data ONTAP, the retrieval will fail.

Snapshot issues: Because the intended purpose of SM to tape is to initialize mirrors for SnapMirror relationships, it maintains snapshots that future SM relationships use to perform updates. When used as a backup method, snapshots are created, but never used; therefore, snapshot and data resources are wasted.

Note

You can manage and delete snapshots using the `snapmirror release` and `snapmirror destinations` commands. See the na_snapmirror(1) man page for more information.

Volume types: The traditional volume file system format is different from the flexible volume file system format. Because of this difference, a backed up flexible volume cannot be retrieved on a traditional volume. Likewise, a backed up traditional volume cannot be retrieved on a flexible volume.

Lack of archive support: Some backups are made to be kept for a long time; as such, they become long term storage of old versions of data. Backups using SnapMirror replication to tape are in a format that is only readable by Data ONTAP and the WAFL file system, and is not meant for long term storage. Keeping a set of tapes for extended periods of time risks the ability to restore them in the future.

Effects of a bad tape: When retrieving a file system from tape, the format of the SnapMirror replication to tape requires the entire file system to be retrieved before the file system is usable. A bad tape when retrieving a file system means that not all of the file system is retrievable; therefore, the file system is not constructible and all of the data is unavailable.

Lack of backup features: Using SnapMirror replication to copy data to tape does not have features that backup software does. For example, features like individual file restore, backup indexing, and incremental backups are not supported.

SnapMirror source-to-tape-to-destination scenario

In this scenario, you want to establish a SnapMirror relationship between a source filer and a destination filer over a low-bandwidth connection. Incremental snapshot mirroring from the source to the destination over the low-bandwidth connection is feasible, but the initial base snapshot mirroring is not. In such a case it might be faster to first transfer the initial base snapshot image from source to destination via tape, and then set up incremental SnapMirror updates to the destination filer via the low-bandwidth connection.

Prerequisites: This scenario assumes the following configuration:

- A low-bandwidth connection between the source and destination filers
- A local tape drive attached to the source filer
- A local tape drive attached to the destination filer

Caution

To prevent extended tape-to-filer transfer time, it is recommended that the destination filer disks be the same size and in the same RAID configuration as the source filer disks.

Sequence of SnapMirror operations: You follow this sequence to set up this arrangement:

1. On the source filer, use the `snapmirror store` command to copy all volume snapshots, including the base snapshot, to tape, and use the `snapmirror use` command to continue the copy if more than one backup tape is necessary.

   For more information, see "Copying source-to-intermediate tape" on page 179.

2. Physically transport the backup tapes from the source filer to the destination filer.

3. On the destination filer, use the `vol create` and `vol restrict` commands to set up a SnapMirror target volume.

4. Use the `snapmirror retrieve` command to copy the initial SnapMirror tape to the destination filer and, if necessary, use the `snapmirror use` command to continue the copy if it is stored on more than one backup tape.

   Then, either use the `snapmirror update` command to manually mirror an incremental update from the source to the destination filer over the low-bandwidth connection, or edit the snapmirror.conf file to set up an incremental update schedule from the source to destination filer.

   For more information, see "Initializing a SnapMirror destination via local tape" on page 181.

5. After completing manual or scheduled incremental update over a connection, you can use the `snapmirror release` command to eliminate the source-to-tape relationship and associated snapshot. For more information see "Releasing a SnapMirror source-to-tape relationship" on page 182.

Copying source-to-intermediate tape: To copy a source volume to an intermediate tape, complete the following steps.

Step	Action
1	If you do not know whether the disk geometries (that is, the size and number of disks) of the source volume and the ultimate destination volume match, use the `snapmirror retrieve -g` command to determine if this is so. In the filer consoles of both the source and destination, enter the following command: **`snapmirror retrieve -g`** ***`vol_name`*** For both the source and the destination volume, the system displays the number and block size of the disks it includes. For example, a return value of `7200000x10 7000x10` means that the specified volume consists of 10 disks of 720,000 blocks and 10 disks of 7,000 blocks. ◆ The most desirable result is for the source and destination volume disk geometries to match. ◆ If the geometries do not match, record the disk geometry of the destination volume for later use.
2	At the source filer, load the tape into a local attached tape device.

Step	Action
3	At the source filer, start the data transfer to tape by entering the following command: **`snapmirror store [-g`** ***`dest_disk_geom`*** **`]`** ***`source_volume dest_tapedevices`*** `-g` *dest_disk_geom* applies if the disk geometry of the destination volume, as determined in Step 1, is different from the disk geometry of the source volume. If they are different, use the `-g` parameter to specify the destination volume disk geometry, for example: **`-g 140000x10,7000x10`** *source_volume* is the volume you are copying. *dest_tapedevices* is a comma-separated list of tape devices to which you are copying the volume. **Result:** SnapMirror starts transferring data to tape. This command creates a baseline transfer. If you have to use more than one tape, SnapMirror prompts you for another tape. **Example:** **`snapmirror store -g 14000x10,7000X10 vol2 nrst0a,rst1a`**
4	If SnapMirror prompts you for another tape, add another tape to the drive, and continue transfer of data to tape by entering the following command: **`snapmirror use`** ***`dest_tapedevices tape_drive`*** *dest_tapedevices* is the tape device or a comma-separated list of tape devices to which you are copying the volume. *tape_drive* is the drive holding the new tape you supplied. **Note** The `snapmirror use` command does not support copying qtrees from tape. If you specify a qtree path as the source or destination, SnapMirror returns an error message.
5	Repeat Step 4 until SnapMirror finishes copying the volume to tape.

Step	Action
6	Follow the procedure described in "Initializing a SnapMirror destination via local tape" on page 181.

Initializing a SnapMirror destination via local tape: To initialize a SnapMirror destination volume via local tape, complete the following steps.

Step	Action
1	Create a volume on the SnapMirror destination filer. See the *System Administration Storage Management Guide* for information about how to create a volume.
2	Put the volume in the restricted state. See the *System Administration Storage Management Guide* for information about how to restrict a volume.
3	Load the tape (made with `snapmirror store`) into the destination filer's local tape device.
4	Start the initialization by entering the following command on the destination filer: **`snapmirror retrieve`** ***`dest_volume tape_drive`*** *dest_volume* is the volume that you are initializing. *tape_drive* is a tape device or a comma-separated list of devices from which you are restoring the volume. **Result:** SnapMirror starts transferring data from the tape. If data is stored on more than one tape, SnapMirror prompts you for the next tape.
5	If Data ONTAP prompts you for another tape, add the next tape to the drive and restart the initialization by entering the following command: **`snapmirror use`** ***`volume tape_list`*** *volume* is the volume to which you are restoring. *tape_list* is the tape device from which you are restoring the volume.

Step	Action
6	Repeat Step 5 until SnapMirror finishes initializing the volume from the tape.
7	If you need to, you can update the data online manually with the following command: **`snapmirror update [-k`** ***`n`*** **`] -S`** ***`source_filer`*** **`:`** ***`source_volume`*** **`[`** ***`dest_filer`*** **`:]`** ***`dest_volume`*** `-k` *n* sets the maximum transfer speed to *n* kilobytes per second. This option has the same effect as the `kbs` argument in the /etc/snapmirror.conf file. `-S` *source_filer*:*source_volume* specifies the source filer and volume for the migration. *source_volume* is the volume you want to copy. *dest_filer* is the name of the destination filer. *dest_volume* is the destination volume. For more information about the `snapmirror update` command, see "Updating a destination manually" on page 126. **Note** Alternatively, you can update the baseline transfer automatically with the schedule you set in the /etc/snapmirror.conf. file. See "Creating and editing the snapmirror.conf file" on page 105.

Releasing a SnapMirror source-to-tape relationship: After a successful update of the destination filer over a connection, you no longer require the source-to-tape SnapMirror relationship and its associated snapshot that you established with the `snapmirror store` command. To end the source-to-tape backup and delete the snapshot from the source filer, complete the following steps.

Step	Action

1	In the source filer console, enter the following command:
	
	`snapmirror status`
	
	Your system will display at least two SnapMirror relationships:
	◆ the source-to-tape relationship, established when you used the `snapmirror store` command
	◆ the source-to-destination relationship, established when you used the `snapmirror update` command
	
	Example:
	`>snapmirror status`
	`....`
	`source         destination     state     ....`
	`s_filer:vol1   snapmirror_tape_01_15_03_20:05:32  ...`
	`s_filer:vol1   d_filer:vol1   snapmirrored  ...`

2	Release the source-to-tape SnapMirror relationship. For this particular operation, the `snapmirror release` syntax is unique: `snapmirror release` *`source_vol tape_snapshot_name`* **Example:** **`snapmirror release vol1 snapmirror_tape_01_15_03_20:05:32`** **Note** Do not release any other SnapMirror relationship. Those relationships are necessary to continue your incremental updates over the low-bandwidth connection.

SnapVault-to-tape backup scenario

In this scenario, you use SnapMirror to replicate snapshots from a SnapVault secondary storage system to an attached local tape drive for the purposes of backup. In case of data loss on the SnapVault secondary system, you can restore a specified snapshot back to the SnapVault secondary system.

Note

This scenario applies only to backup of SnapVault secondary storage. Do not apply this configuration for any other purpose.

Prerequisites: You need a SnapVault secondary system with an attached local tape drive. The volume on which the secondary storage qtrees reside must also be configured as a SnapMirror source volume.

Sequence of SnapMirror operations: You follow these steps to set up this arrangement.

1. On the SnapVault secondary/SnapMirror source filer, use the `snapmirror store` command to copy the base snapshot of the volume to tape and use the `snapmirror use` command to continue the copy if more than one backup tape is necessary.

   For more information, see "Copying from a SnapVault volume to local tape" on page 184.

2. Use the `snapmirror update` command to manually mirror incremental updates to tape.

3. In event of data loss on the SnapVault secondary storage system, convert the volume that holds the SnapVault data from its role as a SnapMirror source volume to a destination volume, mirror the tape contents back, and then convert the restored destination volume back to a regular volume again.

   For more information, see "Restoring to SnapVault from a local tape" on page 185.

Copying from a SnapVault volume to local tape: To back up a SnapVault volume to tape, complete the following steps.

Step	Action
1	At the SnapVault secondary storage system, load the tape into a local attached tape device.

Step	Action
2	At the SnapVault secondary storage system, start the data transfer to tape by entering the following command: **`snapmirror store`** ***`sv_volume dest_tapedevices`*** *sv_volume* is the volume you are copying. *dest_tapedevices* is a comma-separated list of tape devices to which you are copying the volume. **Result:** SnapMirror starts transferring data to tape. This command creates a baseline transfer. If you have to use more than one tape, SnapMirror prompts you for another tape.
3	If SnapMirror prompts you for another tape, add another tape to the drive, and continue transfer of data to tape by entering the following command: **`snapmirror use`** ***`dest_tapedevices tape_drive`*** *dest_tapedevices* is a comma-separated list of tape devices to which you are copying the volume. *tape_drive* is the drive holding the new tape you supplied. **Note** The `snapmirror use` command does not support copying qtrees from tape. If you specify a qtree path as the source or destination, SnapMirror returns an error message.
4	Repeat Step 3 until SnapMirror finishes copying the volume to tape.
5	If the volume on the SnapVault secondary system that you backed up ever needs to be restored, follow the procedure described in “Restoring to SnapVault from a local tape” on page 185.

Restoring to SnapVault from a local tape: In event of data loss on a SnapVault secondary storage volume that is also configured as a SnapMirror source, data that has been SnapMirrored to tape can be restored to the SnapVault secondary storage volume with the `snapmirror retrieve` command.

Note

The `snapmirror retrieve` command restores only full volumes. If you specify a qtree path as the source or destination, SnapMirror returns an error message.

To restore a SnapMirrored volume on a SnapVault secondary storage system from tape, complete the following steps.

Step	Action
1	On the SnapVault secondary storage system, use the `vol restrict` command to put the volume that you want to restore from tape into restricted state. See the *System Administration Storage Management Guide* for information about how to restrict a volume.
2	Load the tape (made with `snapmirror store`) into the local tape device.
3	Start the initialization by entering the following command on the SnapVault secondary storage system: **`snapmirror retrieve [-h]`** ***`rest_volume tape_drive`*** The `-h` option displays the headers of the source tapes but does not transfer data. *rest_volume* is the volume that you are restoring. *tape_drive* is a tape device or a comma-separated list of devices from which you are restoring the volume. **Result:** SnapMirror starts transferring data from the tape. If data is stored on more than one tape, SnapMirror prompts you for the next tape.
4	If Data ONTAP prompts you for another tape, add the next tape to the drive and restart the initialization by entering the following command: **`snapmirror use`** ***`rest_volume tape_list`*** *rest_volume* is the volume to which you are restoring. *tape_list* is the tape device from which you are restoring the volume.
5	Repeat Step 4 until SnapMirror finishes initializing the volume from the tape.

Step	Action
6	If incremental updates to the baseline snapshot data that you just restored exist on tape, insert that tape into the SnapVault secondary storage system's local tape drive and enter the following command: **`snapmirror update [`*`options`*`] [`*`sv_secondary`*`:]`*`sv_volume`*** *options* can be one or more of the following: ◆ `-k` *n* sets the maximum transfer speed to *n* kilobytes per second. This option has the same effect as the `kbs` argument in the /etc/snapmirror.conf file. ◆ `-S` [*source_filer:*]*source_volume* specifies the source filer and volume for the migration. *source_volume* is the volume you want to copy. *sv_secondary* is the name of the SnapVault secondary system that you want to restore to. *sv_volume* is the volume that you want to restore. For more information about the `snapmirror update` command, see "Updating a destination manually" on page 126.
7	After all tape-to-SnapVault retrieve and update operations are complete, use the `snapmirror break` command to turn the restored filer volume back to completely read-writable status. **`snapmirror break /vol/`*`volume_name/`***

How SnapMirror works with the dump command

How to back up data in the destination volume

You can use the `dump` command to back up data from a SnapMirror destination volume. The `dump` command picks the most recent snapshot and copies that to tape.

You can back up any snapshot displayed by the `snap list` command on the destination. You can also create a snapshot on the source volume, copy the snapshot to the destination, and use the `dump` command to back up this snapshot from the destination to tape.

Example: `dump 0f rst0a /vol/vol1/.snapshot/weekly.0`

Effect of the dump command on the destination update schedule

Running the `dump` command on a SnapMirror destination affects SnapMirror operations on that destination in the following ways:

- Scheduled incremental SnapMirror updates of a destination *volume* can occur concurrently with a `dump` operation to tape; however, if a scheduled SnapMirror update to the destination volume involves the deletion of a snapshot that the `dump` operation is currently writing to tape, the SnapMirror update will be delayed until the `dump` operation is complete.

  Note

  SnapMirror updates of a destination *qtree* are not affected by `dump` operations under any circumstances.

- SnapMirror `break`, `resync`, and `migrate` operations cannot be carried out concurrently with the `dump` operation.

Fixing changes to SnapMirror elements

Accidental deletion of SnapMirror snapshots

SnapMirror snapshots stored on either the SnapMirror source or destination location must not be deleted.

If the base snapshot (most recent common snapshot) is accidentally deleted from either the source or destination location, attempt recovery as follows:

You might be able to recover without reinitializing the destination by breaking the SnapMirror relationship and then resynchronizing the source and the destination.

As long as there is at least one snapshot common to both the source and the destination, resynchronization will succeed. See the section on the `snapmirror break` command in “Converting a destination to a writable volume or qtree” on page 156 and the section on the `snapmirror resync` command in “Resynchronizing a SnapMirror relationship” on page 162.

If there is no snapshot common to both the source and the destination, you need to use the `snapmirror initialize` command over the network. Or, if the source and destination are volumes, use the `snapmirror store` command to store the source volume on tape and then use the `snapmirror retrieve` command to restore the volume from the tape to the destination.

If you change a destination volume name

If you change the name of a SnapMirror destination volume, you need to manually correct the SnapMirror relationships affected by the change. SnapMirror will be unable to replicate source volume data to a newly-named destination volume whose configuration information is incomplete.

In the case below, the destination, volJobak, was renamed to volStatbak. After the renaming, the `snapmirror status` command does not show the source. Instead the entry is shown with a '-' appears in the source column.

```
filerB> vol rename volJobak volStatbak
volJbak renamed to volStatbak
you may need to update /etc/exports
filerB> snapmirror status volJobak
Snapmirror is on.
filerB>snapmirror status volStatbak
Snapmirror is on.
```

```
Source        Destination          State          Lag         Status
-             filerB:volStatbak   Snapmirrored   -00:03:22   Idle
```

If you change the volume name of a SnapMirror source or destination, you need to make the following changes:

- ◆ Update the snapmirror.conf file, if there is an old entry.
- ◆ Use the `snapmirror release` command to update the old destination name, so SnapMirror will release the softlock and the old snapshot.
- ◆ Use the `snapmirror update` command on the new volume name, so status registry is updated with the new volume name.
- ◆ Update the /etc/exports file.

Caution

If a filer is running at its limit of concurrent transfers, and you attempt to initiate more transfers through manual `snapmirror update` command, the filer can panic. When this particular panic case is triggered, the system log will contain the message “snapmirror: unexpected need to notify a waiting process in replica_unconfigure”.

Creating SnapLock destination volumes

What SnapLock volumes are

SnapLock volumes are Write-Once-Read-Many (WORM) volumes that you create for data you want to archive permanently. There are two types of SnapLock volumes:

- SnapLock Compliance volume—for strict regulatory environments, such as SEC 17a-4 compliant environments
- SnapLock Enterprise volume—for environments without regulatory restrictions.

For details about licensing and creating SnapLock volumes, see the *Storage Management Guide*.

SnapLock destination volume restrictions

A SnapMirror destination volume created as a SnapLock Compliance volume has the following restrictions:

- You cannot use the `snapmirror resync` command to return a former destination volume and resynchronize its content with the source.
- You cannot reinitialize a SnapLock Compliance destination volume because data on the volume cannot be changed. If you break the SnapMirror relationship between the source and destination volumes using the `snapmirror break` command, Data ONTAP prevents you from ever reestablishing the SnapMirror relationship with the same SnapLock Compliance destination volume.

  You can initalize a new empty SnapLock Compliance destination volume.

A SnapMirror destination volume created as a SnapLock Enterprise volume is the same as a destination volume created for a non-SnapLock volume. You can perform all the same administrative tasks on a SnapLock Enterprise destination that you can perform on a non-SnapLock destination.

Creating a SnapLock Compliance SnapMirror relationship

To create a SnapMirror relationship from a SnapLock Compliance volume to a SnapLock Compliance destination volume, complete the following steps.

Note

This procedure assumes you have read about basic SnapMirror setup and you have experience with creating SnapMirror relationships. See "Setting up a basic SnapMirror operation" on page 93 for a details about first-time SnapMirror setup.

Step	Action
1	On the administration host, create or edit the snapmirror.conf file on the destination filer to specify the SnapLock Compliance volume source and destination. **Example:** The following entry specifies asynchronous mirroring from vol1 of s_filer and vol2 of d_filer using the default replication arguments. `s_filer:vol1 d_filer:vol2 - * * * *`
2	On the destination filer console, create the destination volume using the `vol create` command. **Example:** **`vol create vol2 2`** **Caution** Do not use the `vol create -L` command to create a SnapLock Compliance volume because volume SnapMirror cannot use it as a destination volume.
3	On the destination filer console, mark the volume as restricted using the `vol restrict` command.

Step	Action
4	On the destination filer console, create an initial complete (baseline) copy of the source on the destination and start the mirroring process using the `snapmirror initialize -L` command. **Result:** After successfully completing the baseline transfer, Data ONTAP converts the destination volume to a SnapLock Compliance destination volume before bringing it online. **Example:** **`snapmirror initialize -S s_filer:vol1 -L d_filer:vol2`** **Note** After the destination volume is converted to a SnapLock Compliance destination volume, it will always be a SnapLock Compliance volume and cannot be changed.

Creating a SnapLock Enterprise SnapMirror relationship

Creating a SnapMirror relationship from a SnapLock Enterprise source volume to a SnapLock Enterprise destination volume is the same as creating a SnapMirror relationship for non-SnapLock Enterprise volumes. See "Setting up a basic SnapMirror operation" on page 93 for a details about first-time SnapMirror creation.

Data Protection Using SnapVault 5

About this chapter

This chapter discusses how the optional SnapVault on-line backup feature of Data ONTAP provides a faster, cheaper alternative or a complement to tape for data protection.

Topics in this chapter

This chapter discusses the following topics:

SnapVault overview

What is SnapVault

SnapVault is a disk-based storage backup feature of Data ONTAP that enables data stored on multiple filer and Open Systems (non-filer) storage systems to be backed up to a central, secondary storage system quickly and efficiently as read-only snapshots. In event of data loss or corruption on a filer or open storage system, backed up data can be restored from the SnapVault secondary storage system with less downtime and less of the uncertainty associated with conventional tape backup and restore operations.

Terminology

This chapter uses the following terms to describe the SnapVault feature:

- Primary storage system—a system whose data is to be backed up
- Secondary storage system—a filer or NearStore™ system to which data is backed up
- Primary system qtree—a qtree on a primary storage system whose data is backed up to a secondary qtree on a secondary storage system
- Secondary system qtree—a qtree on a secondary storage system to which data from a primary qtree on a primary storage system is backed up
- Open Systems platform—a non-filer system, such as a server running AIX, Solaris, HP-UX, Linux, Windows NT, Windows 2000, or Windows 2003, whose data can be backed up to a SnapVault secondary storage system
- Open Systems SnapVault agent—a software module that can be installed on a non-filer system that enables the system to back up its data to a SnapVault secondary storage system
- SnapVault relationship—the backup relationship between a qtree on a filer primary system or a directory on an Open Systems primary platform and its corresponding secondary system qtree
- SnapVault snapshot—the backup images that SnapVault creates at intervals on its primary and secondary storage systems. SnapVault snapshots capture the state of primary qtree data on each primary system. This data is transferred to secondary qtrees on the SnapVault secondary system, which creates and maintains versions of snapshots of the combined data for long-term storage and possible restore operations.
- SnapVault snapshot basename—a name that you assign to a set of SnapVault snapshots through the `snapvault snap sched` command. As incremental snapshots for a set are taken and stored on both the primary and secondary

storage systems, the system appends a number (0, 1, 2, 3, and so on) to the basenames to track the most recent and earlier snapshot updates.

- SnapVault baseline transfer—an initial complete backup of a filer primary storage qtree or an Open Systems platform directory to a corresponding qtree on the secondary storage system
- SnapVault incremental transfer—a followup backup to the secondary storage system that contains only the changes to the primary storage data between the current and last transfer actions

Advantages of SnapVault

The SnapVault disk-based backup and restore system enables you to do the following:

- Browse backed-up files online
- Schedule fast, frequent, efficient backup of large amounts of data
- Carry out fast, flexible, selective, and simple data restore operations
- Minimize media consumption and system overhead through incremental backup
- If tape backup is necessary, offload the tape backup task from the primary storage systems to the snapshot secondary storage system, which centralizes the operation and saves resources
- Configure and maintain a single system for backing up data stored on multiple platforms: Network Appliance filer, AIX, Solaris, HP-UX, Linux, Windows NT server, Windows 2000 server, or Windows 2003 server systems

What data gets backed up and restored

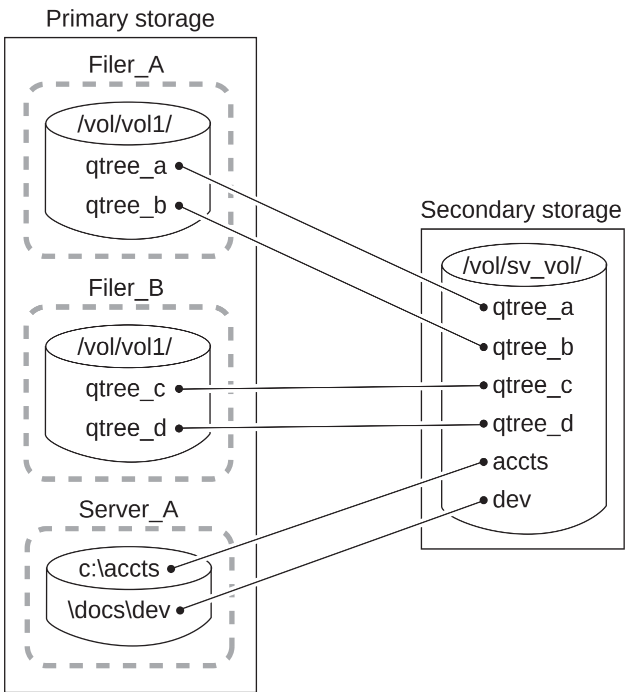
The data structures that are backed up and restored through SnapVault depend on the primary storage system.

- On NetApp filer systems, the qtree is the basic unit of SnapVault backup and restore.

  SnapVault backs up specified qtrees on the primary filer system to associated qtrees on the SnapVault secondary storage system. If necessary, data is restored from the secondary qtrees back to their associated primary qtrees.
- On Open System storage platforms, the directory is the basic unit of SnapVault backup.

  SnapVault backs up specified directories from the native system to specified qtrees in the SnapVault secondary storage system. If necessary SnapVault can restore an entire directory or a specified file to the Open System storage platform.

Primary storage qtrees from multiple filers, or primary directories from multiple Open Systems storage platforms, can all be backed up to associated qtrees on a single SnapVault secondary volume.



Basic SnapVault deployment

The basic SnapVault backup system deployment consists of the following components.

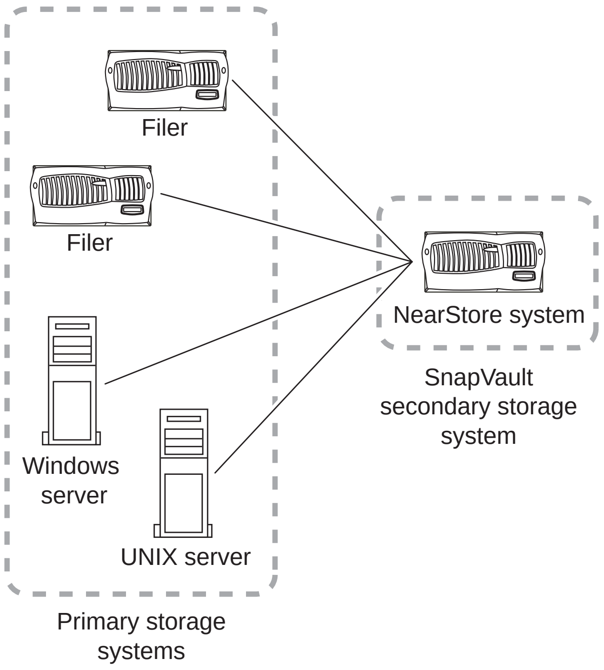
Primary storage systems: Primary storage systems are the filer and Open Systems primary storage platforms to be backed up.

- On filer primary storage systems, SnapVault backs up primary qtree data, non-qtree data, and entire volumes to qtree locations on the SnapVault secondary storage system.
- Supported Open Systems storage platforms include Windows NT servers, Windows 2000 servers, Solaris servers, AIX servers, Linux servers, and HP-UX servers. On Open Systems storage platforms, SnapVault can back up

directories to qtree locations on the secondary storage system. SnapVault can restore directories and single files.

Secondary storage system: The SnapVault secondary storage system is the central disk-based unit that receives and stores backup data from the filer and Open Systems storage data as snapshots. Any filer can be configured as a SnapVault secondary storage system; however, the recommended hardware platform is a Network Appliance NearStore system.

The following figure shows a basic SnapVault deployment.



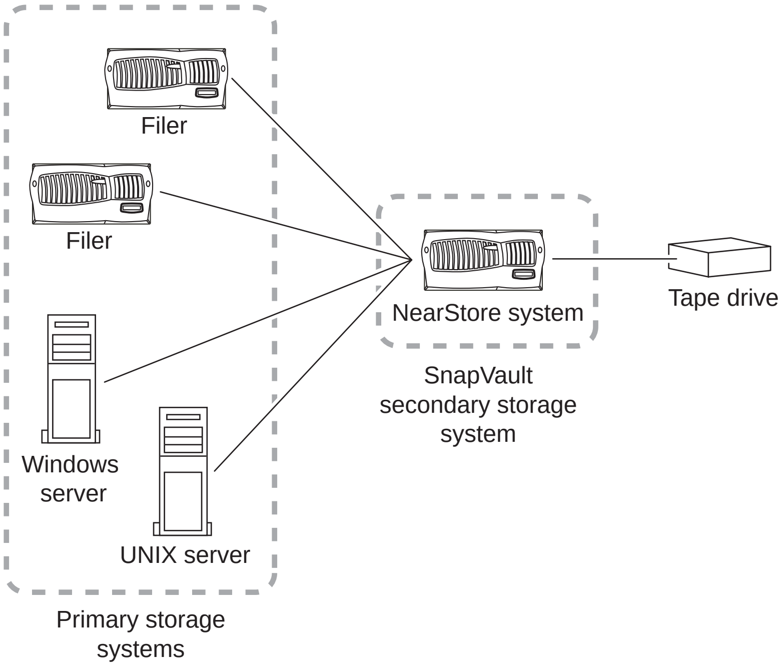
Primary to secondary to tape backup variation

A common variation to the basic SnapVault backup deployment adds a tape backup of the SnapVault secondary storage system. This deployment might serve two purposes:

◆ It enables you to store an unlimited number of network backups offline while keeping your most recent backups available online in secondary storage, if necessary, for quick restoration. If you run a single tape backup off the SnapVault secondary storage system, your filer and Open Systems storage platforms are not subject to the performance degradation, system unavailability, and complexity of direct tape backup of multiple systems.

- It can be used to restore data to a SnapVault secondary storage system in case of data loss or corruption on that system.

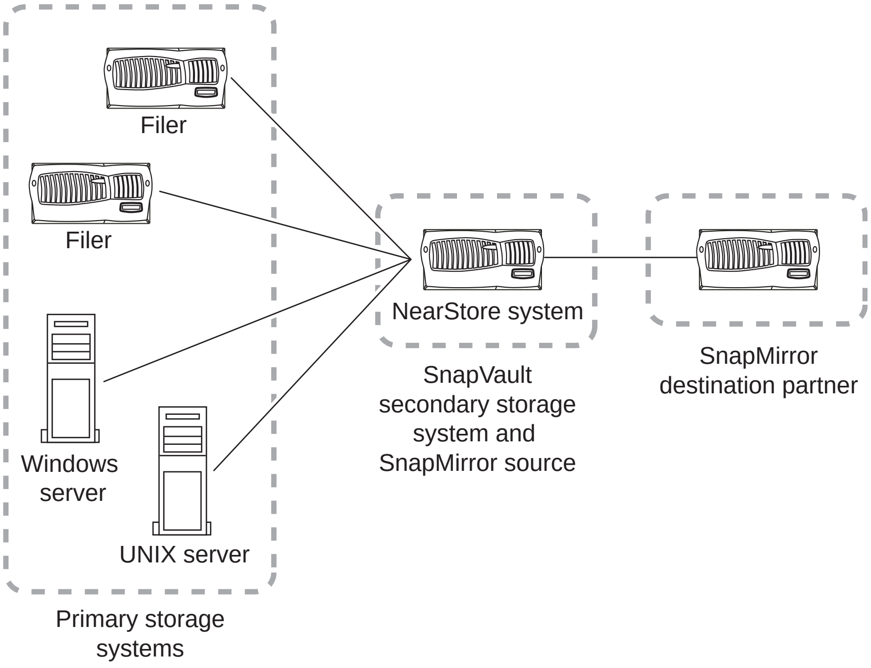
The following figure shows a basic SnapVault deployment plus tape backup.



Primary to secondary to SnapMirror variation

Another variation to the basic SnapVault deployment protects snapshots stored on SnapVault secondary storage against problems with the secondary storage system itself. The data backed up to SnapVault secondary storage is mirrored to a unit configured as a SnapMirror partner, or *destination*. If the secondary storage system fails, the data mirrored to the SnapMirror destination can be converted to a secondary storage system and used to continue the SnapVault backup operation with a minimum of disruption.

The following figure shows a SnapVault deployment with a SnapMirror backup.


Filer
Filer
Windows server
UNIX server
Primary storage systems
NearStore system
SnapVault secondary storage system and SnapMirror source
SnapMirror destination partner

How SnapVault backup of filer systems works

The process of SnapVault backup of filer qtrees is as follows:

Starting the baseline transfers:

- In response to command line input, the SnapVault secondary storage system requests initial base transfers of qtrees specified for backup from a primary storage volume to a secondary storage volume. These transfers establish SnapVault relationships between the primary and secondary qtrees.
- Each filer primary storage system, when requested by the secondary storage system, transfers initial base images of specified primary qtrees to qtree locations on the secondary storage system.

Making scheduled incremental transfers:

- Each filer primary storage system, in response to command line input, creates sets of scheduled SnapVault snapshots (which, for tracking purposes, you might name according to frequency, for example, “sv_hourly,” “sv_nightly,” and so on) of the volumes containing the qtrees to be backed up.

For each snapshot set, SnapVault saves the number of primary storage snapshots you specify and assigns each snapshot a version number (0 for most current, 1 for second most recent, and so on).

- The SnapVault secondary storage system, in response to command line input, carries out a specified set of scheduled data transfer and snapshot actions. For each of its secondary qtrees on a given volume, SnapVault retrieves, from the snapshot data of each corresponding primary qtree, the incremental changes to the primary qtrees made since the last data transfer.

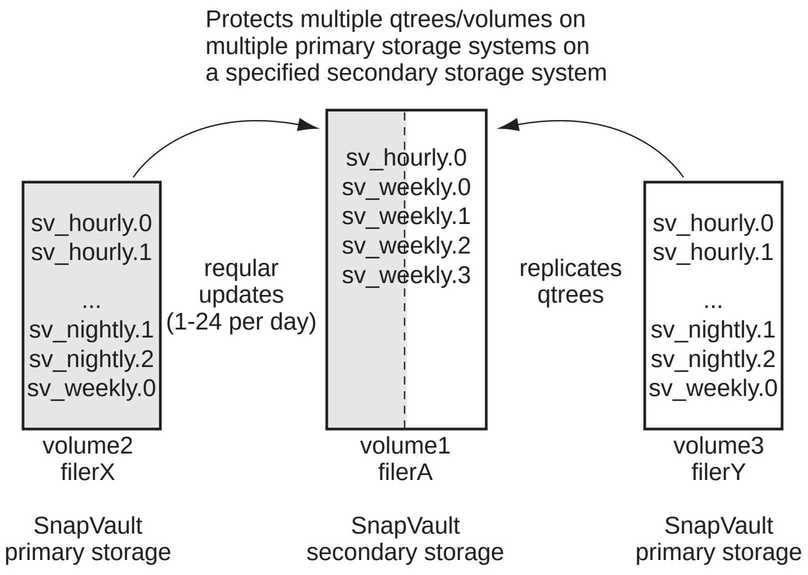
  Then SnapVault creates a volume snapshot of the changes in the secondary qtrees.

  For each transfer and snapshot set, SnapVault saves the number of secondary storage snapshots that you specify and assigns each snapshot a version number (0 for most current, 1 for second most recent, and so on).

Restoration upon request:

- If data needs to be restored to the primary storage system, SnapVault transfers the specified versions of the qtrees back to the primary storage system that requests them.

The following diagram illustrates SnapVault functionality.



How SnapVault backup of Open Systems platforms works

SnapVault backup of Open Systems platform directories works as follows:

Starting the baseline transfers:

- In response to command line input, the SnapVault secondary storage system requests from an Open Systems platform initial baseline image transfers of directories specified for backup. These transfers establish SnapVault relationships between the Open Systems platform directories and the SnapVault secondary qtrees.
- Each Open Systems platform, when prompted by the secondary storage system, transfers initial base images of specified directories to qtree locations on the secondary storage system.

Scheduling incremental transfers:

- The SnapVault secondary storage system, in response to command line input, follows a set of scheduled data transfers (to which, for tracking purposes, you can assign names like “sv_hourly,” “sv_nightly,” and so on). To each secondary qtree on a given volume, from a corresponding primary directory on the Open Systems storage platform, SnapVault transfers the files that have been added or modified since the previous data transfer.

  For each set of scheduled data transfers, SnapVault creates a set of incremental snapshots that capture the changes to the secondary qtrees after each transfer.

  For each set of snapshots, the SnapVault secondary storage system saves the number of secondary storage snapshots you specify and assigns each snapshot in the set a version number (0 for most current, 1 for second most recent, and so on).

Restore upon request:

- If directory or file data needs to be restored to the Open Systems storage platform, SnapVault retrieves the data from one of the retained snapshots and transfers the data back to the Open Systems storage platform that requests it.

Maximum number of simultaneous SnapVault backups on a filer

SnapVault backups have the same limit of simultaneous replications that SnapMirror replications have. See “Maximum number of simultaneous replications on a filer” on page 81.

Planning SnapVault

Planning primary and secondary qtree locations

Some planning of your primary system qtrees or Open Systems directories and their corresponding secondary system qtrees is helpful. Multiple primary system qtrees from multiple volumes and multiple Open Systems directories can all be backed up to corresponding secondary system qtrees in a single volume. The maximum number of secondary system qtrees per volume is 255.

Primary system qtree or directory location example	Corresponding Secondary system qtree location example
filerA:/vol/vol1/qtreeAA	sv_secondary:/vol/sv_vol/qtreeAA
filerA:/vol/vol1/qtreeAB	sv_secondary:/vol/sv_vol/qtreeAB
filerB:/vol/vol1/qtreeBB	sv_secondary:/vol/sv_vol/qtreeBB
winsrvrA:c:\melzdir	sv_secondary:/vol/sv_vol/melzdir
ux_srvrB:/usrs/moz_acct	sv_secondary:/vol/sv_vol/moz_acct

Planning SnapVault backup schedule and snapshot retention

On the filer-based primary systems and the SnapVault secondary storage system, the data to be backed up is captured and preserved in snapshots.

- On your filer-based primary storage systems, plan the intervals at which to create SnapVault snapshots of your primary system qtrees.

  Note

  On Open Systems primary storage platforms, snapshot planning and creation does not apply.

- On your SnapVault secondary system, plan the intervals at which you want to update your secondary system qtrees with data transferred from filer and Open Systems primary storage platforms, and create SnapVault snapshots to retain that information.
- Plan how to limit the combined total of snapshots retained on any one volume of the SnapVault secondary system to 251 or fewer.

Caution

The combined total of snapshots retained on each volume of the SnapVault secondary storage system cannot exceed 251. If the number of snapshots per volume limit is reached and the old snapshots are not deleted, SnapVault will not create new snapshots on that volume.

Before you start SnapVault configuration, use a table like the one below to plan how many snapshots you want per volume, when you want them updated, and how many of each you want to keep. For example:

◆ Hourly (periodically throughout the day)

  Does the data change often enough throughout the day to make it worthwhile to create a snapshot every hour or every two hours or every four hours?

◆ Nightly

  Do you want to create a snapshot every night or just workday nights?

◆ Weekly

  How many weekly snapshots is it useful to keep?

Snapshot intervals	Filer primary storage: when created	Filer primary storage: snapshots retained	Secondary storage: when created	Secondary storage: snapshots retained
weekly	sat @19	4	sat @ 21	8
nightly	mon-fri @19	10	mon-fri @20	60
hourly	@7-18	11	@8-19	120
Total	n/a	21	n/a	188

In the preceding example, the user is assumed to have 12 qtrees on the secondary storage system volume. On the secondary storage system, the user schedules

◆ A weekly update every Saturday at 9:00 p.m. and keeps 8 of them

◆ A daily update every Monday through Friday at 8:00 p.m and keeps 60 of them

◆ An hourly update every hour from 8:00 a.m. to 7:00 p.m. and keeps 120 of them

The result in this example is that 188 snapshots are being kept in the SnapVault secondary storage system volume.

The limit on snapshots per volume is 251, so the 188 snapshots scheduled in this example do not exceed the volume limit.

If you need to retain more than 251 snapshots: If you determine that you need to retain more than 251 SnapVault snapshots on the SnapVault secondary storage system, you can configure additional volumes on the secondary storage system. Each additional volume can support 251 additional snapshots.

Estimating the initial backup time

The backup time required for the initial transfer of data from the primary storage system to the secondary storage system depends on the inode count of the primary data to be backed up. SnapVault can carry out initial backup at an approximate rate of 7 million inodes per hour (110,000 inodes per minute).

Setting up a basic SnapVault backup for filer systems

About configuring SnapVault Backup

Setting up SnapVault backup on the filer primary systems means preparing the primary storage systems and SnapVault secondary storage systems to fulfill their backup tasks.

Note

You must have separate SnapVault licenses for the primary storage and the secondary storage to use SnapVault.

- ◆ On filer primary storage systems, use filer console commands to activate the SnapVault primary license, and specify the SnapVault secondary storage host. See "Configuring a filer primary storage system for SnapVault" on page 208.
- ◆ On the SnapVault secondary storage system, use the filer console commands to license and enable SnapVault, specify the primary storage systems to back up, and start the initial snapshot backup. See "Configuring the SnapVault secondary storage system" on page 209.
- ◆ On the primary filer storage systems, schedule times for local SnapVault snapshots to occur. And on the SnapVault secondary storage system, schedule times for those primary snapshots to be backed up to secondary storage. See "Scheduling SnapVault update backups" on page 211.
- ◆ In event of data loss or corruption on a primary storage qtree, use the `snapvault restore` command to restore the affected qtree to its state at the time of its last SnapVault snapshot. See "In event of data loss or corruption" on page 212.

Note

To set up SnapVault backup on Open Systems storage platforms, see "Setting up SnapVault backup on Open Systems platforms" on page 215.

Configuring a filer primary storage system for SnapVault

On each filer primary storage system to be backed up to the SnapVault secondary storage system, log in to that filer's console and carry out the following steps.

Step	Description
1	Set up the SnapVault primary license on each primary storage system to be backed up. Enter the following command: **`license add`** ***`sv_primary_license`*** For more information, see "Entering license codes" on page 230.
2	Enable the NDMP service on each primary storage system to be backed up. Enter the following command: **`ndmpd on`** For more information see "Setting the ndmpd option" on page 232.
3	Enable SnapVault on each primary storage system to be backed up. Enter the following command: **`options snapvault.enable on`** For more information, see "Setting the enable option" on page 231.
4	Use the `options snapvault.access` command to specify the name of the SnapVault secondary storage system to backup to. Enter the following command: **`options snapvault.access host=`*****`snapvault_secondary`*** The filer must be able to resolve the host name (*snapvault_secondary)* to an IP address in the /etc/hosts file, or else the filer needs to be running DNS or NIS. You can also use the literal IP address instead of the host name. See the na_protocolaccess(8) man page for details. For more information about the `options` command, see the na_options(1) man page. For more information, see "Setting the access option" on page 232.

Configuring the SnapVault secondary storage system

Configure the SnapVault secondary storage system. Carry out the following steps.

Step	Description
1	Set up the SnapVault secondary license. Enter the following command: **`license add`** ***`sv_secondary_license`*** For more information see “Entering license codes” on page 230.
2	Enable the NDMP service on each primary storage system to be backed up. Enter the following command: **`ndmpd on`** For more information see “Setting the ndmpd option” on page 232.
3	Enable SnapVault. Enter the following command: **`options snapvault.enable on`** For more information see “Setting the enable option” on page 231.
4	Use the `options snapvault.access` command to specify the names of the primary storage systems to back up and restore. Enter the following command: **`options snapvault.access host=`*`snapvault_primary1, snapvault_primary2`* `...`** For more information see “Setting the access option” on page 232.

Step	Description
5	For each qtree on the filer primary storage systems to be backed up, use the `snapvault start` command line to execute an initial baseline copy of the qtree from the primary to the secondary storage system. On each command line, specify the primary storage filer, volume, and qtree, and the secondary volume and qtree. Note that the secondary storage host is optional. Use the `-S` prefix to indicate the source qtree path. For example, **`snapvault start -S filer_a:/vol/vol1/tree_a sv_filerb:/vol/sv_vol/tree_a`** **`snapvault start -S filer_a:/vol/vol1/tree_b sv_filerb:/vol/sv_vol/tree_b`** **`snapvault start -S filer_a:/vol/vol1/tree_c sv_filerb:/vol/sv_vol/tree_c`** **Note** Enter each command on a single line. For more information see “Initiating the first backup of data from filer primary storage systems” on page 235.

Scheduling SnapVault update backups

On both the filer primary storage systems and the SnapVault secondary storage system, configure a snapshot schedule. To set up a snapshot schedule, complete the following steps.

Step	Description
1	**On the filer primary storage systems:** On each filer primary storage system that contains qtrees to be backed up to a SnapVault secondary storage system, use the `snapvault snap sched` command to schedule sets of SnapVault snapshots on each volume containing the backed-up qtrees. For each set of snapshots, specify volume name, snapshot basename (for example: “sv_hourly,” or “sv_nightly,” and so on), number of SnapVault snapshots to store locally, and the days and hours to execute the snapshots. For example: **`snapvault snap sched vol1 sv_weekly 1@sat@19`** **`snapvault snap sched vol1 sv_nightly 2@mon-fri@19`** **`snapvault snap sched vol1 sv_hourly 11@mon-fri@7-18`** **Note** When specifying the SnapVault snapshot basename, avoid using “hourly,” “nightly,” or “weekly.” Such naming will conflict with the non-SnapVault `snap sched` snapshots. For more information, see “Scheduling snapshots on the SnapVault filer primary storage system” on page 242.

Step	Description
2	**On the SnapVault secondary storage system:** For each SnapVault volume snapshot set that you scheduled on your filer primary storage systems (see Step 1), use the `snapvault snap sched -x` command line to schedule a set of transfers to and subsequent snapshots of the SnapVault secondary storage system. For example: **`snapvault snap sched -x sv_vol sv_weekly 8@sat@20`** **`snapvault snap sched -x sv_vol sv_nightly 7@mon-fri@20`** **`snapvault snap sched -x sv_vol sv_hourly 11@mon-fri@7-19`** Snapshot basenames on the primary and secondary systems must match, but snapshot times and number of stored snapshots can differ. The `-x` parameter causes SnapVault to copy new or modified data from the primary qtrees to their associated qtrees on the secondary storage system. After all the secondary qtrees on the specified volume have been updated, SnapVault then creates a snapshot of this volume for archiving. **Note** SnapVault snapshots on the secondary system take place 5 minutes after the hour specified in the `snapvault snap sched -x` command. This delay allows time for the transfer of new data from the primary qtrees to the secondary qtrees on the specified volume. For more information see "Scheduling snapshot backups to the SnapVault secondary storage system" on page 244.

In event of data loss or corruption

In the event of data loss or corruption on a qtree in the primary storage system, the administrator can use the `snapvault restore` command to restore that qtree to its state at the time of one of its SnapVault snapshots.

To restore a qtree to its state at the time of one of its SnapVault snapshots, complete the following steps.

Step	Description
1	In the filer console of the primary storage system whose data you want to restore, use the `snapvault restore` command to specify ◆ The path to the qtree on the secondary storage system that you want to restore from ◆ The name of the snapshot that you want to restore from (if it is not the most recent snapshot) ◆ The path on the primary storage system to the qtree you want to restore to **Example:** This example restores qtree tree_b on primary storage system melzhost from qtree tree_b on the secondary storage system, using the most recent snapshot. **`snapvault restore -S secondary_host:/vol/sv_vol/tree_b melzhost:/vol/vol1/tree_b`** **Note** Enter the entire command as a single line. For more information, see “Restoring data to the primary storage system” on page 264.

Step	Description
2	After the restore operation is completed on the primary volume, go to the SnapVault secondary storage system and use the `snapvault start -r` command to restart the SnapVault backup relationship between the primary and secondary qtrees. Specify ◆ The path on the primary storage system to the qtree you want to back up ◆ The path on the secondary storage system to the qtree that you want to back up to For example: **`snapvault start -r -S melzhost/vol/vol1/tree_b secondary_host/vol/vol1/tree_b`** **Note** Enter the entire command as a single line. For more information see "Restoring data to the primary storage system" on page 264.

Setting up SnapVault backup on Open Systems platforms

About configuring SnapVault backup of Open Systems storage platforms

Setting up SnapVault for Open Systems platform backup (backup of Windows NT, Windows 2000, Solaris, AIX, Linux, HP-UX systems, and so on) involves installing special agents on those systems and preparing the SnapVault secondary storage system to fulfill its backup tasks.

See the *Open Systems SnapVault Release Notes* for information about supported systems and installing the SnapVault agent on your system.

Backing up and restoring the agent database

The Open Systems SnapVault agent creates a history database of files and directories that the agent is backing up. You can back up this database and restore it if something happens to the original.

Note

You must have root or admin priveledges on the primary storage system to back up or restore the agent database.

To back up the Open Systems SnapVault agent database, complete the following steps.

Step	Action
1	Using the command-line interface of the Open Systems SnapVault primary storage system, navigate to the snapvault/bin directory. ◆ On Windows systems, the default path is c:\Program Files\snapvault\bin ◆ On UNIX systems, the default path is /usr/snapvault/bin
2	Back up the agent database using the following command: **`./svdb -s`** **Result:** The primary system prints a file name to the console.
3	Save the file in a safe location.

To restore the Open Systems SnapVault agent database, complete the following steps.

Step	Action
1	Using the command-line interface of the Open Systems SnapVault primary storage system, navigate to the snapvault/bin directory. ◆ On Windows systems, the default path is c:\Program Files\ snapvault\bin ◆ On UNIX systems, the default path is /usr/snapvault/bin
2	Restore the agent database by using the following command: **`./svdb -r filename`** *filename* is the file name printed to the console when you backed up the agent database.
3	Stop and restart Open Systems SnapVault services using the svconfigurator utility. See the *Open Systems SnapVault Release Notes* for details.

After installing the agent

On the SnapVault secondary storage system, use the filer console commands to license and enable SnapVault, specify the Open Systems platforms to back up, start the initial baseline copy, and schedule times to download updates from the Open Systems platforms. See "Configuring the SnapVault secondary storage system" on page 217 and "Managing SnapVault backup of Open Systems platforms" on page 220 for information about these operations.

Configuring the SnapVault secondary storage system

SnapVault secondary storage system requirement

The SnapVault secondary storage system must be running Data ONTAP 6.4 or later to support backup of systems installed with Open Systems SnapVault.

Configuring the SnapVault secondary storage system

To configure the SnapVault secondary storage system to support Open Systems platform SnapVault backup, complete the following steps.

Step	Description
1	License the SnapVault secondary storage system. In the filer console of the SnapVault secondary system, enter the following command: **`license add`** ***`sv_secondary_license`*** For more information see “Entering license codes” on page 230.
2	License the SnapVault primary storage system. In the filer console of the SnapVault secondary system, enter the following command: **`license add`** ***`ossv_primary_license`***
3	Enable SnapVault. In the secondary storage system filer console, enter the following command: **`options snapvault.enable on`** For more information see “Setting the enable option” on page 231.
4	Specify the names of the primary storage systems to back up to. Enter the following command: **`options snapvault.access host=`*****`snapvault_primary1, snapvault_primary2`*** **`...`** **Example:** **`options snapvault.access host=melzhost,samzhost,budzhost`** For more information see “Setting the access option” on page 232.

Step	Description

5	For each Open Systems platform directory to be backed up to the SnapVault secondary storage system, execute an initial baseline copy from the primary to secondary storage system.
	◆ Specify the fully qualified path to the Open Systems host directory that you want to back up. Use the `-S` prefix to indicate the source path.
	◆ Even though the Open Systems platform directory to be backed up has no qtree, you *still* need to specify a host and path to the qtree where you will back up this data on the SnapVault secondary storage system.
	Enter the following command:
	`snapvault start -S` ***`prim_host`*** **`:`** ***`dirpath`***
	`sec_host` **`:/vol/`** ***`sec_vol`*** **`/`** ***`sec_tree`***
	Example 1 (Windows):
	`snapvault start -S melzhost:c:\melzdir`
	`sv_secondary:/vol/sv_vol/tree_melz`
	`snapvault start -S samzhost:c:\samzdir`
	`sv_secondary:/vol/sv_vol/tree_samz`
	`snapvault start -S budzhost:c:\budzdir`
	`sv_secondary:/vol/sv_vol/tree_budz`
	Example 2 (UNIX):
	`snapvault start -S melzhost:/usr/melzdir`
	`sv_secondary:/vol/sv_vol/tree_melz`
	`snapvault start -S samzhost:/usr/samzdir`
	`sv_secondary:/vol/sv_vol/tree_samz`
	`snapvault start -S budzhost:/usr/budzdir`
	`sv_secondary:/vol/sv_vol/tree_budz`
	Note
	Enter the entire command as a single line.

Step	Description
6	Use the `snapvault snap sched` command to schedule update copying of new or modified data on all Open Systems platform directories that are backed up to qtrees in SnapVault secondary storage. Specify the name of the secondary storage volume containing the secondary qtrees, a snapshot basename (for example: “sv_hourly” or “sv_nightly”), the number of SnapVault snapshots to store on the secondary storage system, and the days and hours to execute. For example: **`snapvault snap sched -x vol1 sv_weekly 1@sat@19`** **`snapvault snap sched -x vol1 sv_nightly 2@mon-fri@19`** **`snapvault snap sched -x vol1 sv_hourly 11@mon-fri@7-18`** The `-x` parameter causes SnapVault to copy new or modified files from the Open Systems platform directories to their associated qtrees on the secondary storage system. After all the secondary qtrees on the specified volume have been updated, SnapVault then creates a snapshot of this volume for archiving. **Note** If files on an Open Systems platform directory are open when a scheduled SnapVault copy takes place, those files are skipped for the current transfer. Changes to those files will be backed up during the next scheduled transfer.

Managing SnapVault backup of Open Systems platforms

Central management of Open Systems SnapVault agents

Open Systems SnapVault agents for all operating systems can be centrally managed using one of the following management tools:

- DataFabric® Manager 2.2 or later from Network Appliance.

  For more information, see the DataFabric Manager description page of the NetApp On the Web (NOW) site at http://now.netapp.com/NOW/cgi-bin/software.
- NetVault software with a SnapVault Manager plugin module from Bakbone Software, Inc.

  For more information, see the Bakbone web site at http://www/bakbone.com/.

Where to find status and problem reports

You can find primary storage system operational status and problem reports in a log file called snapvault. The snapvault log file is located in the following directory:

```
install_dir/snapvault/etc
```

install_dir is the directory on the primary storage system in which you installed the Open Systems SnapVault agent. On Windows systems, the default location for *install_dir* is the c:\Program Files directory. On UNIX systems, the default location for *install_dir* is the /usr directory.

You can find secondary storage system reports in the /etc/log/snapmirror file in the root volume.

What SnapVault operations are possible on Open Systems platforms

SnapVault backup and restore of an Open Systems platform is carried out using many of the same commands that are used to manage backup and restore of filer primary storage systems.

In many cases, the required input for a command is different, reflecting the fact that Open Systems platforms are not organized in qtrees. Instead of primary qtrees, the user specifies directories or volumes to be backed up from the Open Systems platform.

Where directories and volumes are backed up

Each directory or volume you want backed up from a primary storage system is backed up to its own qtree on the SnapVault secondary storage system.

Secondary side SnapVault commands applied to Open Systems platforms

The following SnapVault commands can be entered at the command line of the SnapVault secondary storage system.

Command	Input for Open Systems platforms
`license add`	Specifies the license information for the SnapVault secondary storage and primary storage systems. **Syntax:** At the secondary storage console, enter **`license add`** ***`license_no`***
`options snapvault.enable`	Enables SnapVault on the secondary storage system. **Syntax:** At the secondary storage console, enter **`options snapvault.enable {on|off}`**
`options snapvault.access`	Specifies the filer or Open Systems platforms that can be backed up by SnapVault. **Syntax:** At the secondary storage console, enter **`options snapvault.access host=`*`prim_host`*`[,`*`prim_host`*`...]`**

`snapvault start`	Starts the initial backup of data from the Open Systems platform volumes and directories to SnapVault secondary storage qtrees.
	Syntax: At the secondary storage console, enter
	`snapvault start -S` ***`prim_host`*****`:`*****`dirpath`*** ***`sec_host`*****`:vol/`*****`sec_vol`*****`/`*****`sec_tree`***
	Example 1 (Windows):
	`snapvault start -S winservrB:c:\documents sv_hostA:vol/sv_vol/documents`
	Example 2 (UNIX):
	`snapvault start -S uxservrB:/usr/documents sv_hostA:vol/sv_vol/documents`

Command	Input for Open Systems platforms
`snapvault snap sched -x`	Schedules sets of incremental backups to all secondary system qtrees on a specified volume from their primary directory sources on Open Systems platforms. After each incremental backup, SnapVault then creates a snapshot of updated secondary system qtrees for long-term storage. **Syntax:** At the secondary storage console, enter **`snapvault snap sched -x`** ***`volume_name snap_name count@weekday-weekday@hour`*** **Example:** **`snapvault snap sched -x sv_vol sv_daily 5@Monday-Friday@23`**
`snapvault update`	Carries out unscheduled incremental SnapVault backups from Open Systems platform drives or directories to the secondary storage qtree. **Syntax:** At the secondary storage console, enter **`snapvault update [-k`** ***`kbs`*** **`] [-s`** ***`snapname`*** **`]`** **`[`*****`sec_host`*****`:]`*****`sec_qtree_path`***
`snapvault modify`	Modifies a SnapVault relationship. **Syntax:** At the secondary storage console, enter **`snapvault modify [-k`** ***`kbs`*** **`] [-t`** ***`n`*** **`]`** **`[-S [`*****`prim_host`*****`:][`*****`dir_path`*****`]`** **`[`*****`sec_host`*****`:]/vol/`*****`sec_vol`*****`/`*****`sec_qtree_path`***
`snapvault abort`	Stops an ongoing transfer of data (scheduled by the `snapvault snap sched -x` command or invoked by the `snapvault start` or `snapvault update` commands) from an Open Systems platform source to a specified secondary system qtree. **Syntax:** At the secondary console, enter **`snapvault abort [`*****`sec_filer`*****`:]`** ***`sec_qtree_path`***

Command	**Input for Open Systems platforms**
`snapvault stop`	Ends a SnapVault relationship between a secondary system qtree and its Open Systems platform source. **Syntax:** At the secondary console, enter **`snapvault stop [`*`sec_filer`*`:]`** ***`sec_qtree_path`***
`snapvault snap unsched`	Unschedules a set of incremental backups to the secondary system qtrees on a specified volume. This set of incremental backups was originally scheduled through the `snapvault snap sched` command. This command does not end the SnapVault relationships between the secondary system qtrees and their Open Systems platform source drives or directories. **Syntax:** At the secondary storage console, enter **`snapvault snap unsched [`*`volume_name`*`] [`*`snap_name`*`]`** **Example:** **`snapvault snap unsched sv_vol sv_daily`**
`snapvault restore`	Restores a directory, subdirectory, or file from the secondary system to the Open System platform. **Syntax:** At the Open System platform console, enter **`snapvault restore -S [`*`sec_filer`*`:]`*`sec_qtree_path prim_qtree_path`***

Restoring a directory or file

Methods for restoring data

In the event of data loss or corruption on the Open Systems platform, the administrator can use one of three different methods for restoring a directory or file:

- Copy files from the secondary storage system to the primary storage system.
- Use the `snapvault restore` command.
- Use the DataFabric Manager restore wizard.

Copying files

You can copy files from the secondary storage system to the NFS or CIFS primary storage system if you want to restore something other than an entire qtree, that is, a single file, a small group of files, or a few directories.

You might want to share the SnapVault destination on the secondary storage system with all the primary storage systems all the time. In this way, end users can perform self-service restores without bothering a backup administrator.

Note

Some Windows NT and UNIX attributes are not preserved using this method, notably, Windows NT sparse files and UNIX access control lists (ACLs).

Copying files to NFS primary storage systems: To restore data by copying files back to an NFS primary storage system, complete the following steps.

Step	Action
1	Mount the backed-up qtree from the SnapVault secondary storage system to the NFS primary storage system.
2	Use the UNIX `cp` command, or an equivalent command, to copy the desired files from the backup to the directory in which you want them.

Copying files to CIFS primary storage systems: To restore data by copying files back to a CIFS primary storage system, complete the following steps.

Step	Action
1	Create a share from the backed-up qtree on the SnapVault secondary storage system to the CIFS primary storage system.
2	Drag and drop the desired files from the backup to the directory in which you want them..

Using the snapvault restore command

You can use the `snapvault restore` command to restore a directory or file on the Open Systems platform to its state at the time of one of its SnapVault snapshots.

To use the `snapvault restore` command, complete the following step.

Step	Description
1	In the command line interface of the Open Systems platform whose data you want to restore, use the `snapvault restore` command to specify ◆ the secondary storage system host and the path to the secondary qtree, directory, and file that you want to restore from ◆ the name of the snapshot that you want to restore from (for example, sv_weekly.0, sv_weekly.1, or sv_weekly.2) ◆ the path on the primary storage system to the directory, or file that you want to restore to **Example 1 (Windows):** **`snapvault restore -s sv_daily.0 -S myvault:/vol/sv_vol/melzdir\evidence.doc a:\melzdir\evidence.doc`** **Example 2 (UNIX system):** **`snapvault restore -s sv_daily.0 -S myvault:/vol/sv_vol/melzdir/evidence.doc /usr/melzdir/evidence.doc`** **Note** Enter the entire command as a single line.

Using the DataFabric Manager restore wizard

The DataFabric Manager restore wizard leads you through the entire restore process. See the *DataFabric Manager Administration Guide* for details.

Restoring an entire primary storage system

Restoring a primary storage system

You can restore an entire NFS or CIFS primary storage system from a SnapVault secondary storage system, but the restore cannot be to a primary storage system that has a blank hard disk. There must be an operating system on the disk.

To restore an entire primary storage system, complete the following steps.

Step	Action
1	Reinstall the operating system on the primary storage system.
2	Reformat the file system as the original file system was formatted.
3	Install the SnapVault agent. See the *Open Systems SnapVault Release Notes* for details.
4	Restore the backed-up directories using the `snapvault restore` command. See “Using the snapvault restore command” on page 225 for details.

Restoring files to a primary storage system from tape

Restoring from tape

To perform a SnapVault restore to an NFS or CIFS primary storage system from tape, complete the following steps.

Step	Action
1	Mount the tape that has the files that need to be restored.
2	Use the `restore` command to restore from the tape to the SnapVault secondary storage system. See the *Tape Backup and Recovery Guide* for details.
3	Copy the files from the SnapVault secondary storage system to the NFS or CIFS primary storage system. See "Restoring a directory or file" on page 224 for details.

Backing up a database

Backing up a database using Open Systems SnapVault

To use Open Systems SnapVault to back up a database, complete the following steps.

Note

The following procedure assumes that all of the files that make up the database reside in a single directory on the primary storage system.

Step	Action
1	On the primary storage system, put the database into hot backup mode.
2	On the SnapVault secondary storage system, back up the data using the `snapvault update` command.
3	On the primary storage system, after the backup completes, take the database out of backup mode.

Enabling SnapVault

What you need to do to enable SnapVault

You need to complete the following tasks to activate SnapVault:

- Enter the license code on both the secondary storage system and the primary storage system.
- Turn on the `options snapvault.enable` option for both the secondary storage system and the primary storage system. Setting this option enables SnapVault data transfers and snapshot creation.
- Turn on the NDMP service on both the secondary storage system and the primary storage system.
- Set the `options snapvault.access` option on both the secondary storage system and the primary storage system to allow primary storage system and secondary storage system access.
  - Setting this option on the SnapVault secondary storage system determines which SnapVault primary storage systems can access the secondary storage system.
  - Setting this option on the SnapVault primary storage system determines which secondary storage system can access data from that primary storage system.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For SnapVault backup of Open Systems drives and directories, see "Setting up SnapVault backup on Open Systems platforms" on page 215 and "Managing SnapVault backup of Open Systems platforms" on page 220.

Entering license codes

Enter the SnapVault secondary storage system license code on the filer that is to be the secondary storage system, and enter the separate SnapVault primary storage system license code on the filer that is to be the primary storage system.

Note

You cannot license the SnapVault secondary storage system and the SnapVault primary storage system on the same filer.

To enter the SnapVault secondary storage system license code, complete the following step.

Step	Action
1	On the secondary storage system, enter the following command: **`license add`** ***`xxxxxxx`*** *xxxxxxx* is the license code you purchased. This setting persists across reboots.

To enter the SnapVault primary storage system license code, complete the following step.

Step	Action
1	On the primary storage system, enter the following command: **`license add`** ***`xxxxxxx`*** *xxxxxxx* is the license code you purchased. This setting persists across reboots.

For more information about entering license codes, see the information about licensing in the *Storage Management Guide.*

Setting the enable option

To set the enable option, complete the following step.

Step	Action
1	On both the primary storage system and the secondary storage system, enter the following command: **`options snapvault.enable on`** This option persists across reboots.

Setting the ndmpd option

To set the ndmpd option, complete the following step.

Step	Action
1	On both the primary storage system and the secondary storage system, enter the following command: **`ndmpd on`** This option persists across reboots.

Setting the access option

The `snapvault.access` option controls which filers can request data transfers. This option persists across reboots.

On the filer primary storage system: To set the primary storage systems to grant access only to the secondary storage system, complete the following step.

Step	Action
1	On the primary storage system, enter the following command: **`options snapvault.access host=`*`snapvault_secondary`*** The filer must be able to resolve the host name entered as *snapvault_secondary* to an IP address in the /etc/hosts file, or else the filer needs to be running DNS or NIS. You can also use the literal IP address instead of the host name. See the na_protocolaccess(8) man page for details. For more information about the `options` command, see the na_options(1) man page.

Example:

```
filerB> options snapvault.access host=filerA
```

On the secondary storage system: To allow the primary storage systems to restore data from the secondary storage system, complete the following step.

Step	Action
1	On the secondary storage system, enter the following command: **`options snapvault.access host=`*`snapvault_primary1, snapvault_primary2`*` ...`** The filer must be able to resolve the host name entered as *snapvault_primary* to an IP address in the /etc/hosts file, or else the filer needs to be running DNS or NIS. You can also use the literal IP address instead of the host name. The syntax for specifying which filers are allowed access to the secondary storage system is described in the na_protocolaccess(8) man page. For more information about the `options` command, see the na_options(1) man page.

Example:

```
filerA> options snapvault.access host=filerB,filerC
```

Note

To grant access to any requester, enter **`options snapvault.access all`**. For security reasons, this configuration is not recommended.

Starting a SnapVault backup relationship

Why you need to specify qtrees for filer primary storage systems

After you have enabled SnapVault on both your primary and secondary storage systems and have given primary and secondary systems access to each other, you must specify the qtrees or volumes whose data you want transferred from the primary storage system to the SnapVault secondary storage device, and you must perform a complete (baseline) transfer of data from primary storage system to secondary storage.

The `snapvault start` command configures the SnapVault relationship between primary qtrees or volumes and secondary qtrees by

- ◆ Specifying the primary system qtrees or volumes to be backed up on the secondary system qtrees
- ◆ Specifying parameters for the updates from the primary qtrees to the secondary system qtrees. The parameters include transfer speed and try count (the number of times SnapVault will attempt to start a transfer before stopping the operation).
- ◆ Initiating the baseline transfer of the primary system qtree data to the secondary system qtree

Guidelines when creating a SnapVault relationship

The following are guidelines for volumes and qtrees in a SnapVault backup relationship:

- ◆ Establish a SnapVault relationship between volumes that have the same `vol lang` settings.
- ◆ Once you establish a SnapVault relationship, do not change the language assigned to the destination volume.
- ◆ Avoid whitespace (space or tab characters) in names of source and destination qtrees.
- ◆ Do not rename volumes or qtrees after establishing a SnapVault relationship.

For detailed information

The following sections discuss the backup relationships you can start:

- ◆ “Backing up qtree data” on page 235
- ◆ “Backing up non-qtree data” on page 237
- ◆ “Backing up volume data” on page 239

Backing up qtree data

Initiating the first backup of data from filer primary storage systems

To specify the qtrees that you want to be backed up and to run a first-time (complete) SnapVault transfer, complete the following step.

Step	Action
1	At the filer console for the secondary storage system, enter the following command: **`snapvault start -S`** ***`prim_filer`*****`:`*****`prim_qtree_path`*** ***`sec_host`*****`:`*****`sec_qtree_path`*** `-S` *prim_filer*:*prim_qtree_path* specifies the qtree on the primary storage system to be backed up. This option must be set the first time the command is run for each primary storage system qtree you want to copy. *sec_host* is the name of the destination (secondary storage system) filer to which the data from the primary system qtree is transferred. If no secondary storage system filer is specified, the local host's name is used. *sec_qtree_path* is the path to and the name of the qtree on the secondary storage system. **Note** The qtree specified for *sec_qtree_name* must not exist on the secondary system before you run the `snapvault start` command. For information about `snapvault start` command options, see the na_snapvault(1) man page.

Example:

```
filerB> snapvault start -S filerA:/vol/vol2/qtree3 /vol/vol1/qtree3
```

Result: SnapVault creates a snapshot of qtree3 in vol2 of the primary storage system (filerA), copies the data in it to the secondary storage system (filerB) at vol/vol1/qtree3, and configures the qtree for future updates.

Note

The time required for this baseline transfer is limited by the total amount of primary data to be backed up and by the inode count. SnapVault can carry out initial backup at an approximate rate of 7 million inodes per hour (110,000 inodes per minute). In the first phase of a large transfer, SnapVault creates inodes, and it may appear that no activity is taking place.

Backing up non-qtree data

What non-qtree data is

Non-qtree data is any data on a filer that is not contained in its qtrees. Non-qtree data can include

- Configuration and logging directories on the filer (for example, /etc or /logs) that are not normally visible to filer clients
- Directories and files on a filer volume that has no qtree configured

Initiating backup of filer primary system non-qtree data

To replicate and protect non-qtree data in a filer primary storage system, complete the following step.

Step	Action
1	From the secondary storage system, enter the following command: **`snapvault start -S [`*`prim_filer`*`:]/vol/`*`volume_name/-`* `/vol/`*`volume_name/qtree_name`*** `-S` *prim_filer*:/`vol`/*volume_name*/- specifies the volume on the primary storage system whose non-qtree data you want to backup. The dash (-) indicates all non-qtree data in the specified volume. /`vol`/*volume_name*/*qtree_name* specifies the qtree in the secondary storage system where you want to store this data. **Note** The qtree that is specified for /`vol`/*volume_name*/*qtree_name* must not exist on the secondary system before you run the `snapvault start` command.

Note
The non-qtree part of the primary storage system volume can be replicated only to the SnapVault secondary storage system. The data can be restored to a qtree on the primary storage system, but cannot be restored as non-qtree data.

Example: `filerB> snapvault start -S filerA:/vol/vol1/-`
`/vol/vol4/non_qtree_data_in_vol7`

Result: SnapVault transfers the non-qtree data on primary filerA to the qtree called non_qtree_data_in_vol7 in vol4 on filerB (the secondary storage system). It also configures the qtree for future updates.

Backing up volume data

Data structure of the backed up data

When you back up a source volume using SnapVault, the volume is backed up to a qtree on the secondary storage system; therefore, any qtrees in the source volume become directories in the destination qtree.

Reasons for backing up a volume using SnapVault

You might want to back up a volume to a qtree using SnapVault for one or more of the following reasons:

- You want to back up a volume that contains many qtrees.
- You want the snapshot management that SnapVault provides.
- You want to consolidate the data from several source volumes on a single destination volume.

Limitations to backing up a volume to a qtree

There are limitations to backing up a volume to a qtree. You should consider the following limitations before performing a volume-to-qtree backup:

- You lose the qtree as a unit of quota management.

  Quota information from the qtrees in the source volume is not saved when they are replicated as directories in the destination qtree.
- You lose qtree security information.

  If the qtrees in the source volume had different qtree security styles, those security styles are lost in the replication to the destination qtree and are replaced by the security style of the volume.
- It is not a simple process to restore data.

  SnapVault cannot restore the data back to a volume. When restoring data, what was a source volume is restored as a qtree.

  See “Restoring the qtree to the original volume structure” on page 240 for a process to restore data.
- Volume-to-qtree backup is not supported for qtrees with LUNs.

Initiating backup of filer primary system volume data

To replicate and protect volume data in a filer primary storage system, complete the following step.

Step	Action
1	From the secondary storage system, enter the following command: **`snapvault start -S [`*`prim_filer`*`:]/vol/`*`volume_name`* `/vol/`*`volume_name`*`/`*`qtree_name`*** `-S` *prim_filer*:/`vol`/*volume_name* specifies the volume on the primary storage system whose non-qtree data you want to backup. /`vol`/*volume_name*/*qtree_name* specifies the qtree in the secondary storage system where you want to store this data. **Note** The qtree that is specified for /`vol`/*volume_name*/*qtree_name* must not exist on the secondary system before you run the `snapvault start` command.

Example: `filerB> snapvault start -S filerA:/vol/vol1 /vol/vol4/vol1_copy`

Result: SnapVault transfers the data in vol1 on primary filerA to the qtree called vol1_copy in vol4 on filerB (the secondary storage system). It also configures the qtree for future updates.

Restoring the qtree to the original volume structure

If you were to use the `snapvault restore` command, the source volume you backed up to a qtree is restored as a qtree on the primary storage system.

To restore the backed-up qtree to the original volume structure with multiple qtrees on the primary storage system, complete the following steps.

Step	Action
1	Re-create all of the qtrees in the volume on the primary storage system using the `qtree create` command. **Example:** The following command creates a project_x qtree in the projs volume: **`pri_filer> qtree create /vol/projs/project_x`**

Step	Action

2	Restore the data for each qtree using the `ndmpcopy` command.
	
	Example: The following command restores data from the backed-up project_x directory on the secondary storage system to the re-created project_x qtree on the primary storage system.
	
	`pri_filer> ndmpcopy -sa` ***`username`*****`:`*****`password`***
	`filer2:/vol/vol1/projs/project_x /vol/projs/project_x`

3	Stop qtree updates and remove the qtree on the secondary storage system using the `snapvault stop` command. **Example:** The following command removes the projs qtree from the secondary storage system: **`sec_filer> snapvault stop /vol/vol1/projs`**
4	Reinitialize a baseline copy of each qtree to the secondary storage system using the `snapvault start` command. **Example:** The following command reinitializes the SnapVault backup: **`snapvault start -S rdfiler:/vol/projs /vol/vol1/projs`**

Scheduling SnapVault snapshot updates

About SnapVault snapshot update schedules

After you have completed the initial baseline backup of qtrees on the filer primary storage systems to qtrees on the SnapVault secondary storage system (see "Starting a SnapVault backup relationship" on page 234), you must use the `snapvault snap sched` command to schedule

- ◆ Regular SnapVault snapshot times of volumes on the primary storage system to capture new and changed data in the qtrees that have a SnapVault relationship configured through the `snapvault start` command
- ◆ Regular transport of new or modified data in the primary qtrees to their associated secondary qtrees on the SnapVault secondary storage system
- ◆ Regular snapshots of the volume containing the updated secondary qtrees for archiving by SnapVault

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For descriptions and procedures pertaining to SnapVault backup of Open Systems drives and directories, see "Setting up SnapVault backup on Open Systems platforms" on page 215 and "Managing SnapVault backup of Open Systems platforms" on page 220.

Scheduling snapshots on the SnapVault filer primary storage system

To schedule a set of snapshots on the SnapVault primary storage system, you use the `snapvault snap sched` command to specify the volume to create snapshots for, the snapshot basename, how many versions of the snapshots to be retained, and the days and hours to execute this set of snapshots.

To set a schedule for the primary storage system, complete the following step.

Step	Action
1	From the primary storage system, enter the following command: **`snapvault snap sched`** ***`volume_name snap_name schedule_spec`*** *volume_name* is the name of the volume on the primary storage system on which to create this set of snapshots. *snap_name* is the basename of a snapshot set, for example, sv_nightly. The name of this snapshot must be the same on the primary storage system and on the secondary storage system. The *snap_name* must not be “hourly,” “nightly,” or “weekly” to avoid conflict with `snap sched` snapshots. *schedule_spec* is made up of *count*[*@day_list*][*@hour_list*]. *count* is the number of snapshots to retain for this snapshot set. A zero (0) in this field means no new instance of this snapshot will be created. **Caution** The combined total of snapshots retained for this and other snapshot sets cannot exceed 251 snapshots per volume. If it does, SnapVault will not create new snapshots. *@day_list* is a comma-separated list that specifies the days on which a new snapshot for this set is created. Valid entries are `mon tue wed thu fri sat sun`. They are not case-sensitive. You can specify a range using a dash (-), for example, `mon-fri`. The dash (-) by itself means no snapshot will be created automatically. You can create the snapshot manually. The default value is `mon-sun`. *@hour_list* specifies the hours at which a new snapshot is created for this set. Valid entries are whole numbers from 0 to 23. You can specify a range using a dash (-), or use a comma-separated list, for example, `7, 8-17, 19, 21, 23`. The default is midnight (`0`).

Example: The following three `snapvault snap sched` command lines schedule three sets of SnapVault snapshots on volume vol1 of the primary storage system.

```
filerB> snapvault snap sched vol1 sv_weekly 1@sat@19
filerB> snapvault snap sched vol1 sv_nightly 2@mon-fri@19
filerB> snapvault snap sched vol1 sv_hourly 11@mon-fri@7-18
```

Result: SnapVault primary storage system creates snapshots on the specified volume, as follows:

- SnapVault creates *sv_weekly.0* every Saturday at 7:00 p.m., and keeps one copy.
- SnapVault creates *sv_nightly.0* every Monday through Friday at 7:00 p.m., and keeps two copies.
- SnapVault creates *sv_hourly.0* every Monday through Friday, every hour from 7:00 a.m. to 6:00 p.m., and keeps eleven copies.

Scheduling snapshot backups to the SnapVault secondary storage system

To schedule the backup of SnapVault snapshots from the filer primary storage systems to the secondary storage system, use the `snapvault snap sched -x` command.

Complete the following step.

Step	Action
1	From the secondary storage system, enter the following command on a single line: **`snapvault snap sched -x`** ***`sec_vol snap_name schedule_spec`***

In the command line just described:

- The `-x` portion of the command is required on the secondary storage system. This parameter specifies that the SnapVault secondary qtrees on the specified volume are updated from their associated primary system qtrees just before the new snapshot of the specified volume is created.
- *sec_vol* is the name of the volume on the secondary storage system for which this snapshot is scheduled.
- *snap_name* is the basename of the set of snapshots to create, for example, sv_nightly. The basename of this snapshot set must match the basename of the corresponding snapshot set configured on the primary storage system volume. Each new snapshot created for this set is numbered 0, the number of

each previous snapshot in this set is increased by 1, and the oldest snapshot in this set is deleted. The *snap_name* must not be "hourly," "nightly," or "weekly" to avoid conflict with regular Data ONTAP `snap sched` snapshots.

◆ *schedule_spec* is made up of *count*[*@day_list*][*@hour_list*].

  ❖ *count* is the number of snapshots to retain for this set. A zero (0) in this field means no new secondary storage system snapshot will be created for this set, although the qtrees on the secondary storage system will be updated by the transfers from the primary storage systems.

Caution

The combined total of snapshots retained for this and other snapshot sets cannot exceed 251 snapshots per volume. If it does, SnapVault will not create new snapshots.

  ❖ *@day_list* is a comma-separated list that specifies the days on which a new snapshot is created for this set. Valid entries are `mon tue wed thu fri sat sun`. They are not case-sensitive. You can specify a range using a dash (-), for example, `mon-sun`. The dash (-) by itself means no snapshot will be created automatically. You can create the snapshot manually.

  ❖ *@hour_list* specifies the hours at which a new snapshot is created for this set. Valid entries are whole numbers from 0 to 23. You can specify a range using a dash (-), or use a comma-separated list, for example, `6, 8-17, 19, 21, 23`. The default is midnight (`0`).

Note

Turn off the SnapVault snapshot schedule on the primary storage system or the secondary storage system at any time with the `snapvault snap unsched` command. (See "Unscheduling SnapVault snapshots" on page 272.)

Example: The following three `snapvault snap sched` command lines schedule three sets of SnapVault transfer updates and snapshots on volume vol1 of the secondary storage system.

```
filerA> snapvault snap sched -x vol1 sv_weekly 5@sat@21
filerA> snapvault snap sched -x vol1 sv_nightly 5@mon-fri@20
filerA> snapvault snap sched -x vol1 sv_hourly 4@mon-fri@8-19
```

Result: SnapVault transfers qtree data from the primary storage system's snapshot as follows:

◆ SnapVault transfers *sv_weekly.0* to the secondary storage system every Saturday at 9:00 p.m., makes a new snapshot with the same name containing all the transferred data, and keeps five copies.

- SnapVault transfers *sv_nightly.0* to the secondary storage system every Monday through Friday at 8:00 p.m., makes a new snapshot with the same name containing all the transferred data, and keeps five copies.
- SnapVault transfers *sv_hourly.0* to the secondary storage system every hour from 8:00 a.m. to 7:00 p.m., Monday through Friday, makes a new snapshot with the same name containing all the transferred data, and keeps four copies.

Note

Snapshots scheduled on the secondary storage system with the `snapvault snap sched -x` command are created five minutes after the hour you specify to give SnapVault on the filer primary storage systems enough time to create snapshots before the secondary storage system updates from them.

Scheduling snapshots on the secondary system for archiving

You might want to schedule some snapshots on the secondary storage system that do not require a transfer from the primary storage system. For example, you might want to maintain hourly and nightly snapshots on the primary storage system and you might want to keep only weekly snapshots on the secondary storage system. To schedule snapshots on the secondary storage system without having to create a corresponding snapshot on the primary storage system, complete the following step.

Step	Action
1	Enter the following command: **`snapvault snap sched`** ***`sec_vol snap_name schedule_spec`*** **Note** The `snapvault snap sched` command is used because it waits for any active SnapVault transfers to finish before creating the snapshot.

Example: The following command schedules a weekly snapshot at 11 p.m. on Saturdays and keeps the last five snapshots:

```
snapvault snap sched vol2 sv_weekly 5@sat@22
```

How to display the currently configured snapshot schedule

Enter the `snapvault snap sched` command without a schedule specification to show the current schedule values. If no *snap_name* is given, the command shows the basenames of all snapshot sets scheduled for the specified volume. If no *volume_name* is given, the command shows the basenames of all snapshots sets for all SnapVault volumes on the filer.

Example: `filerA> snapvault snap sched`

```
xfer   vol1 sv_weekly 5@sat@21
xfer   vol1 sv_nightly 5@mon-fri@20
xfer   vol1 sv_hourly 4@mon-fri@8-19
xfer   vol2 sv_nightly 5@mon-fri@20
xfer   vol2 sv_hourly 4@mon-fri@8-19
```

Checking SnapVault transfers

Why to check data replication status

You check transfer status, using the `snapvault status` command, to make sure SnapVault transfers are taking place as they should.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For descriptions and procedures pertaining to SnapVault backup of Open Systems drives and directories, see "Setting up SnapVault backup on Open Systems platforms" on page 215 and "Managing SnapVault backup of Open Systems platforms" on page 220.

Checking the status

To check the status of a data transfer and see how recently a qtree has been updated, complete the following step.

Step	Action
1	Enter the following command: **`snapvault status [-l] [[[`*`filer_name`*`:]`*`qtree_path`*`] ...] [-s] [-c]`** Options can be one or more of the following: ◆ `-l` displays the long format of the output, which contains more detailed information. ◆ `-s` displays the SnapVault snapshot basename, status, and schedule for each volume. ◆ `-c` displays the configuration parameters of all SnapVault qtrees on the filer. This option can be run only from the secondary storage system. *filer_name* is the name of the filer for which you want to see the status of SnapVault operations. *qtree_path* is the path of the qtree or qtrees for which you want to see the status of SnapVault operations. You can specify more than one qtree path.

Result: The filer displays a message showing whether a transfer is in progress, how much data has been transferred, the state of the destination, and how long ago the last successful transfer took place.

◆ If no options or arguments are given, the output includes the information in Example 1 below.

  If [*filer_name*:]*qtree_path* arguments are specified, then status is displayed only for the specified qtrees.

◆ If the `-l` option is given, the output includes the more detailed information shown in Example 2.

  See the table "What the status fields mean" on page 251 for more information about the command output shown in Example 1 and Example 2.

◆ If the `-s` option is given, the output displays snapshot creation status, as shown in Example 3.

◆ If the `-c` option is given, the output displays current primary storage system configuration parameter settings as shown in Example 4.

Example 1: If you enter `snapvault status` with no option, you see SnapVault qtree relationships involved in the most recent transfer from a primary qtree to a secondary qtree.

```
filerA> snapvault status
Snapvault client is ON.
```

```
Source                   Destination                     State          Lag       Status
filerB:/vol/vol2/qtree3  filerA:/vol/sv_vol/qtree3       Snapvaulted  00:48:24   Idle
```

Example 2: The `snapvault status -l` command displays more detailed information on the most recent SnapVault transfer and snapshot activity.

```
filerA> snapvault status -l
SnapVault client is ON.

Source:                      filerA:/vol/vol2/qtree3
Destination                  filerB:/vol/sv_vol/qtree3
Status                       Idle
Progress:                    -
State:                       SnapVaulted
Lag:                         2:09:58
SnapVault Timestamp:         Thu Jan 31 12:30:01 PST 2002
Base Snapshot:               filerB(0016778780)_sv_vol.59
Current Transfer Type
Current Transfer Error:
```

```
Contents:                    Replica
Last Transfer Type:          Scheduled
Last Transfer Size:          1680 KB
Last Transfer Duration:      00:02:08-
```

Example 3: The `snapvault status -s` command lists all the snapshots scheduled on the primary or secondary storage system. Information includes volume, snapshot basename, current status, and snapshot schedule.

```
filerA> snapvault status -s
Snapvault server is ON.

Volume          Snapshot      Status              Schedule
------          --------      ------              --------
sv_vol          sv_hourly     Idle                1@0-23
vol2            sv_weekly     Idle                8@fri
```

Example 4: The snapvault status `-c` command lists all the secondary system qtrees, their corresponding primary system qtrees, maximum speed of scheduled transfers, and maximum number of times SnapVault attempts to start a scheduled transfer before skipping that transfer.

```
filerA> snapvault status -c
/vol/sv_vol/db_qtree1 source=filerB:/vol/db1/s1 kbs=unlimited
tries=20
/vol/sv_vol/db_qtree2 source=filerC:/vol/db2/s2 kbs=unlimited
tries=20
/vol/sv_vol/qtree1 source=filerC:/vol/users/qtree1 kbs=10000
tries=10
/vol/sv_vol/qtree2 source=filerC:/vol/users/qtree2 kbs=10000
tries=10
/vol/sv_vol/qtree3 source=filerD:/vol/users/qtree1 kbs=10000
tries=10
/vol/sv_vol/qtree4 source=filerD:/vol/db/db_1 kbs=7000 tries=10

filerA> snapvault status -c /vol/sv_vol/qtree3
/vol/sv_vol/qtree3 source=filerA:/vol/vol2/qtree3 kbs=10000
tries=10
```

What the status fields mean

Information fields that SnapVault can display for the `snapvault status` and `snapvault status -l` commands are as follows.

Field	Possible values that might be displayed:
Source	*filer:qtree_path*—The source is the primary storage system filer and qtree path listed. A dash (-) in this field means that the SnapVault secondary storage destination qtree is not yet initialized through the `snapvault start -x` command.
Destination	*filer:qtree_path*—The destination is the secondary storage system filer and qtree path listed.
State	Uninitialized—The destination SnapVault secondary storage qtree is not yet initialized through the `snapvault start` command. Snapvaulted—The qtree is a SnapVault secondary destination. Unknown—It is not known what state the secondary system qtree is in; the volume that contains the secondary system qtree could be offline or restricted. Source—This state is reported when the `snapvault status` command is run on the primary storage system. It also appears if `snapvault status` is run on secondary storage systems after the `snapvault restore` command was run on an associated primary storage system.
Lag	*hh:mm:ss* indicates the time difference between the data currently on the primary system and the latest data stored on the secondary system; that is, the difference between the current time and the timestamp of the snapshot last successfully transferred to the secondary storage system. A dash (-) in this field means that the secondary storage system is not initialized.

Field	**Possible values that might be displayed:**
Status	Aborting—A transfer is being aborted and cleaned up. Idle—No data is being transferred. Pending—The secondary storage system cannot be updated because of a resource issue; the transfer is retried automatically. Quiescing—The specified qtree is waiting for all existing transfers to complete. The destination is being brought into a stable state. Resyncing—The specified qtree is resynchronizing. Transferring—Transfer has been initiated, but has not yet started, or is just finishing.
Progress	Shows the number of KB transferred by the current transfer, or the restart check point if the status is Idle or Pending.
Mirror Timestamp	*hh:mm:ss* indicates the timestamp of the last snapshot successfully transferred from the primary storage system to the secondary storage system. **Note** A resynchronization (`snapvault start -r`) may change the base snapshot to a snapshot with a timestamp older than the original base.
Base Snapshot	The name of the base snapshot for the corresponding qtree on the secondary storage system. For qtrees in a SnapVault relationship, the secondary storage side lists the name of the exported snapshot for that qtree on the storage side. A resynchronization (`snapvault start -r`) may change the name of the base snapshot.
Current Transfer Type	Indicates the type of the current transfer: scheduled, retry, resync, update, initialize, store, or retrieve. This field applies only to the destination side.

Field	**Possible values that might be displayed:**
Current Transfer Error	Displays an error message if the latest transfer attempt failed.
Contents	Indicates whether the contents of the destination volume or qtree in the active file system are up-to-date replicas or are in transition. The field applies only to the destination side. Since a destination is read-only, its contents are always a replica.
Last Transfer Type	Indicates the type of the previous transfer: scheduled, retry, resync, update, initialize, store, or retrieve. This field applies only to the secondary storage side.
Last Transfer Size	Shows the number of KB transferred in the last successful transfer.
Last Transfer Duration	Shows the elapsed time for the last successful transfer. If the transfer failed and restarted, the time includes time waiting to restart the transfer. If a transfer aborted and was retried from the beginning, it includes only the time required for the final successful attempt.
Last Transfer From	This field applies only to the secondary storage side and shows the name of the primary filer and volume or qtree (where the content is transferred from).

Displaying SnapVault snapshots

Why to display SnapVault snapshots and qtrees

You can use the `snap list` command to display a list of snapshots to confirm what versions of your primary qtree data have been backed up, or to locate by date or time a particular version of a qtree to retrieve. Using the `snap list -q` command, you can see the following:

- A list of all snapshots on the secondary storage system (not just SnapVault snapshots)
- The qtrees in the snapshots
- The primary storage system sources of those qtrees
- The timestamp of the primary storage system snapshot that was the source for the data in the secondary storage system snapshot

Using the `snap list -o` command, you can also list the snapshot timestamps, primary qtree origin (if applicable), and snapshot names stored for an individual qtree.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For descriptions and procedures pertaining to SnapVault backup of Open Systems drives and directories, see “Setting up SnapVault backup on Open Systems platforms” on page 215 and “Managing SnapVault backup of Open Systems platforms” on page 220.

How to display SnapVault snapshots on volumes

To see a list of the snapshots and qtrees on your filer volumes, complete the following step.

Step	Action
1	On the filer for which you want to see the information, enter the following command: **`snap list -q [`*`volume_name`*`]`** *volume_name* is the name of the volume for which you want to list the snapshots. If no volume name is given, the snapshots on all this filer’s volumes are displayed.

Sample snap list -q output

Primary storage output example: If you specify the primary volume name, the command lists the information for each snapshot in the volume. This output is from a volume used as a SnapVault primary storage system.

```
filerA> snap list -q vol2
working...

Volume vol2
    qtree                contents       timestamp     source
    -----                --------       ---------     ------
sv_hourly.0  (Jan 22 20:00)
    qtree1               Original       Jan 22 20:00  -
    qtree2               Original       Jan 22 20:00  -
    qtreeZ               Original       Jan 22 20:00  -
sv_hourly.1  (Jan 22 16:00)
    qtree1               Original       Jan 22 16:00  -
    qtree2               Original       Jan 22 16:00  -
    qtreeZ               Original       Jan 22 16:00  -
sv_hourly.2  (Jan 22 12:00)
    qtree1               Original       Jan 22 12:00  -
    qtree2               Original       Jan 22 12:00  -
    qtreeZ               Original       Jan 22 12:00  -
sv_hourly.3  (Jan 22 08:00)
    qtree1               Original       Jan 22 08:00  -
    qtree2               Original       Jan 22 08:00  -
    qtreeZ               Original       Jan 22 08:00  -
sv_nightly.0  (Jan 22 00:00)
    qtree1               Original       Jan 22 00:00  -
    qtree2               Original       Jan 22 00:00  -
    qtreeZ               Original       Jan 22 00:00  -
sv_hourly.4  (Jan 21 20:00)
    qtree1               Original       Jan 21 20:00  -
    qtree2               Original       Jan 21 20:00  -
    qtreeZ               Original       Jan 21 20:00  -
sv_hourly.5  (Jan 21 16:00)
    qtree1               Original       Jan 21 16:00  -
    qtree2               Original       Jan 21 16:00  -
    qtreeZ               Original       Jan 21 16:00  -
sv_nightly.1  (Jan 21 00:00)
    qtree1               Original       Jan 21 00:00  -
    qtree2               Original       Jan 21 00:00  -
    qtreeZ               Original       Jan 21 00:00  -
```

This output shows which qtrees were writable and therefore have original content (the timestamp in these cases is the same as for the snapshot as a whole). It also shows whether any qtrees were transitioning and are therefore neither a faithful replica nor original content. Transitioning qtrees are shown with a dash (-) instead of a timestamp.

Secondary storage output example: If you specify the volume name (in this example, sv_vol) and are running the command from a filer used as a SnapVault secondary storage system, you see a list of all the SnapVault snapshots retained on volume sv_vol and the details of the qtrees contained in those snapshots.

```
filerB> snap list -q sv_vol
Volume sv_vol
working...

   qtree       contents    date           source
   --------    ---------   --------       -------
sv_hourly.0 (Jan 31 20:00)
   qtree1      Replica     Jan 22 20:40   filerA:/vol/vol2/qtree1
   qtree2      Replica     Jan 22 20:40   filerA:/vol/vol2/qtree2
   qtreeZ      Replica     Jan 22 20:40   filerA:/vol/vol2/qtreeZ
sv_hourly.1  (Jan 22 16:00)
   qtree1      Replica     Jan 22 16:00   filerA:/vol/vol2/qtree1
   qtree2      Replica     Jan 22 16:00   filerA:/vol/vol2/qtree2
   qtreeZ      Replica     Jan 22 16:00   filerA:/vol/vol2/qtreeZ
sv_hourly.2  (Jan 22 12:00)
   qtree1      Replica     Jan 22 12:00   filerA:/vol/vol2/qtree1
   qtree2      Replica     Jan 22 12:00   filerA:/vol/vol2/qtree2
   qtreeZ      Replica     Jan 22 12:00   filerA:/vol/vol2/qtreeZ
.....
```

This output shows which qtrees are replicas of another qtree, and the timestamp of the source snapshot.

Note

Qtrees that are transitioning appear with a dash (-) instead of a timestamp. In general, you should not attempt to restore snapshot versions of qtrees that are transitioning.

If you specify no arguments, the output shows the information for each snapshot in each volume.

How to list snapshots for qtrees

To see a list of snapshots associated with a qtree and, if applicable, the snapshots' primary qtree origins, complete the following step.

Step	Action
1	On the filer for which you want to see the information, enter the following command: **`snap list -o [`*`qtree_path`*`]`** *qtree_path* displays one qtree. If no qtree name is given, information for all the qtree names on the volume are displayed.

Sample snap list -o output

If you specify the `-o` parameter with a qtree path, the `snap list` output includes the dates, sources (if any), and names of associated SnapVault snapshots that are retained on the system, for example:

```
filerB> snap list -o /vol/sv_vol/qtree3
working...

Qtree /vol/sv_vol/qtree3
date          source                      name
------------  --------                    --------
Jan 31 18:00  filerA:/vol/vol2/qtree3     hourly.0
Jan 31 17:00  filerA:/vol/vol2/qtree3     hourly.1
Jan 31 16:00  filerA:/vol/vol2/qtree3     hourly.2
Jan 31 15:00  filerA:/vol/vol2/qtree3     hourly.3
Jan 30 14:00  filerA:/vol/vol2/qtree3     hourly.4
Jan 30 13:00  filerA:/vol/vol2/qtree3     hourly.5
Jan 30 12:00  filerA:/vol/vol2/qtree3     hourly.6
Jan 30 11:00  filerA:/vol/vol2/qtree3     hourly.7
Jan 31 10:00  filerA:/vol/vol2/qtree3     hourly.8
Jan 31 9:00   filerA:/vol/vol2/qtree3     hourly.9
Jan 31 8:00   filerA:/vol/vol2/qtree3     hourly.10
Jan 30 20:00  filerA:/vol/vol2/qtree3     nightly.0
Jan 29 20:00  filerA:/vol/vol2/qtree3     nightly.1
Jan 26 16:00  filerA:/vol/vol2/qtree3     weekly.0
```

Changing settings for SnapVault backup relationships

Why you change SnapVault settings

You can use the `snapvault modify` command to change the primary storage system (source) qtree that you specified using the `snapvault start` command.

You can also use the `snapvault modify` command to change the SnapVault settings for transfer speed and number of tries before quitting that you entered using the `snapvault start` command. You might need to make these changes if there are hardware or software changes to your filers.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For descriptions and procedures pertaining to SnapVault backup of Open Systems drives and directories, see "Setting up SnapVault backup on Open Systems platforms" on page 215 and "Managing SnapVault backup of Open Systems platforms" on page 220.

The `snapvault modify` command is available only from the secondary storage system.

If you need to change the SnapVault schedule, use the `snapvault snap sched` command, explained in more detail in "Starting a SnapVault backup relationship" on page 234.

How to change SnapVault settings

To change the values you entered with the `snapvault start` command, complete the following step.

Step	Action
1	From the secondary storage system, enter the following command on a single line: **`snapvault modify [-k`** ***`kbs`*** **`] [-t`** ***`n`*** **`]`** **`[-S [`** ***`prim_filer`*** **`:]`** ***`prim_qtree_path`*** **`]`** **`[`** ***`sec_filer`*** **`:]`** ***`sec_qtree_path`*** `-k` *kbs* specifies a value in kilobytes per second for the throttle (transfer speed) for the primary storage system. A value of *unlimited* lets the transfer run as fast as it can. Other valid values are whole positive numbers. `-t` *n* specifies the number of times to try the transfer before giving up. The default is 2. Valid values are positive integers. `-S` [*prim_filer*:]*prim_qtree_path* specifies the primary storage system for the qtree. [*sec_filer*:]*sec_qtree_path* specifies the secondary storage system for the update.

Example: This example shows how to modify SnapVault so that it will continue backing up the data on the primary storage system filerB:/vol2/qtree3 after the name qtree3 on the primary storage system changes to qtreeBob.

```
filerA> snapvault status

Snapvault server is ON.
Source                     Destination                  State           Lag        Status
filerB:/vol/vol2/qtree3    filerA:/vol/sv_vol/qtree3    Snapvaulted     02:48:24   Idle
```

[The qtree3 on filerB is renamed qtreeBob.]

```
filerA> snapvault modify -S filerB:/vol/vol2/qtreeBob /vol/sv_vol/qtree3

filerA> snapvault status
Snapvault server is ON.
Source                      Destination                  State           Lag        Status
filerB:/vol/vol2/qtreeBob   filerA:/vol/sv_vol/qtree3    Snapvaulted     0:00:31    Idle
```

Manually updating individual secondary storage system qtrees

Why to manually update a qtree on the secondary storage system

You can use the `snapvault update` command to manually update the SnapVault qtree on the secondary storage system from a snapshot on the primary storage system. You might want to update at an unscheduled time to protect the primary storage system data if one of the following conditions exists:

- ◆ A disk failed on the primary storage system and you want extra protection for the data.
- ◆ The nightly backup failed due to a network problem.
- ◆ The primary storage system hardware is going to be reconfigured.
- ◆ You want to transfer a snapshot of a quiesced database.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For SnapVault backup of Opens System drives and directories, see “Setting up SnapVault backup on Open Systems platforms” on page 215 and “Managing SnapVault backup of Open Systems platforms” on page 220.

How to update the snapshot on the secondary storage system

To manually update a SnapVault qtree on the secondary storage system, complete the following step.

Step	Action
1	Enter the following command from the secondary storage system: **`snapvault update [`*`options`*`] [`*`sec_filer`*`:]`*`sec_qtree_path`*** *options* can be one or more of the following: ◆ `-k` *kbs* overrides the configured rate and specifies a value in kilobytes per second for the throttle (transfer speed) for this snapshot transfer. The default value, *unlimited*, lets the transfer run as fast as it can. ◆ `-s` *snapname* enables you to specify a primary storage system snapshot that is more recent than the current base snapshot. [*sec_filer*:]*sec_qtree_path* is the name of the secondary storage system qtree that you want to update.

Example 1: `filerB> snapvault update /vol/vol2/qtree3`

Result: SnapVault updates the qtree on the secondary storage system (filerB) with the data from a new snapshot of the qtree it creates on the primary storage system (filerA). You do not have to specify where the primary storage system data is, because you already did so when you set up the SnapVault relationship using the `snapvault start` command.

Example 2: To update qtree3 on the secondary storage system (filerB) with a particular snapshot of qtree3 on the primary storage system (filerA), you would use the `-s` option to specify the snapshot on the primary storage system, as in the following example.

```
filerB> snapvault update -s my_snap filerB:/vol/vol0/qtree3
```

Result: SnapVault updates the qtree on the secondary storage system (filerB) with the data from a snapshot of qtree3 (my_snap) that you created earlier on the primary storage system (filerA).

Note

The `snapvault update` command does not create a new snapshot on the secondary storage system. Use the `snapvault snap create` command if you want to create a new snapshot on the secondary storage system. See “Creating a snapshot manually” on page 262.

Creating a snapshot manually

Why to create a snapshot manually

You might create a manual (unscheduled) snapshot for the following reasons:

- You anticipate planned downtime or you need to recover from downtime (during which a snapshot was not taken on time).
- You have just carried out a manual update of a secondary qtree, as described in “Manually updating individual secondary storage system qtrees” on page 260, and you want to immediately incorporate that update into the retained snapshots on the secondary storage system.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For descriptions and procedures pertaining to SnapVault backup of Open Systems drives and directories, see “Setting up SnapVault backup on Open Systems platforms” on page 215 and “Managing SnapVault backup of Open Systems platforms” on page 220.

How to create a snapshot manually

You run this command on the SnapVault primary storage system or SnapVault secondary storage system containing the volume on which you want to create the snapshot. To create a manual snapshot of a volume, complete the following step.

Step	Action
1	From the primary storage system or secondary storage system, enter the following command: **`snapvault snap create`** ***`volume_name snap_name`*** *volume_name* is the name of the volume where the snapshot to be created will reside. *snap_name* is the basename of the snapshot to create. If there is already a snapshot being created in the volume at the time this command is invoked, this command will be carried out after the other snapshot is completed.

Example: `filerB> snapvault snap create vol1 sv_nightly`

Result: SnapVault creates a new snapshot and, based on the specified snapshot basename, numbers it just as if that snapshot had been created by the SnapVault schedule process. SnapVault names the new snapshot sv_nightly.0, renames the older snapshots, and deletes the oldest sv_nightly snapshot.

Note

Unlike the `snapvault snap sched -x` command, the `snapvault snap create` command does not update the data in the secondary storage system qtree from the data in the primary storage system qtree prior to creating the new snapshot. If you want to update your secondary qtrees prior to running the `snapvault snap create` command, use the `snapvault update` command as described in "Manually updating individual secondary storage system qtrees" on page 260.

Restoring SnapVault data to the primary storage system

About SnapVault data restoration

In the event of data loss on a primary storage system, restoring data from the SnapVault secondary storage system involves these command-line operations.

- You use the `snapvault restore` command to restore a backed-up qtree from its last update saved to the secondary storage system.
- After successfully restoring data, you use the `snapvault start` command to restart the SnapVault backup relationship between the primary and secondary qtrees (because you want to continue SnapVault protection of the data). If you do not want to continue the backup relationship, you use the `snapvault release` command to cancel any further backups of the restored qtree and to release the resources on the secondary storage system that were used in the SnapVault relationship.

Note

The descriptions and procedures in this section pertain to SnapVault backup of NetApp filers only. For SnapVault backup of Open Systems drives and directories, see "Setting up SnapVault backup on Open Systems platforms" on page 215 and "Managing SnapVault backup of Open Systems platforms" on page 220.

Restoring data to the primary storage system

To restore data from the SnapVault secondary storage system to the primary storage system, complete the following steps.

Step	Action
1	If you intend to restore a primary qtree to the exact qtree location on the primary storage system from which you backed it up, delete the existing qtree from the primary storage system.

Step	Action	
2	Enter the following command on the primary storage system on a single line: **`snapvault restore [`*`options`*`] -S [`*`sec_filer`*`:]`*`sec_qtree_path`* `[`*`prim_filer`*`:]`*`prim_qtree_path`*** `-S` [*sec_filer*:]*sec_qtree_path* specifies the secondary storage system and qtree path from which you want to restore the data. *prim_filer* is the name of the primary storage system filer that you want to restore to. If specified, this filer name must match the name of the host filer. *prim_qtree_path* is the name of the primary storage system qtree that you want to restore to. For options to the `snapvault restore` command, see the na_snapvault(1) man page.	
3	**If you want to...**	**Then...**
	Resume SnapVault backups of the newly restored qtree on the primary storage system	Go to Step 4.
	Discontinue the backup relationship between the newly restored qtree on the primary storage system and its secondary qtree partner	Go to Step 6.

Step	Action
4	If you want to resume the SnapVault relationship between the restored qtree and its backup qtree on the secondary storage system, enter the following command on a single line: **`snapvault start -r [`*`options`*`] -S [`*`prim_filer`*`:]`*`prim_qtree_path`*` [`*`sec_filer`*`:]/vol/`*`sec_volume`*`/`*`sec_qtree_path`*** `-r` is required to restart backups from a restored primary storage system. *options* can be one or more of the following: ◆ `-k` *kbs* sets a value for the throttle (transfer speed) in kilobytes per second for the primary storage system. ◆ `-t` *n* sets the number of times to try to start the transfer before giving up. **Note** This parameter does not enable SnapVault to restart incomplete transfers. `-S` [*prim_filer*:]*prim_qtree_path* specifies the primary storage system and path for the restored qtree. *sec_filer* is the name of the secondary storage system filer to which the snapshots from the primary storage system are transferred. If specified, this filer name must match the name of the host filer. If no secondary storage system filer is specified, the local host’s name is used for the filer name. *sec_volume* is the name of the volume on the secondary storage system to which the data from the primary storage system is transferred. *sec_qtree_path* is the path and name of the qtree on the secondary storage system. It must be the same name and path as used for the *sec_qtree_path* variable in Step 1.
5	You have finished. See Example 1.

Step	Action
6	If you want to discontinue the SnapVault relationship between the restored qtree and its backup qtree on the secondary storage system, enter the following command on the secondary storage system: **`snapvault release`** ***`sec_qtree_path`*** **`[`*`prim_filer`*`:]`*`prim_qtree_path`*** *sec_qtree_path* is the name of the secondary storage system qtree that you want to release from a SnapVault relationship. [*prim_filer*:]*prim_qtree_path* is the name of the primary storage system qtree that you want to release.
7	You have finished. See Example 2.

Example 1: This example shows the primary storage system (filerA) requesting data to be restored from the secondary storage system (filerB) and then the secondary storage system restarting the SnapVault backup relationship.

```
filerA> snapvault restore -S filerB:/vol/sv_vol/qtree3
vol/vol1/qtree3

filerB> snapvault start -r -S filerA:/vol/vol1/qtree3
filerB:/vol/sv_vol/qtree3
```

Example 2: This example shows the primary storage system (filerA) requesting data to be restored from the secondary storage system (filerB) and then the secondary storage system canceling the SnapVault backup relationship on both filers to release the resources used.

```
filerA> snapvault restore -S filerB:/vol/sv_vol/qtree3
/vol/vol1/qtree3

filerB> snapvault release /vol/sv_vol/qtree3
filerA:/vol/vol1/qtree3
```

Deleting the residual snapshot

When you use the `snapvault restore` command to restore a primary qtree, SnapVault places a residual SnapVault snapshot on the volume of the restored primary qtree. This snapshot is not automatically deleted. If you have configured this volume to retain the maximum 251 snapshots allowed by Data ONTAP, you must manually delete this residual snapshot, or else no new snapshots can be created.

To find and delete this residual snapshot, complete the following steps.

Step	Action

1	To display all snapshots (including the residual snapshot) on the volume of the restored qtree, enter the following command:
	`snap list` ***`primaryvolume`***
	The residual snapshot will be distinguished by the following syntax:
	`primaryhost` `(`*`nvram_id`*`)`*`_primaryvolume_restoredqtree`*`-dst.2`
	Example:
	`prim_filer (1880911275)_vol1_mytree-dst.2`

2	To delete the residual snapshot, enter the following command:
	`snap delete` *`primaryvolume extrasnapshotname`*
	Example:
	`snap delete vol1 prim_filer (1880911275)_vol1_mytree-dst.2`

Aborting SnapVault transfers

Why you abort transfers

You can use the `snapvault abort` command to halt an ongoing SnapVault transfer if a later transfer would be more useful or if an immediate shutdown or reboot is necessary.

This command can halt ongoing SnapVault transfers from primary to secondary storage (invoked by the `snapvault start` or `snapvault update` commands or scheduled through the `snapvault snap sched -x` command), or from secondary back to primary storage (invoked by the `snapvault restore` command).

You can enter the `snapvault abort` command at the filer primary storage system or at the secondary storage system.

Aborting primary to secondary storage transfers

The `snapvault abort` command can halt ongoing SnapVault transfers from primary to secondary storage that were invoked by the `snapvault start` or `snapvault update` commands, or that were scheduled through the `snapvault snap sched -x` command. To abort a primary to secondary storage transfer, complete the following step.

Step	Action
1	At the filer console of either the primary or secondary storage system, enter the following command: **`snapvault abort [`*`sec_filer`*`:]/vol/`*`volx/sec_qtree`*** *sec_filer* is the name of the SnapVault secondary storage system. *sec_qtree* is the name of the secondary qtree to which the data is being transferred through SnapVault start or update commands. **Note** If you use the `-h` option (hard abort) with the `snapvault abort` command, you cannot restart the transfer.

Aborting secondary to primary storage transfers

The `snapvault abort` command can halt ongoing SnapVault transfers from secondary to primary storage that were invoked by the `snapvault restore` command. To abort a secondary to primary storage transfer, complete the following step.

Step	Action
1	At the filer console of either the primary or secondary storage system, enter the following command: **`snapvault abort [`*`prim_filer`*`:]/vol/`*`volx`*`/`*`prim_qtree`*** *prim_filer* is the name of the primary storage filer. *prim_qtree* is the name of the primary qtree to which the data is being restored. **Note** If you use the `-h` option (hard abort) with the `snapvault abort` command, you cannot restart the transfer.

Aborting SnapVault snapshot creation

To abort the ongoing creation of a SnapVault snapshot on the secondary storage system, complete the following step.

Step	Action
1	At the filer console of the secondary storage system, enter the following command: **`snapvault abort -s`** ***`volx sv_snapname`*** The `-s` option specifies that the command aborts the attempt to create a snapshot with basename of *sv_snapname* on volume *volx*.

You can obtain the secondary storage system volume and snapshot basenames from the `snapvault status -s` output.

Ending SnapVault backups for a qtree

Why you end SnapVault backups

You can use the `snapvault stop` command to end the SnapVault backup process for a qtree when you no longer need the data in the primary storage system qtree to be protected.

How to end a SnapVault backup

To end SnapVault backups for a specified qtree, complete the following step.

Step	Action
1	From the secondary storage system, enter the following command: **`snapvault stop [`*`sec_filer`*`:]`*`sec_qtree_path`*** [*sec_filer*:]*sec_qtree_path* is the qtree that you no longer want to back up.

Example: `filerB> snapvault stop filerB:/vol/sv_vol/qtree3`

Result: SnapVault stops updating the qtree on the secondary storage system and deletes the qtree. Existing snapshots on the secondary storage system are unaffected, but as new snapshots replace the old ones, the data from the qtree whose backup was stopped will disappear.

Note

After you end the backup process from a SnapVault secondary storage system, you might want to release the now-obsolete snapshots on the primary storage system. For more information, see “Releasing SnapVault relationships” on page 273.

Unscheduling SnapVault snapshots

Why you unschedule snapshots

You might want to unschedule a set of SnapVault snapshots if the data in the qtrees you are backing up has been migrated to another location or is no longer useful.

How to unschedule snapshots

To turn off the SnapVault schedule for a set of snapshots and stop the snapshot process for the SnapVault primary storage system or secondary storage system, complete the following step.

Step	Action
1	From the primary or secondary storage system, enter the following command: **`snapvault snap unsched [-f] [`*`volume`*` [`*`snap_name`*`]]`** `-f` forces the command to run without showing the list of snapshots to stop creating and without asking for confirmation. *volume* is the name of the volume to stop creating snapshots on. *snap_name* is the basename of the snapshot set to stop creating. If no *snap_name* is provided, SnapVault turns off the SnapVault snapshot schedule for all snapshot sets in the volume and does not update any more SnapVault snapshots in the volume. If no volume is provided, SnapVault unschedules all SnapVault snapshot sets for the filer and does not update any more SnapVault snapshots in the filer. If there is already a snapshot being created in the volume, the command fails.

Example: `filerB> snapvault snap unsched vol1 sv_nightly`

Result: SnapVault asks for confirmation. If you confirm the action, SnapVault unschedules all SnapVault snapshots with the basename sv_nightly on vol1 of filerB.

Releasing SnapVault relationships

Why to release SnapVault relationships

You release a SnapVault relationship between a primary qtree and its secondary qtree backup (originally defined through the `snapvault start` command) after the relationship is no longer needed, for example:

- On a primary storage system as a part of shutting down a SnapVault relationship after a `snapvault stop` command was completed on the secondary storage system
- On the secondary storage system after data is restored to a primary storage system and you do not want to reactivate the backup relationship between the primary and secondary qtrees

How to release SnapVault relationships

To release snapshots and free the space they used, complete the following step.

Step	Action
1	Enter one of the following commands. ◆ On a primary storage system filer console, enter: **`snapvault release`** ***`prim_qtree_path sec_filer`*****`:`*****`sec_qtree_path`*** ◆ On the secondary storage system filer console, enter: **`snapvault release`** ***`sec_qtree_path prim_filer`*****`:`*****`prim_qtree_path`***

Example:

```
filerB> snapvault release /vol/sv_vol/qtree3
filerA:/vol/vol1/qtree3
```

Protecting your SnapVault backups through SnapMirror

About SnapVault secondary storage system protection

By setting up a SnapMirror relationship between the SnapVault secondary storage system and a SnapMirror destination filer, NearStore system, or tape backup unit, you can provide backup and standby service or backup and restore protection for the SnapVault secondary storage system data.

- SnapMirror backup and standby service for SnapVault uses the SnapMirror destination device as a standby device to be activated as an alternate SnapVault secondary storage if the original secondary storage device goes down. For details see "Backup and standby service for SnapVault" on page 274.
- SnapMirror backup and restore protection for SnapVault uses the SnapMirror destination device as a source from which you can restore backup data to a SnapVault secondary storage system that has suffered data loss or corruption. For details see "Backup and restore protection for SnapVault" on page 275.

The process of using SnapMirror with SnapVault: The SnapVault secondary storage system will carry out SnapVault operations on its filer sources as usual. Then on a scheduled per-volume basis, the system will replicate the SnapVault data to its SnapMirror destination partner or tape backup unit.

Under this configuration, SnapVault does not delete any snapshot version on the primary storage systems until that version has been successfully mirrored from the SnapVault secondary storage unit to its SnapMirror destination. This guarantees that a snapshot version will always be retrievable even if the SnapVault secondary storage system is disabled.

Backup and standby service for SnapVault

To set up SnapMirror backup and standby protection for the SnapVault secondary storage system:

1. Use the `license` command to confirm that the SnapVault secondary storage device has both SnapVault secondary storage and SnapMirror features licensed.

2. Use the `license` command to confirm that the SnapMirror destination device has both the SnapVault secondary storage and SnapMirror features licensed.

3. Set up SnapMirror replication from the active SnapVault secondary storage system to a disk-based destination device (another filer or NearStore system).

   For a quick description of setting up a SnapMirror relationship with a disk-based destination device, see “Setting up a basic SnapMirror operation” on page 93.

4. If the active SnapVault secondary storage system is damaged or destroyed, convert the SnapMirror destination device to an alternate SnapVault secondary system to carry on the task of backing up data from the primary storage filers.

   For details on converting a SnapMirror destination to a writable volume, see “Converting a destination to a writable volume or qtree” on page 156.

5. Activate the SnapVault license on the new SnapVault systems, and use the `snapvault start` and `snapvault snap sched` commands to complete configuration of the new SnapVault secondary storage system.

Backup and restore protection for SnapVault

To set up SnapMirror backup and restore protection for the SnapVault secondary storage system:

1. Use the `license` command to confirm that the SnapVault secondary storage device has both SnapVault secondary storage and SnapMirror features licensed.

2. If the SnapMirror destination device is a filer or NearStore system, use the `license` command to confirm that its SnapMirror feature is licensed.

3. Set up SnapMirror replication from the active SnapVault secondary storage system to either a disk-based device or a tape device.

   ❖ For a quick description of setting up a SnapMirror relationship with a destination disk-based device, see “Setting up a basic SnapMirror operation” on page 93.

   ❖ For a description of setting up a SnapMirror relationship with a destination tape device, see “SnapVault-to-tape backup scenario” on page 183.

4. If the active SnapVault secondary storage system suffers data loss or corruption, use the appropriate SnapMirror commands to restore the necessary data from the SnapMirror destination device back to SnapVault secondary storage.

- For a description of restoring to a SnapMirror source from a destination (the SnapVault secondary storage unit would be the source, and the partner device would be the destination) see “Disaster recovery: a special use for snapmirror resync” on page 166.
- For a description of restoring to a SnapMirror source from a tape drive (the SnapVault secondary storage unit would be the source, and the tape device would be the destination) see “Restoring to SnapVault from a local tape” on page 185.

Turning SnapVault off

Why you turn SnapVault off

You might want to turn SnapVault off on a filer because the files there are no longer important or current or have been moved to another location. When you no longer want SnapVault to run on a filer, you can turn it off using the `options snapvault.enable off` command.

How to turn SnapVault off

To turn off SnapVault on a filer primary storage system or on a secondary storage system, complete the following step.

Step	Action
1	On both the primary storage system and the secondary storage system, enter the following command: **`options snapvault.enable off`** This option persists across reboots.

Volume Copy 6

About this chapter

This chapter discusses using the `vol copy` set of commands for replicating all of the data from one volume to another. This chapter does not discuss using the SnapMirror option, which automatically maintains copies (replicas) of data in one volume in another volume. For information about replicating a volume using SnapMirror, see "Data Protection Using SnapMirror" on page 73.

Topics of this chapter

This chapter discusses the following topics:

Learning about volume copy

What a volume copy is

Volume copy is a way of copying both data in the active file system and data in snapshots from one volume to another. The source and destination volumes must be the same type (traditional or flex). You can initiate a volume copy with the `vol copy start` command, which enables you to copy data from one volume to another volume, either on the same filer or on a different filer. The result is a restricted volume containing the same data as the source volume at the time you initiated the copy operation.

Benefits of the vol copy command set

Although you can copy data on the filer using client programs such as `cpio` or use the Data ONTAP `dump` and `restore` commands, the `vol copy` command set offers the following benefits:

- When a `vol copy` command reads and writes data, Data ONTAP does not traverse directories on the filer. Data is copied block for block directly from the disks, which means that Data ONTAP can finish the copying faster than it could with other methods.
- Using a `vol copy` command, Data ONTAP preserves the snapshot data of the source volume. If, in the future, users might need to use snapshots that were taken before data was copied from one volume to another, you can use a `vol copy` command for migrating data. For example, if users accidentally delete files and need to recover them, they can do so from the preserved data.

Maximum number of simultaneous volume copies on a filer

Volume copy has the same limit of simultaneous copies that SnapMirror replications have. See “Maximum number of simultaneous replications on a filer” on page 81.

When to copy volumes

The following table describes some situations where you might find copying volumes useful.

Situation	Reasons for copying one volume to another
You want to migrate data from one filer to another.	The destination filer has more storage or is a model that supports newer technology.
You want to move a volume from one set of disks to another on the same filer.	You want to ◆ Split a volume ◆ Expand filer storage **Examples:** You can copy the vol0 volume to the vol1 volume and then delete duplicate files and directories in these volumes so that the original contents of vol0 are split into two volumes. You have six 9-GB disks for the vol0 volume and four 18-GB spare disks. You can migrate vol0 to the four 18-GB disks and replace all the 9-GB disks with larger capacity disks.
You want to copy data from one filer to another regularly to ensure high data availability.	After you copy the data, clients can switch to the destination filer in the following scenarios: ◆ When you shut down the source filer for software or hardware upgrades, or when the source filer is not available for reasons such as natural disasters, you can put the destination volume online to continue file service. ◆ If a network client process accidentally deletes a large number of files on the source filer, clients can continue to have access to the files on the destination filer while you are restoring the files to the source filer. **Note** This is also a good application for SnapMirror.

Preparing to copy a volume

Requirements for copying a volume

The filers involved in a volume copy operation must meet several requirements. The following list provides a brief description of these requirements. The rest of this section provides more detailed information about verifying whether the source and destination volumes meet these requirements.

- The source and destination volumes must be of the same type: either both traditional or both flexible volumes.
- The capacity of the destination volume must be greater than or equal to the capacity of the source volume.
- The source and destination filers must have a trust relationship with each other.
- The destination volume must exist, and must not be the root volume.
- The source volume must be online and the destination volume must be restricted.
- Remote Shell access must be enabled.
- The destination volume must not contain data that you want to preserve.

Take care not to overwrite data that you need

If the destination volume is not a new volume, make sure that it does not contain data that you might need in the future. After Data ONTAP starts copying the source volume, it overwrites the entire destination volume. All data in the active file system and in the snapshots of the destination volume is lost after Data ONTAP starts copying the data.

Where volume copies can reside

The source and destination volumes of the copy can reside on the same filer or on different filers.

Recommendation for copying a volume

When a filer copies data between two volumes on separate filers, it floods the network between the two filers with packets. Users of the filers involved in a volume copy operation might notice a degradation in response time during the copy. To avoid network-related performance problems when copying to a different filer, Network Appliance recommends that you set up a private network for copying between the source and destination filers.

For detailed information

The following sections discuss the ways you prepare for copying a volume:

- "Verifying the size of each volume" on page 284
- "Verifying the relationship between filers" on page 286
- "Verifying and changing status of source and destination volumes" on page 287
- "Enabling remote access" on page 289

Verifying the size of each volume

Verifying the volume size

To see whether the data in one volume can be copied or replicated to another volume, you need to compare the file system size of the two volumes.

Verifying traditional volume size

To compare the file system size of traditional volumes, complete the following steps.

Step	Action
1	On the source filer, enter the following command: **`vol status -b volume_name`** *volume_name* is the name of the source volume. **Result:** Data ONTAP displays the block size of the volume (in bytes), the RAID volume size, and the Write Anywhere File Layout (WAFL) file system size. If no volume name is given, information for all volumes is displayed.
2	On the destination filer, repeat Step 1, replacing *volume_name* with the name of the destination volume.
3	Compare the file system (FS) numbers. If the file system size of the destination is the same as or larger than the file system size of the source, you can use the `vol copy` command (or SnapMirror) to transfer data from the source to the destination.

Example:

```
vol status -b

Volume    Block Size (bytes)   Vol Size (blocks)   FS Size (blocks)
------    ------------------   -----------------   ---------------
sourcevol 4096                 4346752             4346752
destvol   4096                 4346752             4346752
```

Verifying flexible volume size

You can compare the size of volumes using the `vol status -b` command as described above, or you can use the `vol size` command. It might be more convenient to use `vol size` command, because you use this command to change the size of a flexible volume.

To compare the file system size of flexible volumes, complete the following steps.

Step	Action
1	On the source filer, enter the following command: **`vol size`** ***`volume_name`*** *volume_name* is the name of the source volume. **Result:** Data ONTAP displays the block size of the volume (in bytes), the RAID volume size, and the Write Anywhere File Layout (WAFL) file system size. If no volume name is given, information for all volumes is displayed.
2	On the destination filer, repeat Step 1, replacing *volume_name* with the name of the destination volume.
3	Compare the sizes. If the size of the destination is the same as or larger than the size of the source, you can use the `vol copy` command (or SnapMirror) to transfer data from the source to the destination. If the destination volume is smaller than the size of the source, increase the size of the destination volume. For information on increasing the size of a volume see the volume management chapter in the *Storage Management Guide*.

Example:

```
vol status -b

Volume    Block Size (bytes)   Vol Size (blocks)   FS Size (blocks)
------    ------------------   -----------------   ---------------
sourcevol 4096                 4346752             4346752
destvol   4096                 4346752             4346752
```

Verifying the relationship between filers

Creating trust relationships between filers

If the source and destination volumes in a volume copy operation reside on two different filers, the filers must have a trust relationship with each other.

To specify each filer as a trusted host of the other, complete the following steps.

Step	Action
1	By using FilerView or mounting the filer with NFS, enter the destination filer host name in the /etc/hosts.equiv file of the source filer, if it is not present already. The /etc/hosts.equiv file contains a list of host names, each of which is on a separate line. The presence of a host name in this file indicates that the filer allows that host to perform remote operations.
2	Repeat Step 1 on the destination filer, entering the source filer host name in the /etc/hosts.equiv file, if it is not present already.

Verifying and changing status of source and destination volumes

Verifying and changing volume status

To verify that the source volume is online and that the destination volume exists and is restricted, and to change the status of a volume when necessary, complete the following steps.

Step	Action
1	To verify that the destination volume exists and is restricted, enter the following command on the destination filer: **`vol status`** ***`dest_volume`*** *dest_volume* is the name of the volume whose status you want to check. If you do not provide a volume name, the command displays the status of all volumes in the filer. If the volume does not exist, Data ONTAP returns an error. See the *System Administrator's Guide* for information about how to create a volume. **Note** The destination volume cannot be the root volume. This is because the destination volume must be offline when Data ONTAP executes the `vol copy` command, and a root volume must always be online.
2	To verify that the source volume is online, repeat Step 1 on the source filer, replacing *dest_volume* with the name of the source volume.
3	If you need to change the status of a volume because of the results of Step 1, enter the following command on the destination filer: **`vol restrict`** ***`dest_volume`*** *dest_volume* is the name of the destination volume.

Step	Action
4	If you need to change the status of a volume because of the results of Step 2, enter the following command on the source filer: **`vol online`** ***`source_volume`*** *source_volume* is the name of the source volume.
5	If you needed to perform Step 3 or Step 4, you might want to perform Step 1 or Step 2 again to verify the changes that you made.

Example:

```
filerA> vol status

    Volume    State       Status     Options
      vol0    online      normal     root
      vol1    online      normal     raidsize=14
      vol2    online      restricted
  volextra    offline
```

Enabling remote access

Enabling Remote Shell services

To perform a volume copy from one volume to another volume on the same filer, Remote Shell services must be enabled or the volume copy fails.

To enable Remote Shell services, complete the following step.

Step	Action
1	Enter the following command: `options rsh.enable on`

Copying volumes

Command to use to copy volumes

You use the `vol copy start` command to generate volume copy operations, which produce screen messages that show the progress of the operations.

Each `vol copy start` command generates two volume copy operations, each of which is assigned a number:

- ◆ One operation is for reading data from the source volume. Screen messages displayed by a `vol copy` command refer to this operation as the `volcopy dump` operation.
- ◆ One operation is for writing data to the destination volume. Screen messages displayed by a `vol copy` command refer to this operation as the `volcopy restore` operation.

When to use the volume copy operation number

You need the volume copy operation number if you want to stop a volume copy operation or change the volume copy operation speed.

For information about obtaining the volume copy operation number, see "Checking the status of a volume copy operation" on page 295.

Number of vol copy operations supported

Each filer supports a limited number of simultaneous volume copy operations. For the number of copy operations supported per filer type, see "Maximum number of simultaneous replications on a filer" on page 81.

Whether Data ONTAP can execute a `vol copy start` command depends on how many volume copy operations are already in progress on the filers specified in the `vol copy start` command, as illustrated in the following examples.

Example: To copy volumes locally, you can enter the following two `vol copy start` commands on an F85 filer, which supports four simultaneous copy operations:

```
vol copy start vol0 vol1
vol copy start vol2 vol3
```

When these commands are in progress, if you enter additional `vol copy start` commands, they will fail, because four volume copy operations are already running on the filer. Two of the operations are for reading the vol0 and vol2 volumes, and two of the operations are for writing the vol1 and vol3 volumes.

Example: Suppose you enter the following three `vol copy start` commands on an F85 filer named filerA to copy volumes to another F85 filer named filerB:

```
vol copy start vol0 filerB:vol0
vol copy start vol1 filerB:vol1
vol copy start vol2 filerB:vol2
```

When these commands are in progress, filerA runs three volume copy operations to read the volumes, and filerB runs three volume copy operations to write the volumes.

An additional `vol copy start` command to copy *between* filerA and filerB will succeed because the command adds one more volume copy operation to each filer.

However, if you enter an additional `vol copy start` command to copy volumes *locally* on either filerA or filerB, it will fail. This is because the additional command creates two volume copy operations, one for reading and one for writing, on the filer that performs the local copying.

Copying snapshots with the vol copy start command

The following table describes the snapshots that will be copied from the source volume and the resulting snapshots on the destination volume, depending on the option you use with the `vol copy start` command.

Option	Snapshots to copy from the source volume	Snapshots in the snapshot file system of the destination volume
None	No snapshots are copied. Only the snapshot taken after you enter the `vol copy start` command, are copied.	A snapshot named snapshot_for_volcopy.*n* is created, where *n* is a number starting at 0 and incrementing by one whole number with each `vol copy` operation is created.
`-S`	All snapshots in the snapshot file system of the source volume, and the snapshot taken after you enter the `vol copy start command,` are copied.	All snapshots in the source volume, and snapshot_for_volcopy.*n*, where *n* is a number starting at 0 and incrementing by one whole number with each vol copy operation, are created.

Option	Snapshots to copy from the source volume	Snapshots in the snapshot file system of the destination volume
`-s` followed by the name of the snapshot	The specified snapshot will be copied.	The specified snapshot is created.

Note

The `vol copy start -S` command does not copy any snapshots that are created while the copying is in progress. For example, if the copying lasts from 11:45 p.m. to 1:00 a.m. the next day and Data ONTAP creates a snapshot named nightly.0 at midnight, Data ONTAP does not copy the nightly.0 snapshot.

Copying one volume to another

To copy one volume to another, complete the following step.

Step	Action
1	Enter the following command: **`vol copy start [-S	-s`** ***`snapshot_name`*** **`]`** ***`source_volume dest_volume`*** The `-S` and `-s` arguments specify the snapshots to copy. *source_volume* and *dest_volume* are the names of the source and destination volumes. If a volume is on a different filer, precede the volume name with the filer name and a colon. For examples illustrating how to specify volume names, see "Examples of the vol copy start command" in the following table. **Note** If the copying takes place between two filers, you can enter the `vol copy start` command on either the source or destination filer. You cannot, however, enter the command on a third filer that does not contain the source or destination volume.

Examples of the vol copy start command

The following table shows several examples of the `vol copy start` command.

If you want to...	Use...
Copy all snapshots from the vol0 volume to the vol1 volume on the same filer	`vol copy start -S vol0 vol1`
Copy a nightly snapshot from the vol0 volume to the vol1 volume on the same filer	`vol copy start -s nightly.1 vol0 vol1`
Create a snapshot in the vol0 volume to be copied to the vol1 volume on the same filer	`vol copy start vol0 vol1`
Copy all snapshots from the vol0 volume to the vol1 volume on a different filer named filerA	`vol copy start -S vol0 filerA:vol1`

Error messages generated by vol copy start commands

If your filer does not meet the requirements for copying a volume, the `vol copy start` command generates one or more error messages. The following table explains the possible error messages and their meanings.

Error message	Meaning

`Permission denied.`
 	
`VOLCOPY: Could not connect to filer filerB`
	The source filer does not have permission to copy to the destination filer.
	
	Action: Make sure that the filers have a trust relationship with each other.

`VOLCOPY: volcopy restore: volume is online, aborting`	The destination volume is online. **Action:** Take the destination volume offline.

Error message	**Meaning**
`VOLCOPY: volcopy restore: volume is too small, aborting`	The destination volume is smaller than the source volume. **Action:** Add more disk space to the destination volume or choose another destination volume of sufficient capacity.
`write: setting up STDERR broken pipe`	A local volume copy tried to start, but Remote Shell access is not enabled on the filer. **Action:** Enable Remote Shell access on the filer so that it can receive `rsh` commands.

Checking the status of a volume copy operation

Command to use to check status

You use the `vol copy status` command to check the status of volume copy operations.

This command displays the status for a specified volume copy operation. If you do not specify the operation number, the command displays the status of all volume copy operations in progress. In the command output, the operations are differentiated from one another with unique volume copy operation numbers.

Restrictions

Keep the following restrictions in mind when checking volume copy status:

- If you start a volume copy operation from the filer console, you can enter the `vol copy status` command only through the `rsh` command when the copy operation is in progress. This is because you do not have access to the filer prompt on the console when Data ONTAP is copying the volume.
- If data is being copied between two filers, you can enter the `vol copy status` command through a Remote Shell connection to either filer. The operation numbers displayed on the source filer and the destination filer are different because the reading and the writing are considered two different operations.

Checking operation status

To check the status of a volume copy operation in progress, complete the following step.

Step	Action
1	Enter the following command: **`vol copy status [`*`operation_number`*`]`** *operation_number* is the specific volume copy operation. Omit *operation_number* to display the status of all current volume copy operations. The operations are numbered from 0 through 3.

Sample status message from vol copy start command

The following example shows a `vol copy start` command that copies the vol0 volume to the vol1 volume on the same filer. When the operation is in progress, it displays the volume copy operation status.

```
filerA>vol copy start -S vol0 vol1
Copy Volume: vol0 on machine 127.0.0.1 to Volume: vol1
Reading the dump stream
VOLCOPY: Starting on volume 1.
This dump contains 257 blocks
10:04 pm : volcopy restore 1 : begun.
10:04 pm : volcopy restore 1 : 5 % done. Estimate 3 minutes
remaining.
.
.
.
10:04 pm : volcopy restore 1 : 95% done. Estimate 1 minutes
remaining.
```

Example of the vol copy status command using rsh

Before the prompt is displayed again, you can use the `vol copy status` command on a trusted host of the filer, as shown in the following example:

```
rsh filerA vol copy status
10:04 pm : volcopy dump 0 : 99 % done. Estimate 1 minutes remaining.
10:04 pm : volcopy restore 1 : 99 % done. Estimate 1 minutes
remaining.
No operation 2 in progress.
No operation 3 in progress.
```

In the previous examples, volume copy operation 0, shown as `volcopy dump 0` in the display, is for reading the data from the vol0 volume; volume copy operation 1, shown as `volcopy restore 1` in the display, is for writing the data to the vol1 volume.

Displaying the current speed for copying a volume

When to display the volume copy speed

You can display the speed for copying a volume when you want to determine the current setting, and to verify the speed before changing the setting. This procedure enables you to verify the default speed for all volume copy operations.

Displaying the default speed for copying a volume

To display the speed for copying a volume, complete the following step.

Step	Action
1	Enter the following command: `options vol.copy.throttle` **Result:** The value 10 (full speed) through 1 (one-tenth full speed) to be used by all volume copy operations is displayed. The default value is 10.

Controlling the speed of a volume copy operation

When to control volume copy speed

You might want to control the speed of a volume copy operation at two times:

- Before you start the volume copy operation
- During a volume copy operation

Note

The speed for reading data from the source volume and the speed for writing data to the destination volume can be different. The slower of the two values determines the time required for Data ONTAP to finish copying the data.

Why you change volume copy speed

You can change the speed of a volume copy operation when you suspect it might cause performance problems on your filer.

Note

Changing the `vol.copy.throttle` option changes the default speed for all volume copy operations to follow.

Controlling volume copy operation speed

To control volume copy operation speed, complete the following step.

Step	Action	
1	**If you want to control the speed of the volume copy...**	**Then...**
	Before you start the copy operations	Enter the following command: **`options vol.copy.throttle`** ***`value`*** *value* is the specific speed you want.

Step	Action	
	During the copy operation	Enter the following command through a Remote Shell: **`vol copy throttle [`*`operation_number`*`]`** ***`value`*** *operation_number* is the specific volume copy operation whose speed you want to adjust. If you do not specify an operation number, the command applies to all volume copy operations that are in progress. *value* is the specific speed you want.

Example

The following example illustrates changing the speed of all volume copy operations in progress to one-tenth of full speed through a Remote Shell:

```
rsh filerA vol copy throttle 1
volcopy operation 0: Throttle adjusted from 100% to 10%.
volcopy operation 1: Throttle adjusted from 100% to 10%.
```

Aborting a volume copy operation

About aborting a volume copy

If data is being copied between two filers, you can stop copying by executing the `vol copy abort` command on either filer.

If you start the volume copying operation from the filer console, you can enter the `vol copy abort` command only through the `rsh` command. This is because you do not have access to the filer prompt on the console during the copying.

Caution
An incomplete volume copy operation leaves unusable data in the destination volume.

Prerequisite

To abort a specific volume copy operation, you need to specify the volume copy operation number. You can obtain the operation number from the `vol copy status` output.

Aborting a volume copy

To abort a volume copy operation, complete the following step.

Step	Action
1	Enter the following command: **`vol copy abort [all	operation_number]`** *operation_number* is the specific volume copy operation to be aborted. Specify `all` to abort all operations.

SyncMirror Management 7

About this chapter

This chapter describes how to use the optional SyncMirror feature of Data ONTAP to mirror aggregates embedded in traditional volumes and aggregates that contain flexible volumes. It also describes how to manage aggregates in mirrored relationships. For more information about aggregates and volumes, see the *Storage Management Guide*.

Topics in this chapter

This chapter discusses the following topics:

Understanding mirrored aggregates

About the SyncMirror feature

The SyncMirror software creates aggregates or traditional volumes that consist of two copies of the same WAFL file system. The two copies, known as plexes, are simultaneously updated; therefore, the copies are always identical.

Advantages of SyncMirror

A SyncMirror relationship between two aggregates or traditional volumes provides a high level of data availability because the two plexes are physically separated on different shelves and the shelves are connected to the filer with separate cables and adapters. Each plex has its own collection of spare disks.

Physical separation of the plexes protects against data loss in the case of a double-disk error or loss of disk connectivity. The unaffected plex continues to serve data while you fix the cause of the failure. Once fixed, the two plexes can be resynchronized and the mirror relationship reestablished.

Another advantage of mirrored plexes is faster rebuild time.

In contrast, if a SnapMirrored aggregate or traditional volume goes down, its SnapMirror partner cannot automatically take over the file serving functions and can only restore data to its condition at the time the last snapshot was created (you must issue commands to make the partner's data available).

Disadvantage of SyncMirror

The disadvantage of SyncMirror is that a mirrored aggregate or traditional volume requires twice as many disks as an unmirrored aggregate or traditional volume. Each of the two plexes requires a full set of disks. For example, you need 200 GB of disk space to mirror a 100-GB traditional volume—100 GB for each plex of the mirrored traditional volume, or 2,880 GB of disk space to mirror a 1,440-GB aggregate.

Prerequisites for using mirrored volumes

The following are prerequisites for using mirrored aggregates or traditional volumes:

- ◆ You must purchase and enable a SyncMirror license.
- ◆ You must have an F800 series filer, F800 series cluster, or better.

Note

FAS200 series filers and NearStore storage systems do not support SyncMirror configurations.

- ◆ You must have DS14 or FC9 disk shelves.
- ◆ You must connect disk shelves to the filer in a configuration that supports mirrored volumes. For more information about connecting disk shelves to support mirrored aggregates or traditional volumes, see the *Storage Management Guide*, *System Configuration Guide*, *Cluster Guide*, *Fibre Channel DiskShelf14 Hardware Guide*, and *Fibre Channel StorageShelf FC9 Hardware Guide*.

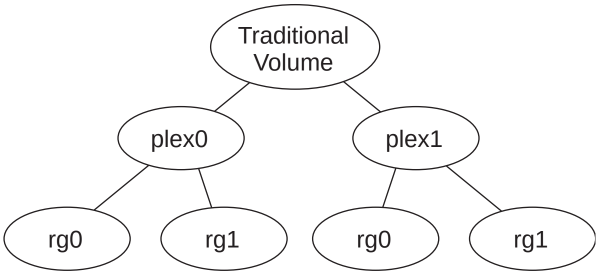
Note

The SyncMirror function is not supported on cluster configurations that use the Servernet cluster card.

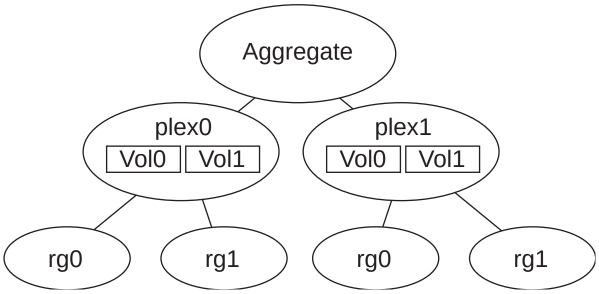
What mirrored aggregates and traditional volumes are

A mirrored aggregate or traditional volume is a single WAFL storage file system with two physically separated and synchronously up-to-date copies on disks. These copies are called plexes. Data ONTAP automatically names the first plex *plex0* and the second plex *plex1*.

Each plex is a physical copy of the same WAFL file system and consists of one or more RAID groups. Because SyncMirror duplicates complete WAFL file systems, you cannot use the SyncMirror feature with a flexible volume—only aggregates (including all contained flexible volumes) or traditional volumes are supported. The following diagram illustrates the concept of the plexes in a traditional SyncMirror relationship.



The following diagram illustrates the concept of the plexes in an aggregate SyncMirror relationship. Note that each plex includes a copy of the flexible volumes contained in the aggregate.



Example: The mirrored volume vol0 has two plexes, vol0/plex0 and vol0/plex1.

Where mirrored volume plexes get their disks

When a mirrored volume is created, Data ONTAP separates spare disks from a collection of disks, called a disk pool, into two disk pools, *pool0* and *pool1*. When assigning a disk to a pool, Data ONTAP determines the shelf for the disk and makes sure that the disks in pool0 are from different shelves than the disks in pool1. So before enabling SyncMirror, make sure disks are installed in at least two shelves and the shelves are connected to the filer with separate cables and adapters. Disk pools must be physically separate to ensure high availability of the mirrored volume.

Disks from pool0 are used to create plex0 while disks from pool1 are used to create plex1.

Plex pool assignment

Plexes local to the host node in a cluster must be connected to the disk pool named pool0. Pool0 consists of the storage attached to host adapters in slots 3 through 7. Pool rules for MetroCluster configurations that use switches are different. See the *System Configuration Guide* at now.netapp.com and the Cluster Guide for your filers for more information about filer slot assignments.

Enabling and disabling the mirroring license

Enabling the mirroring license

The license code name for the mirroring feature is syncmirror_local. To enable the mirroring license, complete the following steps.

Step	Action
1	Enter the following command: `license add xxxxxxx` *xxxxxxx* is the syncmirror_local license code you purchased.
2	Reboot the filer.

Disabling the mirroring license

You cannot disable the mirroring license if mirrored volumes or mirrored aggregates exist and are online. Before disabling the mirroring license, you must take one of the plexes offline for each mirrored volume or mirrored aggregate and destroy it.

Disabling if you have traditional volumes: To disable the mirroring license, complete the following steps.

Step	Action
1	For each mirrored volume, decide which plex you want to take offline. **Note** Every mirrored volume must have one plex taken offline and destroyed before you can disable the mirroring license.

Step	Action
2	Take offline each of the plexes you decided to take offline by entering the following command: **`vol offline`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes. **Note** Only one plex at a time can be taken offline.
3	Destroy the plex you took offline by entering the following command: **`vol destroy`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes.
4	Enter the following command: **`license delete syncmirror_local`**

Disabling if you have mirrored aggregates: To disable the mirroring license, complete the following steps.

Step	Action
1	For each mirrored aggregate, decide which plex you want to take offline. **Note** Every mirrored aggregate must have one plex taken offline and destroyed before you can disable the mirroring license.
2	Take offline each of the flexible volumes in the plexes you decided to take offline by entering the following command: **`vol offline`** ***`vol-name`*** *vol-name* is the name of the flexible volume. **Note** Only one flexible volume at a time can be taken offline.

Step	Action
3	Remove the flexible volumes from the plexes you decided to take offline by entering the following command: **`vol destroy`** ***`vol-name`*** *vol-name* is the name of the flexible volume. **Note** Only one flexible volume at a time can be taken offline.
4	Take offline each of the plexes you decided to take offline by entering the following command: **`aggr offline`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes. **Note** Only one plex at a time can be taken offline.
5	Destroy the plex you took offline by entering the following command: **`aggr destroy`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes.
6	Enter the following command: **`license delete syncmirror_local`**

Creating mirrored aggregates and traditional volumes

Two ways to create mirrored aggregates or traditional volumes

You can create a mirrored aggregate or traditional volume in the following ways:

- ◆ You can create a new aggregate or traditional volume that has two plexes.
- ◆ You can add a plex to an existing unmirrored aggregate or traditional volume.

Note
A mirrored aggregate or traditional volume cannot have more than two plexes.

How Data ONTAP selects disks

Regardless of which way you create a mirrored aggregate or traditional volume, when you create it or add disks to it, Data ONTAP determines what disks to use. Data ONTAP uses the following policies when selecting disks for mirrored aggregate or traditional volumes:

- ◆ Disks selected for each plex come from different disk pools.
- ◆ The number of disks selected for one plex must equal the number of disks selected for the other plex.
- ◆ Disks are selected first on the basis of equivalent bytes per sector (bps) size, then on the basis of the size of the disk.
- ◆ If there is no equivalent-sized disk, Data ONTAP takes a larger-capacity disk and downsizes it.

Note
When creating an unmirrored volume, Data ONTAP selects disks from the plex which has the most available disks. You can override this selection policy by specifying the disks to use.

Disk selection policies if you select disks

Data ONTAP allows you to select disks when creating mirrored aggregate or traditional volumes or adding disks to them. You need to follow the same disk selection policies that Data ONTAP follows when selecting disks for mirrored aggregate or traditional volumes. See the previous section, “How Data ONTAP selects disks” on page 308.

See "Creating a mirrored aggregate or traditional volume" on page 313 and "Adding a plex" on page 319 for information about creating mirrored aggregate or traditional volumes with disks you select.

Data ONTAP names plexes

Regardless of how a mirrored aggregate or traditional volume is created, Data ONTAP names the plexes of the mirrored aggregate or traditional volume. For more information about the plex naming convention, see "What mirrored aggregates and traditional volumes are" on page 303.

Viewing plex and disk pools

Viewing the disks in a plex or the spare disks in a disk pool is useful for adding disks to a plex or for understanding which plex is using which disk pool. To view plexes and disk pools, complete the following step.

Step	Action

1	Enter one of the following commands:
	`sysconfig -r`
	or
	`aggr status -r`
	or
	`vol status -r`

Example: In this example, the `aggr status -r` command is used to view the disks in plexes and spare disks in disk pools:

```
filer1> aggr status -r
Aggregate vol0 (online, raid4) (block checksums)
Plex /vol0/plex0 (online, normal, active, pool1)
RAID group /vol0/plex0/rg0 (normal)

RAID Disk Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks) Ph)
--------- ------  ------------- ---- ---- ---- ----- --------------    ---
parity    9a.16   9a    1   0   FC:A   1  FCAL 10000 34000/69632000    34
data      9a.17   9a    1   1   FC:A   1  FCAL 10000 600/1228800       76
data      9a.20   9a    1   4   FC:A   1  FCAL 10000 34000/69632000    34

Aggregate GreG (online, raid4) (block checksums)
Plex /GreG/plex0 (online, normal, active, pool1)
RAID group /GreG/plex0/rg0 (normal)

RAID Disk Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
--------- ------  ------------- ---- ---- ---- ----- --------------    ---
parity    9a.18   9a    1   2   FC:A   1  FCAL 10000 34000/69632000    34
data      9a.19   9a    1   3   FC:A   1  FCAL 10000 34000/69632000    34


Pool1 spare disks

RAID Disk       Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
---------       ------  ------------- ---- ---- ---- ----- --------------    ---
Spare disks for block or zoned checksum traditional volumes or aggregates
spare           9a.24   9a    1   8   FC:A   1  FCAL 10000 34000/69632000    34
spare           9a.29   9a    1   13  FC:A   1  FCAL 10000 34000/69632000    34

Pool0 spare disks (empty)

Partner disks

RAID Disk       Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
---------       ------  ------------- ---- ---- ---- ----- --------------    ---
partner         9b.25   9b    1   9   FC:B   1  FCAL 10000 0/0               34
partner         9b.16   9b    1   0   FC:B   1  FCAL 10000 0/0               34
partner         9b.17   9b    1   1   FC:B   1  FCAL 10000 0/0               34
partner         9b.21   9b    1   5   FC:B   1  FCAL 10000 0/0               34
partner         9b.18   9b    1   2   FC:B   1  FCAL 10000 0/0               34
partner         9b.22   9b    1   6   FC:B   1  FCAL 10000 0/0               34
tpubs-cf1>
```

Example: In this example, the `vol status -r` command is used to view the disks in plexes and spare disks in disk pools:

```
filer1> vol status -r
Volume vol0 (online, raid4) (block checksums)
  Plex /vol0/plex0 (online, normal, active, pool1)
    RAID group /vol0/plex0/rg0 (normal)

      RAID Disk Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
      --------- ------  ------------- ---- ---- ---- ----- --------------    ---
      parity    9a.16   9a    1   0   FC:A   1  FCAL 10000 34000/69632000    34
      data      9a.17   9a    1   1   FC:A   1  FCAL 10000 600/1228800       76
      data      9a.20   9a    1   4   FC:A   1  FCAL 10000 34000/69632000    34


Aggregate aggrz (online, raid4) (block checksums)
  Plex /aggrz/plex0 (online, normal, active, pool1)
    RAID group /aggrz/plex0/rg0 (normal)

      RAID Disk Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
      --------- ------  ------------- ---- ---- ---- ----- --------------    ---
      parity    9a.25   9a    1   9   FC:A   1  FCAL 10000 34000/69632000    34
      data      9a.26   9a    1   10  FC:A   1  FCAL 10000 34000/69632000    34
      data      9a.27   9a    1   11  FC:A   1  FCAL 10000 34000/69632000    34
      data      9a.28   9a    1   12  FC:A   1  FCAL 10000 34000/69632000    34


Pool1 spare disks
RAID Disk       Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
---------       ------  ------------- ---- ---- ---- ----- --------------    ---
Spare disks for block or zoned checksum traditional volumes or aggregates
spare           9a.24   9a    1   8   FC:A   1  FCAL 10000 34000/69632000    34
spare           9a.29   9a    1   13  FC:A   1  FCAL 10000 34000/69632000    34


Pool0 spare disks (empty)

Partner disks
RAID Disk       Device  HA  SHELF BAY CHAN Pool Type  RPM  Used (MB/blks)    Ph)
---------       ------  ------------- ---- ---- ---- ----- --------------    ---
partner         9b.25   9b    1   9   FC:B   1  FCAL 10000 0/0               34
partner         9b.16   9b    1   0   FC:B   1  FCAL 10000 0/0               34
partner         9b.17   9b    1   1   FC:B   1  FCAL 10000 0/0               34
partner         9b.21   9b    1   5   FC:B   1  FCAL 10000 0/0               34
partner         9b.18   9b    1   2   FC:B   1  FCAL 10000 0/0               34
partner         9b.22   9b    1   6   FC:B   1  FCAL 10000 0/0               34
partner         9b.23   9b    1   7   FC:B   1  FCAL 10000 0/0               34
```

For detailed information

The following sections discuss the ways you can create mirrored aggregate or traditional volumes:

- "Creating a mirrored aggregate or traditional volume" on page 313
- "Adding a plex" on page 319

Creating a mirrored aggregate or traditional volume

Methods for creating a mirrored volume

You can create a mirrored aggregate or traditional volume using one of the following methods:

- ◆ Let Data ONTAP select the disks used to mirror the aggregate or traditional volume.
- ◆ Select the disks used to mirror the aggregate or traditional volume yourself.
- ◆ Preview the disks Data ONTAP would use, then either use those disks or replace one or more of the disks with disks you choose.

Data ONTAP selects the disks

To create a mirrored aggregate or traditional volume by letting Data ONTAP select the disks used, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } create {`** ***`aggrname`*** **`|`** ***`volname}`*** **`-m`** ***`ndisks`*** **`[`** ***`@disk-size`*** **`]`** *aggrname* is the name for the new aggregate. *volname* is the name for the new traditional volume (without the /vol/ prefix). `-m` specifies a mirrored aggregate or traditional volume to Data ONTAP. *ndisks* is the number of disks to use; must be at least 4 and an even number. Even numbers are required because half of the disks specified by *ndisks* go to one plex from one disk pool and half go to the other plex from another disk pool. *disk-size* specifies the disk capacity, in gigabytes. There must be enough disks of this size available to both disk pools if you use this option. **Note** If you use the `disk-size` option, first determine the size of spare disks in the disk pools using the `vol status -r` command. **Example:** In this example, the following command creates a mirrored aggregate that has two plexes, each plex containing three disks: `aggr create aggrA -m 6`

You select the disks

If you select the disks used to create a mirrored aggregate or traditional volume, you must ensure that you choose a correct number and size of disks from each pool.

To create a mirrored aggregate using disks you select, complete the following step.

Step	Action
1	Enter the following command: **`aggr create`** ***`aggrname`*** **`-m -d`** ***`disk-list`*** **`-d`** ***`disk-list`*** *aggrname* is the name of the aggregate you are mirroring. `-m` specifies a mirrored aggregate to Data ONTAP. *disk-list* consists of the disk IDs of two or more available disks; separate multiple disks with a space. Both `-d` options must be used, one for each plex. **Note** Specifying only one disk set using the `-d` option will fail. **Example:** In this example, the following command creates a mirrored aggregate named aggrA with disks 6.1 and 6.2 on one plex and disks 8.1 and 8.2 on the other plex: `aggr create aggrA -m -d 6.1 6.2 -d 8.1 8.2`

To create a mirrored traditional volume using disks you select, complete the following step.

Step	Action
1	Enter the following command: **`vol create`** ***`volname`*** **`-m -d`** ***`disk-list`*** **`-d`** ***`disk-list`*** *volname* is the name of the volume you are mirroring (without the /vol/ prefix). `-m` specifies a mirrored volume to Data ONTAP. *disk-list* consists of the disk IDs of two or more available disks; separate multiple disks with a space. Both `-d` options must be used, one for each plex. **Note** Specifying only one disk set using the `-d` option will fail. **Example:** In this example, the following command creates a mirrored volume named vol A with disks 6.1 and 6.2 on one plex and disks 8.1 and 8.2 on the other plex: `vol create volA -m -d 6.1 6.2 -d 8.1 8.2`

You select the disks with assistance from Data ONTAP

You can preview what disks Data ONTAP would select if it were to create a mirrored aggregate or traditional volume.

To create a mirrored aggregate by previewing the disks Data ONTAP would use in the form of a sample command, complete the following steps.

Step	Action	
1	Enter the following command: **`aggr create`** ***`aggrname`*** **`-n -m`** ***`ndisks`*** **`[`*****`@disk-size`*****`]`** *aggrname* is the name of the mirrored volume you are creating. `-n` specifies to Data ONTAP not to create the mirrored volume, but to show the disks that it would use if it did create the mirrored volume. `-m` specifies a mirrored aggregate to Data ONTAP. *ndisks* is the number of disks to use. *ndisks* must be an even number. *disk-size* specifies the disk capacity, in gigabytes. You must have enough available disks of the size you specify. **Result:** Data ONTAP returns an `aggr create` command that specifies the disks.	
2	**If you want to...**	**Then...**
	Use the disks specified by Data ONTAP	At the prompt, enter the `aggr create` command specified.
	Substitute other disks for one or more disks specified by Data ONTAP	At the prompt, enter the `aggr create` command and substitute other disks for one or more of the disks specified.

Example: In this example, the following command previews the command that Data ONTAP would use when creating a 4-disk mirrored aggregate called aggrB:

```
aggr create aggrB -n -m 4
```

Data ONTAP returns the following command:

```
aggr create aggrB -m -d 5.1 5.3 -d 8.3 8.4
```

To create a mirrored traditional volume by previewing the disks Data ONTAP would use in the form of a sample command, complete the following steps.

Step	Action	
1	Enter the following command: **`vol create`** ***`volname`*** **`-n -m`** ***`ndisks`*****`[`*****`@disk-size`*****`]`** *volname* is the name of the mirrored volume you are creating (without the /vol/ prefix). `-n` specifies to Data ONTAP not to create the mirrored volume, but to show the disks that it would use if it did create the mirrored volume. `-m` specifies a mirrored volume to Data ONTAP. *ndisks* is the number of disks to use. *ndisks* must be an even number. *disk-size* specifies the disk capacity, in gigabytes. You must have enough available disks of the size you specify. **Result:** Data ONTAP returns a `vol create` command that specifies the disks.	
2	**If you want to...**	**Then...**
	Use the disks specified by Data ONTAP	At the prompt, enter the `vol create` command specified.
	Substitute other disks for one or more disks specified by Data ONTAP	At the prompt, enter the `vol create` command and substitute other disks for one or more of the disks specified.

Example: In this example, the following command previews the command that Data ONTAP would use when creating a 4-disk mirrored volume called volB:

```
vol create volB -n -m 4
```

Data ONTAP returns the following command:

```
vol create volB -m -d 5.1 5.3 -d 8.3 8.4
```

Adding a plex

Mirroring an unmirrored aggregate or traditional volume that uses mixed disk capacities

If the unmirrored volume that you want to mirror uses disks of different capacities, Data ONTAP can select disks that match the smallest capacity from a different disk pool. If there are not enough disks of that capacity in the disk pool, Data ONTAP selects a higher capacity disk and downsizes it.

See "How Data ONTAP selects disks" on page 308 for the rules Data ONTAP uses to select disks.

Methods for adding a plex to an unmirrored aggregate or traditional volume

You can add a plex to an existing unmirrored aggregate or traditional volume as follows:

- ◆ Let Data ONTAP select the disks used to mirror the volume.
- ◆ Select the disks used to mirror the aggregate or traditional volume yourself.
- ◆ Preview the disks Data ONTAP would use, then either use those disks or replace one or more of the disks with disks you choose.

Data ONTAP selects the disks

Letting Data ONTAP select the disks is the easiest method of mirroring, but you do not have control over which disks Data ONTAP selects.

To mirror an existing aggregate or traditional volume by letting Data ONTAP select the disks used to mirror the volume, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } mirror {`** ***`aggrname`*** **`|`** ***`volname`*** **`}`** *aggrname* is the name of the aggregate you are mirroring. *volname* is the name of the traditional volume you are mirroring (without the /vol/ prefix).

You select the disks

If you select the disks used to mirror the volume, you must ensure that you choose a correct number and size of disks from a different pool than that used by the aggregate or traditional volume plex that you are mirroring.

To mirror an existing aggregate or traditional volume using disks you select, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } mirror { aggrname	volname } -d disk-list`** *aggrname* is the name of the aggregate you are mirroring. *volname* is the name of the traditional volume you are mirroring (without the /vol/ prefix). *disk-list* consists of disk IDs of one or more available disks; separate multiple disks with a space.

You select the disks with assistance from Data ONTAP

You can preview what disks Data ONTAP would select if it were to mirror the aggregate or traditional volume.

To mirror an existing aggregate or traditional volume by previewing disks Data ONTAP would use in the form of a sample command, complete the following steps.

Step	Action
1	Enter the following command: **`{ aggr	vol } mirror { aggrname	volname } -n`** *aggrname* is the name of the aggregate you are mirroring. *volname* is the name of the traditional volume you are mirroring (without the /vol/ prefix). `-n` specifies to Data ONTAP not to create the mirrored aggregate or traditional volume, but show the disks that it would use if it did create the mirrored aggregate or traditional volume. **Result:** Data ONTAP returns an `aggr mirror` command or a `vol mirror` command that specifies the necessary number and type of disks you can use to mirror the aggregate or traditional volume.

Step	Action	
2	**If you want to...**	**Then...**
	Use the disks specified by Data ONTAP	Enter the `aggr mirror` command or `vol mirror` command specified.
	Substitute other disks for one or more disks specified by Data ONTAP	Enter the `aggr mirror` command or `vol mirror` command and substitute other disks for one or more of the disks specified.

Example: In this example, use the following command to preview the disks that Data ONTAP would use when mirroring the aggrA aggregate:

```
aggr mirror aggrA -n
```

Data ONTAP returns the following command:

```
aggr mirror aggrA -d 8.1 8.2 8.3
```

In this example, use the following command to preview the disks that Data ONTAP would use when mirroring the volA traditional volume:

```
vol mirror volA -n
```

Data ONTAP returns the following command:

```
vol mirror volA -d 8.1 8.2 8.3
```

Adding disks to mirrored aggregate or traditional volumes

Rules to follow when adding disks

When you add disks to a mirrored aggregate or traditional volume, you must follow these rules:

- The number of disks must be even, and the disks must be equally divided between the two plexes.
- The disks for each plex must come from different disk pools.
- The disks you add must have equivalent bytes per sector (bps) sizes.
- The disks must have capacity equal to or greater than existing disks in the plexes.

Note

If you add a larger capacity disk, Data ONTAP downsizes the larger disk to the size of the disks in the volume.

Methods for adding disks

You can add disks to mirrored aggregate or traditional volumes using the following methods:

- Let Data ONTAP select the disks.
- Select the disks yourself.
- Preview the disks Data ONTAP would use, then either use those disks or replace one or more of the disks with ones you choose.

Data ONTAP selects the disks

To add disks to a mirrored volume, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } add {`** ***`aggrname`*** **`|`** ***`volname`*** **`}`** ***`ndisks`*****`[@`*****`disk-size`*****`]`** *aggrname* is the name of the aggregate to which you are adding disks. *volname* is the name of the traditional volume to which you are adding disks. *ndisks* is the number of disks to use. Adding new disks will fail if the number of disks is an odd number. *disk-size* specifies the disk capacity, in gigabytes. Half these disks must be in each disk pool. **Note** If you use the `disk-size` option, first determine the size of spare disks in the disk pools using the `aggr status -r` or `vol status -r` command.

You select the disks

If you select the new disks to add to the mirrored aggregate or traditional volume, you must ensure that you add the correct number of disks of the correct sizes. The disks must have the same checksum compatibility, and disks for each plex must be in different pools.

To add new disks to a mirrored aggregate or traditional volume using disks you select, complete the following steps.

Step	Action
1	Enter the following command to list available spare disks: **`{ aggr	vol } status -r`**
2	Using the rules listed in "Rules to follow when adding disks" on page 322, choose the spare disks you want to add.

Step	Action

3	Enter the following command:
	
	`{ aggr | vol } add {` ***`aggrname | volname`*** **`} -d`** ***`disk-list`*** **`-d`** ***`disk-list`***
	
	aggrname is the name of the aggregate to which you are adding disks. *volname* is the name of the traditional volume to which you are adding disks (without the /vol/ prefix).
	
	disk-list consists of disk IDs of one or more available disks; separate multiple disks with a space. Both `-d` options must be used, one for each plex. Data ONTAP automatically assigns the disks to the appropriate plex.
	
	Note
	Specifying only one disk set using the `-d` option will fail.
	
	Example: In this example, use the following command to add disk 3.1 to one plex and disk 8.1 to the other plex of the aggrD mirrored aggregate:
	
	`aggr add aggrD -d 6.1 -d 8.1`
	
	Example: In this example, use the following command to add disk 3.1 to one plex and disk 8.1 to the other plex of the volD mirrored traditional volume:
	
	`vol add volD -d 6.1 -d 8.1`

You select the disks with assistance from Data ONTAP

You can preview what disks Data ONTAP would select if it were to add new disks to the aggregate or traditional volume.

To add new disks to an existing volume by previewing disks Data ONTAP would add in the form of a sample command, complete the following steps.

Step	Action	
1	Enter the following command: **{ aggr | vol } add { *aggrname* | *volname* } -n *ndisks*[*@disk-size*]** *aggrname* is the name of the mirrored aggregate to which you are adding disks. *volname* is the name of the mirrored traditional volume to which you are adding disks (without the /vol/ prefix). *ndisks* is the number of disks to use. Adding new disks will fail if the number of disks is an odd number. *disk-size* specifies the disk capacity, in gigabytes. Half these disks must be in each disk pool. **Note** If you use the `disk-size` option, first determine the size of spare disks in the disk pools using the `vol status -r` command. **Result:** Data ONTAP returns an `aggr add` or `vol add` command that specifies the disks it would add to each plex of the mirrored volume.	
2	**If you want to...**	**Then...**
	Use the disks specified by Data ONTAP	Enter the `aggr add` or `vol add` command specified.
	Substitute other disks for one or more disks specified by Data ONTAP	Enter the `aggr add` or `vol add` command and substitute other disks for one or more of the disks specified.

Example: In this example, the following command previews the disks that Data ONTAP would use when adding two new 36-GB disks to the aggrA mirrored aggregate:

```
aggr add aggrA -n 2@36
```

Data ONTAP returns the following command:

```
aggr add aggrA -d 6.4 -d 8.6
```

Example: In this example, the following command previews the disks that Data ONTAP would use when adding two new 36-GB disks to the volA mirrored volume:

```
vol add volA -n 2@36
```

Data ONTAP returns the following command:

```
vol add volA -d 6.4 -d 8.6
```

Changing the state of a plex

Plex states

A plex can be in one of the following states:

- Offline—The plex is not available for read or write access.
- Online—The plex is available for read or write access and the contents of the plex are current. An online plex can be in the following states:
    - Active—The plex is available for use.
    - Adding disks—Data ONTAP is adding disks to the RAID group or groups of the plex.
    - Empty—The plex is part of an aggregate or traditional volume that is being created and Data ONTAP needs to zero out one or more of the disks targeted to the aggregate or traditional volume before adding the disks to the plex.
    - Failed—One or more of the RAID groups in the plex failed.
    - Inactive—The plex is not available for use.
    - Normal—All RAID groups in the plex are functional.
    - Out-of-date—The plex contents are out of date and the other plex of the aggregate or traditional volume has failed.
    - Resyncing—The plex contents are being resynchronized with the contents of the other plex of the aggregate or traditional volume.

Viewing the status of plexes

To view the status of plexes, complete the following step.

Step	Action

1	Enter one of the following commands:
	`sysconfig -r`
	or
	`aggr status -r`
	or
	`vol status -r`

Changing the plex state

Data ONTAP specifies the state of a plex as resyncing when synchronizing the two plexes of a mirrored aggregate or traditional volume. For example, when you create a mirrored aggregate or traditional volume by adding a plex to an existing unmirrored aggregate or traditional volume, Data ONTAP puts the added plex in a resyncing state.

Data ONTAP allows you to change the state of a plex from offline to online and from online to offline.

To change the state of a plex, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr | vol } online | offline`** ***`plexname`***

Splitting mirrored aggregates or traditional volumes

What splitting a mirrored aggregate or traditional volume is

Splitting a mirrored aggregate or traditional volume removes the relationship between its two plexes and creates two independent unmirrored aggregates or traditional volumes. After splitting, the new aggregate or traditional volume is in a restricted state and the original aggregate or traditional volume stays in an online state.

Note

To mirror one or both of these aggregates or traditional volumes, see "Adding a plex" on page 319. To rejoin the aggregates or traditional volumes that you split, see "Rejoining split aggregates or traditional volumes" on page 331.

Why you might split a mirrored aggregate or traditional volume

You might split a mirrored aggregate or traditional volume for one of the following reasons:

- You want to stop mirroring an aggregate or traditional volume. This is an alternative to the procedure described in "Removing and destroying a plex" on page 333.
- You want to move a mirrored aggregate or traditional volume to another location.
- You want to modify the mirrored aggregate or traditional volume, and test the modification before applying it. You can apply and test the modifications and test on the split-off copy of the plex, then apply those changes to the untouched original plex.

Note

You do not need to stop applications that are running before splitting a mirrored aggregate or traditional volume.

Splitting a mirrored aggregate or traditional volume

To split a mirrored aggregate or traditional volume into two independent unmirrored aggregates or traditional volumes, complete the following step.

Note

Ensure that both plexes of the mirrored aggregate or traditional volume you are splitting are online and operational. For information about viewing the state of a plex, see "Changing the state of a plex" on page 327.

Step	Action

1	Enter the following command:
	`{ aggr | vol } split plexname { aggrname | volname }`
	plexname is the name of one of the plexes in the mirrored volume.
	aggrname is the name of the new aggregate. *volname* is the name of the new traditional volume.
	Example: In this example, use the following command to split plex0 from mirrored aggregate aggr0 and name the new aggregate aggrNew:
	`aggr split aggr0/plex0 aggrNew`
	After splitting, there are two unmirrored aggregates, aggr0 and aggrNew.
	Example: In this example, use the following command to split plex0 from mirrored volume vol0 and name the new volume volNew:
	`vol split vol0/plex0 volNew`
	After splitting, there are two unmirrored volumes, vol0 and volNew.

Volume structure after splitting

Before splitting, a mirrored aggregate or traditional volume has two plexes, plex0 and plex1. After splitting, the new unmirrored aggregate or traditional volume with the new name has one plex, plex0. The new unmirrored aggregate or traditional volume with the original name also has one plex, either plex0 or plex1.

The plex name for an unmirrored aggregate or traditional volume is unimportant because the aggregate or traditional volume has only one plex. If you remirror one of these unmirrored aggregates or volumes, the resulting plex names will always be plex0 and plex1.

Rejoining split aggregates or traditional volumes

Why you would rejoin split aggregates or traditional volumes

You might rejoin split aggregates or traditional volumes because you were forced to split them. This would be the case if you have configured a cluster in a MetroCluster configuration and a disaster breaks the cluster relationship . See "Recovering from a disaster" on page 383 for more information about rejoining aggregates or traditional volumes in this scenario.

Rejoining a split aggregate or traditional volume

When you rejoin a split traditional volume, Data ONTAP mirrors the data from one volume to the other and destroys data that existed on that volume before the rejoin. Likewise, when you rejoin a split aggregate, Data ONTAP mirrors the data from one aggregate to the other and destroys data that existed on that aggregate before the rejoining.

To rejoin a split aggregate or traditional volume, complete the following steps.

Step	Action
1	Determine the aggregate or traditional volume whose data you will keep and the aggregate or traditional volume whose data will be overwritten.
2	If the volume or aggregate whose data will be overwritten is online, take it offline by entering the following command for each volume or aggregate that was split: **`{aggr	vol} offline {`*`aggrname`*`	`*`volname`*`}`** *aggrname* is the name of the aggregate. *volname* is the name of the traditional volume. **Note** An error message appears if the aggregate or volume is already offline.

Step	Action

3	Re-create the mirrored aggregate or traditional volume by entering the following command for each aggregate or traditional volume that was split:
	
	`{aggr | vol} mirror {aggrname1 |volname1}`
	`-v {aggrname2 |volname2}`
	
	aggrname1 is the name of the aggregate whose data will be kept. *volname1* is the name of the traditional volume whose data will be kept.
	
	aggrname2 is the name of the aggregate whose data will be overwritten by *aggrname1*. *volname2* is the name of the traditional volume whose data will be overwritten by *volname2*.

Removing and destroying a plex

Why you might remove and destroy a plex

You might remove a plex from a mirrored aggregate or traditional volume and destroy it if there was a problem with that plex. For example, if you have a double-disk failure that causes a plex to fail, you can remove the plex from the mirrored aggregate or traditional volume, fix the problem, and then recreate it, or you can recreate it using a different set of disks if the problem cannot be fixed.

You might also remove a plex from a mirrored aggregate or traditional volume and destroy it if you want to stop mirroring the aggregate or traditional volume.

Removing and destroying a plex

You can remove and destroy a plex from a mirrored aggregate or from a traditional volume.

Destroying a plex from a traditional volume: To remove and destroy a plex from a traditional volume, complete the following steps.

Step	Action
1	Ensure that the plex you are removing is offline.
2	Enter the following step: **`vol destroy`** ***`plexname`*** *plexname* is the name of the plex you are removing.

Example: In this example, use the following command to remove and destroy the vol0/plex1 plex from the vol0 mirrored volume after taking the vol0/plex1 plex offline:

```
vol destroy vol0/plex1
```

Destroying a plex from a mirrored aggregate: Removing and destroying a plex from a mirrored aggregate requires the additional steps of taking flexible volumes in the plex offline and destroying them. To remove and destroy a plex from a mirrored aggregate, complete the following steps.

Step	Action
1	Take offline all of the flexible volumes in the plexes you decided to take offline by entering the following command: **`vol offline`** ***`vol-name`*** *vol-name* is the name of the flexible volume. **Note** Only one flexible volume at a time can be taken offline.
2	Remove the flexible volumes from the plexes you decided to take offline by entering the following command: **`vol destroy`** ***`vol-name`*** *vol-name* is the name of the flexible volume. **Note** Only one flexible volume at a time can be taken offline.
3	Take offline all of the plexes you decided to take offline by entering the following command: **`aggr offline`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes. **Note** Only one plex at a time can be taken offline.
4	Destroy the plex you took offline by entering the following command: **`aggr destroy`** ***`plex-name`*** *plex-name* is the name of one of the mirrored plexes.

Result of removing and destroying a plex

Removing and destroying a plex from a mirrored aggregate or traditional volume results in an unmirrored aggregate or traditional volume, because the aggregate or traditional volume now has only one plex.

After removing the plex, Data ONTAP converts the good disks used by the plex into hot spare disks.

Comparing the plexes of mirrored aggregate or traditional volumes

When you should compare plexes

Because the plexes of mirrored aggregate and traditional volumes are almost always synchronized, you rarely need to compare the plexes of a mirrored aggregate or traditional volume. If you do need to compare the plexes of a mirrored aggregate or traditional volume, Data ONTAP enables you to do the following tasks:

- Compare plexes

  Note
  Comparing plexes might affect filer performance.

- Stop comparing plexes
- Suspend and resume comparing plexes
- View the progress of the comparison

Comparing plexes

The mirrored aggregate or traditional volume must be online before you can compare the plexes. When comparing the two plexes of a mirrored aggregate or traditional volume, you can choose one of the following options:

- Data ONTAP compares plexes and corrects differences it finds. NetApp recommends this option.
- Data ONTAP compares plexes without correcting differences. If you choose this option, you must determine which plex is the good one and synchronize the other plex to it. This process can be difficult and can use advanced Data ONTAP commands. You must consult with NetApp Technical Support before correcting differences using this option.

To compare the two plexes of a mirrored aggregate or traditional volume, complete the following step.

Step	Action	
1	**If you...**	**Then...**
	Want Data ONTAP to correct differences	Enter the following command: **`{ aggr | vol } verify start { aggrname | volname }`** *aggrname* is the name of the mirrored aggregate whose plexes you are comparing. *volname* is the name of the mirrored volume whose plexes you are comparing. If neither *aggrname* nor *volname* is specified, the plexes of all online mirrored volumes are compared.
	Do not want Data ONTAP to correct differences	Enter the following command: **`{ aggr | vol } verify start { aggrname | volname } -n`** If neither *aggrname* nor *volname* is specified, Data ONTAP compares the plexes of a mirrored aggregate or traditional volume for all aggregates or traditional volumes.

Stopping plex comparison

You might need to stop Data ONTAP from comparing mirrored aggregate or traditional volume plexes. For example, you might stop verifying because it affects filer performance. To stop Data ONTAP from comparing plexes, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } verify stop {`** ***`aggrname`*** **`|`** ***`volname`*** **`}`** If *aggrname* or *volname* is not specified, Data ONTAP stops comparing plexes for all aggregates or traditional volumes.

Suspending and resuming plex comparison

Rather than stopping Data ONTAP from comparing the plexes of a mirrored aggregate or traditional volume, you can suspend the comparison of plexes. The comparison remains suspended until you resume it, stop it, or reboot the filer.

To suspend or resume a comparison of plexes, complete the following step.

Step	Action	
1	**If you want to...**	**Then...**
	Suspend the comparison	Enter the following command: **`{ aggr	vol } verify suspend {`** ***`aggrname`*** **`|`** ***`volname`*** **`}`** If *aggrname* or *volname* is not specified, Data ONTAP suspends the comparison of plexes for all aggregates or volumes.
	Resume the comparison	Enter the following command: **`{ aggr	vol } vol verify resume {`** ***`aggrname`*** **`|`** ***`volname`*** **`}`** If *aggrname* or *volname* is not specified, Data ONTAP resumes the comparison of plexes for all aggregates or volumes.

Viewing the status of a plex comparison

Plex comparison status tells you what percentage of the plex comparison has been completed and whether plex comparison of a mirrored aggregate or traditional volume is suspended.

To view the status of a plex comparison, complete the following step.

Step	Action
1	Enter the following command: **`{ aggr	vol } verify status { aggrname	volname }`** If *aggrname* or *volname* is not specified, Data ONTAP shows the status of all aggregates or volumes whose plexes are currently being compared.

Database Protection Using NVFAIL 8

About this chapter

This chapter describes how Data ONTAP provides database protection using the `nvfail` option.

Topics in this chapter

This chapter discusses the following topics:

Understanding the nvfail option

How Data ONTAP provides database file protection

Data ONTAP provides database file protection through the `nvfail` option of the `vol options` command. The `nvfail` option enables Data ONTAP to detect nonvolatile RAM (NVRAM) inconsistencies at boot time or while taking over in a cluster failover configuration.You use it to warn database administrators of NVRAM problems that can compromise database validity. If Data ONTAP finds any problems, database instances stop responding or shut down, and Data ONTAP sends error messages to the console to alert you to check the state of the database.

How to provide additional protection for database files

You can provide additional protection to specific database files by adding them to an optional file you create called /etc/nvfail_rename. When you enable `nvfail` and Data ONTAP detects NVRAM errors at boot time, Data ONTAP renames any database files specified in the nvfail_rename file by appending .nvfail to the original file name. When Data ONTAP renames a database file, the database cannot restart automatically. This gives you the opportunity to examine the file for inconsistencies before you remove the .nvfail extension and make the file accessible again.

How nvfail works

When you enable `nvfail`, one of the following processes takes place during bootup.

If Data ONTAP...	Then...
Detects no NVRAM errors	File service starts normally.

If Data ONTAP...	Then...
Detects NVRAM errors and you use the optional nvfail_rename file	**1.** Data ONTAP returns a stale file handle (ESTALE) error to NFS clients trying to access the database, causing the application to stop responding, crash, or shut down. Data ONTAP then sends an error message to the filer console and log file. **2.** Data ONTAP renames database files specified in the nvfail_rename file by appending .nvfail to the original file names, making those files unavailable to both CIFS and NFS clients.
Detects NVRAM errors and you do not use the optional nvfail_rename file	**1.** Data ONTAP returns a stale file handle (ESTALE) error to NFS clients trying to access the database, causing the application to stop responding, crash, or shut down. Data ONTAP then sends an error message to the filer console and log file. **2.** No database files are renamed. When the application restarts, files are available to CIFS clients, even if you have not verified that they are valid. For NFS clients, files remain inaccessible as long as the file system is not remounted.

Where to look for database file verification instructions

See the documentation for your specific database software for instructions about examining database file validity.

Enabling and disabling database file protection

Turning database protection on and off

To enable or disable the `nvfail` option, complete the following step.

Step	Action
1	Enter the following command: **`vol options`** ***`volume_name`*** **`nvfail [on|off]`** *volume_name* is the name of the volume. Use On to enable or Off to disable protection. The default setting is Off.

Using the nvfail_rename file for additional database protection

About the nvfail_rename file

This procedure enables you to use the optional nvfail_rename file when you want to rename database files after Data ONTAP detects NVRAM errors. This enables you to examine the files for consistency before clients can access them.

Creating the nvfail_rename file

To create the nvfail_rename file, complete the following steps.

Step	Action
1	Use an editor to create or modify the nvfail_rename file in the filer’s /etc directory.
2	List the path name and file name of database files you want to protect, one file per line, within the nvfail_rename file. You can list any number of files. **Example:** `/vol/vol1/home/dbs/oracle-WG73.dbf`
3	Save the file.

Virus Protection for CIFS 9

About this chapter

This chapter discusses the virus scan feature for CIFS and how to configure it to scan files on a NetApp filer.

Topics in this chapter

This chapter discusses the following topics:

Understanding CIFS virus protection

What CIFS virus protection is

CIFS virus protection is a Data ONTAP feature that allows a virus-scanning PC client running Network Appliance-compliant antivirus applications to provide on-access virus scanning of files on a filer. On-access virus scanning means that a file is scanned before a CIFS client is allowed to open it.

How CIFS virus scanning works

CIFS virus scanning is carried out on dedicated PC clients running the Network Appliance-compliant antivirus application of your choice. When you enable the virus-scanning process through Data ONTAP on the filer, the virus-scanning application tells the filer to send file scanning requests.

The virus-scanning application watches for requests from the filer. Whenever the types of files you specify are opened or changed on the filer, Data ONTAP sends the PC client a request to scan the file.

The Data ONTAP virus-scanning process can scan multiple filers from a single PC client if your virus-scanning application performs this function. For more information about whether a specific virus-scanning application can accommodate scanning multiple filers, contact the manufacturer of your virus-scanning application.

Obtaining Network Appliance-compliant anti-virus software

In order to successfully scan for viruses on the filer, the anti-virus application installed on your PC client must be a version customized for Network Appliance filers and supplied by one of Network Appliance's anti-virus software partners.

For a listing of the major software vendors who supply anti-virus software compliant with Network Appliance, see the *Partner Solutions Catalog, Data Protection* at the Network Appliance web site, http://www.netapp.com.

You can also see the *Antivirus Scanning Best Practices Guide* in the Windows section of the Network Appliance Technical Library at http://www.netapp.com.

Default file types that are scanned

By default, files with the extensions shown in the following table are scanned when CIFS virus scanning is enabled.

??_	DL?	IM?	OFT	SMM
ARJ	DOC	INI	OLE	SWF
ASP	DOT	JS?	OV?	SYS
BAT	DRV	LZH	PIF	VBS
BIN	EML	MD?	POT	VS?
CAB	EXE	MPP	PP?	VXD
CDR	GMS	MPT	RAR	WBK
CL?	GZ?	MSG	RTF	WPD
COM	HLP	MSO	SCR	XL?
CSC	HT?	OCX	SHS	XML

Note

In this table, the “?” character is a wildcard that matches any character or no character. For example,”C??” matches “C”, “CL”, “CPP”, “C++”, and so on.

Caution

Using “???” will cause all files to be scanned. This may severely degrade performance and is not recommended.

You can also configure Data ONTAP so that file extensions not in the default list are scanned, or only a subset of the default file extensions is scanned. For more information, see “Specifying file types to be scanned” on page 353.

Setting up and starting virus scanning

About virus-scanning clients

The commercial virus-scanning software compliant with Data ONTAP runs on selected PC clients with “Backup Operator” or higher user privilege connections to the target filers.

During installation of the commercial virus scan software on the selected PCs, you indicate the Data ONTAP filers to scan for viruses. Once the relationship is set up, the PCs running the virus scan software act as special *virus-scanning clients* that will virus scan files that are being opened by *other* CIFS clients and, if necessary, block the file operations and send a virus detection warning message to the requesting CIFS client.

Specifying the active virus-scanning clients

To set up PC clients as virus-scanning clients, complete the following steps.

Step	Action
1	If you have not already done so, obtain Network Appliance-compliant virus scan software described in “Obtaining Network Appliance-compliant anti-virus software” on page 348.
2	Make sure that the operators of the PC clients that you want to configure as virus-scanning clients are configured as “Backup Operators” or higher on the filers on which they will conduct virus scanning.
3	Install the NetApp-customized commercial virus scan software on the PCs that you want to configure as virus-scanning clients. Follow the directions that accompany the virus scan software product.
4	After installation and configuration of the virus scan software on the PCs is complete, confirm the success of the installation by listing the IP addresses of the PCs now configured as virus-scanning clients. At the filer console enter the following command: `vscan scanners` **Result:** The system displays a table listing the IP addresses of the active virus-scanning clients for this filer.

Step	Action
5	Leave the virus-scanning client on and connected to the filer or filers on which it is carrying out its virus scan operations.

Enabling and disabling virus scanning

To enable and disable virus scanning, complete the following step.

Step	Action
1	Enter the following command: **`vscan on [-f][on|off]`** `-f` forces virus scanning to be enabled even if no virus-scanning clients are available to scan files.

Designating secondary scanning clients

If you have more than one virus-scanning client configured to scan files on a filer, you can use the `vscan scanners secondary_scanners` command line to designate one or more of the virus-scanning clients as standby virus scanners. Standby virus scanners do not actively scan for viruses unless the other virus-scanning clients go down. At that point the secondary scanners take over the virus-scanning operation.

To set up secondary virus scanning, complete the following steps.

Step	Action

1	List the virus-scanning clients configured to scan on this filer. In the filer console enter the following command:
	`vscan scanners`
	Example:
	`>vscan scanners`
	`Virus scanners(IP and Name)      P/S ...`
	`----------------------------------------`
	`132.132.59.12   \\XLAB-WTS      Pri ....`

Step	Action
2	Specify, by IP address, the PC clients you want to serve as standby virus scanners by entering the following command: **`vscan scanners secondary_scanners`** ***`scanner_ip`*`[,`*`scanner_ip`*`...]`** *scanner_ip* can be either of the following: ◆ IP addresses of one or more of the configured virus-scanning clients displayed in Step 1 ◆ IP addresses of PCs not yet configured as a virus-scanning client **Note** If the IP address you entered belongs to a PC not yet configured as a virus-scanning client for this filer, you must configure it for this setting to take effect. **Example:** `vscan scanners secondary_scanners 132.132.59.14`

3	Use the `vscan scanners` command to confirm your configuration.
	
	Example:
	`>vscan scanners`
	`Virus scanners(IP and Name)      P/S ...`
	`----------------------------------------`
	`132.132.59.14   \\BORIS-PC      Sec ...`
	`132.132.59.12   \\XLAB-WTS      Pri ....`
	
	`Secondary scanners IP address list`
	`132.132.59.14,10.20.30.40`
	
	Note
	In this example, the address 10.20.30.40 belongs to a PC that is enabled as a standby scanner but is not turned on; therefore it is not listed in the Virus scanners (IP and Name) table, but it is listed in the Secondary scanners IP address list.

Specifying file types to be scanned

About specifying file types

Use the `vscan extensions include` command line to list, specify, add to, or remove from the set of file types to be scanned on the filer.

Displaying files to be scanned

To display the list of extensions of file types to be scanned on the filer, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions include`** **Result:** The current list of extensions to be scanned is displayed.

Adding file types to be scanned

A default list of file extensions is made available when you enable virus scanning; however, you can specify additional file extensions not in the default list.

To add the extensions of file types to be scanned, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions include add`** ***`ext`*****`[,`*****`ext...`*****`]`** *ext* is the extension you want to add.

Example: `vscan extensions include add txt`

Note
Up to 255 file extensions can exist in the file extensions list.

Replacing the file types in the extensions list

You can replace the current list of file extensions with a new list of extensions that you choose.

To replace the `vscan` extensions list with a new list, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions include set`** ***`ext`*****`[,`*****`ext`*****`...]`** *ext* is the extension you want to set.

Removing file types from the extensions list

To remove one or more file types from the extensions list, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions include remove`** ***`ext`*****`[,`*****`ext`*****`...]`** *ext* is the extension you want to remove.

Resetting the file extensions list to the default

To reset the file extensions list to the default, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions include reset`** **Result:** The list of file extensions is set to the default list. See "Default file types that are scanned" on page 349.

Excluding file types to be scanned

About excluding file types

Some administrators, because of the proliferation of new file types (with new file name extensions) that might be stored on the filer, prefer to specify what file types to exclude, rather than include, in a virus scan. This practice allows you to pass file types assumed to be safe, at the same time ensuring that any new, unfamiliar file type stored on the filer does get scanned.

Use the `vscan extensions exclude` command line to list, specify, add to, or remove from the set of file types to be excluded from virus scan on the filer.

Typical vscan extensions exclude usage

Typically, you will use the `vscan extensions exclude` command in combination with the `vscan extensions include` command. You use `include` to specify a general virus scan, and use `exclude` to specify the file types to be excepted from the virus scan.

To exclude file types from virus scanning, complete the following steps.

Step	Action
1	At the filer console enter the following command line specifying the extensions of all file types that you want to exclude from virus scan: **`vscan extensions exclude set`** ***`ext`*** **`[,`*`ext...`*`]`**
2	Enter the following command line to specify virus scan of all other file types: **`vscan extensions include set ???`** **Result:** This command line instructs the virus scan program to scan all file types stored on the filer. But the result of both the “exclude” and the “include” command lines together is that all file types are scanned except for the file types whose extensions have been specified in the `vscan extensions exclude` command line described in Step 1.

Displaying files to exclude from a scan

To display the list of extensions of file types to be excluded from a virus scan on the filer, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions exclude`** **Result:** The current list of extensions to be excluded from virus scan is displayed.

Specifying all the file types to exclude from a scan

To specify the list of extensions of file types that you want to exclude from virus scan, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions exclude set`** ***`ext`*****`[,`*****`ext`*****`...]`** *ext* is the extension or extensions that you want to set for the list of file types excluded from virus scan. **Note** Using the `set` parameter will replace completely any existing file type extensions in the exclude list with the extensions you specified in this command. If an exclude list already exists and you merely want to add to it, use the `vscan extensions exclude add` command.

Adding file types to exclude from a scan

To add the extensions of file types to be excluded from a virus scan, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions exclude add`** ***`ext`*****`[,`*****`ext...`*****`]`** *ext* is the extension you want to add to the list of file types excluded from virus scan.

Example: `vscan extensions exclude add txt`

Note
Up to 255 file extensions can exist in the file extensions list.

Removing file types to exclude from a scan

To remove one or more file types to exclude from virus scan, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions exclude remove`** ***`ext`*****`[,`*****`ext`*****`...]`** *ext* is the extension you want to remove from the list of file types excluded from virus scan.

Resetting the exclude file extensions to empty

To reset the file extensions list to the default, which is empty, complete the following step.

Step	Action
1	Enter the following command: **`vscan extensions exclude reset`** **Result:** The list of file extensions is set to the default empty value.

Exclude over include

If an extension for a file type is unintentionally entered in both the vscan include list (through the `vscan extensions include` command) and the vscan exclude list (through the `vscan extensions exclude` command, that file type will be excluded from the virus scan.

Specifying shares for scanning

About specifying shares for scanning

You can turn scanning on or off for shares you specify, either for any access or for read-only access. You can also create shares that you specify are or are not to be scanned, upon either any access or read-only access.

For detailed information

The following sections discuss the ways to turn scanning on and off for shares you specify or create:

- "Turning scanning on or off for shares" on page 360
- "Turning scanning on or off for read-only access for shares" on page 362
- "Adding shares with virus scanning turned on or off" on page 363

Turning scanning on or off for shares

Why you turn virus scanning off for a specified share

You might want to turn virus scanning off for files in a share if it is used only by trusted users, the files are restricted to read-only mode, or speed of access is more important than safety. Virus scanning for a share is turned on by default.

How to turn virus scanning off for shares

To turn virus scanning off for files in a specified share, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -change`** ***`share_name`*** **`-novscan`** *share_name* is the name of the share for which you want to turn off virus scanning. **Result:** The application does not perform a virus scan when clients open files on this share. The setting is persistent after reboot.

How to turn virus scanning on for shares

To turn virus scanning on for files in a specified share, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -change`** ***`share_name`*** **`-vscan`** *share_name* is the name of the share for which you want to turn on virus scanning. **Result:** The application does a virus scan when clients open files on this share. The setting is persistent after reboot.

Note

You can set these share attributes for all CIFS user home directories by using `cifs.homedir` as the share name, for example, `cifs shares -change cifs.homedir -novscan`

Turning scanning on or off for read-only access for shares

Why you turn virus scanning off for read-only access for shares

You might want to turn virus scanning off in a share for users who are opening files for read-only access. This increases the speed of accessing the files. A share has virus scanning turned on by default.

How to turn virus scanning off for shares

To turn virus scanning off for read-only access to files in a specified share, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -change`** ***`share_name`*** **`-novscanread`** *share_name* is the name of the share for which you want to turn off virus scanning. **Result:** The application does not perform a virus scan when clients open files on this share for read access. The setting is persistent after reboot.

How to turn virus scanning on for shares

To turn virus scanning on for read-only access for files in a specified share, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -change`** ***`share_name`*** **`-vscanread`** *share_name* is the name of the share for which you want to turn on virus scanning. **Result:** The application does a virus scan when clients open files on this share for read access. The setting is persistent after reboot.

Adding shares with virus scanning turned on or off

Why you add shares with virus scanning turned off

You might want to create a share with virus scanning turned off. A share has virus scanning turned on by default.

How to add a share with virus scanning turned off

To add a share that has virus scanning turned off, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -add`** ***`share_name /path`*** **`-novscan`** *share_name* is the name of the share with virus scanning turned off that you want to create. *path* specifies where you want the share created. **Result:** Data ONTAP creates a share with virus scanning turned off.

How to add a share with virus scanning turned off for read-only access

To create a share with virus scanning turned off for read-only access, complete the following step.

Step	Action
1	Enter the following command: **`cifs shares -add`** ***`share_name /path`*** **`-novscanread`** *share_name* is the name of the share with virus scanning turned off for read-only access that you want to create. *path* specifies where you want the share created. **Result:** Data ONTAP creates a share with virus scanning turned off for read-only access.

Note

For backup purposes, you can create two shares on the same directory: one share with scanning disabled and a share-level Access Control List (ACL) that allows access only to a backup account; the other share available to normal users and with scanning enabled. Backup can be performed on the share with no virus scanning and improved performance, while normal users continue to access the data through the regular share and get virus protection.

Displaying the scanner list

Displaying the scanner PC list

The scanner list identifies all scanner clients that are currently performing virus scanning for a filer. To display the scanner list, complete the following step.

Step	Action
1	Enter the following command: **`vscan scanners`** **Result:** The filer outputs the names of and information about the virus-scanning clients in a display like the one shown in the following paragraph.

```
Virus scanners(IP and Name)   Connect time (dd:hh:mm)  Reqs  Fails
------------------------------------------------------------------
10.61.155.118  \\WIN2K-NAN    00:02:23                 2     0
10.60.129.152  \\WIN2K-RTB    00:00:01                 0     0
```

The *Connect time* field displays the length of time the scanner has been connected to the filer.

The *Reqs* field displays the total number of requests sent by the filer to that scanner.

The *Fails* field displays the number of requests that failed.

Checking vscan information

How to check current vscan settings and information

You can check whether vscan is on, the current list of file extensions, the names of and information about the virus-scanning clients, the list of extensions to scan, the files scanned, and the number of scan failures on a filer.

To check the status of the vscan appliance, complete the following step.

Step	Action
1	Enter the following command: **`vscan`** **Result:** The filer outputs the on/off status of vscan, information about the virus-scanning clients, the list of extensions to scan, and the number of files scanned and number of scan failures in a display like the one shown in the following paragraph.

```
filerA> vscan

Virus scanning is enabled.

Virus scanners(IP and Name)  Connect time (dd:hh:mm)  Reqs    Fails
------------------------------------------------------------------
10.61.155.118  \\WIN2K-NAN    00:02:23                 2     0
10.60.129.152  \\WIN2K-RTB    00:00:01                 0     0

List of extensions to scan:
??_,ASP,BAT,CDR,COM,CSC,DL?,DOC,DOT,EXE,GMS,GZ?,HLP,HT?,IM?,INI,JS
?,MD?,MPP,MPT,MSG,MSO,OCX,OLE,OV?,POT,PP?,RTF,SCR,SHS,SMM,SYS,VBS,
VS?,VXD,WBK,WPD,XL?,XML

List of extensions not to scan:

Number of files scanned: 28
Number of scan failures: 0
```

Note

The number of scan requests and failures shown in the table at the beginning of the output represents this connection session. The second set of numbers at the end of the output reflects the total scans and failures since vscan was turned on.

Setting and resetting the virus scan request timeout

About the timeout setting

The virus scan timeout setting determines how long Data ONTAP will wait for a scanner to complete file scanning before requesting the status of the scan. The request will be repeated as often as necessary until the scan is complete or the host gives up. By default, the virus scan timeout is 10 seconds.

Note

Prior to Data ONTAP 6.4.3, the default timeout value was 30 seconds. Filers that you upgraded from pre-Data ONTAP 6.4.3 releases will still have the 30 second timeout after the upgrade.

Setting the virus scan request timeout

To set the virus scan request timeout to something other than the default, complete the following step.

Step	Action
1	Enter the following command: **`vscan options timeout set`** ***`value`*** *value* is a setting from 1 to 45 seconds. NetApp recommends a setting between 8 and 12 seconds.

Resetting the virus scan request timeout

To reset the virus scan timeout back to the default, complete the following step.

Step	Action
1	Enter the following command: **`vscan options timeout reset`**

Allowing file access when the scan cannot be performed

About the mandatory_scan option

You can specify whether files can be accessed if the scan cannot be performed by using the `mandatory_scan` option of the `vscan` command. The default setting is On. When this option is set to On, file access is denied if a successful scan cannot be performed. When this option is set to Off, access is allowed even if no successful scan can be performed.

A scan might not be able to be performed if no scanner is available, if the request times out, or for other reasons.

Enabling and disabling mandatory scanning

To enable and disable mandatory file scanning, complete the following step.

Step	Action
1	Enter the following command: **`vscan options mandatory_scan [on|off]`**

Controlling vFiler usage of host filer virus scanners

What a vFiler is

The vFiler™ feature of the MultiStore™ software product is created by partitioning the storage and networking resources of a single filer so that it appears as multiple filers on the network. Each “filer” created as a result of the partitioning is called a vFiler, which stands for virtual filer. A vFiler, using the resources assigned, delivers file services to its clients as a filer does. For more information about vFilers, see the *MultiStore Administration Guide.*

About controlling virus scanner usage

By default, vFilers can scan files using the virus scanners that are connected to the host filer. You can register a virus scanner in one of two ways:

- The virus scanner registers with each vFiler individually. In this way, each vFiler authenticates the virus scanner and the vFiler appears as a separate machine to the virus scanner.
- The virus scanner registers with only the physical filer. In this way, only the filer authenticates the virus scanner.

Note
The virus scanner accesses all files through the filer, not through each vFiler.

If you have several vFilers, each owned by a separate organization, you might want to have the virus scanner register with each vFiler if you have security concerns.

You can control whether vFilers are allowed to use the host filer’s virus scanners by using the `use_host_scanners` option of the `vscan` command.

Enabling and disabling host scanner usage for vFilers

To enable or disable a vFiler's ability to use the virus-scanning clients recognized by the host filer, complete the following step.

Step	Action
1	From the vFiler console, enter the following command: **`vscan options use_host_scanners [on|off]`** **Note** This command does not run on the default vFiler.

Checking the status of virus-scanning options

Why to check the status of virus-scanning options

You check the status of the virus-scanning options to determine whether the options you want to use are set to the values you want.

How to check the status of virus-scanning options

To check the status of virus-scanning options, complete the following step.

Step	Action

1	Enter the following command:
	`vscan options`
	Result: The filer outputs the state of the virus-scanning options in a display like the following:
	`vscan options mandatory_scan     on`
	`vscan options timeout:           12 sec`
	`vscan options use_host_scanners  on`

Note

The `use_host_scanners` option applies only to vfilers and is displayed only if the `vscan options` command was run on a vfiler.

Stopping a virus scanner session

Reason for stopping a virus scanner session

At some time you might want to stop a virus scanner session. For example, you might want to stop a scanner session when you have to terminate CIFS on a filer or upgrade your antivirus program to a new version.

Stopping a specific scanner session

To stop a specific virus scanner session, complete the following step.

Step	Action
1	Enter the following command: **`vscan scanners stop`** ***`scanner_IP`*** *scanner_IP* is the IP address of the virus scanner you want to stop.

Resetting the scanned files cache

When to reset the scanned files cache

Data ONTAP caches information about previously scanned files to avoid rescanning those files. When you have a new virus-scanning signature file, you might want to reset the cache to rescan files that were scanned using an old signature file.

Resetting the cache

To reset the scanned files cache on a filer, complete the following step.

Step	Action
1	Enter the following command: **`vscan reset`**

Enabling virus scan messages to CIFS clients

Virus scan messages

You can enable Data ONTAP to return an explicit virus warning messages to clients if the virus-scanning application running on the filer detects that a file that CIFS client is trying to access is infected with a virus.

If this feature is not enabled, CIFS clients attempting to access virus-infected files that have been detected by virus scanning will simply receive a general “file unavailable” message.

Enabling the virus scan message

To enable the display of explicit virus warning messages to CIFS clients, complete the following step.

Step	Action
1	Enter the following command: **`vscan options client_msgbox {on|off}`** `on` enables the display of virus warning messages. `off` disables the display of virus warning messages.

Disaster Protection Using MetroCluster A

About this appendix

This appendix describes how to use the optional MetroCluster feature of Data ONTAP to protect data that requires continuous backup from loss and unavailability due to situations such as prolonged power outages or natural disasters.

Note

This appendix describes protecting data on Network Appliance filers. For information about protecting data on other networked equipment—application servers, for example—see the documentation for that equipment.

Topics in this appendix

This appendix discusses the following topics:

Configuring disaster recovery using MetroCluster

Definition of terms

This appendix uses the following terms when describing disaster recovery using filers in a MetroCluster configuration:

- Cluster—Two filers that are connected in a cluster configuration
- Cluster node—One of the filers in the cluster
- Local cluster node—The cluster node in the same area as the system administrator
- Remote cluster node—The cluster node in the area affected by the disaster
- Remote storage—The storage that is accessible to the local cluster node, but is located at the remote cluster node site
- Site—The area in which a cluster node is located, for example, a building

Mirrored volume configuration

The MetroCluster configuration employs SyncMirror and a cluster configured for mirrored volumes to build a system that provides data even after complete loss of one of the cluster nodes and the storage at that site. Mirrored volumes, both root volume and data volumes, must be mirrored with one plex at each site to preserve data using MetroCluster.

Caution

The MetroCluster configuration can preserve data only if volumes are mirrored. Unmirrored volumes are not accessible if a disaster occurs; data can be lost.

MetroCluster configurations

There are two MetroCluster configurations. One configuration directly connects the two cluster nodes to each other. The other configuration uses Brocade switches to connect the two cluster nodes. The configuration you use depends on the distance between the two cluster nodes. The following sections discuss the two MetroCluster configurations:

- “MetroCluster configuration up to 500 meters” on page 377
- “MetroCluster configuration for distances up to 30 kilometers” on page 379

MetroCluster configuration up to 500 meters

Configuration overview

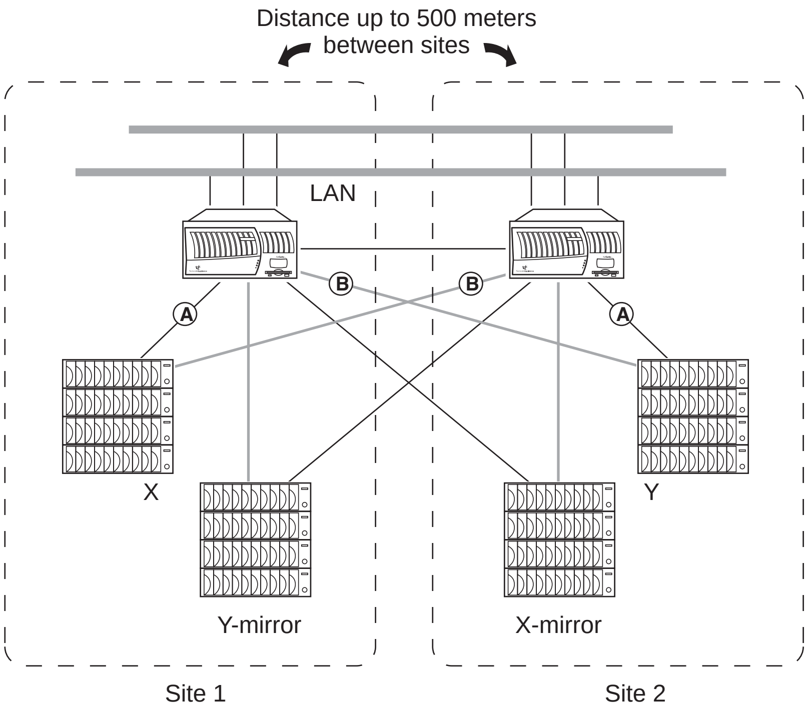
The cluster configuration for distances of up to 500 meters is the standard SyncMirror configuration for clusters using 500-meter interconnect and storage cables between the two cluster nodes.

Note

The distance between cluster nodes is reduced to 300 meters if 2-Gb Fibre Channel is used between the filer and the storage.

See the cluster guide for your filers and "SyncMirror Management" on page 301 for detailed information about configuring clusters and using SyncMirror to mirror volumes, respectively.

The following figure illustrates the MetroCluster configuration.



Configuration requirements for distances under 500 meters

The MetroCluster configurations require the following:

- F800 series or FAS 900 series cluster pair configured for mirrored volume use
- The following licenses:
    - Cluster—Enables the two nodes to communicate and take over each other's functions in the event of a failure.
    - Syncmirror_local—Enables mirrored volumes.
    - Cluster_remote—Enables the local node to take over the functions of the remote node in the event of a failure.

MetroCluster configuration for distances up to 30 kilometers

Configuration overview

The cluster configuration for distances between 500 meters and 30 kilometers uses four Brocade switches in a fabric configuration to connect the two cluster nodes.

See the installation and configuration information in the cluster guide for your filers and “SyncMirror Management” on page 301 for detailed information about configuring clusters in a MetroCluster configuration using switches in a fabric configuration and using SyncMirror to mirror volumes, respectively.

Configuration requirements for distances between 500 meters and 30 kilometers

The MetroCluster configuration using switches in a fabric configuration requires the following:

- F825, F880, or FAS900 series cluster pair configured for mirrored volume use
- All cluster pairs require a QLA2352 VI-MC cluster adapter.
- The following licenses:
  - Cluster—Enables the two nodes to communicate and take over each other’s functions in the event of a failure.
  - Syncmirror_local—Enables mirrored volumes.
  - Cluster_remote—Enables mirrored volumes over long distances and enables the local node to take over the functions of the remote node in the event of a failure.
- Four Brocade Silkworm 3200 or 3800 switches, or a combination of both, that have the following:
  - Firmware version 3.0.2p

  Note
  You can download Brocade switch firmware and obtain details about configuring switches at now.netapp.com.

  - Long distance Small Form-factor Pluggable (SFP) transceivers

  Note
  You can use a combination of two 3200 switches and two 3800 switches; however, the two switches at one site must be the same type. For example, at one site you might use two 3200 switches, and at the other site you might use two 3800 switches.

MetroCluster upgrade from QuickLoop to fabric: If you are upgrading an existing cluster to a MetroCluster configuration, you must upgrade disk firmware to the latest version. After upgrading disk firmware, you must power-cycle the affected disk drives before they work correctly in a MetroCluster fabric configuration. You can download the latest disk firmware from now.netapp.com.

Configuration limitations for distances between 500 meters and 30 kilometers

The MetroCluster configurations have the following limitations:

- The switches used in a MetroCluster configuration cannot be used by Fibre Channel tape devices. Only filers and disk shelves can be connected to the switches in a MetroCluster configuration.
- A tape storage area network (SAN) can be connected to either of the filers in a MetroCluster configuration, but the tape SAN must not use the switches used in the MetroCluster configuration.

Recognizing a disaster

Failures that require disaster recovery

The disaster recovery procedure is an extreme measure that should be used only if the failure disrupts all communication from one MetroCluster site to the other for a prolonged period of time.

The following are examples of disasters that could cause such a failure:

- Fire
- Earthquake
- Prolonged power outages at a site
- Prolonged loss of connectivity from clients to the filers at a site

Ways to determine whether a disaster occurred

It is critical that you follow a predefined procedure to confirm that a disaster has occurred. The procedure should include determining the status of the disaster site by

- Using external interfaces to the filer, such as the following:
    - Ping
    - Remote shell
    - FilerView
- Using network management tools to verify connectivity to the disaster site
- Physically inspecting the disaster site, if possible

You should declare a disaster only after verifying that service cannot be restored.

Failures that do not require disaster recovery

If you can reestablish the cluster relationship after fixing the problem, you should not perform the disaster recovery procedure.

Do not perform the disaster recovery procedure for the following failures:

- A failure of the cluster interconnect between the two sites. This can be caused by the following:
    - Failure of the interconnect cable
    - Failure of one of the VI cluster adapters
    - If using switches, a failure of the SFP connecting a cluster node to the switch

  With this type of failure, both nodes of the cluster remain running. Automatic takeover is disabled because Data ONTAP cannot synchronize the nonvolatile RAM (NVRAM) logs. After you fix the problem and reestablish the connection, the cluster nodes resynchronize their NVRAM logs and the cluster returns to normal operation.

- The storage from one site (site A) is not accessible to the cluster node at the other site (site B). This can be caused by the following:
    - Failure of any of the cables connecting the storage at one site to the cluster node at the other site or switch
    - If using switches, failure of any of the SFPs connecting the storage to the switch or the cluster node to the switch
    - Failure of the Fibre Channel adapter on the cluster node
    - Failure of a storage disk shelf (loop resiliency circuit, power, access to disk shelves, and so on)

  With this type of failure, you see a “mailbox disk invalid” message on the console of the filer that cannot see the storage. After you fix the problem and reestablish the connection, the cluster returns to normal operation.

- If you are using switches, the Inter-Switch Link (ISL) between each pair of switches fails.

  With this type of failure, both nodes of the cluster remain running. You see a “mailbox disk invalid” message because a filer at one site cannot see the storage at the other site. You also see a message because the two filers cannot communicate with each other. After you fix the problem and reestablish the connection, the cluster nodes resynchronize their NVRAM logs and the cluster returns to normal operation.

Recovering from a disaster

What to do

After determining that the failure to the MetroCluster configuration is a disaster, you should do the following:

- ◆ Ensure that the surviving filer is isolated from its partner. See “Restricting access to the disaster site cluster node” on page 383.
- ◆ Force the surviving filer to take over the functions of it partner. See “Forcing a node into takeover mode” on page 385.
- ◆ Recover access to the failed partner’s data by completing one of the following tasks:
  - ❖ If you are using file-access protocols, remount the failed partner’s volumes. See “Remounting volumes of the failed filer” on page 385.
  - ❖ If you are using iSCSI, bring the failed partner’s LUNs online.
- ◆ Fix the cluster at the disaster site. See “Fixing failures caused by the disaster” on page 386.
- ◆ Re-create the MetroCluster configuration. See “Reestablishing the MetroCluster configuration” on page 387.

Restricting access to the disaster site cluster node

You must restrict access to the disaster site cluster node to prevent the cluster node from resuming service. If you do not restrict access, you risk the possibility of data corruption.

You can restrict access to the disaster site cluster node in the following ways:

- ◆ Turn off power to the disaster site cluster node.
- ◆ Use manual fencing.

Turning off power to the disaster site cluster node: To turn off power to the disaster site cluster node, complete the following step.

Step	Action
1	Switch off the power at the back of the filer.

This is the best method for restricting access to the disaster site cluster node. You can perform this task at the disaster site or remotely, if you have that capability. Network Appliance recommends this method.

Using manual fencing: You can use manual fencing as an alternative to turning off power to the disaster site cluster node. The manual fencing method restricts access using software and physical means.

To manually fence off the disaster site cluster node, complete the following steps.

Step	Action	
1	Disconnect the cluster interconnect and Fibre Channel adapter cables of the cluster node at the surviving site.	
2	**If you are using...**	**Then fencing is achieved by...**
	Application failover	Using any application-specified method that either prevents the application from restarting at the disaster site or prevents the application clients from accessing the application servers at the disaster site. Methods can include turning off the application server, removing an application server from the network, or any other method that prevents the application server from running applications.
	IP failover	Using network management procedures to ensure that the filers at the disaster site are isolated from the external public network.

Forcing a node into takeover mode

To force the surviving cluster node into takeover mode, complete the following step.

Step	Action
1	Enter the following command on the surviving cluster node: **`cf forcetakeover -d`** **Result:** Data ONTAP causes the following to occur: ◆ The surviving cluster node takes over the functions of the failed partner. ◆ The mirrored relationships between the two plexes of mirrored volumes are broken, thereby creating two unmirrored volumes. This is called splitting the mirrored volumes. The overall result of using the `cf forcetakeover -d` command is that a cluster node at the surviving site is running in takeover mode with all the data in unmirrored volumes.

Remounting volumes of the failed filer

You must remount the volumes of the failed filer because the volumes will be accessed through the surviving filer. To remount the volumes, complete the following steps.

Step	Action
1	On an NFS client at the surviving site, create a directory to act as a mount point. **Example:** `mkdir /n/toaster/home`
2	Mount the volume. **Example:** `mount toaster:/vol/vol0/home /n/toaster/home`

See the *Software Setup Guide* for details about mounting volumes.

Recovering LUNs of the failed filer

If you have a MetroCluster configuration with iSCSI-attached hosts, you must actively track the state of LUNs (track whether they are online or offline) on the filer at each site. If there is a failure to a MetroCluster configuration that qualifies as a disaster, and the filer at one site is inaccessible, all LUNs in the volumes that were mirrored at the surviving site are offline. There is no way to distinguish the LUNs that were offline before the disaster from the LUNs that were online before the disaster unless you have been tracking their status.

When you recover access to the failed filer's LUNs, it is important to bring only the LUNs that were online before the disaster back online. To avoid igroup mapping conflicts, do not bring a LUN online if it was offline before the disaster. For example, suppose you have two LUNs with IDs of 0 mapped to the same igroup, but one of these LUNs was offline before the disaster. If you bring the previously offline LUN online first, you will not be able to bring the second LUN online because you cannot have two LUNs with the same ID mapped to the same host.

To recover access to the failed filer's LUNs, complete the following steps.

Step	Action
1	Identify the LUNs that were online before the diaster occurred.
2	Make sure the LUNs are mapped to an igroup that contains the hosts attached to the surviving cluster node. For more information about mapping LUNs to igroups, see the *Data ONTAP Block Access Management Guide*.
3	On the surviving cluster node, enter the following command: **`lun online`** ***`lun-path`*** **`...`** *lun-path* is the path to the LUN you want to bring online. You can specify more than one path to bring multiple LUNs online. **Example 1:** `lun online /vol/vol1/lun0` **Example 2:** `lun online /vol/vol1/lun0 /vol/vol1/lun1` **Note** After you bring LUNs back online, you might have to perform some application or host-side recovery procedures. For more information, see the documentation for your application and for your host operating system.

Fixing failures caused by the disaster

You need to fix the failures caused by the disaster, if possible. For example, if a prolonged power outage to one of the MetroCluster sites caused the failure, resetoring the power fixes the failure.

To fix failures, complete the following steps.

Step	Action
1	Ensure that the filer and disk shelves have power.
2	Ensure that all Fibre Channel adapters, VI cluster adapters, cables, and switch ports are connected and working.

You cannot fix failures if the disaster causes a site to be destroyed. For example, a fire or an earthquake could destroy one of the MetroCluster sites. In this case, you fix the failure by creating a new MetroCluster configured partner at a different site.

Once the filer at the surviving site can see the disk shelves at the disaster site, Data ONTAP renames the mirrored volumes that were split by adding a number in parenthesis to the volume name. For example, if the volume name was vol1 before the disaster and the split, the renamed volume name could be vol1(1).

Reestablishing the MetroCluster configuration

Depending on the state of a mirrored volume before you forced the surviving cluster to take over its partner, you use one of two procedures to reestablish the MetroCluster configuration:

- If the mirrored volume was in a normal state before the forced takeover, you can rejoin the two volumes to reestablish the MetroCluster configuration. This is the most typical case.
- If the mirrored volume was in an initial resynchronization state (level-0) before the forced takeover, you cannot rejoin the two volumes. You must re-create the synchronous mirror to reestablish the MetroCluster configuration.

Rejoining the two volumes: To return the cluster to normal operation by rejoining the two volumes, complete the following steps.

Step	Action
1	Validate that you can access the remote storage by entering one of the following commands: **`sysconfig -v`** or **`vol status -r`**

Step	Action
2	Turn on power to the cluster node at the disaster site. **Result:** After the cluster node at the disaster site boots up, it displays the following: `Waiting for Giveback...`
3	Determine which volumes are at the surviving site and which volumes are at the disaster site by entering the following command: **`vol status`** Volumes at the disaster site show plexes that are in a failed state with an out-of-date status. Volumes at the surviving site show plexes as online.
4	If volumes at the disaster site are online, take them offline by entering the following command for each online volume: **`vol offline`** ***`disaster_vol`*** *disaster_vol* is the name of the volume at the disaster site. **Note** An error message appears if the volume is already offline.
5	Re-create the mirrored volumes by entering the following command for each volume that was split: **`vol mirror`** ***`volume_name`*** **`-v`** ***`disaster_vol`*** *volume_name* is the volume on the surviving site's cluster node. *disaster_vol* is the volume on the disaster site's cluster node. **Result:** The *volume_name* volume rejoins the *disaster_vol* volume to reestablish the MetroCluster configuration.
6	Enter the following command at the partner cluster node: **`cf giveback`** **Result:** The cluster node at the disaster site reboots.

For an example of rejoining volumes, see "Example of rejoining traditional volumes" on page 392.

Rejoining the two aggregates: To return the cluster to normal operation by rejoining the two aggregates, complete the following steps.

Step	Action
1	Validate that you can access the remote storage by entering the following command: **`aggr status -r`**
2	Turn on power to the cluster node at the disaster site. **Result:** After the cluster node at the disaster site boots up, it displays the following: `Waiting for Giveback...`
3	Determine which aggregates are at the surviving site and which aggregates are at the disaster site by entering the following command: **`aggr status`** Aggregates at the disaster site show plexes that are in a failed state with an out-of-date status. Aggregates at the surviving site show plexes as online.
4	If the aggregates at the disaster site are online, take them offline by entering the following command for each aggregate that was split: **`aggr offline`** ***`disaster_aggr`*** *disaster_aggr* is the name of the aggregate at the disaster site. **Note** An error message appears if the aggregate is already offline.
5	Re-create the mirrored aggregates by entering the following command for each aggregate that was split: **`aggr mirror`** ***`plex_name`*** **`-v`** ***`disaster_plex`*** *plex_name* is the aggregate plex on the surviving site's cluster node. *disaster_plex* is the aggregate plex on the disaster site's cluster node. **Result:** The *plex_name* aggregate rejoins the *disaster_plex* aggregate to reestablish the MetroCluster configuration.

Step	Action
6	Enter the following command at the partner cluster node: **`cf giveback`** **Result:** The cluster node at the disaster site reboots.

Re-creating mirrored traditional volumes: To return the cluster to normal operation by re-creating the MetroCluster mirror, complete the following steps.

Step	Action
1	Validate that you can access the remote storage by entering one of the following commands: **`sysconfig -v`** or **`vol status -r`** **Note** A `(level-0 resync in progress)` message indicates that a plex cannot be rejoined.
2	Turn on the power to the cluster node at the disaster site. **Result:** After the cluster node at the disaster site boots up, it displays the following: `Waiting for Giveback...`
3	Enter the following command at the partner cluster node: **`cf giveback`** **Result:** The cluster node at the disaster site reboots.
4	Destroy every target plex that is in a level-0 resync state by entering the following command: **`vol destroy`** ***`plex_name`*** For details about the SyncMirror feature, see Chapter 7, “SyncMirror Management,” on page 301.

Step	Action
5	Re-create the mirrored volumes by entering the following command for each volume that was split: **`vol mirror`** ***`volume_name`*** For details about the SyncMirror feature, see Chapter 7, "SyncMirror Management," on page 301.

For an example of rejoining volumes, see "Example of re-creating a mirrored traditional volume" on page 393.

Re-creating mirrored aggregates: To return the cluster to normal operation by re-creating the MetroCluster mirror, complete the following steps.

Step	Action
1	Validate that you can access the remote storage by entering the following command: **`aggr status -r`** **Note** A `(level-0 resync in progress)` message indicates that a plex cannot be rejoined.
2	Turn on the power to the cluster node at the disaster site. **Result:** After the cluster node at the disaster site boots up, it displays the following: `Waiting for Giveback...`
3	If the aggregates at the disaster site are online, take them offline by entering the following command for each aggregate that was split: **`aggr offline`** ***`disaster_aggr`*** *disaster_aggr* is the name of the aggregate at the disaster site. **Note** An error message appears if the aggregate is already offline.

Step	Action
4	Destroy every target plex that is in a level-0 resync state by entering the following command: **aggr destroy *plex_name*** For details about the SyncMirror feature, see Chapter 7, “SyncMirror Management,” on page 301.
5	Re-create the mirrored volumes by entering the following command for each volume that was split: **aggr mirror *plex_name*** For details about the SyncMirror feature, see Chapter 7, “SyncMirror Management,” on page 301.
6	Enter the following command at the partner cluster node: **cf giveback** **Result:** The cluster node at the disaster site reboots.

Example of rejoining traditional volumes

The following example shows the commands and status output when you rejoin volumes to reestablish the MetroCluster configuration.

First, the volume status of the disaster site’s storage after reestablishing access to the cluster partner at the surviving site is shown.

```
filer1> vol status -r
Volume mir (online, normal) (zoned checksums)
  Plex /mir/plex5 (online, normal, active)
    RAID group /filer1/plex5/rg0 (normal)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------ ----- -------------- -------------
parity   8a.2   8a   0     2   FC:B  34500/70656000 35003/71687368
data     8a.8   8a   1     0   FC:B  34500/70656000 35003/71687368

Volume mir(1) (failed, out-of-date) (zoned checksums)
  Plex /mir(1)/plex1 (offline, normal, out-of-date)
    RAID group /mir(1)/plex1/rg0 (normal)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------ ----- -------------- -------------
parity   6a.0   6a   0     0   FC:B  34500/70656000 35003/71687368
data     6a.1   6a   0     1   FC:B  34500/70656000 35003/71687368
```

```
  Plex /mir(1)/plex5 (offline, failed, out-of-date)
```

Next, the mirror is reestablished using the `vol mirror -v` command.

Note

The filer at the surviving site is called filer1; the filer at the disaster site is called filer2.

```
filer1> vol mirror mir -v mir(1)
This will destroy the contents of mir(1).  Are you sure? y
Mon Nov 18 15:36:59 GMT [filer1:
raid.mirror.resync.snapcrtok:info]: mir: created mirror
resynchronization snapshot mirror_resync.1118153658(filer2)
Mon Nov 18 15:36:59 GMT [filer1: raid.rg.resync.start:notice]:
/mir/plex6/rg0: start resynchronization (level 1)
Mon Nov 18 15:36:59 GMT [filer1: raid.mirror.resync.start:notice]:
/mir: start resynchronize to target /mir/plex6
```

After the volumes rejoin, the synchronous mirrors of the MetroCluster configuration are reestablished.

```
filer1> vol status -r mir
Volume mir (online, mirrored) (zoned checksums)
  Plex /mir/plex5 (online, normal, active)
    RAID group /mir/plex5/rg0 (normal)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------- ----- -------------- -------------
parity   8a.2   8a   0     2   FC:B  34500/70656000 35003/71687368
data     8a.8   8a   1     0   FC:B  34500/70656000 35003/71687368

  Plex /mir/plex6 (online, normal, active)
    RAID group /mir/plex6/rg0 (normal)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------- ----- -------------- -------------
parity   6a.0   6a   0     0   FC:B  34500/70656000 35003/71687368
data     6a.1   6a   0     1   FC:B  34500/70656000 35003/71687368
```

Example of re-creating a mirrored traditional volume

The following example shows the commands and status output when re-creating volumes to reestablish the MetroCluster configuration.

The volume status of the disaster site’s storage after reestablishing access to the cluster partner at the surviving site is shown.

```
filer1>vol status -r
Volume mir1 (online, normal) (zoned checksums)
```

```
  Plex /mir1/plex0 (online, normal, active)
  RAID group /mir1/plex0/rg0 (normal)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------ ----- -------------- -------------
parity   8a.3   8a   0     3   FC:B  34500/70656000 35003/71687368
data     8a.4   8a   0     4   FC:B  34500/70656000 35003/71687368
data     8a.6   8a   0     6   FC:B  34500/70656000 35003/71687368
data     8a.5   8a   0     5   FC:B  34500/70656000 35003/71687368

Volume mir1(1) (failed, partial) (zoned checksums)
  Plex /mir1(1)/plex0 (offline, failed, inactive)

  Plex /mir1(1)/plex6 (online, normal, resyncing)
    RAID group /mir1(1)/plex6/rg0 (level-0 resync in progress)

RAID Disk Device HA  SHELF BAY CHAN  Used (MB/blks) Phys (MB/blks)
--------- ------ ------------ ----- -------------- -------------
parity    6a.6   6a  0     6   FC:B  34500/70656000 35003/71687368
data      6a.2   6a  0     2   FC:B  34500/70656000 35003/71687368
data      6a.3   6a  0     3   FC:B  34500/70656000 35003/71687368
data      6a.5   6a  0     5   FC:B  34500/70656000 35003/71687368
```

The mir1(1)/plex6 plex shows that a level-0 resynchronization was in progress; therefore, an attempt to rejoin the plexes fails, as shown in the following.

```
filer1> vol mirror mir1 -v mir1(1)
vol mirror: Illegal mirror state for volume ’mir1(1)’
```

Because the mir1(1)/plex6 plex had a level-0 resynchronization in progress, the mir1(1) volume must be destroyed and the mir volume remirrored to reestablish a synchronous mirror, as shown in the following.

```
filer1> vol mirror mir1 -v mir1(1)
vol mirror: Illegal mirror state for volume ’mir1(1)’
filer1> vol destroy mir1(1)
Are you sure you want to destroy this volume? y
Volume ’mir1(1)’ destroyed.
filer1> vol mirror mir1
Creation of a mirror plex with 4 disks has been initiated.  The
disks need to be zeroed before addition to the volume.  The process
has been initiated and you will be notified via the system log as
disks are added.
```

Glossary

ACL	Access Control List. A list that contains the users' or groups' access rights to each share.
active file system	A file system excluding its snapshots.
aggregate	A manageable unit of RAID-protected storage, consisting of one or two plexes, that can contain one traditional volume or multiple flexible volumes.
blocking factor	The number of tape blocks that are transferred in each write operation.
CIFS	Common Internet File System. A protocol for networking PCs.
console	A terminal that is attached to a filer's serial port and is used to monitor and manage filer operation.
DNS	Domain Naming Service. An Internet service for finding IP addresses.
dump path	A path that specifies one volume, qtree, or subtree to back up.
file mark	Data on a tape that signals a boundary of a tape file.
HTTP	HyperText Transfer Protocol. An Internet Web transfer protocol.
increment chain	A series of incremental backups of the same path.

inode	A data structure containing information about files on a filer and in a UNIX file system.
local tape device	A program-based functionality associated with a tape drive that is directly attached to a filer that is performing a tape operation.
MD5	Message Digest 5. A checksum algorithm described in RFC (Request For Comments) 1321, a proposal for an Internet standard.
mirror	A volume that contains a read-only copy of data in the active file system of another volume, usually on another filer.
NDMP	Network Data Management Protocol. A protocol that allows Network Appliance filers to communicate with backup applications, and provides capabilities for controlling the robotics of multiple tape backup devices.
NFS	Network File System. A protocol for networking UNIX-based computers.
NIS	Network Information Service, formerly called Yellow Pages. An administrative database for networks.
NVFAIL	Software that warns you of compromised database validity and automatically renames the database so that it does not restart automatically.
NVRAM	Nonvolatile RAM in the filer, used for logging incoming write data and NFS requests. Improves system performance and prevents loss of data in case of a filer or power failure.
Open Systems platform	A non-filer system, such as a server running Solaris, HP-UX, Windows NT, or Windows 2000, whose data can be backed up to a SnapVault secondary storage system.

Open Systems SnapVault agent	A software module that can be installed on a non-filer system and enables that system to back up its data to a SnapVault secondary storage system.
plex	A physical copy of a file system. An unmirrored volume has one plex; a mirrored volume has two identical plexes.
primary storage system	A system whose data is to be backed up by SnapVault.
quota	A limit placed on a file system that restricts disk space usage by files with a given User ID (UID) or group ID (GID).
qtree	A special subdirectory of the root of a volume that acts as a virtual subvolume with special attributes.
RAID	Redundant Array of Independent Disks. A technique that protects against disk failure by computing parity information based on the contents of all the disks in the array. NetApp filers use RAID Level 4, which stores all parity information on a single disk.
Remote Shell	A program that enables a user on one system to execute a program on another system. Remote Shell connections are usually not interactive.
remote tape device	A program-based functionality associated with a tape drive that is not directly attached to a filer that is performing a tape operation.
root volume	The volume that contains information that controls the entire filer, usually in the /etc/rc file.
secondary storage system	A filer or NearStore system to which data is backed up by SnapVault.

share	A directory or directory structure on the filer that has been made available to network users and can be mapped to a drive letter on a CIFS client.
SnapMirror	Software that performs automated file system replication of a volume onto the same or a separate disk or filer.
SnapRestore	Software that restores an entire volume to the state recorded in a previously taken snapshot.
snapshot	An online, read-only copy of the entire file system that protects against accidental deletions or modifications of files without duplicating file contents. Snapshots enable users to restore files and enable administrators to back up the filer to tape while the filer is in use.
snapshot reserve	The portion of a volume’s disk space that is reserved for snapshots.
source filer	The filer from which you are replicating data.
subtree	A directory in a volume or qtree.
tape block	1,024 bytes of data.
tape device	A specific functionality of a physical tape drive that you create by specifying information in the tape device name when you install a tape drive or tape stacker.
tape file	Data on a tape delimited by file marks.
tape library	Hardware that can access tape cartridges randomly.

tape medium changer	Hardware that can load and unload tape cartridges.
tape stacker	Hardware that can access tape cartridges from a stack.
vFiler	A virtual filer created using the MultiStore software that allows you to partition the storage and networking resources of a single filer so that it appears as multiple filers on the network.
volume	A file system.
volume copy	A way of copying both data in the active file system and data in snapshots from one volume to another.

Index