VERITAS™

VERITAS FlashSnap™ Point-In-Time Copy Solutions

Administrator's Guide 2.0

N10013F

Disclaimer

The information contained in this publication is subject to change without notice. VERITAS Software Corporation makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. VERITAS Software Corporation shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this manual.

VERITAS Copyright

VERITAS Software Corporation
350 Ellis Street
Mountain View, CA 94043
USA
Phone 650–527–8000
Fax 650–527–2908
http://www.veritas.com

Third-Party Copyrights

Data Encryption Standard (DES) Copyright

Contents

Preface

The purpose of this guide is to demonstrate how to use VERITAS FlashSnap™ to implement point-in-time copy solutions on enterprise systems. FlashSnap offers you flexible solutions for the efficient management of multiple point-in-time copies of your data, and for reducing resource contention on your business-critical servers.

Note This guide supersedes the *VERITAS Off-Host Processing Using FastResync Administrator's Guide.*

Audience and Scope

The *VERITAS FlashSnap Point-In-Time Copy Solutions Administrator's Guide* provides information about how to implement solutions for online backup of databases and cluster-shareable file systems, for decision support on enterprise systems, and for Storage Rollback of databases to implement fast database recovery.

This guide is intended for experienced system administrators responsible for installing, configuring, and maintaining high-availability clustered systems under the control of VERITAS software.

This guide assumes that you have a good understanding of the following topics:

- The AIX, HP-UX, Linux or Solaris operating system.
- AIX, HP-UX, Linux or Solaris system administration.
- Cluster hardware and its configuration in enterprise installations (for scenarios that use clusters).
- Storage management.
- Configuration and administration of DB2, Oracle or Sybase databases (for operating systems and scenarios that use these databases).

Organization

This guide is organized as follows:

- Point-In-Time Copy Solutions
- Setting up Volumes for Instant Snapshots
- Online Database Backup
- Off-Host Cluster File System Backup
- Decision Support
- Database Recovery
- Files and Scripts for Sample Scenarios
- Preparing a Replica Oracle Database

Related Documents

The following documents provide more information related to the installation, configuration and administration of the products described in this guide:

- *VERITAS NetBackup BusinesServer Getting Started Guide*
- *VERITAS NetBackup BusinesServer System Administrator's Guide*
- *VERITAS Storage Foundation for Cluster File System Installation and Configuration Guide*
- *VERITAS Storage Foundation for DB2 Database Administrator's Guide*
- *VERITAS Storage Foundation for DB2 Installation Guide*
- *VERITAS Storage Foundation for Oracle Database Administrator's Guide*
- *VERITAS Storage Foundation for Oracle Installation Guide*
- *VERITAS Storage Foundation for Sybase Database Administrator's Guide*
- *VERITAS Storage Foundation for Sybase Installation Guide*
- *VERITAS File System Administrator's Guide*
- *VERITAS File System Installation Guide*
- *VERITAS NetBackup DataCenter Installation Guide*
- *VERITAS NetBackup DataCenter System Administrator's Guide*
- *VERITAS NetBackup for Oracle ServerFree Agent System Administrator's Guide*
- *VERITAS NetBackup ServerFree Agent System Administrator's Guide*
- *VERITAS Volume Manager Administrator's Guide*
- *VERITAS Volume Manager Installation Guide*

The following guides describe the usage of the FlashSnap agent with the EMC TimeFinder product on EMC Symmetrix storage systems:

- *VERITAS FlashSnap Agent for Symmetrix Installation Guide*
- *VERITAS FlashSnap Agent for Symmetrix Administrator's Guide*
- *VERITAS Cluster Server Agents for VERITAS FlashSnap Agent for Symmetrix Installation and Configuration Guide*

Conventions

The following table describes the typographic conventions used in this guide.

Typeface	Usage	Examples
`monospace`	Computer output, file contents, files, directories, software elements such as command options, function names, and parameters	Read tunables from the `/etc/vx/tunefstab` file. See the `ls`(1) manual page for more information.
italic	New terms, book titles, emphasis, variables to be replaced by a name or value	See the *User's Guide* for details. The variable *ncsize* determines the value of...
`monospace` (**`bold`**)	User input; the "#" symbol indicates a command prompt[1]	# **`mount -F vxfs /h/filesys`**
`monospace` ***(bold and italic)***	Variables to be replaced by a name or value in user input[1]	# **`mount -F`** ***`fstype mount_point`***

Symbol	Usage	Examples
%	C shell prompt	
$	Bourne/Korn/Bash shell prompt	
#	Superuser prompt (all shells)	
\	Continued input on the following line[1]	# **`mount -F vxfs`** \ **`/h/filesys`**
[]	In a command synopsis, brackets indicates an optional argument	`ls [ -a ]`
|	In a command synopsis, a vertical bar separates mutually exclusive arguments	`mount [suid | nosuid ]`

1 On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

Getting Help

If you have any comments or problems with the VERITAS products, contact VERITAS Technical Support:

- U.S. and Canadian Customers: 1-800-342-0652
- International Customers: +1 (650) 527-8555
- Email: support@veritas.com

For license information (U.S. and Canadian Customers):

- Phone: 1-650-527-0300
- Email: license@veritas.com
- Fax: 1-650-527-0952

For information on purchasing VERITAS products:

- Phone: 1-800-327-2232
- Email: sales.mail@veritas.com

For software updates:

- Email: swupdate@veritas.com

For additional technical support information, such as TechNotes, product alerts, and hardware compatibility lists, visit the VERITAS Technical Support Web site at:

- http://support.veritas.com

For information about VERITAS products, VERITAS Education Services and VPro Consulting Services, visit the VERITAS Web site at:

- http://www.veritas.com

Unique Message Number

If you encounter a product error message, record the unique message number preceding the text of the message. When contacting VERITAS Technical Support, either by telephone or by visiting the VERITAS Technical Support website, be sure to provide the relevant message number. VERITAS Technical Support will use this message number to quickly determine if there are TechNotes or other information available for you.

A unique message number is an alpha-numeric string beginning with the letter “V”. For example, in the message number:

```
V-5-732-8018 At least one disk must be specified
```


the “V” indicates that this is a VERITAS product error message. The text of the error message follows the unique message number.

Point-In-Time Copy Solutions 1

This chapter introduces the point-in-time copy solutions that you can implement using the VERITAS FlashSnap™ technology.

Note To implement the Point-In-Time Copy solutions presented in this document, a valid license for VERITAS FlashSnap must be present on all the systems to which the solutions are applied.

VERITAS FlashSnap offers a flexible and efficient means of managing business critical data. It allows you to capture an online image of actively changing data at a given instant: a *point-in-time copy.* You can perform system backup, upgrade and other maintenance tasks on point-in-time copies while providing continuous availability of your critical data. If required, you can offload processing of the point-in-time copies onto another host to avoid contention for system resources on your production server.

Two kinds of point-in-time copy solution are supported by the FlashSnap license:

- Volume-level solutions are made possible by the Persistent FastResync and Disk Group Split/Join features of VERITAS Volume Manager™ 4.0. These features are suitable for implementing solutions where the I/O performance of the production server is critical.

  The Persistent FastResync and Disk Group Split/Join features are described in "Persistent FastResync of Volume Snapshots" on page 6 and "Disk Group Split/Join" on page 7.

- File system-level solutions use the Storage Checkpoint™ feature of VERITAS File System™ 4.0. Storage Checkpoints are suitable for implementing solutions where storage space is critical for:

  - File systems that contain a small number of mostly large files.
  - Application workloads that change a relatively small proportion of file system data blocks (for example, web server content and some databases).
  - Applications where multiple writable copies of a file system are required for testing or versioning.

  The Storage Checkpoints feature is described in "Storage Checkpoints" on page 8.

The FlashSnap license also supports the VERITAS FlashSnap Agent for *Symmetrix*. This feature is described in "VERITAS FlashSnap Agent for Symmetrix" on page 9.

Applications of Point-in-Time Copy Solutions

The following typical activities are suitable for Point-In Time Copy solutions implemented using VERITAS FlashSnap:

- *Data Backup*—Many enterprises require 24 x 7 data availability. They cannot afford the downtime involved in backing up critical data offline. By taking snapshots of your data, and backing up from these snapshots, your business-critical applications can continue to run without extended downtime or impacted performance.
- *Decision Support Analysis and Reporting*—Operations such as decision support analysis and business reporting may not require access to real-time information. You can direct such operations to use a replica database that you have created from snapshots, rather than allow them to compete for access to the primary database. When required, you can quickly resynchronize the database copy with the data in the primary database.
- *Testing and Training*—Development or service groups can use snapshots as test data for new applications. Snapshot data provides developers, system testers and QA groups with a realistic basis for testing the robustness, integrity and performance of new applications.
- *Database Error Recovery*—Logic errors caused by an administrator or an application program can compromise the integrity of a database. You can recover a database more quickly by restoring the database files by using Storage Checkpoints or a snapshot copy than by full restoration from tape or other backup media.

Note To provide continuity of service in the event of hardware failure in a cluster environment, you can use point-in-time copy solutions in conjunction with the high availability cluster functionality of VERITAS Storage Foundation™ for Cluster File System HA or VERITAS Storage Foundation™ HA for the DB2, Oracle and Sybase databases.

Point-In-Time Copy Scenarios

Point-in-time copies of volumes allow you to capture an image of a database or file system at a selected instant for use in applications such as backups, decision support, reporting, and development testing.

Point-in-time copy solutions may additionally be configured to use off-host processing to remove much of the performance overhead on a production system.

The following chapters describe how you can use FlashSnap to implement regular online backup of database and cluster file system volumes, to set up a replica of a production database for decision support:

- Online Database Backup
- Off-Host Cluster File System Backup
- Decision Support

Three types of point-in-time copy solution are considered in this document:

- Primary host solutions where the copy is processed on the same system as the active data. See "Implementing Point-In Time Copy Solutions on a Primary Host" on page 10 for more information.
- Off-host solutions where the copy is processed on a different system from the active data. If implemented correctly, such solutions have almost no impact on the performance of the primary production system. See "Implementing Off-Host Point-In-Time Copy Solutions" on page 12 for more information.
- Using Storage Checkpoints to quickly roll back a database instance to an earlier point in time. See "Database Recovery" on page 61 for more information.

VERITAS Software Used in Point-In-Time Copy Scenarios

This guide provides a number of example scenarios that illustrate how to implement point-in-time copy solutions. The following table shows the VERITAS products that may be used with a VERITAS FlashSnap license to provide the required functionality in different environments:

Environment	Database server	Other applications
Standalone primary host	VERITAS Storage Foundation for DB2, Oracle or Sybase	VERITAS Storage Foundation
Cluster without automatic failover	VERITAS Storage Foundation for DB2, Oracle or Sybase	VERITAS Storage Foundation for Cluster File System
Cluster with automatic failover	VERITAS Storage Foundation for DB2, Oracle or Sybase HA	VERITAS Storage Foundation for Cluster File System HA

VERITAS Storage Foundation for databases is required if you want to use the VERITAS Quick I/O™, VERITAS Extension for Oracle Disk Manager (ODM), VERITAS QuickLog™, Storage Checkpoints and management interface features to enhance database performance and manageability.

Note The Cached Quick I/O and QuickLog features are not supported on the Linux platform.

The VERITAS Clustering Functionality for VxVM and VERITAS Cluster File System 4.0 features of VERITAS Storage Foundation 4.0 for Cluster File System and Cluster File System HA allow you to share data within a cluster. The HA version uses VERITAS Cluster Server to allow you to configure automated application and storage failover to provide continuous availability of service.

Note The Cached Quick I/O and QuickLog features are not supported for use with VERITAS Cluster File System 4.0. However, Quick I/O is supported for use with VERITAS Cluster File System 4.0.

The following non-cluster specific components are used in the sample scenarios:

- VERITAS Volume Manager 4.0 (VxVM) is a disk management subsystem that supports disk striping, disk mirroring, and simplified disk management for improved data availability and superior performance. The FlashSnap license enables the use of the Persistent FastResync and Disk Group Split/Join features of VxVM.
- VERITAS File System 4.0 (VxFS) is a high-performance, fast-recovery file system that is optimized for business-critical database applications and data-intensive workloads. VxFS offers online administration, letting you perform most frequently scheduled maintenance tasks (including online backup, resizing, and file system changes) without interrupting data or system availability. The FlashSnap license enables the use of the Storage Checkpoints feature of VxFS.

You can also use the following cluster-specific components with the sample scenarios where required:

- On AIX, Linux and Solaris, VERITAS Cluster Server™ 4.0 (VCS) is a high-availability (HA) solution for cluster configurations. VCS monitors systems and application services, and restarts services on a different cluster node (failover) in the event of either hardware or software failure. It also allows you to perform general administration tasks such as making nodes join or leave a cluster.

  Note On HP-UX, MC/ServiceGuard may be configured as the cluster monitor.

- VERITAS Clustering Functionality for VxVM (CVM) allows multiple hosts to simultaneously access and manage a given set of disks that are under the control of VERITAS Volume Manager.
- VERITAS Cluster File System 4.0 (CFS) allows cluster nodes to share access to the same VxFS file system. CFS is especially useful for sharing read-intensive data between cluster nodes.

If you require a backup solution, the following VERITAS software is recommended:

- VERITAS NetBackup™ DataCenter provides mainframe-class data protection for corporate data centers. NetBackup DataCenter allows you to manage all aspects of backup and recovery, and allows consistent backup policies to be enforced across your organization. Optional NetBackup ServerFree Agents enhance NetBackup DataCenter to provide data protection for frozen image data such as VxVM snapshot mirrors. They can also be used to offload backup processing to NetBackup media servers or third-party copy devices over Fibre Channel networks.
- VERITAS NetBackup BusinesServer™ provides protection for small to medium-size server installations. It does not provide integrated support for backing up VxVM snapshot mirrors or for offloading backup processing. However, you can use it to back up snapshot volumes that have been created from snapshot mirrors.

For more information about installing and configuring these products see the following documentation:

- *VERITAS Storage Foundation for Cluster File System Installation and Configuration Guide*
- *VERITAS Storage Foundation for DB2 Database Administrator's Guide*
- *VERITAS Storage Foundation for DB2 Installation Guide*
- *VERITAS Storage Foundation for Oracle Database Administrator's Guide*
- *VERITAS Storage Foundation for Oracle Installation Guide*
- *VERITAS Storage Foundation for Sybase Database Administrator's Guide*
- *VERITAS Storage Foundation for Sybase Installation Guide*
- *VERITAS File System Administrator's Guide*
- *VERITAS File System Installation Guide*
- *VERITAS NetBackup DataCenter Installation Guide*
- *VERITAS NetBackup DataCenter System Administrator's Guide*
- *VERITAS NetBackup ServerFree Agent System Administrator's Guide*
- *VERITAS NetBackup for Oracle ServerFree Agent System Administrator's Guide*
- *VERITAS NetBackup BusinesServer Getting Started Guide*
- *VERITAS NetBackup BusinesServer System Administrator's Guide*
- *VERITAS Volume Manager Administrator's Guide*
- *VERITAS Volume Manager Installation Guide*

Persistent FastResync of Volume Snapshots

VERITAS Volume Manager allows you to take multiple snapshots of your data at the level of a volume. A snapshot volume contains a stable copy of a volume's data at a given moment in time that you can use for online backup or decision support. If Persistent FastResync is enabled on a volume, VxVM uses a *FastResync map* to keep track of which blocks are updated in the volume and in the snapshot. If the data in one mirror is not updated for some reason, it becomes out-of-date, or *stale*, with respect to the other mirrors in the volume. The presence of the FastResync map means that only those updates that the mirror has missed need be reapplied to resynchronize it with the volume. A full, and thereby much slower, resynchronization of the mirror from the volume is unnecessary.

When snapshot volumes are reattached to their original volumes, Persistent FastResync allows the snapshot data to be quickly refreshed and re-used. Persistent FastResync uses disk storage to ensure that FastResync maps survive both system and cluster crashes. If Persistent FastResync is enabled on a volume in a private disk group, incremental resynchronization can take place even if the host is rebooted.

Persistent FastResync can track the association between volumes and their snapshot volumes after they are moved into different disk groups. After the disk groups are rejoined, Persistent FastResync allows the snapshot plexes to be quickly resynchronized.

For more information, see the *VERITAS Volume Manager Administrator's Guide*.

Instant Volume Snapshots

The traditional type of volume snapshot that was provided in VxVM is of the *third-mirror* type. This name comes from its original implementation by adding an additional plex to a mirrored volume. After the contents of the third-mirror (or snapshot plex) had been synchronized from the original plexes of the volume, it could be detached as a snapshot volume for use in backup or decision support applications. Enhancements to the snapshot model allowed snapshot volumes to contain more than a single plex, reattachment of a subset of a snapshot volume's plexes, and persistence of FastResync across system reboots or cluster restarts.

VxVM 4.0 introduces instant snapshots that offer advantages over traditional third-mirror snapshots. The benefits of instant snapshots include immediate availability for use, quick refreshment, and easier configuration and administration. Full-sized instant snapshots are similar to third-mirror snapshots in that they are the same length as the original volume.

Space-optimized instant snapshots require less space than full-sized snapshots by recording changed regions in the original volume to a storage cache. As the original volume is written to, VxVM preserves its data in the cache before the write is committed.

For more information, see the *VERITAS Volume Manager Administrator's Guide*.

Disk Group Split/Join

One or more volumes, such as snapshot volumes, can be split off into a separate disk group and deported. They are then ready for importing on another host that is dedicated to off-host processing. This host need not be a member of a cluster but it must have access to the disks on which the volumes are configured. At a later stage, the disk group can be deported, re-imported, and joined with the original disk group, or with a different disk group.

Note As space-optimized instant snapshots only record information about changed regions in the original volume, they cannot be moved to a different disk group. They are therefore unsuitable for the off-host processing applications that are described in this document.

The contents of full-sized instant snapshots must be fully synchronized with the unchanged regions in the original volume before such snapshots can be moved into a different disk group and deported from a host.

For more information, see the *VERITAS Volume Manager Administrator's Guide*.

Storage Checkpoints

A Storage Checkpoint is a persistent image of a file system at a given instance in time. Storage Checkpoints use a *copy-on-write* technique to reduce I/O overhead by identifying and maintaining only those file system blocks that have changed since a previous Storage Checkpoint was taken. Storage Checkpoints have the following important features:

- Storage Checkpoints persist across system reboots and crashes.
- A Storage Checkpoint can preserve not only file system metadata and the directory hierarchy of the file system, but also user data as it existed when the Storage Checkpoint was taken.
- After creating a Storage Checkpoint of a mounted file system, you can continue to create, remove, and update files on the file system without affecting the image of the Storage Checkpoint.
- Unlike file system snapshots, Storage Checkpoints are writable.
- To minimize disk space usage, Storage Checkpoints use free space in the file system.

Storage Checkpoints and the Storage Rollback feature of VERITAS Storage Foundation for Databases enable rapid recovery of databases from logical errors such as database corruption, missing files and dropped table spaces. You can mount successive Storage Checkpoints of a database to locate the error, and then roll back the database to a Storage Checkpoint before the problem occurred. For more information, see "Database Recovery" on page 61 and the *VERITAS Storage Foundation Administrator's Guide.*

VERITAS NetBackup for Oracle Advanced BLI Agent uses Storage Checkpoints to enhance the speed of backing up Oracle databases. For more information, see the *VERITAS NetBackup for Oracle Advanced BLI Agent System Administrator's Guide.*

For more information about the implementation of Storage Checkpoints, see the *VERITAS File System Administrator's Guide*.

VERITAS FlashSnap Agent for Symmetrix

The EMC TimeFinder product from EMC is a business continuance solution that allows you to create and use copies of EMC Symmetrix devices while the standard devices remain online and accessible. Business Continuance Volume (BCV) devices contain copies of Symmetrix standard (STD) devices and provide redundancy. You can temporarily detach the BCV mirrors and use the BCVs to perform backups, testing, and other administrative tasks.

VERITAS FlashSnap Agent for *Symmetrix* provides a set of commands that allow you to use the EMC TimeFinder split and restore operations in conjunction with VxFS file systems and VxVM disk groups and volumes that have been created on Symmetrix STD devices.

You can use the commands in VERITAS FlashSnap Agent for *Symmetrix* to:

- Associate a disk group with a BCV disk group, or Symmetrix STD devices in a disk group with identical BCV devices.
- Initiate TimeFinder mirroring for Symmetrix STD devices in a disk group.
- Split Symmetrix STD devices from their BCV devices and create duplicate volumes on the BCV devices. You can use the resulting BCV volumes for administrative tasks such as backups and testing.
- Reattach and resynchronize the STD and BCV devices. The devices can be remirrored from the STD copy or restored from the BCV copy.
- Detach the STD devices from their BCV devices.

Note The VERITAS FlashSnap Agent for *Symmetrix* software is available with the VERITAS Storage Foundation for *Oracle* product. It is not currently available for DB2 or Sybase databases.

A valid SYMCLI license key must be obtained from EMC to use EMC TimeFinder.

The VERITAS Cluster Server Agents for VERITAS FlashSnap Agent for *Symmetrix* are add-ons to VERITAS Cluster Server that enable automatic recovery of FlashSnap Agent for Symmetrix operations.

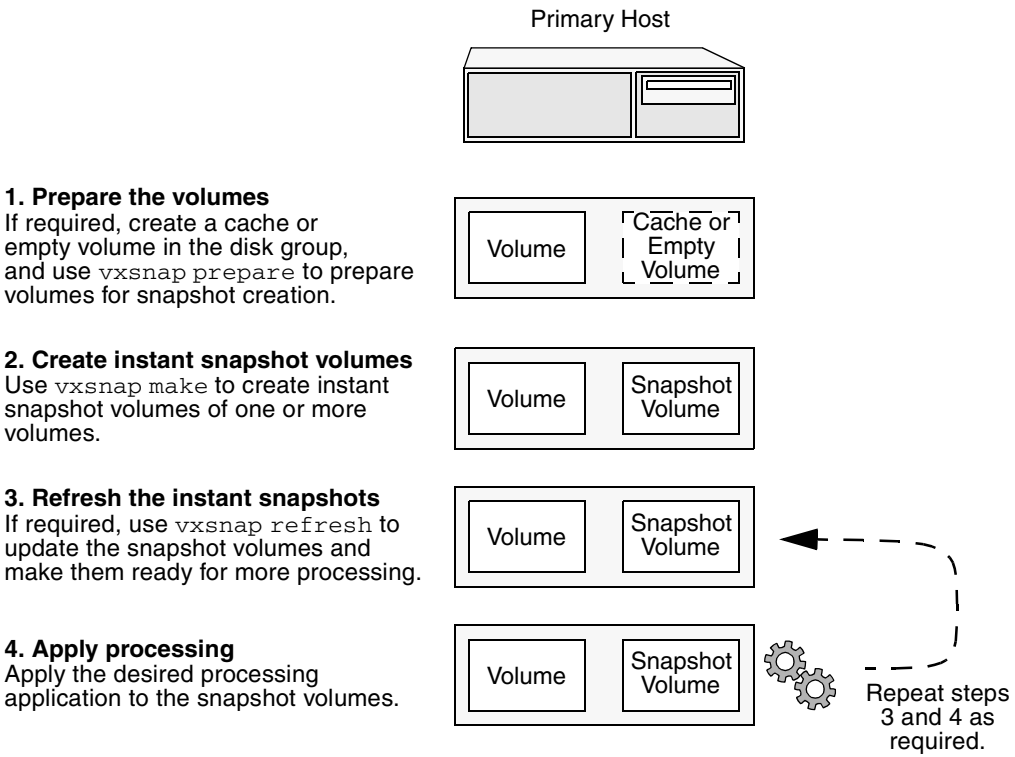
For more information, see the following documents that are installed with the `VRTSfasdc` package:

- *VERITAS FlashSnap Agent for Symmetrix Installation Guide*
- *VERITAS FlashSnap Agent for Symmetrix Administrator's Guide*
- *VERITAS Cluster Server Agents for VERITAS FlashSnap Agent for Symmetrix Installation and Configuration Guide*

Implementing Point-In Time Copy Solutions on a Primary Host

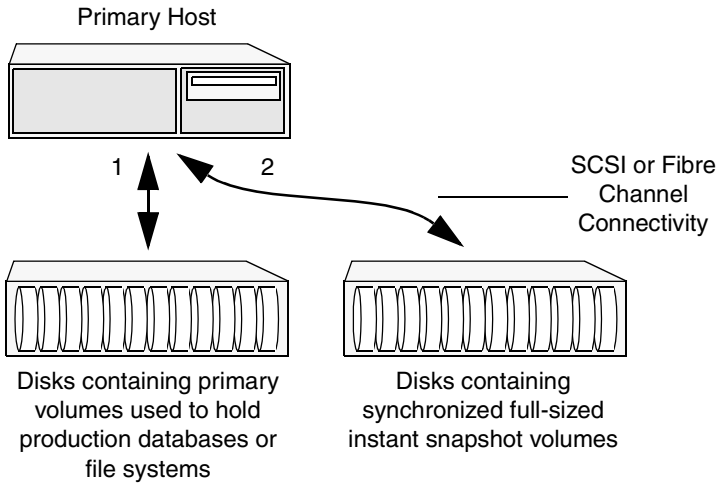
The figure, "Using Snapshots and FastResync to Implement Point-In-Time Copy Solutions on a Primary Host" on page 10, illustrates the steps that are needed to set up the processing solution on the primary host. Note that the Disk Group Split/Join functionality is not used. As all processing takes place in the same disk group, synchronization of the contents of the snapshots from the original volumes is not usually required unless you want to prevent disk contention. Snapshot creation and updating are practically instantaneous.

Using Snapshots and FastResync to Implement Point-In-Time Copy Solutions on a Primary Host



The figure, "Example Point-In-Time Copy Solution on a Primary Host" on page 11, shows the suggested arrangement for implementing solutions where the primary host is used and disk contention is to be avoided.

Example Point-In-Time Copy Solution on a Primary Host



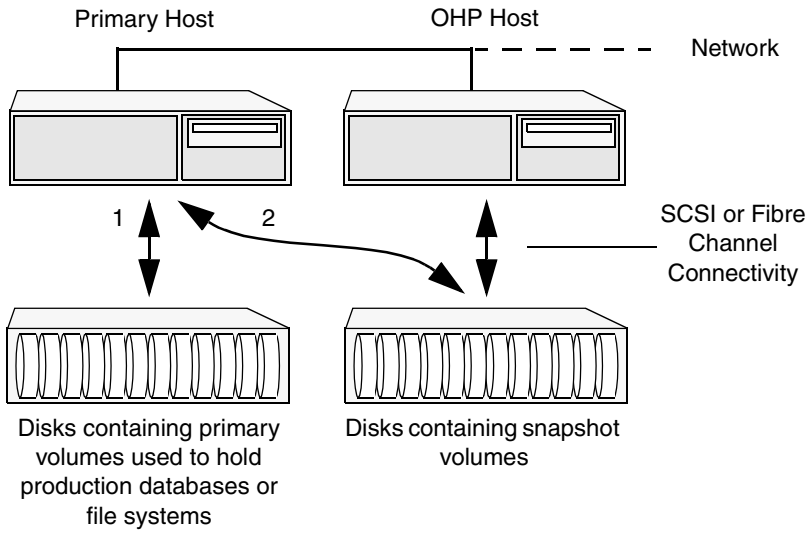
In this setup, it is recommended that separate paths (shown as 1 and 2) from separate controllers be configured to the disks containing the primary volumes and the snapshot volumes. This avoids contention for disk access, but the primary host's CPU, memory and I/O resources are more heavily utilized when the processing application is run.

Note For space-optimized or unsynchronized full-sized instant snapshots, it is not possible to isolate the I/O pathways in this way. This is because such snapshots only contain the contents of changed regions from the original volume. If applications access data that remains in unchanged regions, this is read from the original volume.

Implementing Off-Host Point-In-Time Copy Solutions

As shown in "Example Implementation of an Off-Host Point-In-Time Copy Solution" on page 12, by accessing snapshot volumes from a lightly loaded host (shown here as the *OHP host*), CPU- and I/O-intensive operations for online backup and decision support are prevented from degrading the performance of the primary host that is performing the main production activity (such as running a database). Also, if you place the snapshot volumes on disks that are attached to host controllers other than those for the disks in the primary volumes, it is possible to avoid contending with the primary host for I/O resources. To implement this, paths 1 and 2 shown in the following figures should be connected to different controllers.

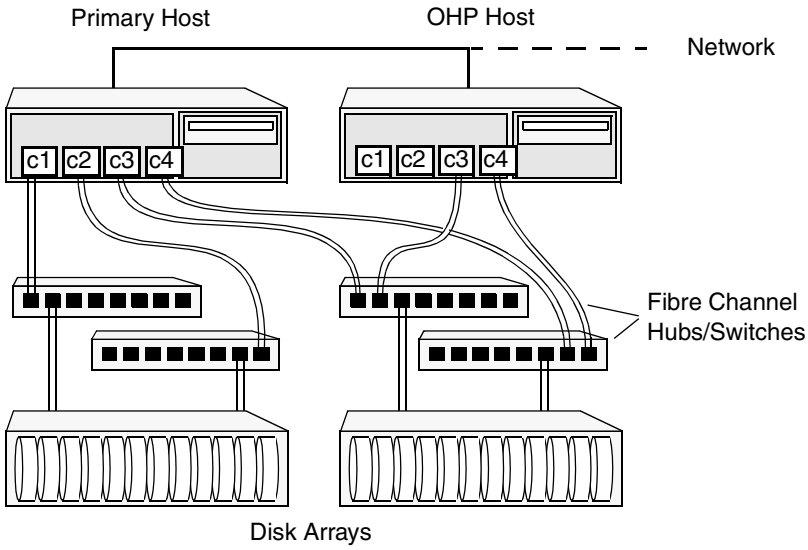
Example Implementation of an Off-Host Point-In-Time Copy Solution



The figure "Example Connectivity for Off-Host Solution Using Redundant-Loop Access" on page 13 gives an example of how you might achieve such connectivity using Fibre Channel technology with 4 Fibre Channel controllers in the primary host. This layout uses redundant-loop access to deal with the potential failure of any single component in the path between a system and a disk array.

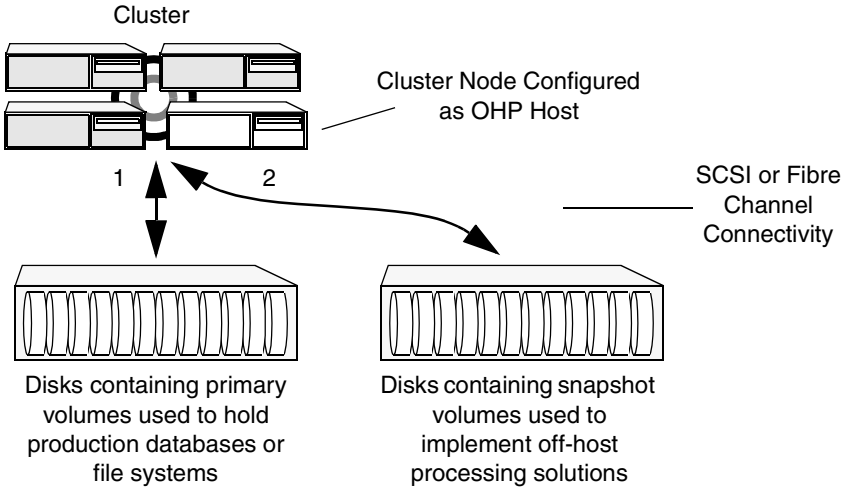
Note On some operating systems, controller names may differ from what is shown here.

Example Connectivity for Off-Host Solution Using Redundant-Loop Access



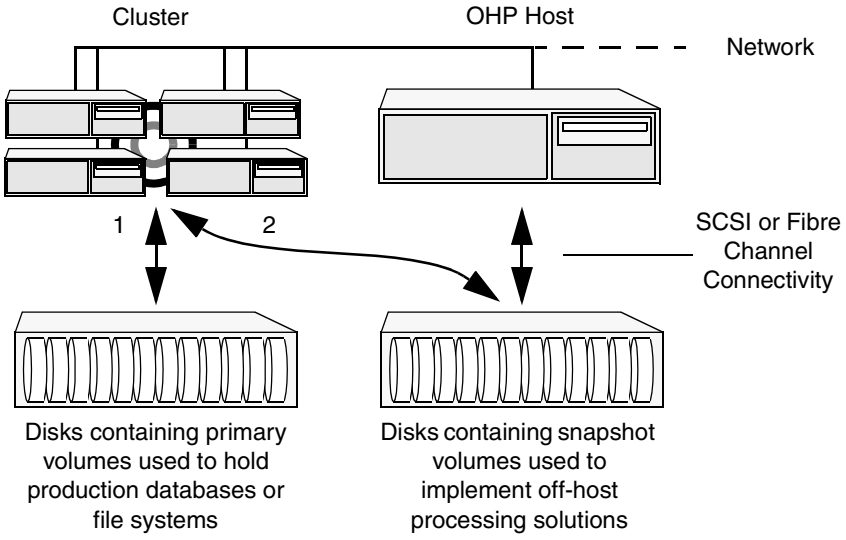
"Example Implementation of an Off-Host Point-In-Time Copy Solution Using a Cluster Node" on page 13 shows how off-host processing might be implemented in a cluster by configuring one of the cluster nodes as the OHP node.

Example Implementation of an Off-Host Point-In-Time Copy Solution Using a Cluster Node



Alternatively, the OHP node could be a separate system that has a network connection to the cluster, but which is not a cluster node and is not connected to the cluster's private network. This arrangement is illustrated in "Example implementation of an Off-Host Point-In-Time Copy Solution Using a Separate OHP Host" on page 14.

Example implementation of an Off-Host Point-In-Time Copy Solution Using a Separate OHP Host

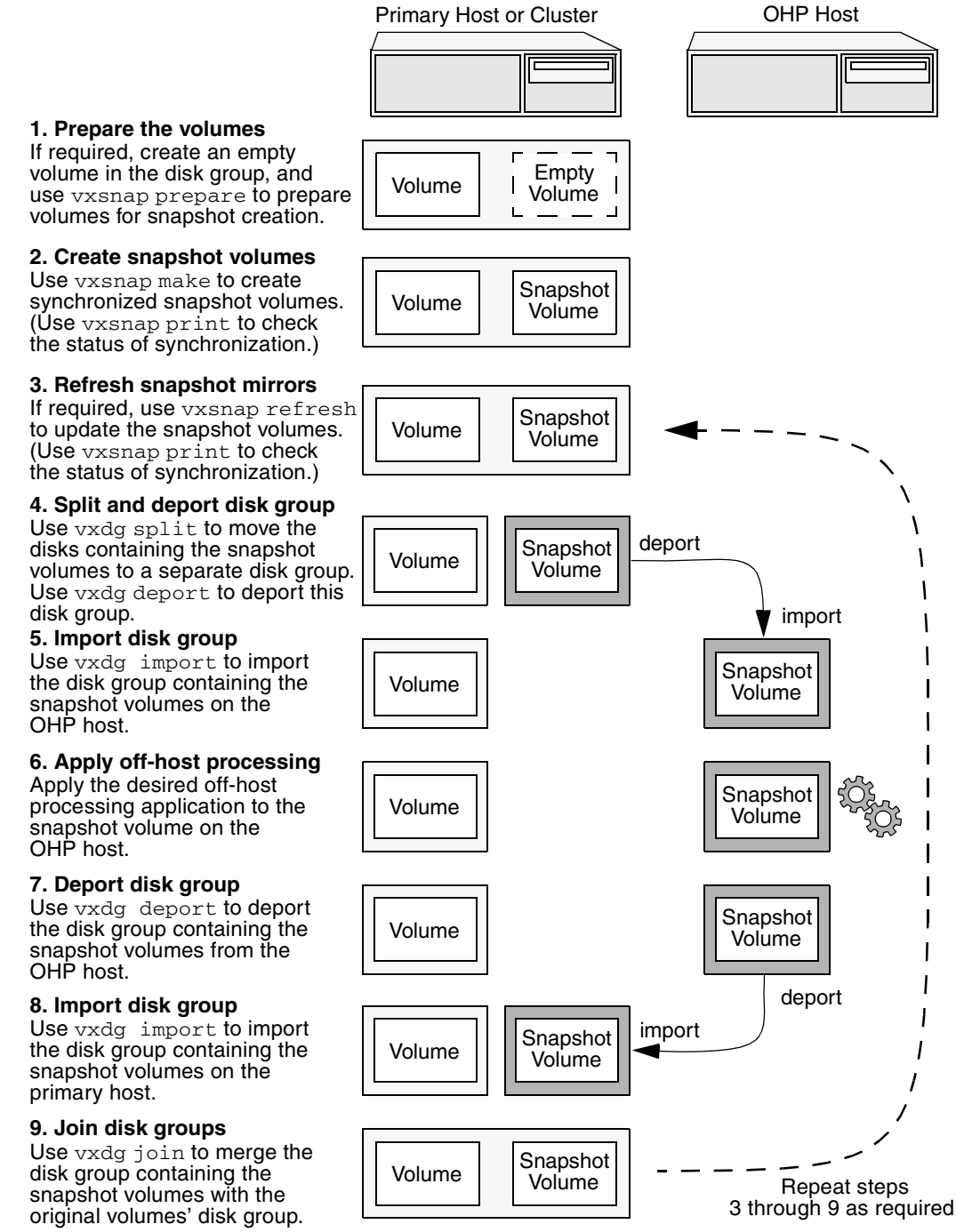


Note For off-host processing, the example scenarios in this document assume that a separate OHP host is dedicated to the backup or decision support role. For clusters, it may be simpler to configure an OHP host that is not a member of the cluster.

"Using VERITAS FlashSnap to implement Off-Host Processing Solutions" on page 15 illustrates the steps that are needed to set up the processing solution on the primary host. Disk Group Split/Join is used to split off snapshot volumes into a separate disk group that is imported on the OHP host.

Note As the snapshot volumes are to be moved into another disk group and then imported on another host, their contents must first be synchronized with the parent volumes. On reimporting the snapshot volumes, refreshing their contents from the original volume is speeded by using FastResync.

Using VERITAS FlashSnap to implement Off-Host Processing Solutions



Data Integrity in Volume Snapshots

A volume snapshot represents the data that exists in a volume at a given point in time. As such, VxVM does not have any knowledge of data that is cached by the overlying file system, or by applications such as databases that have files open in the file system. If the `fsgen` volume usage type is set on a volume that contains a VERITAS File System (VxFS), intent logging of the file system metadata ensures the internal consistency of the file system that is backed up. For other file system types, depending on the intent logging capabilities of the file system, there may potentially be inconsistencies between in-memory data and the data in the snapshot image.

For databases, a suitable mechanism must additionally be used to ensure the integrity of tablespace data when the volume snapshot is taken. The facility to temporarily suspend file system I/O is provided by most modern database software. The examples provided in this document illustrate how to perform this operation. For ordinary files in a file system, which may be open to a wide variety of different applications, there may be no way to ensure the complete integrity of the file data other than by shutting down the applications and temporarily unmounting the file system. In many cases, it may only be important to ensure the integrity of file data that is not in active use at the time that you take the snapshot.

Choices for Snapshot Resynchronization

When a snapshot volume is reattached to its original volume within a shared disk group, there are two choices for resynchronizing the data in the volume:

- *Resynchronize the snapshot from the original volume*—updates the snapshot with data from the primary volume that has changed since the snapshot was taken. The snapshot is then again ready to be taken for the purposes of backup or decision support.
- *Resynchronize the original volume from the snapshot*—updates the original volume with data from the snapshot volume that has changed since the snapshot was taken. This may be necessary to restore the state of a corrupted database or file system, or to implement upgrades to production software, and is usually much quicker than using alternative approaches such as full restoration from backup media.

Setting up Volumes for Instant Snapshots 2

This chapter describes how to make volumes ready for instant snapshot creation. These may be volumes that you want to back up, or that you want to use for decision support or reporting.

If a snapshot volume is to be used on the same host, and will not be moved to another host for off-host processing, you can use space-optimized instant snapshots rather than full-sized instant snapshots. Depending on the application, space-optimized snapshots typically require 10% of the disk space that is required for full-sized instant snapshots.

For more information about administering instant snapshots and FastResync, see the *VERITAS Volume Manager Administrator's Guide*.

The following table summarizes which volumes require the creation of snapshot mirrors for backup, decision support, and database error recovery.

Creation of Snapshot Mirrors

Point-In-Time Copy Application	Create Snapshot Mirrors for Volumes containing...
Online Database Backup	VxFS file systems for database datafiles to be backed up.
Off-Host Cluster File System Backup	VxFS cluster file systems to be backed up.
Decision Support	VxFS file systems for database datafiles to be replicated.

Caution To avoid data inconsistencies, do not use the same snapshot with different point-in-time copy applications. If you require snapshot mirrors for more than one application, configure at least one snapshot mirror that is dedicated to each application.

If the existing volume was created before release 4.0 of VxVM, and it has any attached snapshot plexes, is associated with any snapshot volumes, or has any dedicated DRL logs, follow the procedure given in the section "Upgrading Existing Volumes to Use VxVM 4.0

Features" in the "Administering Volumes" chapter of the *VERITAS Volume Manager Administrator's Guide*. The procedure given in this section assumes that no snapshot plexes, snapshot volumes, or DRL logs are associated with the volumes.

Additional Preparation Activities

Depending on the type of snapshots that you want to create, you may need to perform additional preparatory tasks.

When creating a full-sized instant snapshot, you can use one of the following two methods:

- Break off one or more spare plexes from the original volume to form a snapshot volume with the required redundancy. These plexes must be in or `SNAPDONE` state. (You can also break off named plexes of a volume that are in the `ACTIVE` state, but that method is not described here. For more information, see the "Administering Volume Snapshots" chapter in the *VERITAS Volume Manager Administrator's Guide*.)
- Use a separate empty volume that you have prepared in advance as described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23.

When creating space-optimized instant snapshots that share a cache, you must set up the cache before creating the snapshots. See "Creating a Shared Cache Object" on page 24 for details.

If a space-optimized instant snapshot uses a dedicate cache, this can also be set up when the snapshot is created. No additional preparation is required in this case.

Note The off-host processing solutions in this book require full-sized snapshots.

Preparing a Volume for Instant Snapshot Operations

To prepare a volume for instant snapshot operations, a version 20 Data Change Object (DCO) and DCO volume must first be associated with that volume.

To add a version 20 DCO object and DCO volume to an existing volume, use the following procedure:

1. Ensure that the disk group containing the existing volume has been upgraded to at least version 110. Use the following command to check the version of a disk group:

   ```
   # vxprint -l diskgroup | egrep 'version:'
   ```

To upgrade a disk group, use the following command:

```
# vxdg upgrade diskgroup
```

2. Use the following command to add a version 20 DCO and DCO volume to an existing volume:

```
# vxsnap [-g diskgroup] prepare volume [ndcomirs=number] \
   [regionsize=size] [alloc=storage_attribute[,...]]
```

The `ndcomirs` attribute specifies the number of DCO plexes that are created in the DCO volume. It is recommended that you configure as many DCO plexes as there are data and snapshot plexes in the volume. The DCO plexes are used to set up a DCO volume for any snapshot volume that you subsequently create from the snapshot plexes. For example, specify `ndcomirs=5` for a volume with 3 data plexes and 2 snapshot plexes.

The value of the `regionsize` attribute specifies the size of the tracked regions in the volume. A write to a region is tracked by setting a bit in the change map. The default value is `64k` (64KB). A smaller value requires more disk space for the change maps, but the finer granularity provides faster resynchronization.

You can also specify `vxassist`-style storage attributes to define the disks that can and/or cannot be used for the plexes of the DCO volume.

Note The `vxsnap prepare` command automatically enables Persistent FastResync on the volume. Persistent FastResync is also set automatically on any snapshots that are generated from a volume on which this feature is enabled.

If the volume is a RAID-5 volume, it is converted to a layered volume that can be used with instant snapshots and Persistent FastResync.

By default, a new-style DCO volume contains 32 per-volume maps. If you require more maps than this, you can use the `vxsnap addmap` command to add more maps. See the `vxsnap`(1M) manual page for details of this command.

3. If you are going to create a snapshot volume by breaking off existing plexes, use the following command to add one or more snapshot mirrors to the volume:

```
# vxsnap [-b] [-g diskgroup] addmir volume [nmirror=N] \
  [alloc=storage_attribute[,...]]
```

By default, one snapshot plex is added unless you specify a number using the `nmirror` attribute. For a backup, you should usually only require one plex. The mirrors remain in the `SNAPATT` state until they are fully synchronized. The `-b` option can be used to perform the synchronization in the background. Once synchronized, the mirrors are placed in the `SNAPDONE` state.

For example, the following command adds 2 mirrors to the volume, `vol1`, on disks `mydg10` and `mydg11`:

```
# vxsnap -g mydg addmir vol1 nmirror=2 alloc=mydg10,mydg11
```

Note Do not perform this step if you create a full-sized snapshot volume using a suitably prepared empty volume (see "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23), or if you create space-optimized snapshots that use a cache (see "Creating a Shared Cache Object" on page 24).

If the disks that contain volumes and their snapshots are to be moved into different disk groups, you must ensure that the disks that contain their DCO plexes can accompany them. You can use storage attributes to specify which disks to use for the DCO plexes. (If you do not want to use dirty region logging (DRL) with a volume, you can specify the same disks as those on which the volume is configured, assuming that space is available on the disks). For example, to add a DCO object and DCO volume with plexes on `disk05` and `disk06`, and a region size of 32KB, to the volume, `myvol`, use the following command:

```
# vxsnap -g mydg prepare myvol ndcomirs=2 regionsize=32k \
  alloc=disk05,disk06
```

If required, you can use the `vxassist move` command to relocate DCO plexes to different disks. For example, the following command moves the plexes of the DCO volume for volume `vol1` from `disk03` and `disk04` to `disk07` and `disk08`:

```
# vxassist -g mydg move vol1_dcl !disk03 !disk04 disk07 disk08
```

To view the details of the DCO object and DCO volume that are associated with a volume, use the `vxprint` command. The following is example `vxprint -vh` output for the volume named `zoo` (the TUTIL0 and PUTIL0 columns are omitted for clarity):

```
TY NAME         ASSOC       KSTATE    LENGTH    PLOFFS  STATE   ...
v  zoo          fsgen       ENABLED   1024      -       ACTIVE
pl zoo-01       zoo         ENABLED   1024      -       ACTIVE
sd disk01-01    zoo-01      ENABLED   1024      0       -
pl foo-02       zoo         ENABLED   1024      -       ACTIVE
sd disk02-01    zoo-02      ENABLED   1024      0       -
dc zoo_dco      zoo         -         -         -       -
v  zoo_dcl      gen         ENABLED   132       -       ACTIVE
pl zoo_dcl-01   zoo_dcl     ENABLED   132       -       ACTIVE
sd disk03-01    zoo_dcl-01  ENABLED   132       0       -
pl zoo_dcl-02   zoo_dcl     ENABLED   132       -       ACTIVE
sd disk 04-01   zoo_dcl-02  ENABLED   132       0       -
```

In this output, the DCO object is shown as `zoo_dco`, and the DCO volume as `zoo_dcl` with 2 plexes, `zoo_dcl-01` and `zoo_dcl-02`.

For more information, see "Considerations for Placing DCO Plexes" on page 21, and the `vxassist`(1M) and `vxsnap`(1M) manual pages.

Considerations for Placing DCO Plexes

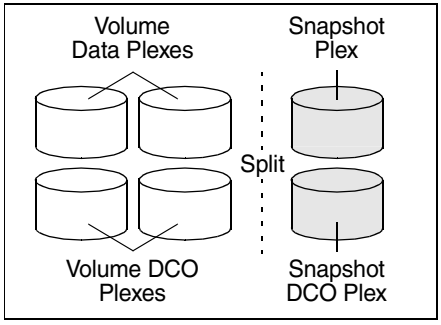
If you use the `vxassist` command or the VERITAS Enterprise Administrator (VEA) to create both a volume and its DCO, or the `vxsnap prepare` command to add a DCO to a volume, the DCO plexes are automatically placed on different disks from the data plexes of the parent volume. In previous releases, version 0 DCO plexes were placed on the same disks as the data plexes for convenience when performing disk group split and move operations. As the version 20 DCOs in VxVM 4.0 support dirty region logging (DRL) in addition to Persistent FastResync, it is preferable for the DCO plexes to be separated from the data plexes. This improves the performance of I/O from/to the volume, and provides resilience for the DRL logs.

If you use the `vxsnap prepare` command to set up a DCO, you must ensure that the disks that contain the plexes of the DCO volume accompany their parent volume during the move. Use the `vxprint` command on a volume to examine the configuration of its associated DCO volume.

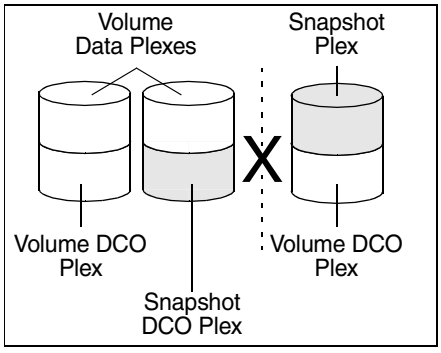
"Examples of Disk Groups That Can and Cannot be Split" on page 22 illustrates some instances in which it is not be possible to split a disk group because of the location of the DCO plexes.

For more information about relocating DCO plexes, see "Preparing a Volume for Instant Snapshot Operations" on page 18.

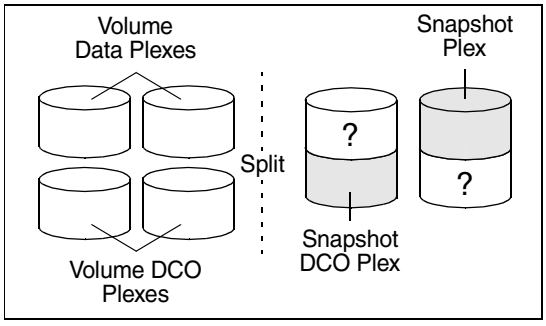
Examples of Disk Groups That Can and Cannot be Split



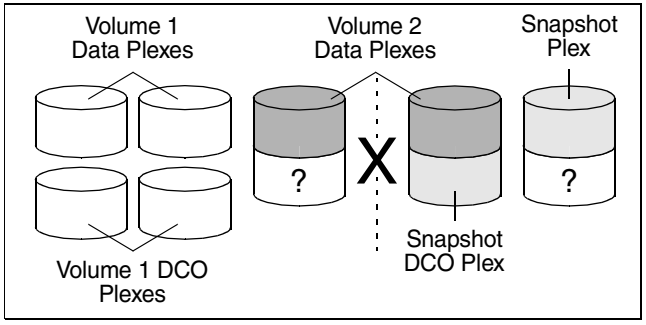
The disk group can be split as the DCO plexes are on dedicated disks, and can therefore accompany the disks that contain the volume data.



The disk group cannot be split as the DCO plexes cannot accompany their volumes. One solution is to relocate the DCO plexes. In this example, use an additional disk in the disk group as an intermediary to swap the misplaced DCO plexes. Alternatively, to improve DRL performance and resilience, allocate the DCO plexes to dedicated disks.



The disk group can be split as the DCO plexes can accompany their volumes. However, you may not wish the data in the portions of the disks marked "?" to be moved as well.



The disk group cannot be split as this would separate the disks that contain the data plexes of Volume 2. Possible solutions are to relocate the snapshot DCO plex to the disk containing the snapshot plex, or to another suitable disk that can be moved.

Creating a Volume for Use as a Full-Sized Instant Snapshot

If you want to create a full-sized instant snapshot for an original volume that does not contain any spare plexes, you can use an empty volume with the required degree of redundancy, and with the same size and same region size as the original volume.

To create an empty volume for use by a full-sized instant snapshot:

1. Use the `vxprint` command on the original volume to find the required size for the snapshot volume.

   ```
   # LEN=`vxprint [-g diskgroup] -F%len volume`
   ```

   Note The command shown in this and subsequent steps assumes that you are using a Bourne-type shell such as `sh`, `ksh` or `bash`. You may need to modify the command for other shells such as `csh` or `tcsh`.

2. Use the `vxprint` command on the original volume to discover the name of its DCO:

   ```
   # DCONAME=`vxprint [-g diskgroup] -F%dco_name volume`
   ```

3. Use the `vxprint` command on the DCO to discover its region size (in blocks):

   ```
   # RSZ=`vxprint [-g diskgroup] -F%regionsz $DCONAME`
   ```

4. Use the `vxassist` command to create a volume, *snapvol*, of the required size and redundancy, together with a version 20 DCO volume with the correct region size:

   ```
   # vxassist [-g diskgroup] make snapvol $LEN \
     [layout=mirror nmirror=number] logtype=dco drl=no dcoversion=20 \
     [ndcomirror=number] regionsz=$RSZ init=active \
     [storage_attributes]
   ```

   Specify the same number of DCO mirrors (`ndcomirror`) as the number of mirrors in the volume (`nmirror`). The `init=active` attribute is used to make the volume available immediately. You can use storage attributes to specify which disks should be used for the volume.

As an alternative to creating the snapshot volume and its DCO volume in a single step, you can first create the volume, and then prepare it for instant snapshot operations as shown here:

```
# vxassist [-g diskgroup] make snapvol $LEN \
  [layout=mirror nmirror=number] init=active [storage_attributes]
# vxsnap [-g diskgroup] prepare snapvol [ndcomirs=number] \
  regionsize=$RSZ [storage_attributes]
```

Creating a Shared Cache Object

If you need to create several instant space-optimized snapshots for the volumes in a disk group, you may find it more convenient to create a single shared cache object in the disk group rather than a separate cache object for each snapshot.

To create a shared cache object:

1. Decide on the following characteristics that you want to allocate to the cache volume that underlies the cache object:
   - The size of the cache volume should be sufficient to record changes to the parent volumes during the interval between snapshot refreshes. A suggested value is 10% of the total size of the parent volumes for a refresh interval of 24 hours.
   - If redundancy is a desired characteristic of the cache volume, it should be mirrored. This increases the space that is required for the cache volume in proportion to the number of mirrors that it has.
   - If the cache volume is mirrored, space is required on at least as many disks as it has mirrors. These disks should not be shared with the disks used for the parent volumes. The disks should also be chosen to avoid impacting I/O performance for critical volumes, or hindering disk group split and join operations.
2. Having decided on its characteristics, use the `vxassist` command to create the volume that is to be used for the cache volume. The following example creates a mirrored cache volume, `cachevol`, with size 1GB in the disk group, `mydg`, on the disks `mydg16` and `mydg17`:

   ```
   # vxassist -g mydg make cachevol 1g layout=mirror init=active \
     mydg16 mydg17
   ```

   The attribute `init=active` is specified to make the cache volume immediately available for use.

3. Use the `vxmake cache` command to create a cache object on top of the cache volume that you created in the previous step:

   ```
   # vxmake [-g diskgroup] cache cache_object cachevolname=volume \
     [regionsize=size] [autogrow=on] [highwatermark=hwmk] \
     [autogrowby=agbvalue] [maxautogrow=maxagbvalue]]
   ```

   If the region size, `regionsize`, is specified, it must be a power of 2, and be greater than or equal to 16KB (`16k`). If not specified, the region size of the cache is set to 64KB.

   Note All space-optimized snapshots that share the cache must have a region size that is equal to or an integer multiple of the region size set on the cache. Snapshot creation also fails if the original volume's region size is smaller than the cache's region size.

   If the cache is to be allowed to grow in size as required, specify `autogrow=on`. By default, the ability to automatically grow the cache is turned off.

   In the following example, the cache object, `cobjmydg`, is created over the cache volume, `cachevol`, the region size of the cache is set to 32KB, and the `autogrow` feature is enabled:

   ```
   # vxmake -g mydg cache cobjmydg cachevolname=cachevol \
     regionsize=32k autogrow=on
   ```

4. Having created the cache object, use the following command to enable it:

   ```
   # vxcache [-g diskgroup] start cache_object
   ```

   For example to start the cache object, `cobjmydg`:

   ```
   # vxcache -g mydg start cobjmydg
   ```

Tuning the autogrow Attributes

The `highwatermark`, `autogrowby` and `maxautogrow` attributes determine how the VxVM cache daemon (`vxcached`) maintains the cache if the `autogrow` feature has been enabled:

- When cache usage reaches the high watermark value, `highwatermark` (default value is 90 percent), and the new required cache size would not exceed the value of `maxautogrow` (default value is twice the size of the cache volume in blocks), `vxcached` grows the size of the cache volume by the value of `autogrowby` (default value is 20% of the size of the cache volume in blocks).
- When cache usage reaches the high watermark value, and the new required cache size would exceed the value of `maxautogrow`, `vxcached` deletes the oldest snapshot in the cache. If there are several snapshots with the same age, the largest of these is deleted.

If the `autogrow` feature has been disabled:

- When cache usage reaches the high watermark value, `vxcached` deletes the oldest snapshot in the cache. If there are several snapshots with the same age, the largest of these is deleted. If there is only a single snapshot, this snapshot is detached and marked as invalid.

Note The `vxcached` daemon does not remove snapshots that are currently open, and it does not remove the last or only snapshot in the cache.

If the cache space becomes exhausted, the snapshot is detached and marked as invalid. If this happens, the snapshot is unrecoverable and must be removed. Enabling the `autogrow` feature on the cache helps to avoid this situation occurring. However, for very small caches (of the order of a few megabytes), it is possible for the cache to become exhausted before the system has time to respond and grow the cache. In such cases, use the `vxcache` command to increase the size of the cache, or to reduce the value of `highwatermark`.

If necessary, you can use the `vxcache set` command to change other `autogrow` attribute values for a cache. For example, you can use the `maxautogrow` attribute to limit the maximum size to which a cache can grow. To estimate this size, consider how much the contents of each source volume are likely to change between snapshot refreshes, and allow some additional space for contingency.

Caution Ensure that the cache is sufficiently large, and that the `autogrow` attributes are configured correctly for your needs.

See the `vxcache`(1M) manual page and the "Administering Volume Snapshots" chapter in the *VERITAS Volume Manager Administrator's Guide* for more information including how to grow, shrink and remove a storage cache.

Online Database Backup 3

Online backup of a database can be implemented by configuring either the primary host or a dedicated separate host to perform the backup operation on snapshot mirrors of the primary host's database.

Two backup methods are described in the following sections:

- Making a Backup of an Online Database on the Same Host
- Making an Off-Host Backup of an Online Database

Note All commands require superuser (`root`) or equivalent privileges, except where it is explicitly stated that a command must be run by the database administrator.

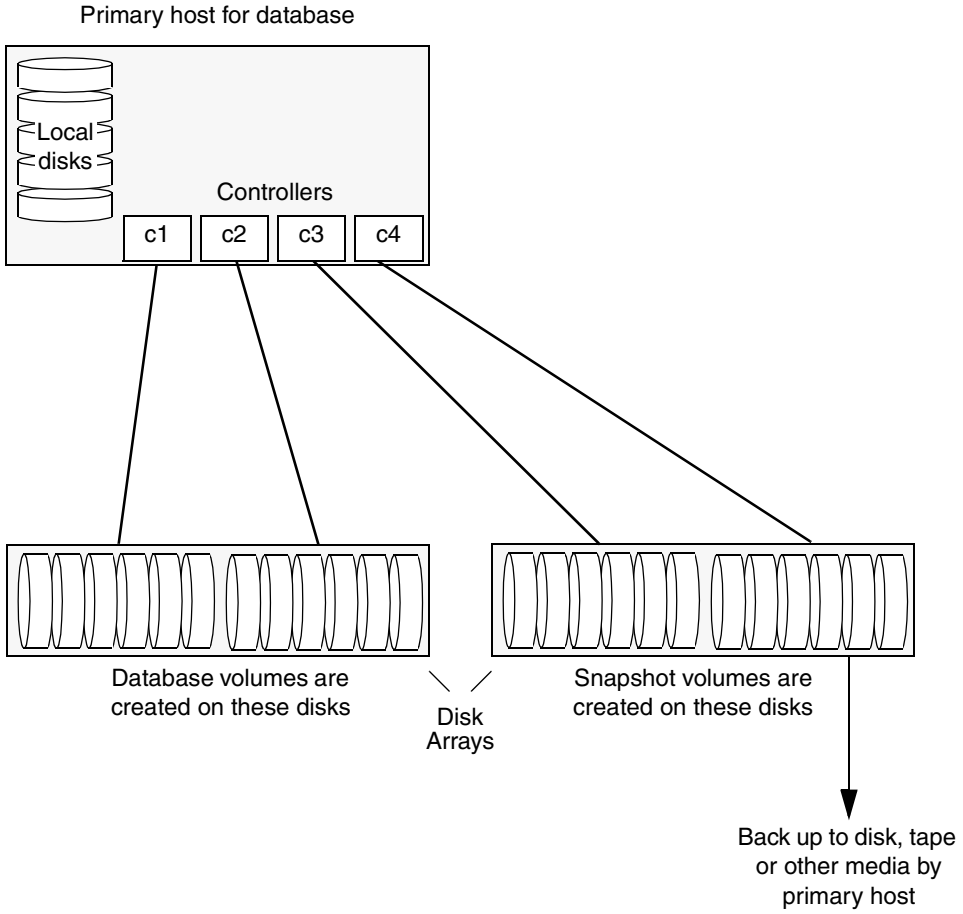
For more information about using snapshots to back up DB2, Oracle and Sybase databases, see the chapter "Using Snapshots for Database Backup" in the *VERITAS Storage Foundation Database Administrator's Guide*.

The following sections include sample scripts:

Making a Backup of an Online Database on the Same Host

As illustrated in "Example System Configuration for Database Backup on the Primary Host" on page 28, the primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, are configured on disks attached to controllers `c1` and `c2`, and the snapshots are created on disks attached to controllers `c3` and `c4`.

Example System Configuration for Database Backup on the Primary Host



Note It is assumed that you have already prepared the volumes containing the file systems for the datafiles to be backed up as described in "Setting up Volumes for Instant Snapshots" on page 17. For an Oracle database, it is not necessary to create snapshots of the volumes containing the file systems for the redo log volumes or archived logs.

To make a backup of an online database on the same host:

1. Use the following command to make a full-sized snapshot, *snapvol*, of the tablespace volume by breaking off plexes from the original volume:

   ```
   # vxsnap -g volumedg make source=volume/newvol=snapvol/nmirror=N
   ```

   The `nmirror` attribute specifies the number of mirrors, *N*, in the snapshot volume.

   If the volume does not have any available plexes, or its layout does not support plex break-off, use the procedure described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23 to prepare an empty volume for the snapshot, and then use the following command to create the snapshot:

   ```
   # vxsnap -g volumedg make source=volume/snapvol=snapvol
   ```

   If a database spans more than one volume, specify all the volumes and their snapshot volumes as separate tuples on the same line, for example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1 \
     source=vol2/newvol=svol2 source=vol3/newvol=svol3
   ```

   If you want to save disk space, you can use the following command to create a space-optimized snapshot instead:

   ```
   # vxsnap -g volumedg make \
     source=volume/newvol=snapvol/cache=cacheobject
   ```

   The argument *cacheobject* is the name of a pre-existing cache that you have created in the disk group for use with space-optimized snapshots. See "Creating a Shared Cache Object" on page 24 for more information.

   If several space-optimized snapshots are to be created at the same time, these can all specify the same cache object as shown in this example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1/cache=dbaseco \
     source=vol2/newvol=svol2/cache=dbaseco \
     source=vol3/newvol=svol3/cache=dbaseco
   ```

   Note This step sets up the snapshot volumes ready for the backup cycle, and starts tracking changes to the original volumes. When you are ready to make a backup, proceed to step 2.

2. If the volumes to be backed up contain database tables in file systems, suspend updates to the volumes:

   - DB2 provides the `write suspend` command to temporarily suspend I/O activity for a database. As the DB2 database administrator, use a script such as that shown in "Script to Suspend I/O for a DB2 Database" on page 71. Note that to allow recovery from any backups taken from snapshots, the database must be in LOGRETAIN RECOVERY mode.
   - Oracle supports online backup by temporarily suspending updates to the datafiles of the tablespaces, provided that the database is running in archive mode and the tablespaces are online. As the Oracle database administrator, put each tablespace into hot backup mode using a script such as that shown in "Script to Put Oracle Database into Hot Backup Mode" on page 69.
   - Sybase ASE from version 12.0 onward provides the Quiesce feature to allow temporary suspension of writes to a database. As the Sybase database administrator, put the database in quiesce mode by using a script such as that shown in "Script to Quiesce Sybase ASE Database" on page 70.

3. Refresh the contents of the snapshot volumes from the original volume using the following command:

   ```
   # vxsnap -g volumedg refresh snapvol source=vol \
     [snapvol2 source=vol2]...
   ```

   For example, to refresh the snapshots `svol1`, `svol2` and `svol3`:

   ```
   # vxsnap -g dbasedg refresh svol1 source=vol1 svol2 source=vol2 \
     svol3 source=vol3
   ```

4. If you temporarily suspended updates to volumes in step 1, release all the tablespaces or databases from suspend, hot backup or quiesce mode:

   - As the DB2 database administrator, use a script such as that shown in "Script to Resume I/O for a DB2 Database" on page 74.
   - As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in "Script to End Oracle Database Hot Backup Mode" on page 72.
   - As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in "Script to Release Sybase ASE Database from Quiesce Mode" on page 73.

5. Back up the snapshot volume. If you need to remount the file system in the volume to back it up, first run `fsck` on the volume. The following are sample commands for checking and mounting a file system:

```
# fsck -F vxfs /dev/vx/rdsk/snapvoldg/snapvol
# mount -F vxfs /dev/vx/dsk/snapvoldg/snapvol mount_point
```

Note On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

Back up the file system at this point using a command such as `bpbackup` in VERITAS NetBackup. After the backup is complete, use the following command to unmount the file system.

```
# unmount mount_point
```

Repeat steps 2 through 5 each time that you need to back up the volume.

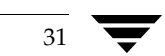
In some instances, such as recovering the contents of a corrupted volume, it may be useful to resynchronize a volume from its snapshot volume (which is used as a hot standby):

```
# vxsnap -g diskgroup restore volume source=snapvol destroy=yes|no
```

The `destroy` attribute specifies whether the plexes of the snapshot volume are to be reattached to the original volume. For example, to resynchronize the volume `dbase_vol` from its snapshot volume `snap2_dbase_vol` without removing the snapshot volume:

```
# vxsnap -g dbasedg restore dbase_vol source=snap2_dbase_vol \
destroy=no
```

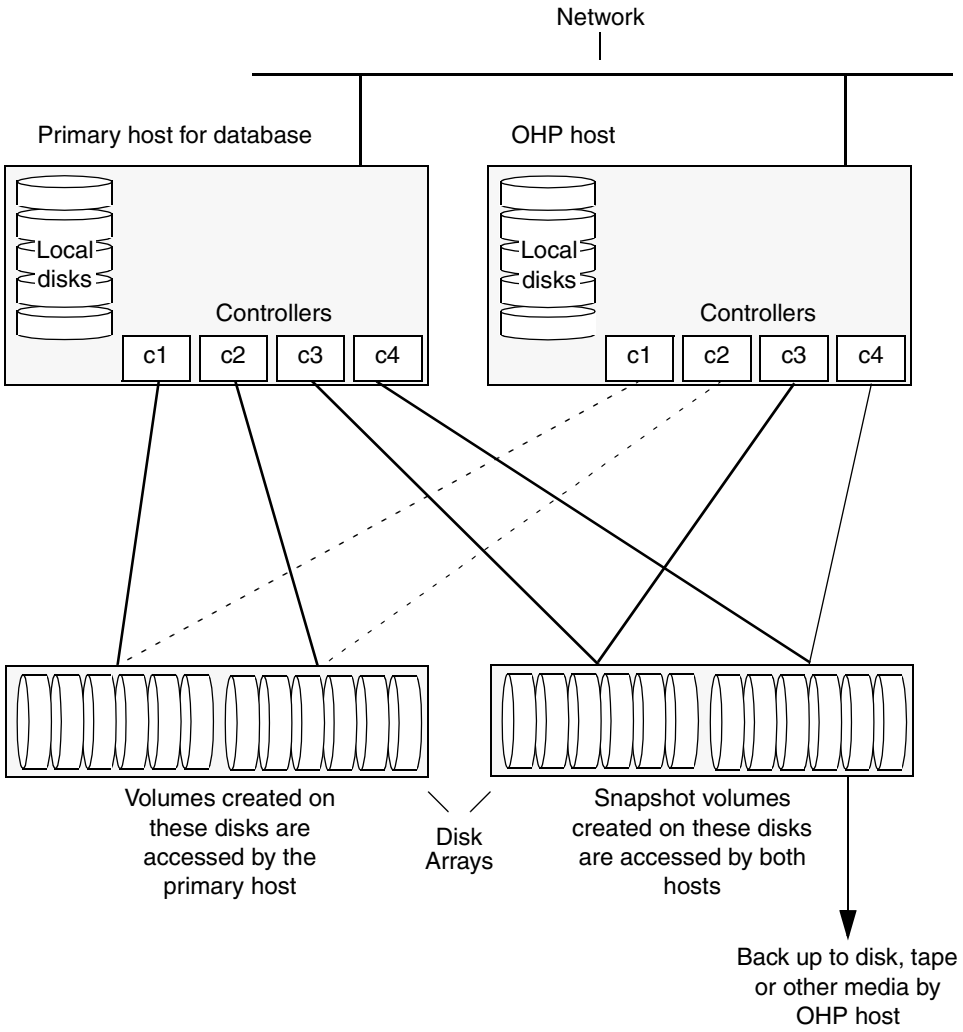
Note You must shut down the database and unmount the file system that is configured on the original volume before attempting to resynchronize its contents from a snapshot.



Making an Off-Host Backup of an Online Database

As illustrated in "Example System Configuration for Off-Host Database Backup" on page 32, the primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, are configured on disks attached to controllers `c1` and `c2`, and the snapshots are created on disks attached to controllers `c3` and `c4`. There is no requirement for the OHP host to have access to the disks that contain the primary database volumes.

Example System Configuration for Off-Host Database Backup



Note It is assumed that you have already prepared the volumes containing the file systems for the datafiles to be backed up as described in "Setting up Volumes for Instant Snapshots" on page 17. For an Oracle database, it is not necessary to create snapshots of the volumes containing the file systems for the redo log volumes or archived logs.

If the database is configured on volumes in a cluster-shareable disk group, it is assumed that the primary host for the database is the master node for the cluster. If the primary host is not also the master node, all VxVM operations on shared disk groups must be performed on the master node.

To make an off-host backup of an online database:

1. On the primary host, use the following command to make a full-sized snapshot, *snapvol*, of the tablespace volume by breaking off plexes from the original volume:

   ```
   # vxsnap -g volumedg make source=volume/newvol=snapvol/nmirror=N
   ```

   The `nmirror` attribute specifies the number of mirrors, *N*, in the snapshot volume.

   If the volume does not have any available plexes, or its layout does not support plex break-off, use the procedure described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23 to prepare an empty volume for the snapshot, and then use the following command to create the snapshot:

   ```
   # vxsnap -g volumedg make source=volume/snapvol=snapvol
   ```

   If a database spans more than one volume, specify all the volumes and their snapshot volumes as separate tuples on the same line, for example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1 \
     source=vol2/newvol=svol2 source=vol3/newvol=svol3
   ```

   Note This step sets up the snapshot volumes ready for the backup cycle, and starts tracking changes to the original volumes. When you are ready to make a backup, proceed to step 2.

2. If the volumes to be backed up contain database tables in file systems, suspend updates to the volumes:

   - DB2 provides the `write suspend` command to temporarily suspend I/O activity for a database. As the DB2 database administrator, use a script such as that shown in "Script to Suspend I/O for a DB2 Database" on page 71. Note that to allow recovery from any backups taken from snapshots, the database must be in LOGRETAIN RECOVERY mode.

- Oracle supports online backup by temporarily suspending updates to the datafiles of the tablespaces, provided that the database is running in archive mode and the tablespaces are online. As the Oracle database administrator, put each tablespace into hot backup mode using a script such as that shown in "Script to Put Oracle Database into Hot Backup Mode" on page 69.
- Sybase ASE from version 12.0 onward provides the Quiesce feature to allow temporary suspension of writes to a database. As the Sybase database administrator, put the database in quiesce mode by using a script such as that shown in "Script to Quiesce Sybase ASE Database" on page 70.

3. On the primary host, refresh the contents of the snapshot volumes from the original volume using the following command:

```
# vxsnap -g volumedg refresh snapvol source=vol \
  [snapvol2 source=vol2]... syncing=yes
```

The `syncing=yes` attribute starts a synchronization of the snapshot in the background.

For example, to refresh the snapshots `svol1`, `svol2` and `svol3`:

```
# vxsnap -g dbasedg refresh svol1 source=vol1 svol2 source=vol2 \
  svol3 source=vol3 syncing=yes
```

4. If you temporarily suspended updates to volumes in step 1, release all the tablespaces or databases from suspend, hot backup or quiesce mode:
   - As the DB2 database administrator, use a script such as that shown in "Script to Resume I/O for a DB2 Database" on page 74.
   - As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in "Script to End Oracle Database Hot Backup Mode" on page 72.
   - As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in "Script to Release Sybase ASE Database from Quiesce Mode" on page 73.

5. Use the following command to wait for the contents of the snapshot to be fully synchronous with the contents of the original volume:

```
# vxsnap -g volumedg syncwait snapvol
```

For example, to wait for synchronization to finish for all the snapshots `svol1`, `svol2` and `svol3`, you would issue three separate commands:

```
# vxsnap -g dbasedg syncwait svol1
# vxsnap -g dbasedg syncwait svol2
# vxsnap -g dbasedg syncwait svol3
```

Note You cannot move a snapshot volume into a different disk group until synchronization of its contents is complete. You can use the `vxsnap print` command to check on the progress of synchronization.

6. On the primary host, use the following command to split the disks containing the snapshot volumes into a separate disk group, *snapvoldg*, from the original disk group, *volumedg*:

   ```
   # vxdg split volumedg snapvoldg snapvol ...
   ```

7. On the primary host, deport the snapshot volume's disk group using the following command:

   ```
   # vxdg deport snapvoldg
   ```

8. On the OHP host where the backup is to be performed, use the following command to import the snapshot volume's disk group:

   ```
   # vxdg import snapvoldg
   ```

9. The snapshot volumes are initially disabled following the split. Use the following commands on the OHP host to recover and restart the snapshot volumes:

   ```
   # vxrecover -g snapvoldg -m snapvol ...
   # vxvol -g snapvoldg start snapvol ...
   ```

10. On the OHP host, back up the snapshot volumes. If you need to remount the file system in the volume to back it up, first run `fsck` on the volumes. The following are sample commands for checking and mounting a file system:

    ```
    # fsck -F vxfs /dev/vx/rdsk/snapvoldg/snapvol
    # mount -F vxfs /dev/vx/dsk/snapvoldg/snapvol mount_point
    ```

    Note On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

    Back up the file system at this point using a command such as `bpbackup` in VERITAS NetBackup. After the backup is complete, use the following command to unmount the file system.

    ```
    # unmount mount_point
    ```


11. On the OHP host, use the following command to deport the snapshot volume's disk group:

```
# vxdg deport snapvoldg
```

12. On the primary host, re-import the snapshot volume's disk group using the following command:

```
# vxdg [-s] import snapvoldg
```

Note Specify the `-s` option if you are reimporting the disk group to be rejoined with a shared disk group in a cluster.

13. On the primary host, use the following command to rejoin the snapshot volume's disk group with the original volume's disk group:

```
# vxdg join snapvoldg volumedg
```

14. The snapshot volume is initially disabled following the join. Use the following commands on the primary host to recover and restart a snapshot volume:

```
# vxrecover -g volumedg -m snapvol
# vxvol -g volumedg start snapvol
```

Repeat steps 2 through 14 each time that you need to back up the volume.

For an example of a script that uses this method, see "Script to Initiate Online Off-Host Oracle Database Backup" on page 67.

In some instances, such as recovering the contents of a corrupted volume, it may be useful to resynchronize a volume from its snapshot volume (which is used as a hot standby):

```
# vxsnap -g diskgroup restore volume source=snapvol destroy=yes|no
```

The `destroy` attribute specifies whether the plexes of the snapshot volume are to be reattached to the original volume. For example, to resynchronize the volume `dbase_vol` from its snapshot volume `snap2_dbase_vol` without removing the snapshot volume:

```
# vxsnap -g dbasedg restore dbase_vol source=snap2_dbase_vol \
  destroy=no
```

Note You must shut down the database and unmount the file system that is configured on the original volume before attempting to resynchronize its contents from a snapshot.

Reattaching Snapshot Plexes

Note This operation is not supported for space-optimized instant snapshots.

Using the following command, some or all plexes of an instant snapshot may be reattached to the specified original volume, or to a source volume in the snapshot hierarchy above the snapshot volume:

```
# vxsnap [-g diskgroup] reattach snapvol source=vol \
  [nmirror=number]
```

By default, all the plexes are reattached, which results in the removal of the snapshot. If required, the number of plexes to be reattached may be specified as the value assigned to the `nmirror` attribute.

Note The snapshot being reattached must not be open to any application. For example, any file system configured on the snapshot volume must first be unmounted.

For example the following command reattaches 1 plex from the snapshot volume, `snapmyvol`, to the volume, `myvol`:

```
# vxsnap -g mydg reattach snapmyvol source=myvol nmirror=1
```

While the reattached plexes are being resynchronized from the data in the parent volume, they remain in the `SNAPTMP` state. After resynchronization is complete, the plexes are placed in the `SNAPDONE` state.

Off-Host Cluster File System Backup 4

VERITAS Cluster File System (CFS) allows cluster nodes to share access to the same file system. CFS is especially useful for sharing read-intensive data between cluster nodes.

Off-host backup of cluster file systems may be implemented by taking a snapshot of the volume containing the file system and performing the backup operation on a separate host.

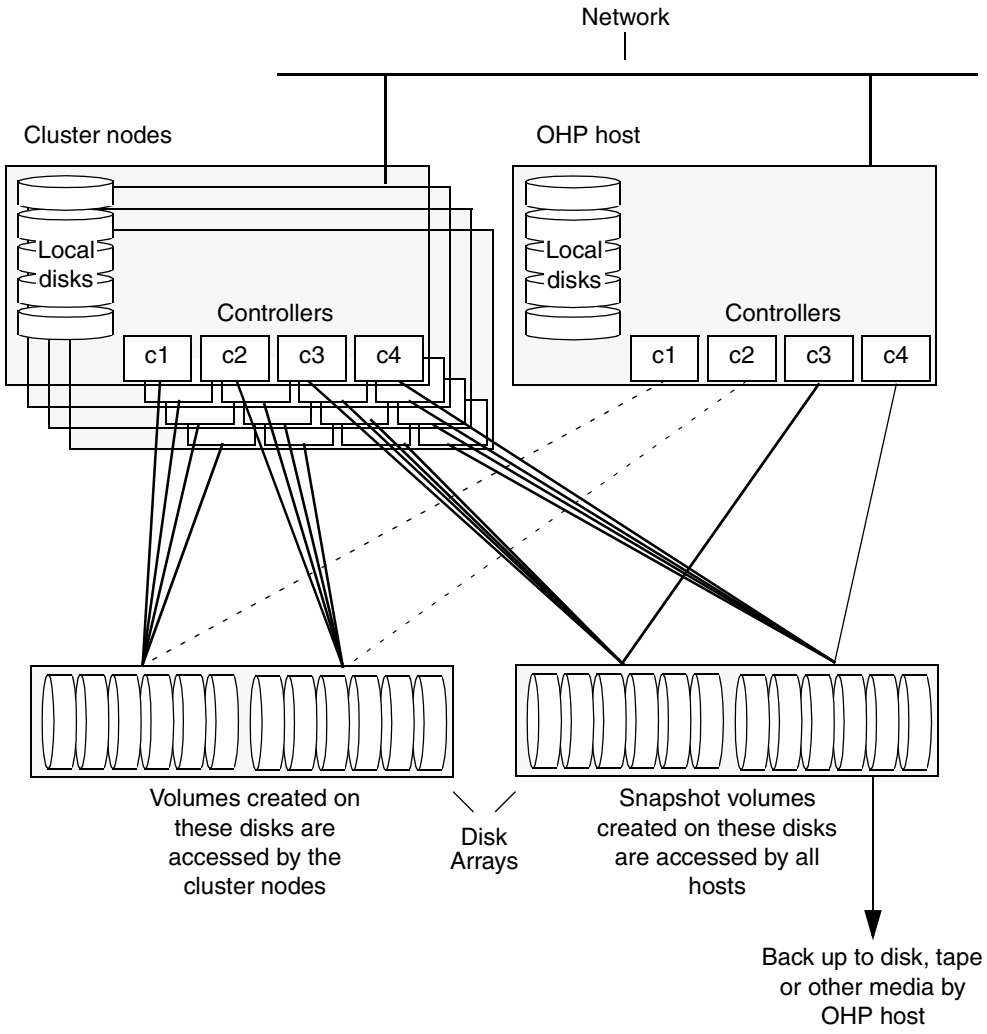
As illustrated in "System Configuration for Off-Host File System Backup Scenarios" on page 40, the primary volume that contains the file system to be backed up is configured on disks attached to controllers `c1` and `c2`, and the snapshots are created on disks attached to controllers `c3` and `c4`.

See "Mounting a File System for Shared Access" on page 41 for a description of how to mount a VxFS file system for shared access by the nodes of a cluster.

See "Using Off-Host Processing to Back Up Cluster File Systems" on page 41 for a description of how to perform off-host backups of cluster-shared file systems.

Note All commands require superuser (`root`) or equivalent privileges.

System Configuration for Off-Host File System Backup Scenarios



Mounting a File System for Shared Access

To mount a VxFS file system for shared access, use the following command on each cluster node where required:

```
# mount -F vxfs -o cluster /dev/vx/dsk/diskgroup/volume mount_point
```

For example, to mount the volume `cfs_vol` in the disk group `exampledg` for shared access on the mount point, `/mnt_pnt`:

```
# mount -F vxfs -o cluster /dev/vx/dsk/exampledg/cfs_vol /mnt_pnt
```

Using Off-Host Processing to Back Up Cluster File Systems

Note It is assumed that you have already prepared the volumes containing the file systems that are to be backed up as described in "Setting up Volumes for Instant Snapshots" on page 17.

To back up a snapshot of a mounted file system which has shared access, perform the following steps:

1. On the master node of the cluster, use the following command to make a full-sized snapshot, *snapvol*, of the volume containing the file system by breaking off plexes from the original volume:

   ```
   # vxsnap -g volumedg make source=volume/newvol=snapvol/nmirror=N
   ```

   The `nmirror` attribute specifies the number of mirrors, *N*, in the snapshot volume.

   For example, to take a snapshot of the volume `cfs_vol` in the shared disk group `exampledg`:

   ```
   # vxsnap -g exampledg make source=cfs_vol/newvol=scfs_vol
   ```

   If the volume does not have any available plexes, or its layout does not support plex break-off, use the procedure described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23 to prepare an empty volume for the snapshot, and then use the following command to create the snapshot:

   ```
   # vxsnap -g volumedg make source=volume/snapvol=snapvol
   ```

   Note This step sets up the snapshot volumes ready for the backup cycle, and starts tracking changes to the original volumes. When you are ready to make a backup, proceed to step 2.

2. On the master node, refresh the contents of the snapshot volumes from the original volume using the following command:

```
# vxsnap -g volumedg refresh snapvol source=vol \
  [snapvol2 source=vol2]... syncing=yes
```

The `syncing=yes` attribute starts a synchronization of the snapshot in the background.

For example, to refresh the snapshot `scfs_vol`:

```
# vxsnap -g exampledg refresh scfs_vol source=cfs_vol syncing=yes
```

3. On the master node, use the following command to wait for the contents of the snapshot to be fully synchronous with the contents of the original volume:

```
# vxsnap -g volumedg syncwait snapvol
```

For example, to wait for synchronization to finish for the snapshots `scfs_vol`:

```
# vxsnap -g exampledg syncwait scfs_vol
```

Note You cannot move a snapshot volume into a different disk group until synchronization of its contents is complete. You can use the `vxsnap print` command to check on the progress of synchronization.

4. On the master node, use the following command to split the snapshot volume into a separate disk group, *snapvoldg*, from the original disk group, *volumedg*:

```
# vxdg split volumedg snapvoldg snapvol
```

For example, to place the snapshot of the volume `cfs_vol` into the shared disk group `splitdg`:

```
# vxdg split exampledg splitdg scfs_vol
```

5. On the master node, deport the snapshot volume's disk group using the following command:

```
# vxdg deport snapvoldg
```

For example, to deport the disk group `splitdg`:

```
# vxdg deport splitdg
```

6. On the OHP host where the backup is to be performed, use the following command to import the snapshot volume's disk group:

```
# vxdg import snapvoldg
```

For example, to import the disk group `splitdg`:

```
# vxdg import splitdg
```

7. The snapshot volume is initially disabled following the split. Use the following commands on the OHP host to recover and restart the snapshot volume:

```
# vxrecover -g snapvoldg -m snapvol
# vxvol -g snapvoldg start snapvol
```

8. On the OHP host, use the following commands to check and *locally* mount the snapshot volume:

```
# fsck -F vxfs /dev/vx/rdsk/diskgroup/volume
# mount -F vxfs /dev/vx/dsk/diskgroup/volume mount_point
```

Note On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

For example, to check and mount the volume `scfs_vol` in the disk group `exampledg` for shared access on the mount point, `/bak/mnt_pnt`:

```
# fsck -F vxfs /dev/vx/rdsk/exampledg/scfs_vol
# mount -F vxfs /dev/vx/dsk/exampledg/scfs_vol /bak/mnt_pnt
```

9. Back up the file system at this point using a command such as `bpbackup` in VERITAS NetBackup. After the backup is complete, use the following command to unmount the file system.

```
# unmount mount_point
```

10. On the OHP host, use the following command to deport the snapshot volume's disk group:

```
# vxdg deport snapvoldg
```

11. On the master node, re-import the snapshot volume's disk group as a shared disk group using the following command:

```
# vxdg -s import snapvoldg
```

12. On the master node, use the following command to rejoin the snapshot volume's disk group with the original volume's disk group:

```
# vxdg join snapvoldg volumedg
```

For example, to join disk group `splitdg` with `exampledg`:

```
# vxdg join splitdg exampledg
```

13. The snapshot volume is initially disabled following the join. Use the following commands on the primary host to recover and restart the snapshot volume:

```
# vxrecover -g volumedg -m snapvol
```


```
# vxvol -g volumedg start snapvol
```

14. When the backup is complete, use the following command to unmount the snapshot volume, and make it ready for its contents to be refreshed from the primary volume:

    ```
    # umount mount_point
    ```

    When synchronization is complete, the snapshot is ready to be re-used for backup.

Caution Before attempting to unmount the snapshot, shut down all applications that access a file system in the snapshot volume, and also unmount any such file system.

Repeat steps 2 through 14 each time that you need to back up the volume.

In some instances, such as recovering the contents of a corrupted volume, it may be useful to resynchronize a volume from its snapshot volume (which is used as a hot standby):

```
# vxsnap -g diskgroup restore volume source=snapvol destroy=yes|no
```

The `destroy` attribute specifies whether the plexes of the snapshot volume are to be reattached to the original volume. For example, to resynchronize the volume `cfs_vol` from its snapshot volume `scfs_vol`:

```
# vxsnap -g exampledg restore cfs_vol source=scfs_vol destroy=no
```

Note You must unmount the file system that is configured on the original volume before attempting to resynchronize its contents from a snapshot.

Reattaching Snapshot Plexes

Note This operation is not supported for space-optimized instant snapshots.

Using the following command, some or all plexes of an instant snapshot may be reattached to the specified original volume, or to a source volume in the snapshot hierarchy above the snapshot volume:

```
# vxsnap [-g diskgroup] reattach snapvol source=vol \
  [nmirror=number]
```

By default, all the plexes are reattached, which results in the removal of the snapshot. If required, the number of plexes to be reattached may be specified as the value assigned to the `nmirror` attribute.

Note The snapshot being reattached must not be open to any application. For example, any file system configured on the snapshot volume must first be unmounted.

For example the following command reattaches 1 plex from the snapshot volume, `snapmyvol`, to the volume, `myvol`:

```
# vxsnap -g mydg reattach snapmyvol source=myvol nmirror=1
```

While the reattached plexes are being resynchronized from the data in the parent volume, they remain in the `SNAPTMP` state. After resynchronization is complete, the plexes are placed in the `SNAPDONE` state.

Decision Support 5

You can use snapshots of a primary database to create a replica of the database at a given moment in time. You can then implement decision support analysis and report generation operations that take their data from the database copy rather than from the primary database. The FastResync functionality of VERITAS Volume Manager (VxVM) allows you to quickly refresh the database copy with up-to-date information from the primary database. Reducing the time taken to update decision support data also lets you generate analysis reports more frequently.

Two methods are described for setting up a replica database for decision support:

- Creating a Replica Database on the Same Host
- Creating up an Off-Host Replica Database

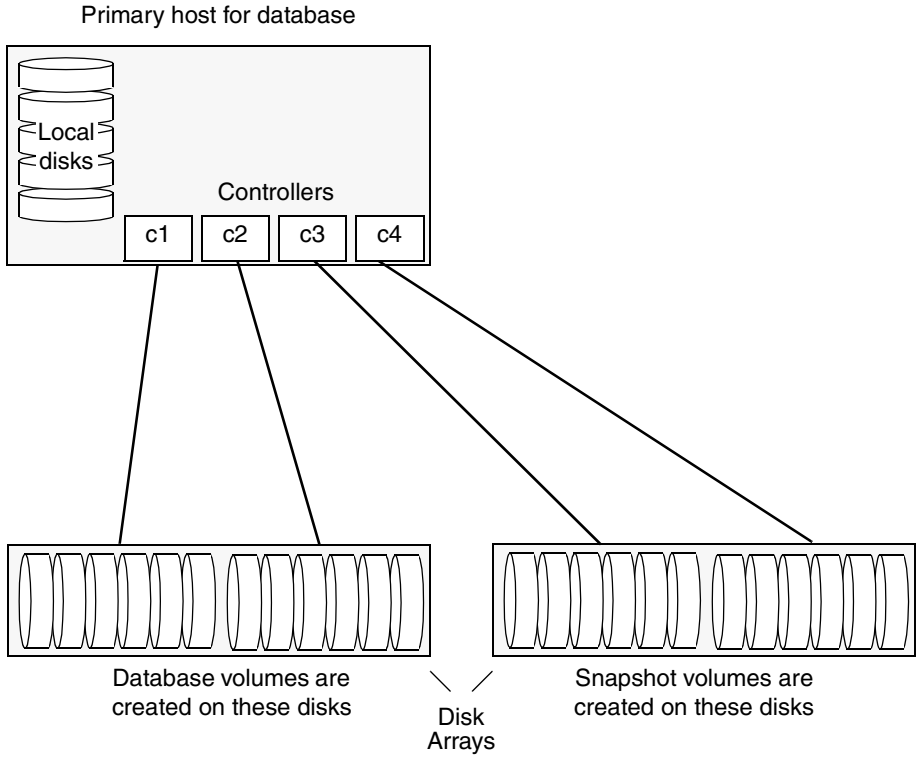
Note All commands require superuser (`root`) or equivalent privileges, except where it is explicitly stated that a command must be run by the database administrator.

The following sections include sample scripts:

Creating a Replica Database on the Same Host

As illustrated in "Example System Configuration for Decision Support on the Primary Host" on page 48, the primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, are configured on disks attached to controllers `c1` and `c2`, and the snapshots are created on disks attached to controllers `c3` and `c4`.

Example System Configuration for Decision Support on the Primary Host



Note It is assumed that you have already prepared the database volumes to be replicated as described in "Setting up Volumes for Instant Snapshots" on page 17.

To set up a replica database to be used for decision support on the primary host:

1. If you have not already done so, prepare the host to use the snapshot volume that contains the copy of the database tables. Set up any new database logs and configuration files that are required to initialize the database.

2. Use the following command to make a full-sized snapshot, *snapvol*, of the tablespace volume by breaking off plexes from the original volume:

   ```
   # vxsnap -g volumedg make source=volume/newvol=snapvol/nmirror=N
   ```

   The `nmirror` attribute specifies the number of mirrors, *N*, in the snapshot volume.

   If the volume does not have any available plexes, or its layout does not support plex break-off, use the procedure described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23 to prepare an empty volume for the snapshot, and then use the following command to create the snapshot:

   ```
   # vxsnap -g volumedg make source=volume/snapvol=snapvol
   ```

   If a database spans more than one volume, specify all the volumes and their snapshot volumes as separate tuples on the same line, for example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1/nmirror=2 \
     source=vol2/newvol=svol2/nmirror=2 \
     source=vol3/newvol=svol3/nmirror=2
   ```

   If you want to save disk space, you can use the following command to create a space-optimized snapshot instead:

   ```
   # vxsnap -g volumedg make \
     source=volume/newvol=snapvol/cache=cacheobject
   ```

   The argument *cacheobject* is the name of a pre-existing cache that you have created in the disk group for use with space-optimized snapshots. See "Creating a Shared Cache Object" on page 24 for more information.

   If several space-optimized snapshots are to be created at the same time, these can all specify the same cache object as shown in this example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1/cache=dbaseco \
     source=vol2/newvol=svol2/cache=dbaseco \
     source=vol3/newvol=svol3/cache=dbaseco
   ```

   See the section "Creating a Share Cache Object" in the "Administering Volume Snapshots" chapter of the *VERITAS Volume Manager Administrator's Guide* for more information.

Note This step sets up the snapshot volumes, and starts tracking changes to the original volumes. When you are ready to create the replica database, proceed to step 3.

3. If the volumes to be backed up contain database tables in file systems, suspend updates to the volumes:

   - DB2 provides the `write suspend` command to temporarily suspend I/O activity for a database. As the DB2 database administrator, use a script such as that shown in "Script to Suspend I/O for a DB2 Database" on page 71. Note that to allow recovery from any backups taken from snapshots, the database must be in LOGRETAIN RECOVERY mode.
   - Oracle supports online backup by temporarily suspending updates to the datafiles of the tablespaces, provided that the database is running in archive mode and the tablespaces are online. As the Oracle database administrator, put each tablespace into hot backup mode using a script such as that shown in "Script to Put Oracle Database into Hot Backup Mode" on page 69.
   - Sybase ASE from version 12.0 onward provides the Quiesce feature to allow temporary suspension of writes to a database. As the Sybase database administrator, put the database in quiesce mode by using a script such as that shown in "Script to Quiesce Sybase ASE Database" on page 70.

     If you are using Sybase ASE 12.5, you can specify the `for external dump` clause to the `quiesce` command. This *warm standby* method allows you to update a replica database using transaction logs dumped from the primary database. See "Updating a Warm Standby Sybase ASE 12.5 Database" on page 59 for more information.

4. Refresh the contents of the snapshot volumes from the original volume using the following command:

   ```
   # vxsnap -g volumedg refresh snapvol source=vol \
     [snapvol2 source=vol2]...
   ```

   For example, to refresh the snapshots `svol1`, `svol2` and `svol3`:

   ```
   # vxsnap -g dbasedg refresh svol1 source=vol1 svol2 source=vol2 \
     svol3 source=vol3
   ```

5. If you temporarily suspended updates to volumes in step 2, release all the tablespaces or databases from suspend, hot backup or quiesce mode:

   - As the DB2 database administrator, use a script such as that shown in "Script to Resume I/O for a DB2 Database" on page 74.

- As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in "Script to End Oracle Database Hot Backup Mode" on page 72.
- As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in "Script to Release Sybase ASE Database from Quiesce Mode" on page 73.

  If you are using Sybase ASE 12.5, you can specify the `for external dump` clause to the `quiesce` command. This *warm standby* method allows you to update a replica database using transaction logs dumped from the primary database. See "Updating a Warm Standby Sybase ASE 12.5 Database" on page 59 for more information.

6. For each snapshot volume containing tablespaces, check the file system that it contains, and mount the volume using the following commands:

```
# fsck -F vxfs /dev/vx/rdsk/diskgroup/snapvol
# mount -F vxfs /dev/vx/dsk/diskgroup/snapvol  mount_point
```

Note On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

For example, to check the file system in the snapshot volume `snap1_dbase_vol`, and mount it on `/rep_dbase_vol`:

```
# fsck -F vxfs /dev/vx/rdsk/dbasedg/snap1_dbase_vol
# mount -F vxfs /dev/vx/dsk/dbasedg/snap1_dbase_vol \
  /rep_dbase_vol
```

7. Copy any required log files from the primary database to the replica database.
   - For an Oracle database, copy the archived log files that were generated while the database was in hot backup mode to the new database's archived log directory (for example, `/rep_archlog`).
   - For a Sybase ASE database, if you specified the `for external dump` clause when you quiesced the database, use the following `isql` command as the database administrator to dump the transaction log for the database:

     ```
     dump transaction to dump_device with standby_access
     ```

     Then copy the dumped transaction log to the appropriate replica database directory.

8. As the database administrator, start the new database:
   - For an Oracle database, use a script such as that shown in "Script to Complete, Recover and Start Replica Oracle Database" on page 78.

- For a Sybase ASE database, use a script such as that shown in "Script to Start Replica Sybase ASE Database" on page 80.

  If you are using the warm standby method, specify the `-q` option to the `dataserver` command. Use the following `isql` commands to load the dump of the transaction log and put the database online:

  ```
  load transaction from dump_device with standby_access
  online database database_name for standby_access
  ```

  If you are not using the warm standby method, use the following `isql` command to recover the database, roll back any uncommitted transactions to the time that the quiesce command was issued, and put the database online:

  ```
  online database database_name
  ```

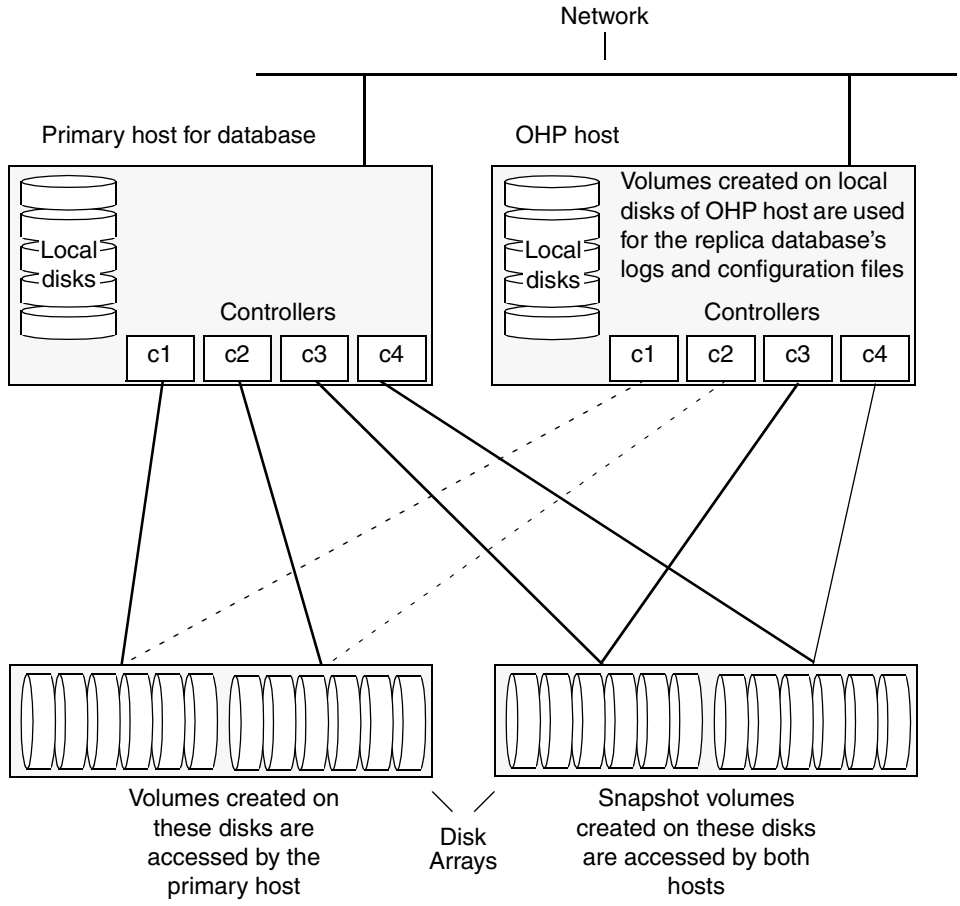
When you want to resynchronize a snapshot with the primary database, shut down the replica database, unmount the snapshot volume, and go back to step 3 to refresh the contents of the snapshot from the original volume.

Creating up an Off-Host Replica Database

As illustrated in "Example System Configuration for Off-Host Decision Support" on page 53, the primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, are configured on disks attached to controllers `c1` and `c2`, and the snapshots are created on disks attached to controllers `c3` and `c4`. There is no requirement for the OHP host to have access to the disks that contain the primary database volumes.

Note If the database is configured on volumes in a cluster-shareable disk group, it is assumed that the primary host for the database is the master node for the cluster. If the primary host is not also the master node, all VxVM operations on shared disk groups must be performed on the master node.

Example System Configuration for Off-Host Decision Support



Note It is assumed that you have already prepared the database volumes to be replicated as described in "Setting up Volumes for Instant Snapshots" on page 17.

To set up a replica database to be used for decision support on an OHP host:

1. If you have not already done so, prepare the OHP host to use the snapshot volume that contains the copy of the database tables. Set up any new database logs and configuration files that are required to initialize the database. See "Preparing a Replica Oracle Database" on page 81 for details of this procedure for an Oracle database.

2. On the primary host, use the following command to make a full-sized snapshot, *snapvol*, of the tablespace volume by breaking off plexes from the original volume:

   ```
   # vxsnap -g volumedg make \
     source=volume/newvol=snapvol/nmirror=N
   ```

   The `nmirror` attribute specifies the number of mirrors, *N*, in the snapshot volume.

   If the volume does not have any available plexes, or its layout does not support plex break-off, use the procedure described in "Creating a Volume for Use as a Full-Sized Instant Snapshot" on page 23 to prepare an empty volume for the snapshot, and then use the following command to create the snapshot:

   ```
   # vxsnap -g volumedg make source=volume/snapvol=snapvol
   ```

   If a database spans more than one volume, specify all the volumes and their snapshot volumes as separate tuples on the same line, for example:

   ```
   # vxsnap -g dbasedg make source=vol1/newvol=svol1 \
     source=vol2/newvol=svol2 source=vol3/newvol=svol3
   ```

   Note This step sets up the snapshot volumes, and starts tracking changes to the original volumes. When you are ready to create the replica database, proceed to step 3.

3. If the volumes to be backed up contain database tables in file systems, suspend updates to the volumes:

   - DB2 provides the `write suspend` command to temporarily suspend I/O activity for a database. As the DB2 database administrator, use a script such as that shown in "Script to Suspend I/O for a DB2 Database" on page 71. Note that if the replica database must be able to be rolled forward (for example, it is to be used a standby database), the primary database must be in LOGRETAIN RECOVERY mode.

- Oracle supports online backup by temporarily suspending updates to the datafiles of the tablespaces, provided that the database is running in archive mode and the tablespaces are online. As the Oracle database administrator, put each tablespace into hot backup mode using a script such as that shown in "Script to Put Oracle Database into Hot Backup Mode" on page 69.
- Sybase ASE from version 12.0 onward provides the Quiesce feature to allow temporary suspension of writes to a database. As the Sybase database administrator, put the database in quiesce mode by using a script such as that shown in "Script to Quiesce Sybase ASE Database" on page 70.

  If you are using Sybase ASE 12.5, you can specify the `for external dump` clause to the `quiesce` command. This *warm standby* method allows you to update a replica database using transaction logs dumped from the primary database. See "Updating a Warm Standby Sybase ASE 12.5 Database" on page 59 for more information.

4. On the primary host, refresh the contents of the snapshot volumes from the original volume using the following command:

   ```
   # vxsnap -g volumedg refresh snapvol source=vol \
     [snapvol2 source=vol2]... syncing=yes
   ```

   The `syncing=yes` attribute starts a synchronization of the snapshot in the background.

   For example, to refresh the snapshots `svol1`, `svol2` and `svol3`:

   ```
   # vxsnap -g dbasedg refresh svol1 source=vol1 svol2 source=vol2 \
     svol3 source=vol3
   ```

5. If you temporarily suspended updates to volumes in step 2, release all the tablespaces or databases from suspend, hot backup or quiesce mode:
   - As the DB2 database administrator, use a script such as that shown in "Script to Resume I/O for a DB2 Database" on page 74.
   - As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in "Script to End Oracle Database Hot Backup Mode" on page 72.
   - As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in "Script to Release Sybase ASE Database from Quiesce Mode" on page 73.

6. Use the following command to wait for the contents of the snapshot to be fully synchronous with the contents of the original volume:

   ```
   # vxsnap -g volumedg syncwait snapvol
   ```


For example, to wait for synchronization to finish for all the snapshots `svol1`, `svol2` and `svol3`, you would issue three separate commands:

```
# vxsnap -g dbasedg syncwait svol1
# vxsnap -g dbasedg syncwait svol2
# vxsnap -g dbasedg syncwait svol3
```

Note You cannot move a snapshot volume into a different disk group until synchronization of its contents is complete. You can use the `vxsnap print` command to check on the progress of synchronization.

7. On the primary host, use the following command to split the disks containing the snapshot volumes into a separate disk group, *snapvoldg*, from the original disk group, *volumedg*:

```
# vxdg split volumedg snapvoldg snapvol ...
```

8. On the primary host, deport the snapshot volume's disk group using the following command:

```
# vxdg deport snapvoldg
```

9. On the OHP host where the replica database is to be set up, use the following command to import the snapshot volume's disk group:

```
# vxdg import snapvoldg
```

10. The snapshot volumes are initially disabled following the split. Use the following commands on the OHP host to recover and restart the snapshot volumes:

```
# vxrecover -g snapvoldg -m snapvol ...
# vxvol -g snapvoldg start snapvol ...
```

11. On the OHP host, for each snapshot volume containing tablespaces, check the file system that it contains, and mount the volume using the following commands:

```
# fsck -F vxfs /dev/vx/rdsk/diskgroup/snapvol
# mount -F vxfs /dev/vx/dsk/diskgroup/snapvol mount_point
```

Note On Linux, use the `-t` option, and on AIX, use the `-V` option, instead of the `-F` option.

For example, to check the file system in the snapshot volume `snap1_dbase_vol`, and mount it on `/rep/dbase_vol`:

```
# fsck -F vxfs /dev/vx/rdsk/dbasedg/snap1_dbase_vol
# mount -F vxfs /dev/vx/dsk/dbasedg/snap1_dbase_vol \
  /rep/dbase_vol
```

Note For a replica DB2 database, the database volume must be mounted in the same location as on the primary host.

12. Copy any required log files from the primary host to the OHP host:
    - For an Oracle database on the OHP host, copy the archived log files that were generated while the database was in hot backup mode to the new database's archived log directory (for example, `/rep/archlog`).
    - For a Sybase ASE database on the primary host, if you specified the `for external dump` clause when you quiesced the database, use the following `isql` command as the database administrator to dump the transaction log for the database:

      `dump transaction to dump_device with standby_access`

      Then copy the dumped transaction log to the appropriate database directory on the OHP host.

13. As the database administrator, start the new database:
    - If the replica DB2 database is not to be rolled forward, use the following commands to start and recover it:

      `db2start`
      `db2inidb database as snapshot`

      If the replica DB2 database is to be rolled forward (the primary must have been placed in LOGRETAIN RECOVERY mode before the snapshot was taken), use the following commands to start it, and put it in roll-forward pending state:

      `db2start`
      `db2inidb database as standby`

      Obtain the latest log files from the primary database, and use the following command to roll the replica database forward to the end of the logs:

      `db2 rollforward db database to end of logs`
    - For an Oracle database, use a script such as that shown in "Script to Complete, Recover and Start Replica Oracle Database" on page 78. (This script also creates the control file for the new database by executing the SQL script that you created using the procedure in "Preparing a Replica Oracle Database" on page 81.)
    - For a Sybase ASE database, use a script such as that shown in "Script to Start Replica Sybase ASE Database" on page 80.

      If you are using the warm standby method, specify the `-q` option to the `dataserver` command. Use the following `isql` commands to load the dump of the transaction log and put the database online:

```
load transaction from dump_device with standby_access
online database database_name for standby_access
```

If you are not using the warm standby method, use the following `isql` command to recover the database, roll back any uncommitted transactions to the time that the quiesce command was issued, and put the database online:

```
online database database_name
```

Resynchronizing the Data with the Primary Host

When you want to resynchronize a snapshot with the primary database, follow these steps:

1. On the OHP host, shut down the replica database, and use the following command to unmount each of the snapshot volumes:

   ```
   # unmount mount_point
   ```

2. On the OHP host, use the following command to deport the snapshot volume's disk group:

   ```
   # vxdg deport snapvoldg
   ```

3. On the primary host, re-import the snapshot volume's disk group using the following command:

   ```
   # vxdg [-s] import snapvoldg
   ```

   Note Specify the `-s` option if you are reimporting the disk group to be rejoined with a shared disk group in a cluster.

4. On the primary host, use the following command to rejoin the snapshot volume's disk group with the original volume's disk group:

   ```
   # vxdg join snapvoldg volumedg
   ```

5. The snapshot volumes are initially disabled following the join. Use the following commands on the primary host to recover and restart a snapshot volume:

   ```
   # vxrecover -g volumedg -m snapvol
   # vxvol -g volumedg start snapvol
   ```

6. Use step 3 on page 54 through step 5 on page 55 to refresh the contents of the snapshot from the original volume.

The snapshots are now ready to be re-used for backup or for other decision support applications.

Updating a Warm Standby Sybase ASE 12.5 Database

If you specified the `for external dump` clause when you quiesced the primary database, and you started the replica database by specifying the `-q` option to the `dataserver` command, you can use transaction logs to update the replica database. As the database administrator, perform the following steps each time that you want to update the replica database:

1. On the primary host, use the following `isql` command to dump the transaction log for the database:

   `dump transaction to dump_device with standby_access`

   Copy the transaction log dump to the appropriate database directory on the OHP host.

2. On the OHP host, use the following `isql` command to load the new transaction log:

   `load transaction from dump_device with standby_access`

3. On the OHP host, use the following `isql` command to put the database online:

   `online database database_name for standby_access`

Reattaching Snapshot Plexes

Note This operation is not supported for space-optimized instant snapshots.

Using the following command, some or all plexes of an instant snapshot may be reattached to the specified original volume, or to a source volume in the snapshot hierarchy above the snapshot volume:

```
# vxsnap [-g diskgroup] reattach snapvol source=vol \
  [nmirror=number]
```

By default, all the plexes are reattached, which results in the removal of the snapshot. If required, the number of plexes to be reattached may be specified as the value assigned to the `nmirror` attribute.

Note The snapshot being reattached must not be open to any application. For example, any file system configured on the snapshot volume must first be unmounted.

For example the following command reattaches 1 plex from the snapshot volume, `snapmyvol`, to the volume, `myvol`:

```
# vxsnap -g mydg reattach snapmyvol source=myvol nmirror=1
```


While the reattached plexes are being resynchronized from the data in the parent volume, they remain in the `SNAPTMP` state. After resynchronization is complete, the plexes are placed in the `SNAPDONE` state.

Database Recovery 6

You can use VERITAS Storage Checkpoints to implement efficient backup and recovery of databases that have been laid out on VxFS file systems. A Storage Checkpoint allows you to roll back an entire database, a tablespace, or a single database file to the time that the Storage Checkpoint was taken. Rolling back to or restoring from any Storage Checkpoint is generally very fast because only the changed data blocks need to be restored.

Storage Checkpoints can also be mounted, allowing regular file system operations to be performed or secondary databases to be started.

This chapter provides an introduction to using Storage Checkpoints for Storage Rollback of an Oracle database. For full information on how to administer the Storage Checkpoints feature for the database software that you are using, see the appropriate *VERITAS Storage Foundation Database Administrator's Guide*.

Note Storage Checkpoints can only be used to restore from logical errors such as human mistakes or software faults. You cannot use them to restore files after a disk failure because all the data blocks are on the same physical device. Disk failure requires restoration of a database from a backup copy of the database files kept on a separate medium. Combining data redundancy (for example, disk mirroring) with Storage Checkpoints is recommended for highly critical data to protect against both physical media failure and logical errors.

Storage Checkpoints require space in the file systems where they are created, and the space required grows over time as copies of changed file system blocks are made. If a file system runs out of space, and there is no disk space into which the file system and any underlying volume can expand, VxFS automatically removes the oldest Storage Checkpoints if they were created with the removable attribute.

If available, it is recommended that you use the VxDBA utility to administer Storage Checkpoints when they are applied to database applications. See the *VERITAS Storage Foundation Database Administrator's Guide* for details.

For information on how Storage Checkpoints work, see the *VERITAS File System Administrator's Guide*.

Creating Storage Checkpoints

To create storage checkpoints, select `3 Storage Checkpoint Administration > Create New Storage Checkpoints` in the VxDBA utility. This can be done with a database either online or offline.

Note To create a Storage Checkpoint while the database is online, `ARCHIVELOG` mode must be enabled in Oracle. During the creation of the Storage Checkpoint, the tablespaces are placed in backup mode. Because it only takes a few seconds to take a Storage Checkpoint, the extra redo logs generated while the tablespaces are in online backup mode are very small. To optimize recovery, it is recommended that you keep `ARCHIVELOG` mode enabled.

Caution Changes to the structure of a database, such as the addition or removal of datafiles, make Storage Rollback impossible if they are made after a Storage Checkpoint was taken. A backup copy of the control file for the database is saved under the `/etc/vx/vxdba/`*`ORACLE_SID`*`/checkpoint_dir` directory immediately after a Storage Checkpoint is created. If necessary, you can use this file to assist with database recovery. If possible, both an ASCII and binary copy of the control file are made, with the binary version being compressed to conserve space. Use extreme caution if you attempt to recover your database using these control files. It is recommended that you remove old Storage Checkpoints and create new ones whenever you restructure a database.

Rolling Back a Database

To roll back a database, for example, after a logical error has occurred:

1. Ensure that the database is offline. You can use the VxDBA utility to display the status of the database and its tablespaces, and to shut down the database:
   - Select `2 Display Database/VxDBA Information` to access the menus that display status information.
   - Select `1 Database Administration > Shutdown Database Instance` to shut down a database.
2. Select `4 Storage Rollback Administration > Roll Back the Database to a Storage Checkpoint` in the VxDBA utility, and choose the appropriate Storage Checkpoint. This restores all data files used by the database, except redo logs and control files, to their state at the time that the Storage Checkpoint was made.

3. Start up, *but do not open*, the database instance by selecting `1 Database Administration > Startup Database Instance` in the VxDBA utility.

4. Use one of the following commands to perform an incomplete media recovery of the database:

   - Recover the database until you stop the recovery:

     ```
     recover database until cancel;
     ...
     alter database [database] recover cancel;
     ```

   - Recover the database to the point just before a specified system change number, *scn*:

     ```
     recover database until change scn;
     ```

   - Recover the database to the specified time:

     ```
     recover database until time 'yyyy-mm-dd:hh:mm:ss';
     ```

   - Recover the database to the specified time using a backup control file:

     ```
     recover database until time 'yyyy-mm-dd:hh:mm:ss' using \
     backup controlfile;
     ```

   Note To find out when an error occurred, check the `../bdump/alert*.log` file.

   See the Oracle documentation for complete and detailed information on database recovery.

5. To open the database after an incomplete media recovery, use the following command:

   ```
   alter database open resetlogs;
   ```

   Note The `resetlogs` option is required after an incomplete media recovery to reset the log sequence. Remember to perform a full database backup and create another Storage Checkpoint after log reset.

6. Perform a full database backup, and use the VxDBA utility to remove any existing Storage Checkpoints that were taken before the one to which you just rolled back the database. These Storage Checkpoints can no longer be used for Storage Rollback. If required, use the VxDBA utility to delete the old Storage Checkpoints and to create new ones.

Files and Scripts for Sample Scenarios A

Note These scripts are not supported by VERITAS, and are provided for informational use only. You can purchase customization of the environment through VERITAS Vpro Consulting Services. For contact details, see "Getting Help" on page xi.

This appendix contains the following configuration files and scripts for the sample point-in-time copy processing scenarios described in this guide:

File or Script	Used for...
"Script to Initiate Online Off-Host Oracle Database Backup" on page 67	- Online off-host backup; see "Making an Off-Host Backup of an Online Database" on page 32.
"Script to Put Oracle Database into Hot Backup Mode" on page 69,	- Online backup; see "Online Database Backup" on page 27
"Script to Quiesce Sybase ASE Database" on page 70 or	- Decision support; see "Decision Support" on page 47.
"Script to Suspend I/O for a DB2 Database" on page 71	
"Script to End Oracle Database Hot Backup Mode" on page 72,	- Online backup; see "Online Database Backup" on page 27
"Script to Release Sybase ASE Database from Quiesce Mode" on page 73 or	- Decision support; see "Decision Support" on page 47.
"Script to Resume I/O for a DB2 Database" on page 74	
"Script to Perform Off-Host Backup" on page 75	- Online off-host backup; see "Making an Off-Host Backup of an Online Database" on page 32.
"Script to Create an Off-Host Replica Oracle Database" on page 76	- Decision support; see "Creating up an Off-Host Replica Database" on page 53.

File or Script	Used for...
"Script to Complete, Recover and Start Replica Oracle Database" on page 78 or "Script to Start Replica Sybase ASE Database" on page 80	- Decision support; see "Creating up an Off-Host Replica Database" on page 53

Script to Initiate Online Off-Host Oracle Database Backup

```
#!/bin/ksh
#
# script: backup_online.sh <dbnode>
#
# Sample script for online, off-host backup.
#
# Note: This is not a production level script, its intention is to help
# you understand the procedure and commands for implementing
# an off-host point-in-time copy solution.

export ORACLE_SID=dbase
export ORACLE_HOME=/oracle/816
export PATH=$ORACLE_HOME/bin:$PATH

dbnode=$1
dbasedg=dbasedg
snapvoldg=snapdbdg
vollist="dbase_vol"
snapvollist="snap_dbase_vol"
volsnaplist="snap_dbase_vol source=dbase_vol"
exit_cnt=0
arch_loc=/archlog

# Put the Oracle database in hot-backup mode;
# see the backup_start.sh script for information.

su oracle -c backup_start.sh

# Refresh the snapshots of the volumes.
#
# Note: If the volume is not mounted, you can safely ignore the
# following message that is output by the snapshot operation:
#
# ERROR: Volume dbase_vol: No entry in /etc/mnttab for volume

vxsnap -g $dbasedg refresh $volsnaplist

# Take the database out of hot-backup mode;
# see the backup_end.sh script for information.

su oracle -c backup_end.sh

# Back up the archive logs that were generated while the database
# was in hot backup mode (as reported by the Oracle Server Manager).

# Wait for synchronization of the snapshot volumes to complete.

for i in `echo $snapvollist`
do
  vxsnap -g $dbasedg syncwait $i
done
```


```
# Move the snapshot volumes into a separate disk group.

vxdg split $dbasedg $snapdg $vollist

# Deport the snapshot disk group.

vxdg deport $snapdg

# The snapshots of the database can be imported and backed up
# on the OHP node and then deported.

rsh $dbnode -c "do_backup.sh $vollist"

# Import the snapshot disk group -- if the database disk group is
# cluster-shareable, you must also specify the -s option.

vxdg import $snapdg

# Join the snapshot disk group to the original volume disk group.

vxdg join $snapdg $dbasedg

# Restart the snapshot volumes ready for the next backup cycle.

for i in `echo $snapvollist`
do
  vxrecover -g $dbasedg -m $i
  vxvol -g $dbasedg start $i
done
```

Script to Put Oracle Database into Hot Backup Mode

```
#!/bin/ksh
#
# script: backup_start.sh
#
# Sample script to put example Oracle database into hot backup mode.

export ORACLE_SID=dbase
export ORACLE_HOME=/oracle/816
export PATH=$ORACLE_HOME/bin:$PATH

svrmgrl <<!
connect internal
archive log list;
alter tablespace ts1 begin backup;

# .
# . Put all required tablespaces into hot backup mode
# .

alter tablespace tsN begin backup;
quit
!
```


Script to Quiesce Sybase ASE Database

```
#!/bin/ksh
#
# script: backup_start.sh
#
# Sample script to quiesce example Sybase ASE database.
#
# Note: The "for external dump" clause was introduced in Sybase
# ASE 12.5 to allow a snapshot database to be rolled forward.
# See the Sybase ASE 12.5 documentation for more information.

isql -Usa -Ppassword -SFMR <<!
quiesce database tag hold database1[, database2]... [for external dump]
go
quit
!
```

Script to Suspend I/O for a DB2 Database

```
#!/bin/ksh
#
# script: backup_start.sh
#
# Sample script to suspend I/O for a DB2 database.
#
# Note: To recover a database using backups of snapshots, the database
# must be in LOGRETAIN mode.

db2 <<!
connect to database
set write suspend for database
quit
!
```


Script to End Oracle Database Hot Backup Mode

```
#!/bin/ksh
#
# script: backup_end.sh
#
# Sample script to end hot backup mode for example Oracle database.

export ORACLE_SID=dbase
export ORACLE_HOME=/oracle/816
export PATH=$ORACLE_HOME/bin:$PATH

svrmgrl <<!
connect internal
alter tablespace ts1 end backup;
# .
# . End hot backup mode for all required tablespaces.
# .
alter tablespace tsN end backup;
alter system switch logfile;
alter system switch logfile;
archive log list;
quit
!

# Note: The repeated line alter system switch logfile; forces a checkpoint and
#       archives the contents of the redo logs recorded during the backup.
```

Script to Release Sybase ASE Database from Quiesce Mode

```
#!/bin/ksh
#
# script: backup_end.sh
#
# Sample script to release example Sybase ASE database from quiesce mode.

isql -Usa -Ppassword -SFMR <<!
quiesce database tag release
go
quit
!
```


Script to Resume I/O for a DB2 Database

```
#!/bin/ksh
#
# script: backup_end.sh
#
# Sample script to resume I/O for a DB2 database.
#

db2 <<!
connect to database
set write resume for database
quit
!
```

Script to Perform Off-Host Backup

```
#!/bin/ksh
#
# script: do_backup.sh <list_of_database_volumes>
#
# Sample script for off-host backup
#
# Note: This is not a production level script, its intention is to help
# you understand the procedure and commands for implementing
# an off-host point-in-time copy solution.

# Modify the following procedure according to your environment
# and backup method.

snapvoldg=snapdbdg

# Import the snapshot volume disk group.

vxdg import $snapvoldg

# Mount the snapshot volumes (the mount points must already exist).

for i in $*
do
  fsck -F vxfs /dev/vx/rdsk/$dbasedg/snap_$i
  mount -F vxfs /dev/vx/dsk/$dbasedg/snap_$i /bak/$i
done

# Back up each tablespace.
# back up /bak/ts1 &
...
# back up /bak/tsN &

wait

# Unmount snapshot volumes.

for i in `echo $vollist`
do
  umount /bak/$i
done

# Deport snapshot volume disk group.

vxdg deport $snapvoldg

echo "do_backup over"
echo "\007 \007 \007 \007 \007 \007"
```


Script to Create an Off-Host Replica Oracle Database

```
#!/bin/ksh
#
# script: create_dss.sh <dbnode>
#
# Sample script to create a replica Oracle database on an OHP host.
#
# Note: This is not a production level script, its intention is to help
# you understand the procedure and commands for implementing
# an off-host point-in-time copy solution.

export ORACLE_SID=dbase
export ORACLE_HOME=/oracle/816
export PATH=$ORACLE_HOME/bin:$PATH

dbnode=$1
localdg=localdg
dbasedg=dbasedg
snapvoldg=snapdbdg
vollist="dbase_vol"
snapvollist="snap_dbase_vol"
volsnaplist="snap_dbase_vol source=dbase_vol"
exit_cnt=0
arch_loc=/archlog
rep_mnt_point=/rep

# Put the Oracle database in hot-backup mode;
# see the backup_start.sh script for information.

su oracle -c backup_start.sh

# Refresh the snapshots of the volumes.
#
# Note: If the volume is not mounted, you can safely ignore the
# following message that is output by the snapshot operation:
#
# vxvm:vxsync: ERROR: Volume dbase_vol: No entry in /etc/mnttab for volume

vxsnap -g $dbasedg refresh $volsnaplist

# Take the Oracle database out of hot-backup mode;
# see the backup_end.sh script for information.

su oracle -c backup_end.sh

# Move the snapshot volumes into a separate disk group.

vxdg split $dbasedg $snapdg $vollist

# Deport the snapshot disk group.

vxdg deport $snapdg
```

```
# Copy the archive logs that were generated while the database was
# in hot backup mode (as reported by the Oracle Server Manager) to the
# archive log location for the replica database on the OHP node
# (in this example, /rep/archlog).

rcp ${arch_loc}/* $dbnode:${rep_mnt_point}${arch_loc}

# The snapshots of the database can be now imported on the OHP node
# and used to complete, recover and start the replica database.

rsh $dbnode -c "startdb.sh $vollist"
```


Script to Complete, Recover and Start Replica Oracle Database

```
#!/bin/ksh
#
# script: startdb.sh <list_of_database_volumes>
#
# Sample script to complete, recover and start replica Oracle database.
#
# It is assumed that you have already performed the following
# steps:
# 1. Create the local volumes, file systems, and mount points for the
#     redo and archived logs, and then mount them.
# 2. Based on the text control file for the production database,
#     write a SQL script that creates a control file for the replica
#     database.
# 3. Create an initialization file for the replica database and place
#     this in the replica database's $ORACLE_HOME/dbs directory.
# 4. Copy the Oracle password file for the production database to the
#     replica database's $ORACLE_HOME/dbs directory.

export ORACLE_SID=REP1
export ORACLE_HOME=/rep/oracle/816
export PATH=$ORACLE_HOME/bin:$PATH
snapvoldg=snapdbdg
rep_mnt_point=/rep

# Import the snapshot volume disk group.

vxdg import $snapvoldg

# Mount the snapshot volumes (the mount points must already exist).

for i in $*
do
  fsck -F vxfs /dev/vx/rdsk/$snapvoldg/snap_$i
  mount -F vxfs /dev/vx/dsk/$snapvoldg/snap_$i ${rep_mnt_point}/$i
done

# Fix any symbolic links required by the database.

cd ${rep_mnt_point}/dbase_vol
for i in 1 2 3 4 5 6 # adjust as required
do
  rm -f ./log$i
  ln -s ${rep_mnt_point}/dbase_logs/log$i ./log$i
done

# Remove the existing control file.

rm -f ${rep_mnt_point}/dbase_vol/cntrl1
```

```
# Create a new control file, recover and start the replica database.

svrmgrl <<!
connect internal
@c_file_create.sql
set autorecovery on
recover database until cancel using backup controlfile;
alter database open resetlogs;
quit
!
```


Script to Start Replica Sybase ASE Database

```
#!/bin/ksh
#
# script: startdb.sh <list_of_database_volumes>
#
# Sample script to recover and start replica Sybase ASE database.

# Import the snapshot volume disk group.

vxdg import $snapvoldg

# Mount the snapshot volumes (the mount points must already exist).

for i in $*
do
  fsck -F vxfs /dev/vx/rdsk/$snapvoldg/snap_$i
  mount -F vxfs /dev/vx/dsk/$snapvoldg/snap_$i ${rep_mnt_point}/$i
done

# Start the replica database.
# Specify the -q option if you specified the "for external dump"
# clause when you quiesced the primary database.
# See the Sybase ASE 12.5 documentation for more information.

/sybase/ASE-12_5/bin/dataserver \
[-q] \
-sdatabase_name \
-d /sybevm/master \
-e /sybase/ASE-12_5/install/dbasename.log \
-M /sybase

# Online the database. Load the transaction log dump and
# specify "for standby_access" if you used the -q option
# with the dataserver command.

isql -Usa -Ppassword -SFMR <<!
[load transaction from dump_device with standby_access
go]
online database database_name [for standby_access]
go
quit
!
```

Preparing a Replica Oracle Database B

This appendix describes how to set up a replica off-host Oracle database to be used for decision support as described in "Creating up an Off-Host Replica Database" on page 53.

To prepare a replica Oracle database on a host other than the primary host:

1. If not already present, install the Oracle software onto the host's local disks. The location of the Oracle home directory ($ORACLE_HOME) is used for the database instance that is created from the snapshot volumes.

   Note In the examples shown here, the home directory is `/rep/oracle` in the local disk group, `localdg`. If required, you could instead choose to use the same file paths and database name as on the primary host.

2. In the local disk group, `localdg`, use the following command to create the volumes that are to be used for the redo logs and archived logs of the replicated database:

   ```
   # vxassist -g diskgroup make volume size
   ```

   For example, to create a 1-gigabyte redo log volume `rep_dbase_logs` and a 2-gigabyte archived log volume `rep_dbase_arch`:

   ```
   # vxassist -g localdg make rep_dbase_logs 1g
   # vxassist -g localdg make rep_dbase_arch 2g
   ```

3. Make the file systems for the redo logs and archive logs in the volumes created in the previous step using the following command:

   ```
   # mkfs -F vxfs /dev/vx/rdsk/diskgroup/volume
   ```

   In this example, the commands would be:

   ```
   # mkfs -F vxfs /dev/vx/rdsk/localdg/rep_dbase_logs
   # mkfs -F vxfs /dev/vx/rdsk/localdg/rep_dbase_arch
   ```


4. Create the mount points that are to be used to mount the new database. For example, create `/rep/dbase_vol` for the snapshot of the tablespace volume, `/rep/dbase_logs` for the redo logs, and `/rep/dbase_arch` for the archived logs:

```
# mkdir -p /rep/dbase_vol
# mkdir -p /rep/dbase_logs
# mkdir -p /rep/dbase_arch
```

5. Mount the redo log and archive log volumes on their respective mount points using the following command:

```
# mount -F vxfs /dev/vx/dsk/diskgroup/volume  mount_point
```

In this example, the commands would be:

```
# mount -F vxfs /dev/vx/dsk/localdg/rep_dbase_logs \
  /rep/dbase_logs
# mount -F vxfs /dev/vx/dsk/localdg/rep_dbase_arch \
  /rep/dbase_arch
```

6. As the Oracle database administrator on the primary host, obtain an ASCII version of the current Oracle control file using the following SQL command:

```
alter database backup controlfile to trace;
```

This command writes a text version of the control file to the directory `$ORACLE_HOME/admin/dbase/udump`. See "Text Control File for Original Production Database" on page 84 for an example.

7. Modify the text version of the control file created in the previous step as described below to create a new SQL script to set up the replica database:

   a. If required, change the locations defined under LOGFILE for the log files. For example, change lines of the form:

   `GROUP` *N* `'/dbase_vol/log`*N*`'  SIZE 52428288,`

   so that they read:

   `GROUP` *N* `'/rep/dbase_vol/log`*N*`'  SIZE 52428288,`

   b. If required, change the locations defined under DATAFILE for the tablespaces. For example, change lines of the form:

   `'/dbase_vol/`*table*`',`

   so that they read:

   `'/rep/dbase_vol/`*table*`',`

c. If required, change the following line:

```
CREATE CONTROLFILE REUSE DATABASE "odb" NORESETLOGS ARCHIVELOG
```

so that it reads:

```
CREATE CONTROLFILE SET DATABASE "ndb" RESETLOGS NOARCHIVELOG
```

where *odb* is the name of the original database and *ndb* is the name of the replica database (DBASE and REP1 in the example). Note that to reduce unnecessary overhead, the new database is not run in archive log mode.

See "SQL Script to Create Control File" on page 85 for an example.

8. Copy the Oracle initialization file (for example, `initdbase.ora`; see "Initialization File for Original Production Database" on page 86) for the original database to a new initialization file for the replica database (for example, `initREP1.ora`; see "Initialization File for Replica Oracle Database" on page 87).

Edit the copied file and change the definitions of the following parameters:

`background_dump_dest`	Background dump location.
`core_dump_dest`	Core dump location.
`db_name`	Database name to the name of the replica database.
`log_archive_dest`	Archive log location, set equal to the path created in step 4 (for example, `/rep/dbase_arch`).
`log_archive_start`	Archive log mode, log_archive_start, to FALSE.
`user_dump_dest`	User dump location.

You may also wish to reduce the resource usage of the new database by adjusting the values of parameters such as `db_block_buffers`. See the *Oracle Database Administrator's Guide* for more information.

9. Copy the Oracle remote password file (for example, `orapwdbase`) in `$ORACLE_HOME/dbs` to a new file (for example, `orapwREP1`).

Text Control File for Original Production Database

```
/oracle/816/admin/dbase/udump/dbase_ora_20480.trc
Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
ORACLE_HOME = /oracle/816
System name:    SunOS
Node name:      node01
Release:        5.8
Version:        Generic_108528-02
Machine:        sun4u
Instance name: dbase
Redo thread mounted by this instance: 1
Oracle process number: 8
Unix process pid: 20480, image: oracle@node01

*** SESSION ID:(#.##) YYYY-MM-DD hh:mm:ss.sss
*** YYYY-MM-DD hh:mm:ss.sss
# The following commands will create a new control file and use it
# to open the database.
# Data used by the recovery manager will be lost. Additional logs may
# be required for media recovery of offline data files. Use this
# only if the current version of all online logs are available.
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "DBASE" NORESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 2
    MAXDATAFILES 70
    MAXINSTANCES 1
    MAXLOGHISTORY 226
LOGFILE
    GROUP 1 '/dbase_vol/log1'  SIZE 52428288,
    # .
    # . List of log files
    # .
    GROUP N '/dbase_vol/logN'  SIZE 52428288
DATAFILE
    '/dbase_vol/ts1',
    # .
    # . List of tablespace datafiles
    # .
    '/dbase_vol/tsN'
CHARACTER SET US7ASCII
;
# Recovery is required if any of the datafiles are restored backups,
# or if the last shutdown was not normal or immediate.
RECOVER DATABASE
# All logs need archiving and a log switch is needed.
ALTER SYSTEM ARCHIVE LOG ALL;
# Database can now be opened normally.
ALTER DATABASE OPEN;
# No tempfile entries found to add.
#
```

SQL Script to Create Control File

```
STARTUP NOMOUNT
CREATE CONTROLFILE SET DATABASE "REP1" RESETLOGS NOARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 2
    MAXDATAFILES 70
    MAXINSTANCES 1
    MAXLOGHISTORY 226
LOGFILE
    GROUP 1 '/rep/dbase_vol/log1'  SIZE 52428288,
    # .
    # . List of log files
    # .
    GROUP N '/rep/dbase_vol/logN'  SIZE 52428288
DATAFILE
    '/rep/dbase_vol/ts1',
    # .
    # . List of tablespace datafiles
    # .
    '/rep/dbase_vol/tsN'
CHARACTER SET US7ASCII
;
```


Initialization File for Original Production Database

```
#===============================================================+
# FILENAME         initdbase.ora
# DESCRIPTION      Oracle parameter file for primary database, dbase.
#===============================================================

db_block_size                              = 8192
parallel_max_servers               = 30
recovery_parallelism               = 20
# db_writers                               = 25
# use_async_io                             = TRUE
# async_io                                 = 1
control_files                      = (/dbase_vol/cntrl1)
sort_area_size                     = 15728640
parallel_max_servers               = 10
recovery_parallelism               = 4
compatible                         = 8.1.5
db_name                            = dbase
db_files                           = 200
db_file_multiblock_read_count      = 32
db_block_buffers                   = 30720 # 8k * 30720 approx 250MB
dml_locks                          = 500
hash_join_enabled                  = FALSE

# Uncommenting the line below will cause automatic archiving if
# archiving has been enabled using ALTER DATABASE ARCHIVELOG.
log_archive_dest                   = /archlog
log_archive_format                 = dbase%t_%s.dbf
log_archive_start                  = TRUE
# log_checkpoint_interval           = 1000000000

log_checkpoint_timeout             = 300
log_checkpoints_to_alert           = TRUE
log_buffer                         = 1048576
max_rollback_segments              = 220
processes                          = 300
sessions                           = 400
open_cursors                       = 200
transactions                       = 400
distributed_transactions           = 0
transactions_per_rollback_segment = 1
rollback_segments                  =
(s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,s20,s21,s22,s23,s24,s25,s
26,s27,s28,s29,s30)
shared_pool_size                           = 7000000
cursor_space_for_time                      = TRUE
audit_trail                                = FALSE
cursor_space_for_time                      = TRUE
background_dump_dest       = /oracle/816/admin/dbase/bdump
core_dump_dest             = /oracle/816/admin/dbase/cdump
user_dump_dest             = /oracle/816/admin/dbase/udump
```

Initialization File for Replica Oracle Database

```
##=================================================================+
# FILENAME         initREP1.ora
# DESCRIPTION      Oracle parameter file for replica database, REP1.
#=================================================================

db_block_size  = 8192
parallel_max_servers              = 30
recovery_parallelism              = 20
# db_writers                       = 25
# use_async_io                     = TRUE
# async_io                         = 1
control_files                     = (/rep/dbase_vol/cntrl1)
sort_area_size                    = 15728640
parallel_max_servers              = 10
recovery_parallelism              = 4
compatible                        = 8.1.5
db_name                           = REP1
db_files                          = 200
db_file_multiblock_read_count     = 32
db_block_buffers                  = 10240
dml_locks                         = 500
hash_join_enabled                 = FALSE
log_archive_start                 = FALSE
log_archive_dest                  = /rep/archlog
log_archive_format                = dbase%t_%s.dbf
log_checkpoint_timeout            = 300
log_checkpoints_to_alert          = TRUE
log_buffer                        = 1048576
max_rollback_segments             = 220
processes                         = 300
sessions                          = 400
open_cursors                      = 200
transactions                      = 400
distributed_transactions          = 0
transactions_per_rollback_segment = 1
rollback_segments                 =
(s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,s20,s21,s22,s23,s24,s25,s
26,s27,s28,s29,s30)
shared_pool_size                  = 7000000
cursor_space_for_time             = TRUE
audit_trail                              = FALSE
cursor_space_for_time             = TRUE
background_dump_dest  = /rep/oracle/816/admin/REP1/bdump
core_dump_dest             = /rep/oracle/816/admin/REP1/cdump
user_dump_dest             = /rep/oracle/816/admin/REP1/udump
```


Index