Veritas Storage Foundation™ Advanced Features Administrator's Guide

Solaris

5.1

Veritas Storage Foundation™ Advanced Features Administrator's Guide

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Product version: 5.1

Document version: 5.1.0

Legal Notice

Copyright © 2009 Symantec Corporation. All rights reserved.

Symantec, the Symantec Logo, Veritas, Veritas Storage Foundation are trademarks or registered trademarks of Symantec Corporation or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Symantec Corporation and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. SYMANTEC CORPORATION SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, "Rights in Commercial Computer Software or Commercial Computer Software Documentation", as applicable, and any successor regulations. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Symantec Corporation
350 Ellis Street
Mountain View, CA 94043

http://www.symantec.com

Technical Support

Symantec Technical Support maintains support centers globally. Technical Support's primary role is to respond to specific queries about product features and functionality. The Technical Support group also creates content for our online Knowledge Base. The Technical Support group works collaboratively with the other functional areas within Symantec to answer your questions in a timely fashion. For example, the Technical Support group works with Product Engineering and Symantec Security Response to provide alerting services and virus definition updates.

Symantec's maintenance offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers automatic software upgrade protection
- Global support that is available 24 hours a day, 7 days a week
- Advanced features, including Account Management Services

For information about Symantec's Maintenance Programs, you can visit our Web site at the following URL:

www.symantec.com/business/support/index.jsp

Contacting Technical Support

Customers with a current maintenance agreement may access Technical Support information at the following URL:

www.symantec.com/business/support/contact_techsupp_static.jsp

Before contacting Technical Support, make sure you have satisfied the system requirements that are listed in your product documentation. Also, you should be at the computer on which the problem occurred, in case it is necessary to replicate the problem.

When you contact Technical Support, please have the following information available:

- Product release level
- Hardware information
- Available memory, disk space, and NIC information
- Operating system

- Version and patch level
- Network topology
- Router, gateway, and IP address information
- Problem description:
  - Error messages and log files
  - Troubleshooting that was performed before contacting Symantec
  - Recent software configuration changes and network changes

Licensing and registration

If your Symantec product requires registration or a license key, access our non-technical support Web page at the following URL:

customercare.symantec.com

Customer service

Customer Care information is available at the following URL:

www.symantec.com/customercare

Customer Service is available to assist with the following types of issues:

- Questions regarding product licensing or serialization
- Product registration updates, such as address or name changes
- General product information (features, language availability, local dealers)
- Latest information about product updates and upgrades
- Information about upgrade assurance and maintenance contracts
- Information about the Symantec Buying Programs
- Advice about Symantec's technical support options
- Nontechnical presales questions
- Issues that are related to CD-ROMs or manuals

Documentation feedback

Your feedback on product documentation is important to us. Send suggestions for improvements and reports on errors or omissions. Include the title and document version (located on the second page), and chapter and section titles of the text on which you are reporting. Send feedback to:

sfha_docs@symantec.com

Maintenance agreement resources

If you want to contact Symantec regarding an existing maintenance agreement, please contact the maintenance agreement administration team for your region as follows:

Asia-Pacific and Japan	customercare_apac@symantec.com
Europe, Middle-East, and Africa	semea@symantec.com
North America and Latin America	supportsolutions@symantec.com

Additional enterprise services

Symantec offers a comprehensive set of services that allow you to maximize your investment in Symantec products and to develop your knowledge, expertise, and global insight, which enable you to manage your business risks proactively.

Enterprise services that are available include the following:

Symantec Early Warning Solutions
These solutions provide early warning of cyber attacks, comprehensive threat analysis, and countermeasures to prevent attacks before they occur.

Managed Security Services
These services remove the burden of managing and monitoring security devices and events, ensuring rapid response to real threats.

Consulting Services
Symantec Consulting Services provide on-site technical expertise from Symantec and its trusted partners. Symantec Consulting Services offer a variety of prepackaged and customizable options that include assessment, design, implementation, monitoring, and management capabilities. Each is focused on establishing and maintaining the integrity and availability of your IT resources.

Educational Services
Educational Services provide a full array of technical training, security education, security certification, and awareness communication programs.

To access more information about Enterprise services, please visit our Web site at the following URL:

www.symantec.com

Select your country or language from the site index.

Contents

Chapter 1

Introducing Veritas Storage Foundation™ Advanced Features

This chapter includes the following topics:

Overview

This guide covers the advanced features of Volume Manager (VxVM) and File System (VxFS).

The guide covers the following storage management features of Veritas Storage Foundation:

- Snapshots - Snapshots enable you to capture an instantaneous image of actively changing data: a point-in-time copy. Storage Foundation offers two main approaches to Snapshots:
  - FlashSnap – VxVM implements snapshots through its FlashSnap feature.
  - Storage Checkpoints – VxFS and SFCFS implement snapshots through the Storage Checkpoint feature. Storage Checkpoints reduce the I/O overhead by identifying and main streaming only the file system blocks that have changed since the last Storage Checkpoint or backup by a copy-on-write technique.
- Migration

    - Cross-Platform Migration – Cross-Platform Data Sharing (CDS) provides you with a foundation for moving data between different systems that are running the HP-UX, AIX, Linux, or Solaris operating system.
    - Migrating to Storage Foundation from other tools – Veritas Volume Manager (VxVM) provides utilities that you can use to convert objects, file systems, and device partitions controlled by the Sun Microsystems Solaris Volume Manager software (Solstice Disk Suite) to VxVM control. The tools convert all Solaris Volume Manager objects into VxVM objects and encapsulates file systems and swap device partitions used by Solaris Volume Manager software.
- Thin Provisioning - Thin Storage is an array vendor solution for allocating storage to applications from a pool of free storage only when the storage is truly needed.
    - SmartMove – With Storage Foundation's SmartMove feature, Veritas File System (VxFS) lets Veritas Volume Manager (VxVM) know which blocks have data. VxVM, which is the copy engine for migration, copies only the used blocks and avoids the empty spaces. This behavior helps optimize the Thin Storage utilization.
    - Thin Reclamation - The Thin Reclamation feature optimizes storage administration by reclaiming space deleted or shrunk on the VxVM Volume or the VxFS file system and returns the space to the Thin Storage pool.
- Dynamic Storage Tiering - VxFS uses multi-tier online storage by way of the Dynamic Storage Tiering (DST) feature, which functions on top of multi-volume file systems.

Advanced features in Storage Foundation

The following table lists the Veritas Storage Foundation product suites in which each advanced feature is available.

Table 1-1 Advanced features in Storage Foundation

Storage Foundation feature	Products in which the feature is available
SmartMove Thin Reclamation	Storage Foundation Basic Storage Foundation Standard Storage Foundation Standard HA Storage Foundation Enterprise Storage Foundation Enterprise HA Storage Foundation Cluster File System HA Storage Foundation for Oracle RAC
Cross Platform Data Sharing	Storage Foundation Basic Storage Foundation Standard Storage Foundation Standard HA Storage Foundation Enterprise Storage Foundation Enterprise HA Storage Foundation Cluster File System HA Storage Foundation for Oracle RAC
FlashSnap Storage Checkpoints Dynamic Storage Tiering	Storage Foundation Enterprise Storage Foundation Enterprise HA Storage Foundation Cluster File System HA Storage Foundation for Oracle RAC

Chapter 2

Storage Foundation Snapshots

This chapter includes the following topics:

- About Snapshots

About Snapshots

Two trends dominate the evolution of digital data used to conduct and manage business. First, more and more data must be continuously available for 24x7 transaction processing, decision making, intellectual property creation, and so forth. Second, as digital data assumes a larger role in the conduct of business, protecting it from loss or destruction becomes increasingly important. These trends run counter to each other. To protect it properly, data must be taken out of service so that it stands still while it is backed up. But taking data out of service means that it's not there when the business needs it. To solve this problem, information technology has developed a set of techniques for freezing instantaneous images of data—for taking snapshots of it—that can be used to make backups, perform analyses, and test software updates, all while production applications continue to operate on the live data. The Veritas Storage Foundation offers two main approaches to Snapshots: Volume snapshots (FlashSnap) and file system Snapshots (Storage Checkpoint).

Snapshot Technique

A snapshot is a virtual image of the content of a set of data at the instant of creation. Physically, a snapshot may be a full (complete bit-for-bit) copy of the data set, or it may contain only those elements of the data set that have been updated since snapshot creation. The latter are sometimes referred to as allocate-on-first-write snapshots, because space for data elements is added to the snapshot image only when the elements are updated (overwritten) for the first time in the original data set. Storage Foundation allocate-on-first-write snapshots are called space-optimized snapshots.

A full-copy snapshot of a virtual volume requires storage capacity equal to that of the original volume, as well as sufficient I/O bandwidth to copy the entire contents of the original volume to the snapshot within the required time. Space-optimized snapshots, on the other hand, consume storage and I/O bandwidth in proportion to how much data on the original volume is updated during a snapshot's life. Thus, full-copy snapshots are inherently greedier in terms of storage and I/O bandwidth than space-optimized ones. They are generally more suitable for recovering from storage device failures, and for off-host auxiliary processing where impact on production applications is a concern. They are generally less optimal for protecting against data corruption due to human or application error.

Because space-optimized snapshots do not contain complete physical images of the original data objects they represent, they cannot be taken off-host for auxiliary processing. But because they consume relatively little storage and I/O bandwidth, they can be taken much more frequently than full-copy snapshots. This makes them well-suited for recovering from data corruption. Space-optimized snapshots naturally tend to grow with age, as more of the data in the original objects changes, so they are inherently better-suited for shorter lifetimes.

Chapter 3

Flashsnap

This chapter includes the following topics:

About point-in-time copy solutions

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

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.

The following 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. These features are suitable for implementing solutions where the I/O performance of the production server is critical.

- File system-level solutions use the Storage Checkpoint feature of Veritas File System. 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 FlashSnap license also supports the Veritas FlashSnap Agent for Symmetrix.

Point-in-time copy solutions

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.

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:

- About online database backup
- About off-host cluster file system backup
- About decision support

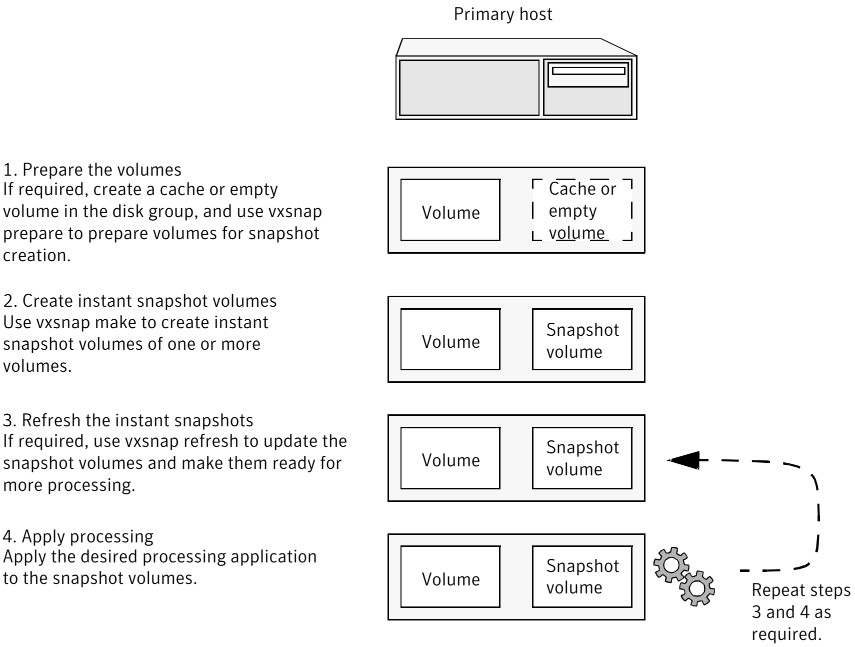
The following types of point-in-time copy solution are considered in this document:

- 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 21.
- 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 23.
- Using Storage Checkpoints to quickly roll back a database instance to an earlier point in time.
  See "Storage Checkpoints" on page 74.

Implementing point-in time copy solutions on a primary host

Figure 3-1 illustrates the steps that are needed to set up the processing solution on the primary host.

Figure 3-1 Using snapshots and FastResync to implement point-in-time copy solutions on a primary host



1. Prepare the volumes
If required, create a cache or empty volume in the disk group, and use vxsnap prepare to prepare volumes for snapshot creation.

2. Create instant snapshot volumes
Use vxsnap make to create instant snapshot volumes of one or more volumes.

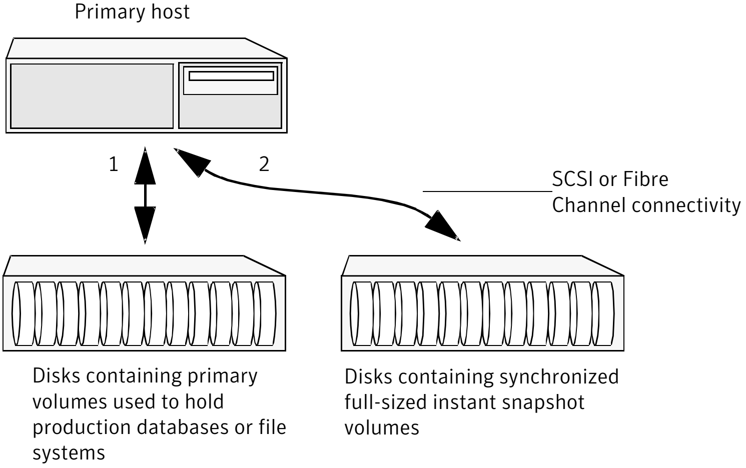
3. Refresh the instant snapshots
If required, use vxsnap refresh to update the snapshot volumes and make them ready for more processing.

4. Apply processing
Apply the desired processing application to the snapshot volumes.

Note: 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.

Figure 3-2 shows the suggested arrangement for implementing solutions where the primary host is used and disk contention is to be avoided.

Figure 3-2 Example point-in-time copy solution on a primary host


Primary host
1
2
SCSI or Fibre Channel connectivity
Disks containing primary volumes used to hold production databases or file systems
Disks containing synchronized full-sized instant snapshot volumes

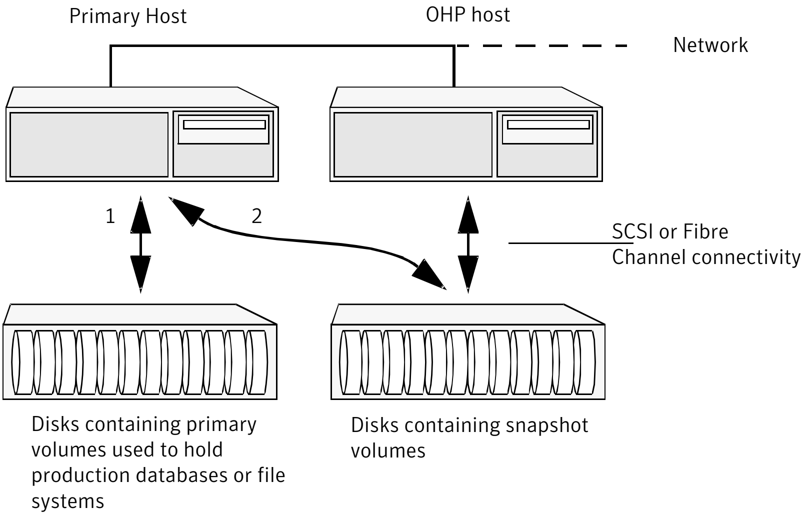
In this setup, it is recommended that separate paths (shown as 1 and 2) from separate controllers be configured to the disks containing the primary volumes and the snapshot volumes. This avoids contention for disk access, but the primary host's CPU, memory and I/O resources are more heavily utilized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

Figure 3-3 illustrates that, 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).

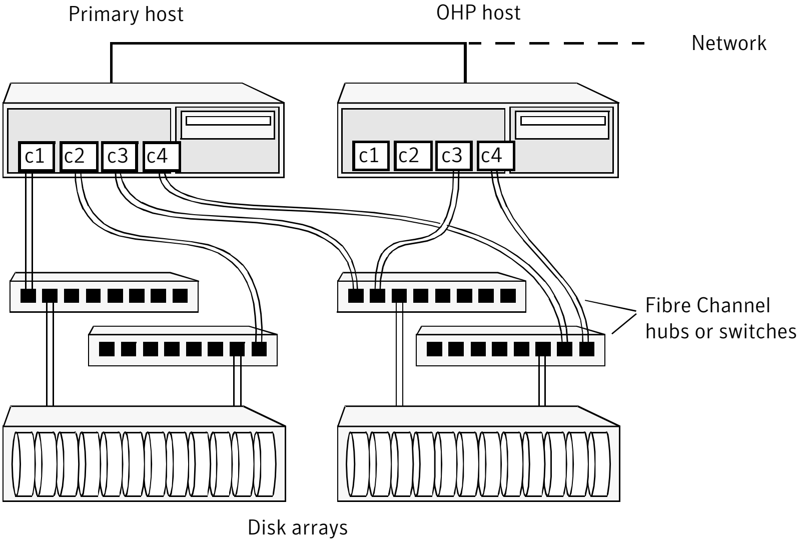
Figure 3-3 Example implementation of an off-host point-in-time copy solution


Primary Host
OHP host
Network
1
2
SCSI or Fibre Channel connectivity
Disks containing primary volumes used to hold production databases or file systems
Disks containing snapshot volumes

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 Figure 3-3 should be connected to different controllers.

Figure 3-4 shows an example of how you might achieve such connectivity using Fibre Channel technology with 4 Fibre Channel controllers in the primary host.

Figure 3-4 Example connectivity for off-host solution using redundant-loop access



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.

Figure 3-5 shows how off-host processing might be implemented in a cluster by configuring one of the cluster nodes as the OHP node.

Figure 3-5 Example implementation of an off-host point-in-time copy solution using a cluster node

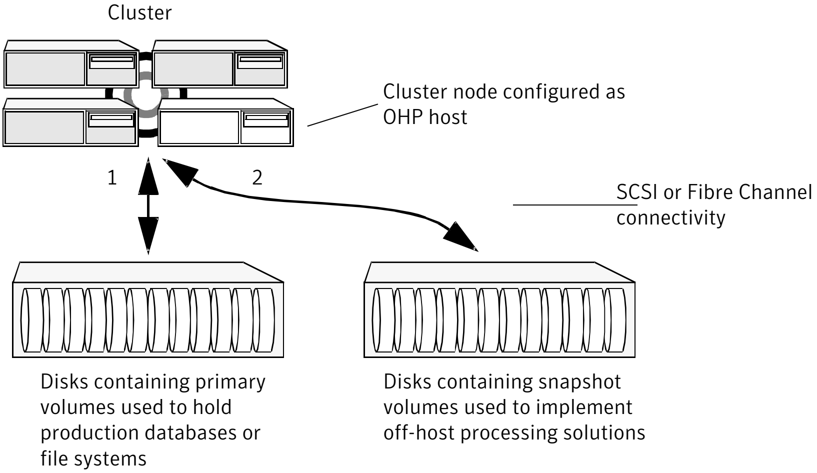
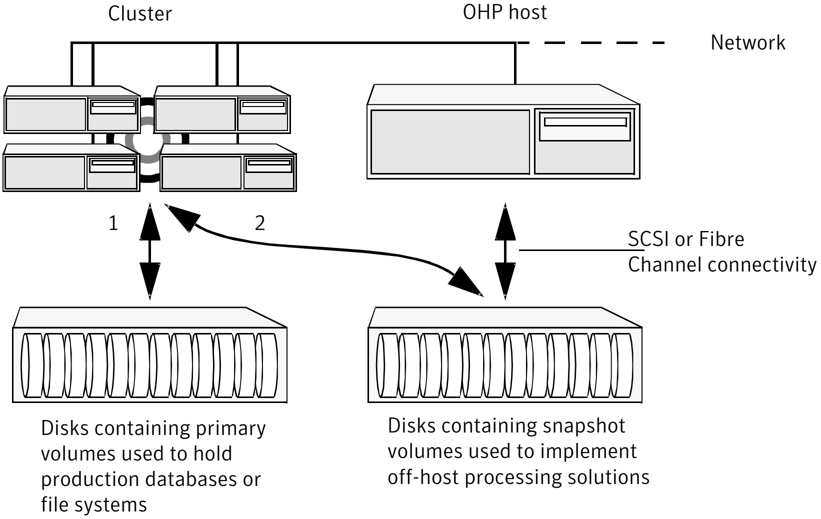

Cluster
Cluster node configured as OHP host
1
2
SCSI or Fibre Channel connectivity
Disks containing primary volumes used to hold production databases or file systems
Disks containing snapshot volumes used to implement off-host processing solutions

Figure 3-6 shows an alternative arrangement, where 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.

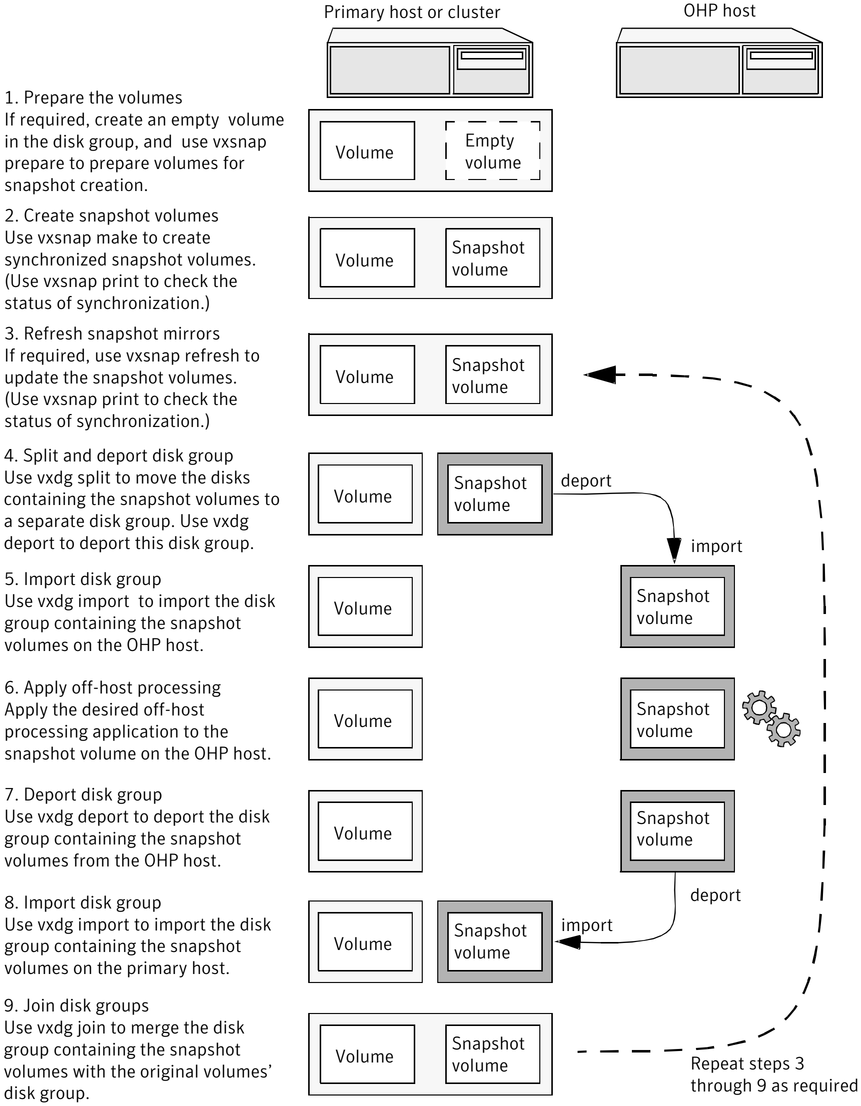
Figure 3-6 Example implementation of an off-host point-in-time copy solution using a separate OHP host


Cluster
OHP host
Network
1
2
SCSI or Fibre Channel connectivity
Disks containing primary volumes used to hold production databases or file systems
Disks containing snapshot volumes used to implement off-host processing solutions

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, and more efficient, to configure an OHP host that is not a member of the cluster.

Figure 3-7 illustrates the steps that are needed to set up the processing solution on the primary host.

Figure 3-7 Implementing off-host processing solutions



1. Prepare the volumes
If required, create an empty volume in the disk group, and use vxsnap prepare to prepare volumes for snapshot creation.

2. Create snapshot volumes
Use vxsnap make to create synchronized snapshot volumes. (Use vxsnap print to check the status of synchronization.)

3. Refresh snapshot mirrors
If required, use vxsnap refresh to update the snapshot volumes. (Use vxsnap print to check the status of synchronization.)

4. Split and deport disk group
Use vxdg split to move the disks containing the snapshot volumes to a separate disk group. Use vxdg deport to deport this disk group.

5. Import disk group
Use vxdg import to import the disk group containing the snapshot volumes on the OHP host.

6. Apply off-host processing
Apply the desired off-host processing application to the snapshot volume on the OHP host.

7. Deport disk group
Use vxdg deport to deport the disk group containing the snapshot volumes from the OHP host.

8. Import disk group
Use vxdg import to import the disk group containing the snapshot volumes on the primary host.

9. Join disk groups
Use vxdg join to merge the disk group containing the snapshot volumes with the original volumes' disk group.

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.

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

About setting up volumes for instant snapshots

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 information on administering instant snapshots and FastResync, see the *Veritas Volume Manager Administrator's Guide*.

Table 3-1 summarizes which volumes require the creation of snapshot mirrors for backup, decision support, and database error recovery.

Table 3-1 Creation of snapshot mirrors

Point-in-time copy application	Create snapshot mirrors for volumes containing...
Online database backup See "About online database backup" on page 41.	VxFS file systems for database datafiles to be backed up.
Off-host cluster file system backup See "About off-host cluster file system backup" on page 51.	VxFS cluster file systems to be backed up.
Decision support See "About decision support" on page 59.	VxFS file systems for database datafiles to be replicated.

Warning: 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 Version 20 DCOs" 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 the `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 about Administering Volume Snapshots, see the *Veritas Volume Manager Administrator's Guide*.
- Use a separate empty volume that you have prepared in advance.
  See "Creating a volume for use as a full-sized instant snapshot" on page 36.

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 37.

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

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 `disk01` and `disk02`:

```
# vxsnap -g mydg addmir vol1 nmirror=2 alloc=disk01,disk02
```

Note: Do not perform this step if you create a full-sized snapshot volume using a suitably prepared empty volume, or if you create space-optimized snapshots that use a cache .

See “Creating a volume for use as a full-sized instant snapshot” on page 36.

See “Creating a shared cache object” on page 37.

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):

```
TTY    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     zoo-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     disk04-01     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.

See “Considerations for placing DCO plexes” on page 34.

For more information on DCO objects, see the vxassist(1M) and vxsnap(1M) manual pages.

Considerations for placing DCO plexes

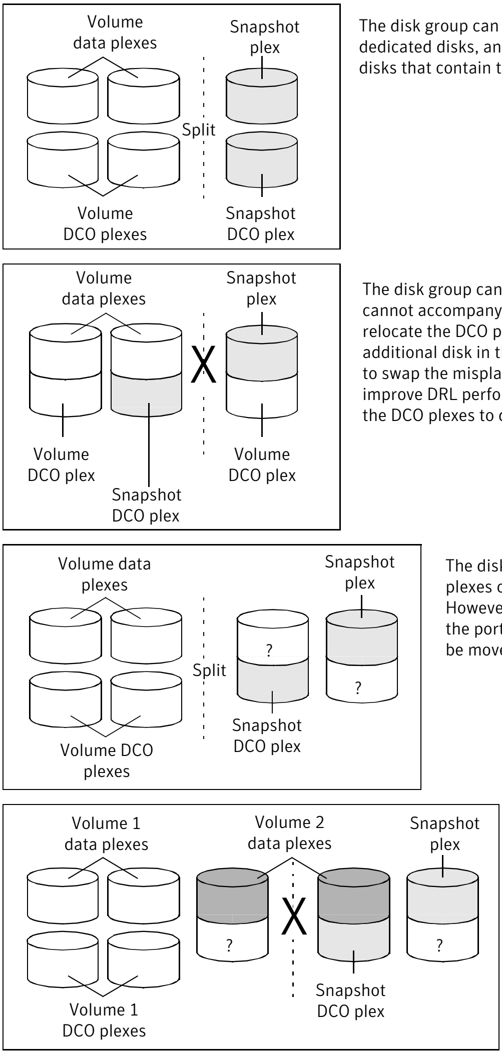
If you use the vxassist command or the Storage Foundation Manager (SFM) to create both a volume and its DCO, or the vxsnap prepare command to add a DCO to a volume, the DCO plexes are automatically placed on different disks from the data plexes of the parent volume. In previous releases, version 0 DCO 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 and later releases 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 I/O performance of 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.

Figure 3-8 illustrates some instances in which it is not be possible to split a disk group because of the location of the DCO plexes. Relocate DCO plexes as needed.

See “Preparing a volume for instant snapshot operations” on page 31.

Figure 3-8 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]
```

It is recommended that you 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 $LEN \
  [layout=mirror nmirror=number] init=active \
  [storage_attributes]
# vxsnap [-g diskgroup] prepare snapvol [ndcomirs=number] \
  regionsz=$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 `disk16` and `disk17`:

   ```
   # vxassist -g mydg make cachevol 1g layout=mirror \
     init=active disk16 disk17
   ```

   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 from 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.

Warning: 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

About online database backup

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.

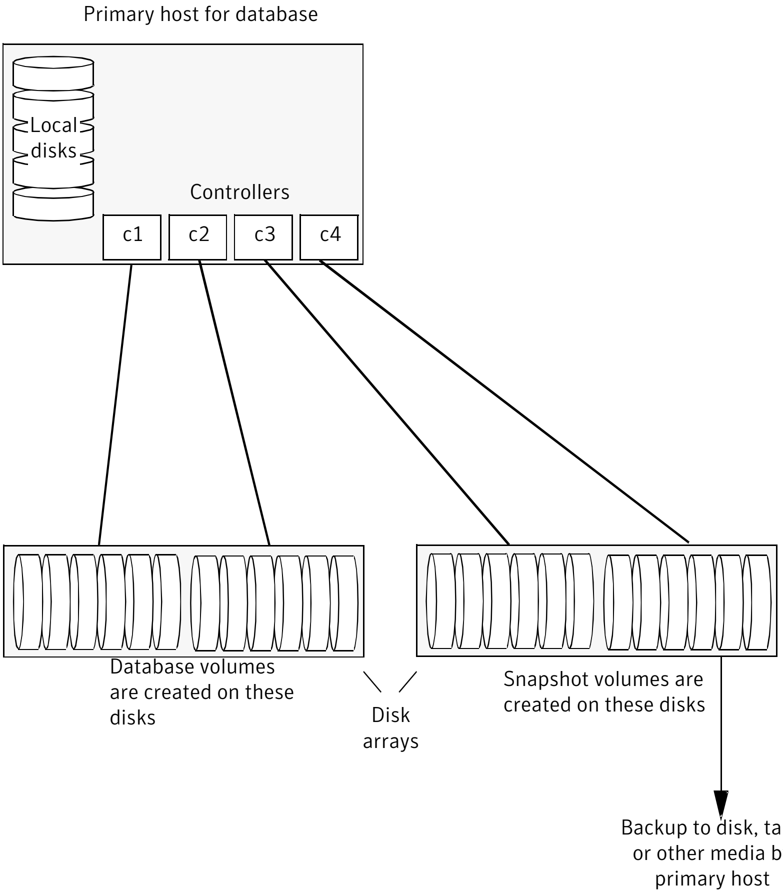
The following sections include sample scripts:

- Script to initiate online off-host backup of an Oracle database
- Script to put an Oracle database into hot backup mode
- Script to quiesce a Sybase ASE database
- Script to suspend I/O for a DB2 database
- Script to end Oracle database hot backup mode
- Script to release a Sybase ASE database from quiesce mode
- Script to resume I/O for a DB2 database
- Script to perform off-host backup

Making a backup of an online database on the same host

Figure 3-9 shows an example with two primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, which are configured on disks attached to controllers `c1` and `c2`, and the snapshots to be created on disks attached to controllers `c3` and `c4`.

Figure 3-9 Example system configuration for database backup on the primary host


Primary host for database
Local disks
Controllers
c1
c2
c3
c4
Database volumes are created on these disks
Disk arrays
Snapshot volumes are created on these disks
Backup to disk, tape or other media by 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 the example.

See "About setting up volumes for instant snapshots" on page 30.

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 commands to add one or more snapshot plexes to the volume, and to make a full-sized break-off snapshot, *snapvol*, of the tablespace volume by breaking off these plexes:

```
# vxsnap -g volumedg addmir volume [nmirror=N] \
  [alloc=storage_attributes]
# vxsnap -g volumedg make \
  source=volume/newvol=snapvol[/nmirror=N]\
  [alloc=storage_attributes]
```

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. You can specify storage attributes (such as a list of disks) to determine where the plexes are created.

If the volume layout does not support plex break-off, prepare an empty volume for the snapshot.

See "Creating a volume for use as a full-sized instant snapshot" on page 36.

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/snapvol=svol1 \
source=vol2/newvol=svol2 source=vol3/snapvol=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 37.

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 the example.
  See "Script to suspend I/O for a DB2 database" on page 234.
  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 the example.
  See "Script to put an Oracle database into hot backup mode" on page 233.
- 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 the example.
  See "Script to quiesce a Sybase ASE database" on page 233.

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 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 the example.
  See "Script to resume I/O for a DB2 database" on page 235.
- As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in the example.
  See "Script to end Oracle database hot backup mode" on page 234.

- As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in the example.

  See "Script to release a Sybase ASE database from quiesce mode" on page 235.

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 for both commands.

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

```
# umount 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
```

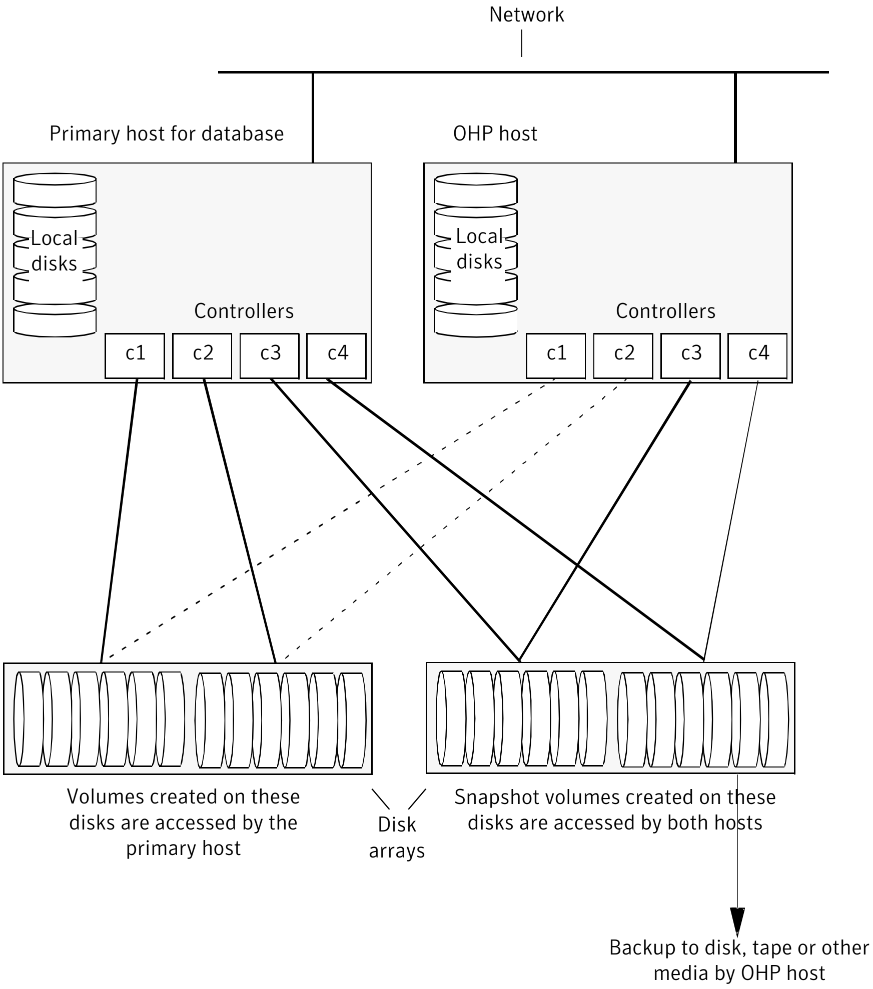
Note: You must shut down the database and unmount the file system that is configured on the original volume before attempting to resynchronize its contents from a snapshot.

Making an off-host backup of an online database

Figure 3-10 shows an example of two primary database volumes to be backed up, `dbase_vol` and `dbase_logs`, which are configured on disks attached to controllers `c1` and `c2`, and the snapshots to be 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.

Figure 3-10 Example system configuration for off-host database backup


Network
Primary host for database
OHP host
Local disks
Local disks
Controllers
Controllers
c1
c2
c3
c4
c1
c2
c3
c4
Volumes created on these disks are accessed by the primary host
Disk arrays
Snapshot volumes created on these disks are accessed by both hosts
Backup to disk, tape or other media by OHP 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 the example.

See "About setting up volumes for instant snapshots" on page 30.

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.

The procedure in this section is designed to minimize copy-on-write operations that can impact system performance. You can also implement this procedure on a single host by omitting steps 5 through 8 and 10 through 13 that split, deport, reimport and rejoin the snapshot disk group.

To make an off-host backup of an online database

1. On the primary host, add one or more snapshot plexes to the volume using this command:

   ```
   # vxsnap -g volumedg addmir volume [nmirror=N] \
    [alloc=storage_attributes]
   ```

   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. You can specify storage attributes (such as a list of disks) to determine where the plexes are created.

2. 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 the example.
     See "Script to suspend I/O for a DB2 database" on page 234.
     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 the example.
     See "Script to put an Oracle database into hot backup mode" on page 233.

- 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 the example.
  See "Script to quiesce a Sybase ASE database" on page 233.

3 Use the following command to make a full-sized snapshot, *snapvol*, of the tablespace volume by breaking off the plexes that you added in step 1 from the original volume:

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

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

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 \
alloc=ctlr:c3,ctlr:c4
```

This step sets up the snapshot volumes ready for the backup cycle, and starts tracking changes to the original volumes.

4 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 the example.
  See "Script to resume I/O for a DB2 database" on page 235.
- As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in the example.
  See "Script to end Oracle database hot backup mode" on page 234.
- As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in the example.
  See "Script to release a Sybase ASE database from quiesce mode" on page 235.

5 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 ...
```

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

```
# vxdg deport snapvoldg
```

7 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
```

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

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

9 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 for both commands.

Back up the file system using a command such as bpbackup in Symantec NetBackup. After the backup is complete, use the following command to unmount the file system.

```
# umount 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 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.

---

12 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
```

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

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

14 On the primary host, reattach the snapshot volumes to their original volume using the following command:

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

For example, to reattach the snapshot volumes svol1, svol2 and svol3:

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

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. You can use the vxsnap print command to check on the progress of synchronization.

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 backup of an Oracle database.

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.

Off-host cluster file system backup

About off-host cluster file system backup

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.

Figure 3-11 shows an example where 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 to be created on disks attached to controllers c3 and c4.

You can mount a VxFS file system for shared access by the nodes of a cluster.

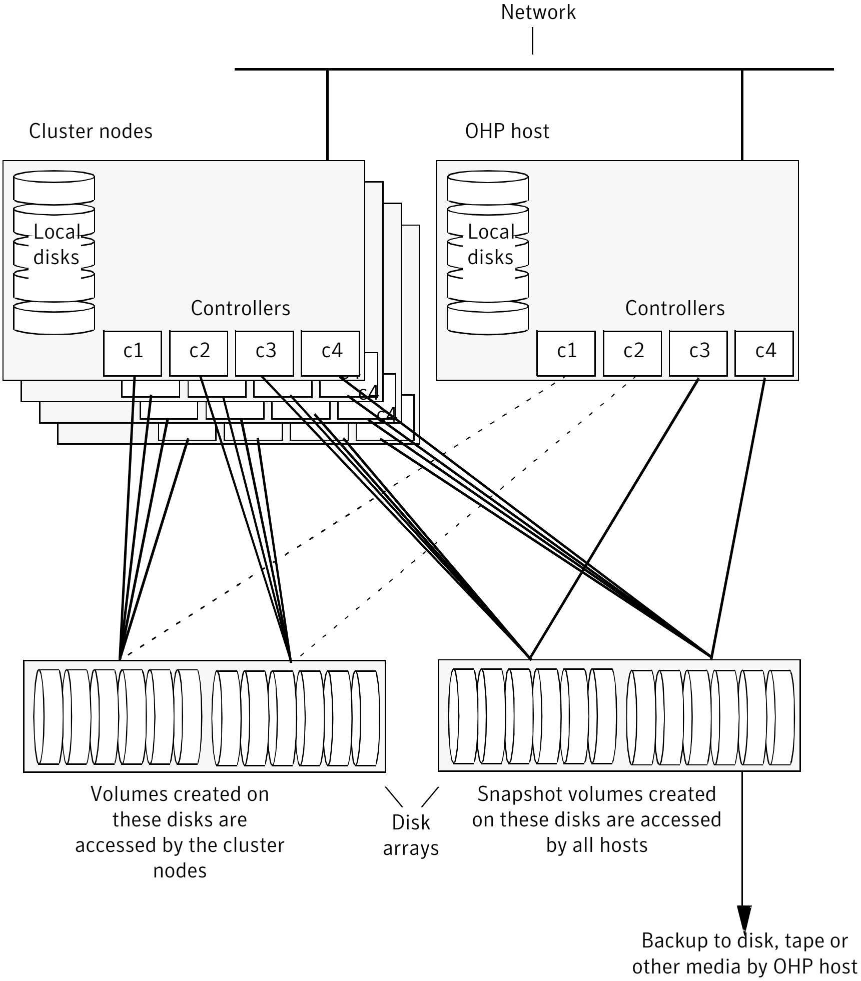
See "Mounting a file system for shared access" on page 52.

You can perform off-host backups of cluster-shared file systems.

See "Using off-host processing to back up cluster file systems" on page 53.

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

Figure 3-11 System configuration for off-host file system backup scenarios


Network
Cluster nodes
OHP host
Local disks
Controllers
c1
c2
c3
c4
Local disks
Controllers
c1
c2
c3
c4
Volumes created on these disks are accessed by the cluster nodes
Disk arrays
Snapshot volumes created on these disks are accessed by all hosts
Backup to disk, tape or other media by OHP host

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

Before using this procedure, you must prepare the volumes containing the file systems that are to be backed up.

See "About setting up volumes for instant snapshots" on page 30.

Warning: This procedure assumes that you have already prepared the volumes containing the file systems that are to be backed up.

To back up a snapshot of a mounted file system which has shared access

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\
   /nmirror=1
   ```

   If the volume does not have any available plexes, or its layout does not support plex break-off, prepare an empty volume for the snapshot.

   See "Creating a volume for use as a full-sized instant snapshot" on page 36.

   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 command on the OHP host to recover and restart the snapshot volume:

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

For example, to start the volume snapvol:

```
# vxrecover -g splitdg -m 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 both commands.

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

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

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

```
# umount mount_point
```

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

```
# vxdg deport snapvoldg
```

For example, to deport splitdg:

```
# vxdg deport splitdg
```

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
```

For example, to import splitdg:

```
# vxdg -s import splitdg
```

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 command on the primary host to recover and restart the snapshot volume:

```
# vxrecover -g volumedg -m 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.

Warning: 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

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.

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

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.

To reattach a snapshot

◆ Use the following command to reattach an instant snapshot 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]
```

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.

The `vxsnap refresh` and `vxsnap reattach` commands have slightly different behaviors.

The `vxsnap reattach` command reattaches a snapshot volume to its source volume and begins copying the volume data to the snapshot volume.

The `vxsnap refresh` commands updates the snapshot volumes contents view. The updated snapshot is available immediately with the new contents while synchronization occurs in the background.

Decision support

About decision support

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 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:

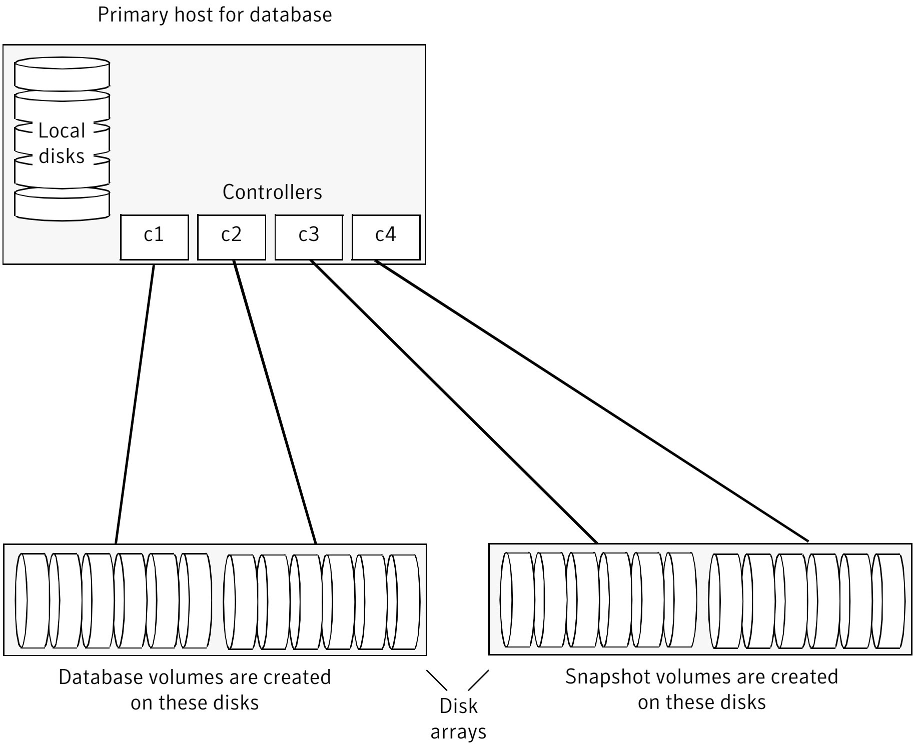
- Script to put an Oracle database into hot backup mode
- Script to quiesce a Sybase ASE database
- Script to suspend I/O for a DB2 database
- Script to end Oracle database hot backup mode
- Script to release a Sybase ASE database from quiesce mode
- Script to resume I/O for a DB2 database
- Script to create an off-host replica Oracle database

- Script to complete, recover and start a replica Oracle database
- Script to start a replica Sybase ASE database

Creating a replica database on the same host

Figure 3-12 shows an example where 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 to be created on disks attached to controllers `c3` and `c4`.

Figure 3-12 Example system configuration for decision support on the primary host


Primary host for database
Local disks
Controllers
c1
c2
c3
c4
Database volumes are created on these disks
Disk arrays
Snapshot volumes are created on these disks

Note: It is assumed that you have already prepared the database volumes to be replicated as described in the example.

See "About setting up volumes for instant snapshots" on page 30.

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, prepare an empty volume for the snapshot.

See "Creating a volume for use as a full-sized instant snapshot" on page 36.

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/snapvol=svol1/nmirror=2 \
source=vol2/snapvol=svol2/nmirror=2 \
source=vol3/snapvol=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 37.

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 the example. Note that to allow recovery from any backups taken from snapshots, the database must be in LOGRETAIN RECOVERY mode.
  See "Script to suspend I/O for a DB2 database" on page 234.
- 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 the example.
  See "Script to put an Oracle database into hot backup mode" on page 233.
- 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 the example.
  See "Script to quiesce a Sybase ASE database" on page 233.
  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 72.

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 3, perform the following steps.

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 the example.
  See "Script to resume I/O for a DB2 database" on page 235.
- As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in the example.
  See "Script to end Oracle database hot backup mode" on page 234.
- As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in the example.
  See "Script to release a Sybase ASE database from quiesce mode" on page 235.
  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 72.

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 both commands.

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 the example. See "Script to complete, recover and start a replica Oracle database" on page 239.
- For a Sybase ASE database, use a script such as that shown in the example. See "Script to start a replica Sybase ASE database" on page 240. 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.

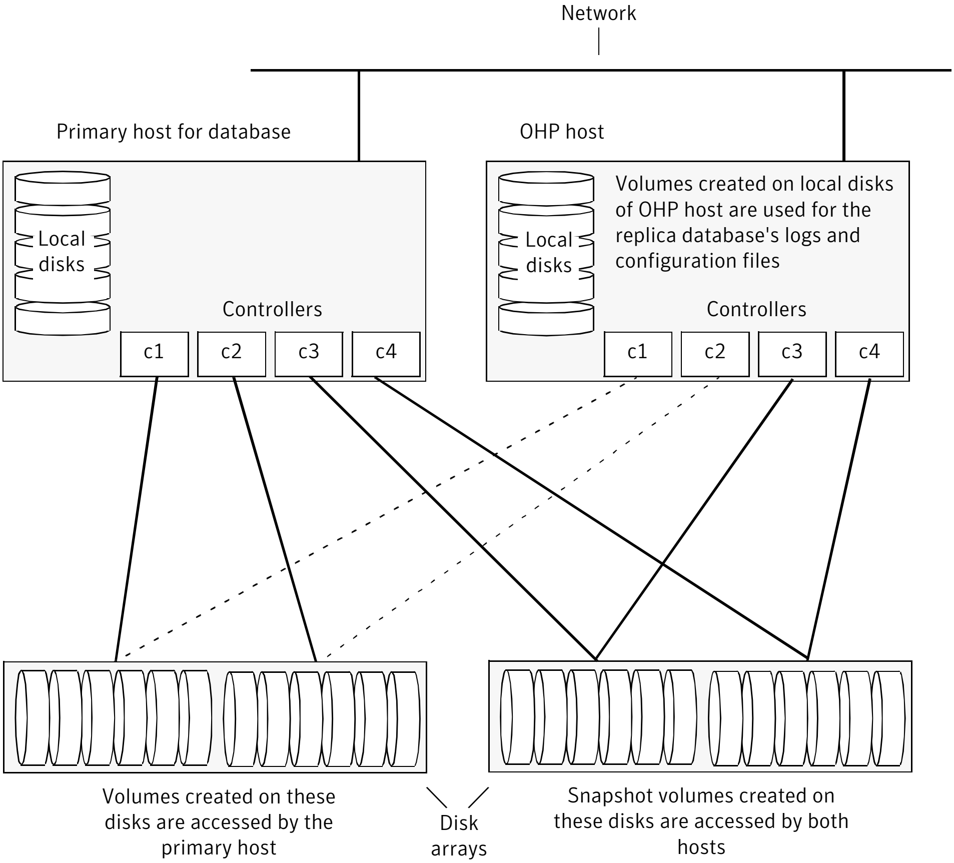
Creating an off-host replica database

Figure 3-13 shows an example where 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 to be 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.

Figure 3-13 Example system configuration for off-host decision support


Network
Primary host for database
OHP host
Local disks
Controllers
c1
c2
c3
c4
Local disks
Volumes created on local disks of OHP host are used for the replica database's logs and configuration files
Controllers
c1
c2
c3
c4
Volumes created on these disks are accessed by the primary host
Disk arrays
Snapshot volumes created on these disks are accessed by both hosts

Note: It is assumed that you have already prepared the database volumes to be replicated as described in the example.

See "About setting up volumes for instant snapshots" on page 30.

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 "About preparing a replica Oracle database" on page 243.

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, prepare an empty volume for the snapshot.

   See "Creating a volume for use as a full-sized instant snapshot" on page 36.

   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/snapvol=svol1 \
   source=vol2/snapvol=svol2 source=vol3/snapvol=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 the example. Note that if the replica database must be able to be rolled forward (for example, if it is to be used as a standby database), the primary database must be in LOGRETAIN RECOVERY mode.
     See "Script to suspend I/O for a DB2 database" on page 234.

- 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 the example.
  See "Script to put an Oracle database into hot backup mode" on page 233.
- 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 the example.
  See "Script to quiesce a Sybase ASE database" on page 233.
  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 72.

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 the example.
  See "Script to resume I/O for a DB2 database" on page 235.
- As the Oracle database administrator, release all the tablespaces from hot backup mode using a script such as that shown in the example.
  See "Script to end Oracle database hot backup mode" on page 234.
- As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in the example.

See "Script to release a Sybase ASE database from quiesce mode" on page 235.

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 ...
```

For example to split the snap volumes from dbasedg:

```
# vxdg split dbasedg snapvoldg svol1 svol2 svol3
```

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 command on the OHP host to recover and restart the snapshot volumes:

```
# vxrecover -g snapvoldg -m 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 both commands.

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/snapvoldg/snap1_dbase_vol
# mount -F vxfs /dev/vx/dsk/snapvoldg/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 the example.
  See "Script to complete, recover and start a replica Oracle database" on page 239.
  (This script also creates the control file for the new database by executing the SQL script that you created using the procedure in About preparing a replica Oracle database.)
- For a Sybase ASE database, use a script such as that shown in the example.
  See "Script to start a replica Sybase ASE database" on page 240.
  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

This procedure describes how to resynchronize the data in a snapshot with the primary host.

To resynchronize a snapshot with the primary database

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

   ```
   # umount 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 command on the primary host to recover and restart a snapshot volume:

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

6. Use step 3 through step 5 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.

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

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.

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

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.

To reattach a snapshot

- Use the following command, to reattach some or all plexes of an instant snapshot 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]
  ```

  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.

Storage Checkpoints

Storage Checkpoints

You can use 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.

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.

See "About Storage Checkpoints" on page 79.

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.

Warning: 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

The procedure in this section describes how to roll back a database using a Storage Checkpoint, for example, after a logical error has occurred.

To roll back a database

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.

Chapter 4

Storage Checkpoints

This chapter includes the following topics:

About Storage Checkpoints

Veritas File System (VxFS) provides a Storage Checkpoint feature that quickly creates a persistent image of a file system at an exact point in time. Storage Checkpoints significantly reduce I/O overhead by identifying and maintaining only the file system blocks that have changed since the last Storage Checkpoint or backup via a copy-on-write technique.

See “Copy-on-write” on page 83.

Storage Checkpoints provide:

- Persistence through reboots and crashes.
- The ability for data to be immediately writeable by preserving the file system metadata, the directory hierarchy, and user data.

Storage Checkpoints are actually data objects that are managed and controlled by the file system. You can create, remove, and rename Storage Checkpoints because they are data objects with associated names.

See “How a Storage Checkpoint works” on page 81.

Unlike a disk-based mirroring technology that requires a separate storage space, Storage Checkpoints minimize the use of disk space by using a Storage Checkpoint within the same free space available to the file system.

After you create a Storage Checkpoint of a mounted file system, you can also continue to create, remove, and update files on the file system without affecting the logical image of the Storage Checkpoint. A Storage Checkpoint preserves not only the name space (directory hierarchy) of the file system, but also the user data as it existed at the moment the file system image was captured.

You can use a Storage checkpoint in many ways. For example, you can use them to:

- Create a stable image of the file system that can be backed up to tape.
- Provide a mounted, on-disk backup of the file system so that end users can restore their own files in the event of accidental deletion. This is especially useful in a home directory, engineering, or email environment.
- Create a copy of an application's binaries before installing a patch to allow for rollback in case of problems.
- Create an on-disk backup of the file system in that can be used in addition to a traditional tape-based backup to provide faster backup and restore capabilities.
- Test new software on a point-in-time image of the primary fileset without jeopardizing the live data in the current primary fileset by mounting the Storage Checkpoints as writable.

How Storage Checkpoints differ from snapshots

Storage Checkpoints differ from Veritas File System snapshots in the following ways because they:

- Allow write operations to the Storage Checkpoint itself.
- Persist after a system reboot or failure.
- Share the same pool of free space as the file system.
- Maintain a relationship with other Storage Checkpoints by identifying changed file blocks since the last Storage Checkpoint.
- Can have multiple, read-only Storage Checkpoints that reduce I/O operations and required storage space because the most recent Storage Checkpoint is the only one that accumulates updates from the primary file system.

- Can restore the file system to its state at the time that the Storage Checkpoint was taken.

Various backup and replication solutions can take advantage of Storage Checkpoints. The ability of Storage Checkpoints to track the file system blocks that have changed since the last Storage Checkpoint facilitates backup and replication applications that only need to retrieve the changed data. Storage Checkpoints significantly minimize data movement and may promote higher availability and data integrity by increasing the frequency of backup and replication solutions.

Storage Checkpoints can be taken in environments with a large number of files, such as file servers with millions of files, with little adverse impact on performance. Because the file system does not remain frozen during Storage Checkpoint creation, applications can access the file system even while the Storage Checkpoint is taken. However, Storage Checkpoint creation may take several minutes to complete depending on the number of files in the file system.

How a Storage Checkpoint works

The Storage Checkpoint facility freezes the mounted file system (known as the primary fileset), initializes the Storage Checkpoint, and thaws the file system. Specifically, the file system is first brought to a stable state where all of its data is written to disk, and the freezing process momentarily blocks all I/O operations to the file system. A Storage Checkpoint is then created without any actual data; the Storage Checkpoint instead points to the block map of the primary fileset. The thawing process that follows restarts I/O operations to the file system.

You can create a Storage Checkpoint on a single file system or a list of file systems. A Storage Checkpoint of multiple file systems simultaneously freezes the file systems, creates a Storage Checkpoint on all of the file systems, and thaws the file systems. As a result, the Storage Checkpoints for multiple file systems have the same creation timestamp. The Storage Checkpoint facility guarantees that multiple file system Storage Checkpoints are created on all or none of the specified file systems, unless there is a system crash while the operation is in progress.

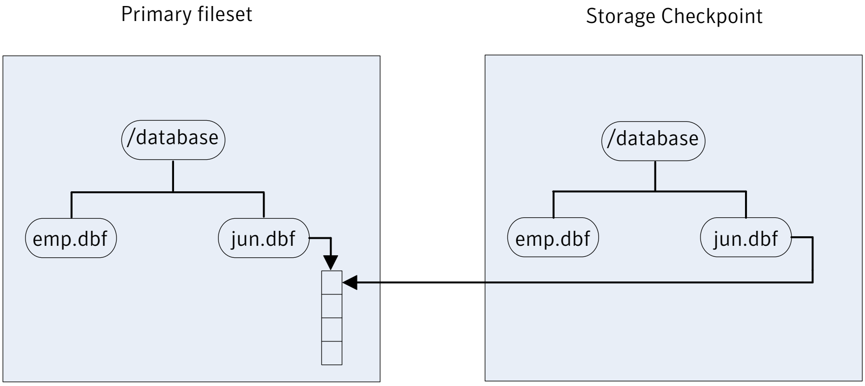
Note: The calling application is responsible for cleaning up Storage Checkpoints after a system crash.

A Storage Checkpoint of the primary fileset initially contains only pointers to the existing data blocks in the primary fileset, and does not contain any allocated data blocks of its own.

Figure 4-1 shows the file system `/database` and its Storage Checkpoint.

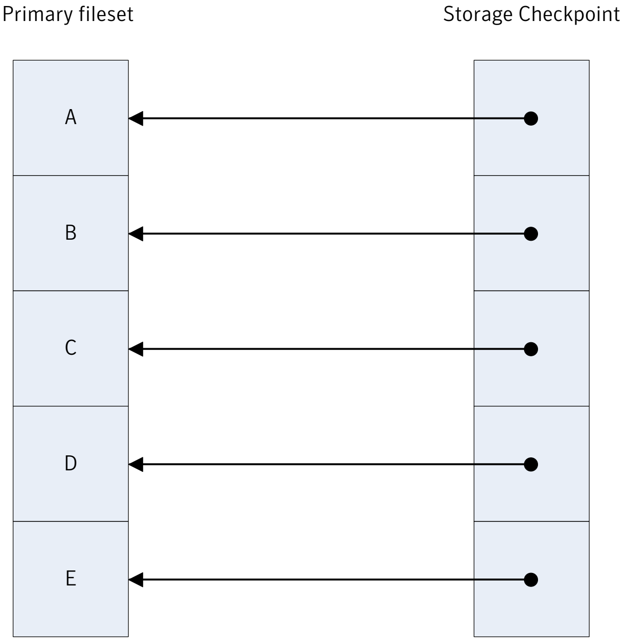
The Storage Checkpoint is logically identical to the primary fileset when the Storage Checkpoint is created, but it does not contain any actual data blocks.

Figure 4-1 Primary fileset and its Storage Checkpoint



In Figure 4-2, a square represents each block of the file system. This figure shows a Storage Checkpoint containing pointers to the primary fileset at the time the Storage Checkpoint is taken, as in Figure 4-1.

Figure 4-2 Initializing a Storage Checkpoint



The Storage Checkpoint presents the exact image of the file system by finding the data from the primary fileset. VxFS updates a Storage Checkpoint by using the copy-on-write technique.

See “Copy-on-write” on page 83.

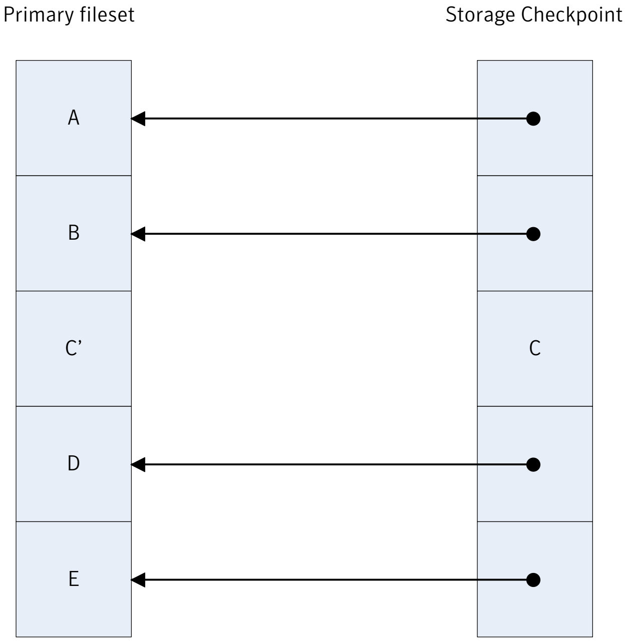
Copy-on-write

In Figure 4-3, the third data block in the primary fileset originally containing C is updated.

Before the data block is updated with new data, the original data is copied to the Storage Checkpoint. This is called the copy-on-write technique, which allows the Storage Checkpoint to preserve the image of the primary fileset when the Storage Checkpoint is taken.

Every update or write operation does not necessarily result in the process of copying data to the Storage Checkpoint because the old data needs to be saved only once. As blocks in the primary fileset continue to change, the Storage Checkpoint accumulates the original data blocks. In this example, subsequent updates to the third data block, now containing C', are not copied to the Storage Checkpoint because the original image of the block containing C is already saved.

Figure 4-3 Updates to the primary fileset



Types of Storage Checkpoints

You can create the following types of Storage Checkpoints:

- Data Storage Checkpoints
- Nodata Storage Checkpoints
- Removable Storage Checkpoints
- Non-mountable Storage Checkpoints

Data Storage Checkpoints

A data Storage Checkpoint is a complete image of the file system at the time the Storage Checkpoint is created. This type of Storage Checkpoint contains the file system metadata and file data blocks. You can mount, access, and write to a data Storage Checkpoint just as you would to a file system. Data Storage Checkpoints are useful for backup applications that require a consistent and stable image of an active file system. Data Storage Checkpoints introduce some overhead to the system and to the application performing the write operation. For best results, limit the life of data Storage Checkpoints to minimize the impact on system resources.

See "Showing the difference between a data and a nodata Storage Checkpoint" on page 90.

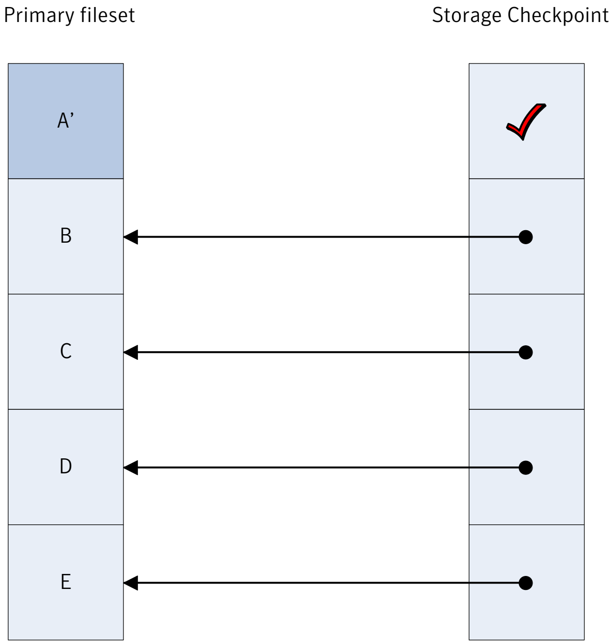
Nodata Storage Checkpoints

A nodata Storage Checkpoint only contains file system metadata—no file data blocks. As the original file system changes, the nodata Storage Checkpoint records the location of every changed block. Nodata Storage Checkpoints use minimal system resources and have little impact on the performance of the file system because the data itself does not have to be copied.

In Figure 4-4, the first block originally containing A is updated.

The original data is not copied to the storage checkpoint, but the changed block is marked in the Storage Checkpoint. The marker indicates which data has changed.

Figure 4-4 Updates to a nodata clone



See “Showing the difference between a data and a nodata Storage Checkpoint” on page 90.

Removable Storage Checkpoints

A removable Storage Checkpoint can “self-destruct” under certain conditions when the file system runs out of space.

See “Space management considerations” on page 97.

After encountering certain out-of-space (`ENOSPC`) conditions, the kernel removes Storage Checkpoints to free up space for the application to continue running on the file system. In almost all situations, you should create Storage Checkpoints with the removable attribute.

Non-mountable Storage Checkpoints

You can create Storage Checkpoints that cannot be mounted by using the `fsckptadm set nomount` command.

See the `fsckptadm`(1M) manual page.

Use this type of Storage Checkpoint as a security feature which prevents other applications from accessing the Storage Checkpoint and modifying it.

Storage Checkpoint administration

Storage Checkpoint administrative operations require the `fsckptadm` utility.

See the `fsckptadm`(1M) manual page.

You can use the `fsckptadm` utility to create and remove Storage Checkpoints, change attributes, and ascertain statistical data. Every Storage Checkpoint has an associated name, which allows you to manage Storage Checkpoints; this name is limited to 127 characters and cannot contain a colon (:).

Storage Checkpoints require some space for metadata on the volume or set of volumes specified by the file system allocation policy or Storage Checkpoint allocation policy. The `fsckptadm` utility displays an error if the volume or set of volumes does not have enough free space to contain the metadata. You can roughly approximate the amount of space required by the metadata using a method that depends on the disk layout version of the file system.

For disk layout Version 6 or 7, multiply the number of inodes by 1 byte, and add 1 or 2 megabytes to get the approximate amount of space required. You can determine the number of inodes with the `fsckptadm` utility.

Use the `fsvoladm` command to determine if the volume set has enough free space.

See the `fsvoladm`(1M) manual page.

```
# fsvoladm list /mnt0
devid     size       used    avail       name
0     20971520       8497658    12473862       mnt1
1     20971520       6328993    14642527       mnt2
2     20971520       4458462    16513058       mnt3
```

Creating a Storage Checkpoint

The following example shows the creation of a nodata Storage Checkpoint named `thu_7pm` on `/mnt0` and lists all Storage Checkpoints of the `/mnt0` file system:

```
# fsckptadm -n create thu_7pm /mnt0
# fsckptadm list /mnt0
/mnt0
thu_7pm:
 ctime   = Thu 3 Mar 2005 7:00:17 PM PST
 mtime   = Thu 3 Mar 2005 7:00:17 PM PST
 flags   = nodata, largefiles
```

The following example shows the creation of a removable Storage Checkpoint named `thu_8pm` on `/mnt0` and lists all Storage Checkpoints of the `/mnt0` file system:

```
# fsckptadm -r create thu_8pm /mnt0
# fsckptadm list /mnt0
/mnt0
thu_8pm:
 ctime   = Thu 3 Mar 2005 8:00:19 PM PST
 mtime   = Thu 3 Mar 2005 8:00:19 PM PST
 flags   = largefiles, removable
thu_7pm:
 ctime   = Thu 3 Mar 2005 7:00:17 PM PST
 mtime   = Thu 3 Mar 2005 7:00:17 PM PST
 flags   = nodata, largefiles
```

Removing a Storage Checkpoint

You can delete a Storage Checkpoint by specifying the remove keyword of the `fsckptadm` command. Specifically, you can use either the synchronous or asynchronous method of removing a Storage Checkpoint; the asynchronous method is the default method. The synchronous method entirely removes the Storage Checkpoint and returns all of the blocks to the file system before completing the `fsckptadm` operation. The asynchronous method simply marks the Storage Checkpoint for removal and causes `fsckptadm` to return immediately. At a later time, an independent kernel thread completes the removal operation and releases the space used by the Storage Checkpoint.

In this example, `/mnt0` is a mounted VxFS file system with a Version 6 disk layout. This example shows the asynchronous removal of the Storage Checkpoint named `thu_8pm` and synchronous removal of the Storage Checkpoint named `thu_7pm`. This example also lists all the Storage Checkpoints remaining on the `/mnt0` file system after the specified Storage Checkpoint is removed:

```
# fsckptadm remove thu_8pm /mnt0
# fsckptadm list /mnt0
/mnt0
thu_7pm:
 ctime   = Thu 3 Mar 2005 7:00:17 PM PST
 mtime   = Thu 3 Mar 2005 7:00:17 PM PST
 flags   = nodata, largefiles
# fsckptadm -s remove thu_7pm /mnt0
# fsckptadm list /mnt0
/mnt0
```

Accessing a Storage Checkpoint

You can mount Storage Checkpoints using the `mount` command with the mount option `-o ckpt=ckpt_name`.

See the `mount_vxfs`(1M) manual page.

Observe the following rules when mounting Storage Checkpoints:

- Storage Checkpoints are mounted as read-only Storage Checkpoints by default. If you must write to a Storage Checkpoint, mount it using the `-o rw` option.
- If a Storage Checkpoint is currently mounted as a read-only Storage Checkpoint, you can remount it as a writable Storage Checkpoint using the `-o remount` option.
- To mount a Storage Checkpoint of a file system, first mount the file system itself.
- To unmount a file system, first unmount all of its Storage Checkpoints.

Warning: If you create a Storage Checkpoint for backup purposes, do not mount it as a writable Storage Checkpoint. You will lose the point-in-time image if you accidently write to the Storage Checkpoint.

If older Storage Checkpoints already exist, write activity to a writable Storage Checkpoint can generate copy operations and increased space usage in the older Storage Checkpoints.

A Storage Checkpoint is mounted on a special pseudo device. This pseudo device does not exist in the system name space; the device is internally created by the system and used while the Storage Checkpoint is mounted. The pseudo device is removed after you unmount the Storage Checkpoint. A pseudo device name is formed by appending the Storage Checkpoint name to the file system device name using the colon character (:) as the separator.

For example, if a Storage Checkpoint named `may_23` belongs to the file system residing on the special device `/dev/vx/dsk/fsvol/vol1`, the Storage Checkpoint pseudo device name is:

```
/dev/vx/dsk/fsvol/vol1:may_23
```

- To mount the Storage Checkpoint named `may_23` as a read-only Storage Checkpoint on directory `/fsvol_may_23`, type:

```
# mount -F vxfs -o ckpt=may_23 /dev/vx/dsk/fsvol/vol1:may_23 \
/fsvol_may_23
```

Note: The vol1 file system must already be mounted before the Storage Checkpoint can be mounted.

- To remount the Storage Checkpoint named may_23 as a writable Storage Checkpoint, type:

  ```
  # mount -F vxfs -o ckpt=may_23,remount,rw \
  /dev/vx/dsk/fsvol/vol1:may_23 /fsvol_may_23
  ```

- To mount this Storage Checkpoint automatically when the system starts up, put the following entries in the /etc/vfstab file:

```
#device              device to     mount point    FS   fsck mount   mount
#to mount            fsck                         type pass at boot options
/dev/vx/dsk/fsvol/   /dev/vx/rdsk/ /fsvol         vxfs 1    yes     -
  vol1                 fsvol/vol1
/dev/vx/dsk/fsvol/   -             /fsvol_may_23  vxfs 0    yes     ckpt=may_23
  vol1:may_23
```

- To mount a Storage Checkpoint of a cluster file system, you must also use the -o cluster option:

  ```
  # mount -F vxfs -o cluster,ckpt=may_23 \
  /dev/vx/dsk/fsvol/vol1:may_23 /fsvol_may_23
  ```

  You can only mount a Storage Checkpoint cluster-wide if the file system that the Storage Checkpoint belongs to is also mounted cluster-wide. Similarly, you can only mount a Storage Checkpoint locally if the file system that the Storage Checkpoint belongs to is mounted locally.

You can unmount Storage Checkpoints using the umount command.

See the umount_vxfs(1M) manual page.

Storage Checkpoints can be unmounted by the mount point or pseudo device name:

```
# umount /fsvol_may_23
# umount /dev/vx/dsk/fsvol/vol1:may_23
```

Note: You do not need to run the `fsck` utility on Storage Checkpoint pseudo devices because pseudo devices are part of the actual file system.

Converting a data Storage Checkpoint to a nodata Storage Checkpoint

A nodata Storage Checkpoint does not contain actual file data. Instead, this type of Storage Checkpoint contains a collection of markers indicating the location of all the changed blocks since the Storage Checkpoint was created.

See “Types of Storage Checkpoints” on page 84.

You can use either the synchronous or asynchronous method to convert a data Storage Checkpoint to a nodata Storage Checkpoint; the asynchronous method is the default method. In a synchronous conversion, `fsckptadm` waits for all files to undergo the conversion process to “nodata” status before completing the operation. In an asynchronous conversion, `fsckptadm` returns immediately and marks the Storage Checkpoint as a nodata Storage Checkpoint even though the Storage Checkpoint's data blocks are not immediately returned to the pool of free blocks in the file system. The Storage Checkpoint deallocates all of its file data blocks in the background and eventually returns them to the pool of free blocks in the file system.

If all of the older Storage Checkpoints in a file system are nodata Storage Checkpoints, use the synchronous method to convert a data Storage Checkpoint to a nodata Storage Checkpoint. If an older data Storage Checkpoint exists in the file system, use the asynchronous method to mark the Storage Checkpoint you want to convert for a delayed conversion. In this case, the actual conversion will continue to be delayed until the Storage Checkpoint becomes the oldest Storage Checkpoint in the file system, or all of the older Storage Checkpoints have been converted to nodata Storage Checkpoints.

Note: You cannot convert a nodata Storage Checkpoint to a data Storage Checkpoint because a nodata Storage Checkpoint only keeps track of the location of block changes and does not save the content of file data blocks.

Showing the difference between a data and a nodata Storage Checkpoint

The following example shows the difference between data Storage Checkpoints and nodata Storage Checkpoints.

Note: A nodata Storage Checkpoint does not contain actual file data.

See "Converting a data Storage Checkpoint to a nodata Storage Checkpoint" on page 90.

To show the difference between Storage Checkpoints

1. Create a file system and mount it on /mnt0, as in the following example:

   ```
   # mkfs -F vxfs /dev/vx/rdsk/dg1/test0

   version 7 layout
   134217728 sectors, 67108864 blocks of size 1024, log \

   size  65536 blocks, largefiles supported
   # mount -F /dev/vx/rdsk/dg1/test0 /mnt0
   ```

2. Create a small file with a known content, as in the following example:

   ```
   # echo "hello, world" > /mnt0/file
   ```

3. Create a Storage Checkpoint and mount it on /mnt0@5_30pm, as in the following example:

   ```
   # fsckptadm create ckpt@5_30pm /mnt0
   # mkdir /mnt0@5_30pm
   # mount -F vxfs -o ckpt=ckpt@5_30pm \
   /dev/vx/dsk/dg1/test0:ckpt@5_30pm /mnt0@5_30pm
   ```

4. Examine the content of the original file and the Storage Checkpoint file:

   ```
   # cat /mnt0/file
   hello, world
   # cat /mnt0@5_30pm/file
   hello, world
   ```

5. Change the content of the original file:

   ```
   # echo "goodbye" > /mnt0/file
   ```

6. Examine the content of the original file and the Storage Checkpoint file. The original file contains the latest data while the Storage Checkpoint file still contains the data at the time of the Storage Checkpoint creation:

   ```
   # cat /mnt0/file
   goodbye
   # cat /mnt0@5_30pm/file
   hello, world
   ```

7 Unmount the Storage Checkpoint, convert the Storage Checkpoint to a nodata Storage Checkpoint, and mount the Storage Checkpoint again:

```
# umount /mnt0@5_30pm
# fsckptadm -s set nodata ckpt@5_30pm /mnt0
# mount -F vxfs -o ckpt=ckpt@5_30pm \
/dev/vx/dsk/dg1/test0:ckpt@5_30pm /mnt0@5_30pm
```

8 Examine the content of both files. The original file must contain the latest data:

```
# cat /mnt0/file
goodbye
```

You can traverse and read the directories of the nodata Storage Checkpoint; however, the files contain no data, only markers to indicate which block of the file has been changed since the Storage Checkpoint was created:

```
# ls -l /mnt0@5_30pm/file
-rw-r--r--   1 root     other 13 Jul 13 17:13 \
# cat /mnt0@5_30pm/file
cat: input error on /mnt0@5_30pm/file: I/O error
```

Converting multiple Storage Checkpoints

You can convert Storage Checkpoints to nodata Storage Checkpoints, when dealing with older Storage Checkpoints on the same file system.

To convert multiple Storage Checkpoints

1. Create a file system and mount it on `/mnt0`:

   ```
   # mkfs -F vxfs /dev/vx/rdsk/dg1/test0
   version 7 layout
   13417728 sectors, 67108864 blocks of size 1024, log \
   size 65536 blocks largefiles supported
   # mount -F vxfs /dev/vx/dsk/dg1/test0 /mnt0
   ```

2. Create four data Storage Checkpoints on this file system, note the order of creation, and list them:

   ```
   # fsckptadm create oldest /mnt0
   # fsckptadm create older /mnt0
   # fsckptadm create old /mnt0
   # fsckptadm create latest /mnt0
   # fsckptadm list /mnt0
   /mnt0
   latest:
     ctime              =  Mon 26 Jul 11:56:55 2004
     mtime              =  Mon 26 Jul 11:56:55 2004
     flags              =  largefiles
   old:
     ctime              =  Mon 26 Jul 11:56:51 2004
     mtime              =  Mon 26 Jul 11:56:51 2004
     flags              =  largefiles
   older:
     ctime              =   Mon 26 Jul 11:56:46 2004
     mtime              =  Mon 26 Jul 11:56:46 2004
     flags              =  largefiles
   oldest:
     ctime              =  Mon 26 Jul 11:56:41 2004
     mtime              =  Mon 26 Jul 11:56:41 2004
     flags              =  largefiles
   ```

3. Try to convert synchronously the `latest` Storage Checkpoint to a nodata Storage Checkpoint. The attempt will fail because the Storage Checkpoints older than the `latest` Storage Checkpoint are data Storage Checkpoints, namely the Storage Checkpoints `old`, `older`, and `oldest`:

   ```
   # fsckptadm -s set nodata latest /mnt0
   UX:vxfs fsckptadm: ERROR: V-3-24632: Storage Checkpoint
   set failed on latest.  File exists (17)
   ```

4 You can instead convert the `latest` Storage Checkpoint to a nodata Storage Checkpoint in a delayed or asynchronous manner.

```
# fsckptadm set nodata latest /mnt0
```

5 List the Storage Checkpoints, as in the following example. You will see that the `latest` Storage Checkpoint is marked for conversion in the future.

```
# fsckptadm list /mnt0
/mnt0
latest:
  ctime                = Mon 26 Jul 11:56:55 2004
  mtime                = Mon 26 Jul 11:56:55
  flags                = nodata, largefiles, delayed
old:
  ctime                = Mon 26 Jul 11:56:51 2004
  mtime                = Mon 26 Jul 11:56:51 2004
  flags                = largefiles
older:
  ctime                = Mon 26 Jul 11:56:46 2004
  mtime                = Mon 26 Jul 11:56:46 2004
  flags                = largefiles
oldest:
  ctime                = Mon 26 Jul 11:56:41 2004
  mtime                = Mon 26 Jul 11:56:41 2004
  flags                = largefiles
```

Creating a delayed nodata Storage Checkpoint

You can combine the three previous steps and create the `latest` Storage Checkpoint as a nodata Storage Checkpoint. The creation process will detect the presence of the older data Storage Checkpoints and create the `latest` Storage Checkpoint as a delayed nodata Storage Checkpoint.

To create a delayed nodata Storage Checkpoint

1 Remove the `latest` Storage Checkpoint.

```
# fsckptadm remove latest /mnt0
# fsckptadm list /mnt0
/mnt0
old:
  ctime                 = Mon 26 Jul 11:56:51 2004
  mtime                 = Mon 26 Jul 11:56:51 2004
  flags                 = largefiles
older:
  ctime                 = Mon 26 Jul 11:56:46 2004
  mtime                 = Mon 26 Jul 11:56:46 2004
  flags                 = largefiles
oldest:
  ctime                 = Mon 26 Jul 11:56:41 2004
  mtime                 = Mon 26 Jul 11:56:41 2004
  flags                 = largefiles
```

2 Recreate the `latest` Storage Checkpoint as a nodata Storage Checkpoint.

```
# fsckptadm -n create latest /mnt0
# fsckptadm list /mnt0
/mnt0
latest:
  ctime                 = Mon 26 Jul 12:06:42 2004
  mtime                 = Mon 26 Jul 12:06:42 2004
  flags                 = nodata, largefiles, delayed
old:
  ctime                 = Mon 26 Jul 11:56:51 2004
  mtime                 = Mon 26 Jul 11:56:51 2004
  flags                 = largefiles
older:
  ctime                 = Mon 26 Jul 11:56:46 2004
  mtime                 = Mon 26 Jul 11:56:46 2004
  flags                 = largefiles
oldest:
  ctime                 = Mon 26 Jul 11:56:41 2004
  mtime                 = Mon 26 Jul 11:56:41 2004
  flags                 = largefiles
```

3 Convert the `oldest` Storage Checkpoint to a nodata Storage Checkpoint because no older Storage Checkpoints exist that contain data in the file system.

Note: This step can be done synchronously.

```
# fsckptadm -s set nodata oldest /mnt0
# fsckptadm list /mnt0
/mnt0
latest:
  ctime                = Mon 26 Jul 12:06:42 2004
  mtime                = Mon 26 Jul 12:06:42 2004
  flags                = nodata, largefiles, delayed
old:
  ctime                = Mon 26 Jul 11:56:51 2004
  mtime                = Mon 26 Jul 11:56:51 2004
  flags                = largefiles
older:
  ctime                = Mon 26 Jul 11:56:46 2004
  mtime                = Mon 26 Jul 11:56:46 2004
  flags                = largefiles
oldest:
  ctime                = Mon 26 Jul 11:56:41 2004
  mtime                = Mon 26 Jul 11:56:41 2004
  flags                = nodata, largefiles
```

4 Remove the older and old Storage Checkpoints.

```
# fsckptadm remove older /mnt0
# fsckptadm remove old /mnt0
# fsckptadm list /mnt0
/mnt0
latest:
  ctime                = Mon 26 Jul 12:06:42 2004
  mtime                = Mon 26 Jul 12:06:42 2004
  flags                = nodata, largefiles
oldest:
  ctime                = Mon 26 Jul 11:56:41 2004
  mtime                = Mon 26 Jul 11:56:41 2004
  flags                = nodata, largefiles
```

Note: After you remove the older and old Storage Checkpoints, the latest Storage Checkpoint is automatically converted to a nodata Storage Checkpoint because the only remaining older Storage Checkpoint (oldest) is already a nodata Storage Checkpoint:

Space management considerations

Several operations, such as removing or overwriting a file, can fail when a file system containing Storage Checkpoints runs out of space. If the system cannot allocate sufficient space, the operation will fail.

Database applications usually preallocate storage for their files and may not expect a write operation to fail. If a file system runs out of space, the kernel automatically removes Storage Checkpoints and attempts to complete the write operation after sufficient space becomes available. The kernel removes Storage Checkpoints to prevent commands, such as rm, from failing under an out-of-space (ENOSPC) condition.

When the kernel automatically removes the Storage Checkpoints, it applies the following policies:

- Remove as few Storage Checkpoints as possible to complete the operation.
- Never select a non-removable Storage Checkpoint.
- Select a nodata Storage Checkpoint only when data Storage Checkpoints no longer exist.
- Remove the oldest Storage Checkpoint first.

Restoring from a Storage Checkpoint

Mountable data Storage Checkpoints on a consistent and undamaged file system can be used by backup and restore applications to restore either individual files or an entire file system. Restoration from Storage Checkpoints can also help recover incorrectly modified files, but typically cannot recover from hardware damage or other file system integrity problems.

Note: For hardware or other integrity problems, Storage Checkpoints must be supplemented by backups from other media.

Files can be restored by copying the entire file from a mounted Storage Checkpoint back to the primary fileset. To restore an entire file system, you can designate a mountable data Storage Checkpoint as the primary fileset using the `fsckpt_restore` command.

See the `fsckpt_restore`(1M) manual page.

When using the `fsckpt_restore` command to restore a file system from a Storage Checkpoint, all changes made to that file system after that Storage Checkpoint's creation date are permanently lost. The only Storage Checkpoints and data preserved are those that were created at the same time, or before, the selected Storage Checkpoint's creation. The file system cannot be mounted at the time that `fsckpt_restore` is invoked.

Note: Individual files can also be restored very efficiently by applications using the `fsckpt_fbmap`(3) library function to restore only modified portions of a files data.

Restoring a file from a Storage Checkpoint

The following example restores a file, `MyFile.txt`, which resides in your home directory, from the Storage Checkpoint `CKPT1` to the device `/dev/vx/dsk/dg1/vol-01`. The mount point for the device is `/home`.

To restore a file from a Storage Checkpoint

1 Create the Storage Checkpoint `CKPT1` of `/home`.

```
$ fckptadm create CKPT1 /home
```

2 Mount Storage Checkpoint `CKPT1` on the directory `/home/checkpoints/mar_4`.

```
$ mount -F vxfs -o ckpt=CKPT1 /dev/vx/dsk/dg1/vol- \
01:CKPT1 /home/checkpoints/mar_4
```

3 Delete the file `MyFile.txt` from your home directory.

```
$ cd /home/users/me
$ rm MyFile.txt
```

4 Go to the `/home/checkpoints/mar_4/users/me` directory, which contains the image of your home directory.

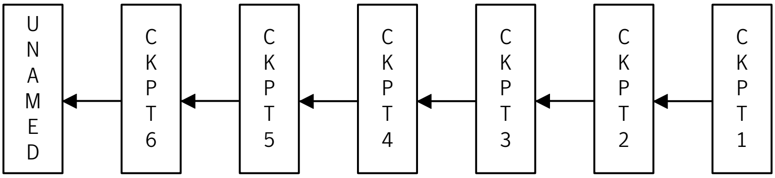
```
$ cd /home/checkpoints/mar_4/users/me
$ ls -l
-rw-r--r--   1  me   staff  14910   Mar 4   17:09   MyFile.txt
```

5 Copy the file `MyFile.txt` to your home directory.

```
$ cp MyFile.txt /home/users/me
$ cd /home/users/me
$ ls -l
-rw-r--r--   1 me   staff  14910   Mar 4  18:21   MyFile.txt
```

Restoring a file system from a Storage Checkpoint

The following example restores a file system from the Storage Checkpoint `CKPT3`. The filesets listed before the restoration show an unnamed root fileset and six Storage Checkpoints.



To restore a file system from a Storage Checkpoint

1 Run the `fsckpt_restore` command:

```
# fsckpt_restore -l /dev/vx/dsk/dg1/vol2
/dev/vx/dsk/dg1/vol2:
UNNAMED:
  ctime        = Thu 08 May 2004 06:28:26 PM PST
  mtime        = Thu 08 May 2004 06:28:26 PM PST
  flags        = largefiles, file system root
CKPT6:
   ctime        = Thu 08 May 2004 06:28:35 PM PST
  mtime        = Thu 08 May 2004 06:28:35 PM PST
  flags        = largefiles
CKPT5:
  ctime        = Thu 08 May 2004 06:28:34 PM PST
  mtime        = Thu 08 May 2004 06:28:34 PM PST
  flags        = largefiles, nomount
CKPT4:
  ctime        = Thu 08 May 2004 06:28:33 PM PST
  mtime        = Thu 08 May 2004 06:28:33 PM PST
  flags        = largefiles
CKPT3:
  ctime        = Thu 08 May 2004 06:28:36 PM PST
  mtime        = Thu 08 May 2004 06:28:36 PM PST
  flags        = largefiles
CKPT2:
  ctime        = Thu 08 May 2004 06:28:30 PM PST
  mtime        = Thu 08 May 2004 06:28:30 PM PST
  flags        = largefiles
CKPT1:
  ctime        = Thu 08 May 2004 06:28:29 PM PST
  mtime        = Thu 08 May 2004 06:28:29 PM PST
  flags        = nodata, largefiles
```

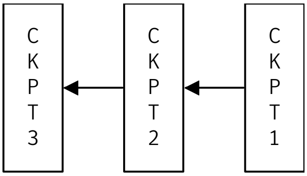
2 In this example, select the Storage Checkpoint `CKPT3` as the new root fileset:

```
Select Storage Checkpoint for restore operation
 or <Control/D> (EOF) to exit
 or <Return> to list Storage Checkpoints: CKPT3
CKPT3:
  ctime        = Thu 08 May 2004 06:28:31 PM PST
  mtime        = Thu 08 May 2004 06:28:36 PM PST
  flags        = largefiles
UX:vxfs fsckpt_restore: WARNING: V-3-24640: Any file system
changes or Storage Checkpoints made after
Thu 08  May 2004 06:28:31 PM PST will be lost.
```

3 Type **y** to restore the file system from CKPT3:

```
Restore the file system from Storage Checkpoint CKPT3 ?
(ynq) y
(Yes)
UX:vxfs fsckpt_restore: INFO: V-3-23760: File system
restored from CKPT3
```

If the filesets are listed at this point, it shows that the former UNNAMED root fileset and CKPT6, CKPT5, and CKPT4 were removed, and that CKPT3 is now the primary fileset. CKPT3 is now the fileset that will be mounted by default.



4 Run the fsckpt_restore command:

```
# fsckpt_restore -l /dev/vx/dsk/dg1/vol2
/dev/vx/dsk/dg1/vol2:
CKPT3:
  ctime        = Thu 08 May 2004 06:28:31 PM PST
  mtime        = Thu 08 May 2004 06:28:36 PM PST
  flags        = largefiles, file system root
CKPT2:
  ctime        = Thu 08 May 2004 06:28:30 PM PST
  mtime        = Thu 08 May 2004 06:28:30 PM PST
  flags        = largefiles
CKPT1:
  ctime        = Thu 08 May 2004 06:28:29 PM PST
  mtime        = Thu 08 May 2004 06:28:29 PM PST
  flags        = nodata, largefiles
Select Storage Checkpoint for restore operation
 or <Control/D> (EOF) to exit
 or <Return> to list Storage Checkpoints:
```

Storage Checkpoint quotas

VxFS provides options to the `fsckptadm` command interface to administer Storage Checkpoint quotas. Storage Checkpoint quotas set the following limits on the amount of space used by all Storage Checkpoints of a primary file set:

hard limit

An absolute limit that cannot be exceeded. If a hard limit is exceeded, all further allocations on any of the Storage Checkpoints fail, but existing Storage Checkpoints are preserved.

soft limit

Must be lower than the hard limit. If a soft limit is exceeded, no new Storage Checkpoints can be created. The number of blocks used must return below the soft limit before more Storage Checkpoints can be created. An alert and console message are generated.

In case of a hard limit violation, various solutions are possible, enacted by specifying or not specifying the `-f` option for the `fsckptadm` utility.

See the `fsckptadm`(1M) manual page.

Specifying or not specifying the `-f` option has the following effects:

- If the `-f` option is not specified, one or many removable Storage Checkpoints are deleted to make space for the operation to succeed. This is the default solution.
- If the `-f` option is specified, all further allocations on any of the Storage Checkpoints fail, but existing Storage Checkpoints are preserved.

Note: Sometimes if a file is removed while it is opened by another process, the removal process is deferred until the last close. Because the removal of a file may trigger pushing data to a “downstream” Storage Checkpoint (that is, the next older Storage Checkpoint), a fileset hard limit quota violation may occur. In this scenario, the hard limit is relaxed to prevent an inode from being marked bad. This is also true for some asynchronous inode operations.

Chapter 5

Using Cross-Platform Data Sharing

This chapter includes the following topics:

Overview of CDS

General concepts

This section presents an overview of the Cross-Platform Data Sharing (CDS) feature of Symantec's Veritas Storage Foundation™ software. CDS provides you with a foundation for moving data between different systems within a heterogeneous environment. The machines may be running HP-UX, AIX, Linux or the Solaris™ operating system (OS), and they may all have direct access to physical devices holding data. CDS allows Symantec's Veritas products and applications to access data storage independently of the operating system platform, enabling them to work transparently in heterogeneous environments.

The Cross-Platform Data Sharing feature is also known as Portable Data Containers (PDC). For consistency, this document uses the name Cross-Platform Data Sharing throughout.

The following levels in the device hierarchy, from disk through file system, must provide support for CDS to be used:

End-user applications	Application level.
Veritas™ File System (VxFS)	File system level.
Veritas™ Volume Manager (VxVM)	Volume level.
Operating system	Device level.

CDS is a license-enabled feature that is supported at the disk group level by VxVM and at the file system level by VxFS.

CDS utilizes a new disk type (`auto:cdsdisk`). To effect data sharing, VxVM supports a new disk group attribute (`cds`) and also supports different OS block sizes.

Note: CDS allows data volumes and their contents to be easily migrated between heterogeneous systems. It does not enable concurrent access from different types of platform unless such access is supported at all levels that are required.

Shared data across platforms

While volumes can be exported across platforms, the data on the volumes can be shared only if data sharing is supported at the application level. That is, to make data sharing across platforms possible, it must be supported throughout the entire software stack.

For example, if a VxFS file system on a VxVM volume contains files comprising a database, then the following functionality applies:

- Disks can be recognized (as `cds` disks) across platforms.
- Disk groups can be imported across platforms.
- The file system can be mounted on different platforms.

However, it is very likely that, because of the inherent characteristics of databases, you may not be able to start up and use the database on a platform different from the one on which it was created. (A notable exception is Oracle 10g's Cross-Platform Transportable Tablespace feature.)

An example is where an executable file, compiled on one platform, can be accessed across platforms (using CDS), but may not be executable on a different platform.

Note: You do not need a file system in the stack if the operating system provides access to raw disks and volumes, and the application can utilize them. Databases and other applications can have their data components built on top of raw volumes without having a file system to store their data files.

Disk drive sector size

Sector size is an attribute of a disk drive (or SCSI LUN for an array-type device), which is set when the drive is formatted. Sectors are the smallest addressable unit of storage on the drive, and are the units in which the device performs I/O. The sector size is significant because it defines the atomic I/O size at the device level. Any multi-sector writes which VxVM submits to the device driver are not guaranteed to be atomic (by the SCSI subsystem) in the case of system failure.

Block size issues

The block size is a platform-dependent value that is greater than or equal to the sector size. Each platform accesses the disk on block boundaries and in quantities that are multiples of the block size.

Data that is created on one platform, and then accessed by a platform of a different block size, can suffer from the following problems:

Addressing issues

- The data may not have been created on a block boundary compatible with that used by the accessing platform.
- The accessing platform cannot address the start of the data.

Bleed-over issues

The size of the data written may not be an exact multiple of the block size used by the accessing platform. Therefore the accessing platform cannot constrain its I/O within the boundaries of the data on disk.

Operating system data

Some operating systems (OS) require OS-specific data on disks in order to recognize and control access to the disk.

CDS disk access and format

For a disk to be accessible by multiple platforms, the disk must be consistently recognized by the platforms, and all platforms must be capable of performing I/O on the disk. CDS disks contain specific content at specific locations to identify or control access to the disk on different platforms. The same content and location are used on all CDS disks, independent of the platform on which the disks are initialized.

In order for a disk to be initialized as, or converted to a CDS disk, it must satisfy the following requirements:

- Must be a SCSI disk
- Disk size cannot exceed 1 TB
- Cannot be an EFI disk
- Must be the entire physical disk (LUN)
- Only one volume manager (such as VxVM) can manage a physical disk (LUN)
- There can be no disk partition (slice) which is defined, but which is not configured on the disk
- Cannot contain a volume whose use-type is either `root` or `swap` (for example, it cannot be a boot disk)

The CDS conversion utility, `vxcdsconvert`, is provided to convert non-CDS VM disk formats to CDS disks, and disk groups with a version number less than 110 to disk groups that support CDS disks.

See "Converting non-CDS disks to CDS disks" on page 116.

Note: Disk groups with version numbers less than 110 are not supported for the Solaris OS on the x64 platform.

CDS disk types

The CDS disk format, `cdsdisk`, is recognized by all VxVM platforms (including Windows). The `cdsdisk` disk format is the default for all newly-created VM disks unless overridden in a defaults file. The `vxcdsconvert` utility is provided to convert other disk formats and types to CDS.

See "Defaults files" on page 120.

Note: Disks with format `cdsdisk` can only be added to disk groups with version 110 or later.

Private and public regions

A VM disk usually has a private and a public region.

The private region is a small area on the disk where VxVM configuration information is stored, such as a disk header label, configuration records for VxVM objects (such as volumes, plexes and subdisks), and an intent log for the configuration database. The default private region size is 32MB, which is large enough to record the details of several thousand VxVM objects in a disk group.

The public region covers the remainder of the disk, and is used for the allocation of storage space to subdisks.

The private and public regions are aligned and sized in multiples of 8K to permit the operation of CDS. The alignment of VxVM objects within the public region is controlled by the disk group alignment attribute. The value of the disk group alignment attribute must also be 8k to permit the operation of CDS.

Note: With other (non-CDS) VxVM disk formats, the private and public regions are aligned to the platform-specific OS block size.

Disk access type auto

The disk access (DA) disk type auto supports multiple disk formats, including cdsdisk, which is supported across all platforms. It is associated with the DA records created by the VxVM auto-configuration mode. Disk type auto automatically determines which format is on the disk.

Platform block

The platform block resides on disk sector 0, and contains data specific to the operating system for the platforms. It is necessary for proper interaction with each of those platforms. The platform block allows a disk to perform as if it was initialized by each of the specific platforms.

AIX coexistence label

The AIX coexistence label resides on the disk, and identifies the disk to the AIX logical volume manager (LVM) as being controlled by VxVM.

HP-UX coexistence label

The HP-UX coexistence label resides on the disk, and identifies the disk to the HP logical volume manager (LVM) as being controlled by VxVM.

VxVM ID block

The VxVM ID block resides on the disk, and indicates the disk is under VxVM control. It provides dynamic VxVM private region location and other information.

CDS disk groups

A CDS disk group allows cross-platform data sharing of VxVM objects, so that data written on one of the supported platforms may be accessed on any other supported platform. A CDS disk group is composed only of CDS disks (VM disks with the disk format `cdsdisk`), and is only available for disk group version 110 and greater.

Note: The CDS conversion utility, `vxcdsconvert`, is provided to convert non-CDS VM disk formats to CDS disks, and disk groups with a version number less than 110 to disk groups that support CDS disks.

See “Converting non-CDS disks to CDS disks” on page 116.

All VxVM objects in a CDS disk group are aligned and sized so that any system can access the object using its own representation of an I/O block. The CDS disk group uses a platform-independent alignment value to support system block sizes of up to 8K.

See “Disk group alignment” on page 111.

CDS disk groups can be used in the following ways:

- Initialized on one system and then used “as-is” by VxVM on a system employing a different type of platform.
- Imported (in a serial fashion) by Linux, Solaris, AIX, and HP-UX systems.
- Imported as private disk groups, or shared disk groups (by CVM).

You cannot include the following disks or volumes in a CDS disk group:

- Volumes of usage type `root` and `swap`. You cannot use CDS to share boot devices.
- Encapsulated disks.

Note: On Solaris and Linux systems, the process of disk encapsulation places the slices or partitions on a disk (which may contain data or file systems) under VxVM control. On AIX and HP-UX systems, LVM volumes may similarly be converted to VxVM volumes.

Device quotas

Device quotas limit the number of objects in the disk group which create associated device nodes in the file system. Device quotas are useful for disk groups which to be transferred between Linux with a pre-2.6 kernel and other supported platforms. Prior to the 2.6 kernel, Linux supported only 256 minor devices per major device.

You can limit the number of devices that can be created in a given CDS disk group by setting the device quota.

See "Setting the maximum number of devices for CDS disk groups" on page 127.

When you create a device, an error is returned if the number of devices would exceed the device quota. You then either need to increase the quota, or remove some objects using device numbers, before the device can be created.

See "Displaying the maximum number of devices in a CDS disk group" on page 131.

Minor device numbers

Importing a disk group will fail if it will exceed the maximum devices for that platform.

Note: There is a large disparity between the maximum number of devices allowed for devices on the Linux platform with a pre-2.6 kernel, and that for other supported platforms.

Non-CDS disk groups

Any version 110 (or greater) disk group (DG) can contain both CDS and non-CDS disks. However, only version 110 (or greater) disk groups composed entirely of CDS disks have the ability to be shared across platforms. Whether or not that ability has been enabled is a controlled by the `cds` attribute of the disk group. Enabling this attribute causes a non-CDS disk group to become a CDS disk group.

Although a non-CDS disk group can contain a mixture of CDS and non-CDS disks having dissimilar private region alignment characteristics, its disk group alignment will still direct how all subdisks are created.

Disk group alignment

One of the attributes of the disk group is the block alignment, which represents the largest block size supported by the disk group.

The alignment constrains the following attributes of the objects within a disk group:

- Subdisk offset
- Subdisk length
- Plex offset
- Volume length
- Log length

- Stripe width

The offset value specifies how an object is positioned on a drive.

The disk group alignment is assigned at disk group creation time.

See "Disk group tasks" on page 123.

Alignment values

The disk group block alignment has two values: 1 block or 8k (8 kilobytes).

All CDS disk groups must have an alignment value of 8k.

All disk group versions before version 110 have an alignment value of 1 block, and they retain this value if they are upgraded to version 110 or later.

A disk group that is not a CDS disk group, and which has a version of 110 and later, can have an alignment value of either 1 block or 8k.

The alignment for all newly initialized disk groups in VxVM 4.0 and later releases is 8k. This value, which is used when creating the disk group, cannot be changed. However, the disk group alignment can be subsequently changed.

See "Changing the alignment of a non-CDS disk group" on page 124.

Note: The default usage of `vxassist` is to set the `layout=diskalign` attribute on all platforms. The `layout` attribute is ignored on 8K-aligned disk groups, which means that scripts relying on the default may fail.

Dirty region log alignment

The location and size of each map within a dirty region log (DRL) must not violate the disk group alignment for the disk group (containing the volume to which the DRL is associated). This means that the region size and alignment of each DRL map must be a multiple of the disk group alignment, which for CDS disk groups is 8K. (Features utilizing the region size can impose additional minimums and size increments over and above this restriction, but cannot violate it.)

In a version 110 disk group, a traditional DRL volume has the following region requirements:

- Minimum region size of 512K
- Incremental region size of 64K

In a version 110 disk group, a version 20 DCO volume has the following region requirements:

- Minimum region size of 16K

- Incremental region size of 8K

Note: The map layout within a Data Change Object (DCO) volume changed with the release of VxVM 4.0 to version 20. This can accommodate both FastResync and DRL maps within the DCO volume. The original version 0 layout for DCO volumes only accommodates FastResync maps.

Object alignment during volume creation

For CDS disk groups, VxVM objects that are used in volume creation are automatically aligned to 8K. For non-CDS disk groups, the `vxassist` attribute, `dgalign_checking`, controls how the command handles attributes that are subject to disk group alignment restrictions. If set to `strict`, the volume length and values of attributes must be integer multiples of the disk group alignment value, or the command fails and an error message is displayed. If set to `round` (default), attribute values are rounded up as required. If this attribute is not specified on the command-line or in a defaults file, the default value of `round` is used.

The `diskalign` and `nodiskalign` attributes of `vxassist`, which control whether subdisks are aligned on cylinder boundaries, is honored only for non-CDS disk groups whose alignment value is set to 1.

Setting up your system

Creating CDS disks from uninitialized disks

You can create a CDS disk from an uninitialized disk by using one of the following methods:

- Creating CDS disks by using vxdisksetup
- Creating CDS disks by using vxdiskadm

Creating CDS disks by using vxdisksetup

To create a CDS disk by using the vxdisksetup command

- Type the following command:

```
# vxdisksetup -i disk [format=disk_format]
```

  The format defaults to `cdsdisk` unless this is overridden by the `/etc/default/vxdisk` file, or by specifying the disk format as an argument to the `format` attribute.

See "Defaults files" on page 120.

See the vxdisksetup(1M) manual page.

Creating CDS disks by using vxdiskadm

To create a CDS disk by using the vxdiskadm command

- Run the `vxdiskadm` command, and select the "`Add or initialize one or more disks`" item from the main menu. You are prompted to specify the format.

Warning: On CDS disks, the CDS information occupies the first sector of that disk, and there is no `fdisk` partition information. Attempting to create an `fdisk` partition (for example, by using the `fdisk` or `format` commands) erases the CDS information, and can cause data corruption.

Creating CDS disks from initialized VxVM disks

How you create a CDS disk depends on the current state of the disk, as follows:

- Creating a CDS disk from a disk that is not in a disk group
- Creating a CDS disk from a disk that is already in a disk group

Creating a CDS disk from a disk that is not in a disk group

To create a CDS disk from a disk that is not in a disk group

1 Run the following command to remove the VM disk format for the disk:

```
# vxdiskunsetup disk
```

This is necessary as non-auto types cannot be reinitialized by `vxdisksetup`.

2 If the disk is listed in the `/etc/vx/darecs` file, remove its disk access (DA) record using the command:

```
# vxdisk rm disk
```

(Disk access records that cannot be configured by scanning the disks are stored in an ordinary file, `/etc/vx/darecs`, in the root file system. Refer to the `vxintro`(1M) manual page for more information.)

3 Rescan for the disk using this command:

```
# vxdisk scandisks
```

4 Type this command to set up the disk:

```
# vxdisksetup -i disk
```

Creating a CDS disk from a disk that is already in a disk group

To create a CDS disk from a disk that is already in a disk group

- Run the `vxcdsconvert` command.
  See "Converting non-CDS disks to CDS disks" on page 116.

Creating CDS disk groups

You can create a CDS disk group in the following ways:

- Creating a CDS disk group by using vxdg init
- Creating a CDS disk group by using vxdiskadm

Creating a CDS disk group by using vxdg init

Note: The disk group version must be 110 or greater.

To create a CDS disk group by using the vxdg init command

- Type the following command:

  ```
  # vxdg init diskgroup disklist [cds={on|off}]
  ```

  The format defaults to a CDS disk group, unless this is overridden by the `/etc/default/vxdg` file, or by specifying the `cds` argument.
  See the `vxdg`(1M) manual page for more information.

Creating a CDS disk group by using vxdiskadm

You cannot create a CDS disk group when encapsulating an existing disk, or when converting an LVM volume.

When initializing a disk, if the target disk group is an existing CDS disk group, `vxdiskadm` will only allow the disk to be initialized as a CDS disk. If the target disk group is a non-CDS disk group, the disk can be initialized as either a CDS disk or a non-CDS disk.

If you use the vxdiskadm command to initialize a disk into an existing CDS disk group, the disk must have be added with the cdsdisk format.

The CDS attribute for the disk group remains unchanged by this procedure.

To create a CDS disk group by using the vxdiskadm command

- Run the vxdiskadm command, and select the “Add or initialize one or more disks” item from the main menu. Specify that the disk group should be a CDS disk group when prompted.

Converting non-CDS disks to CDS disks

Note: The disks must be of type of auto in order to be re-initialized as CDS disks.

To convert non-CDS disks to CDS disks

1 If the conversion is not going to performed on-line (that is, while access to the disk group continues), stop any applications that are accessing the disks.

2 Type one of the following forms of the CDS conversion utility (vxcdsconvert) to convert non-CDS disks to CDS disks.

```
# vxcdsconvert -g diskgroup [-A] [-d defaults_file] \
  [-o novolstop] disk name [attribute=value] ...
# vxcdsconvert -g diskgroup [-A] [-d defaults_file] \
  [-o novolstop] alldisks [attribute=value] ...
```

The alldisks and disk keywords have the following effect

alldisks — Converts all non-CDS disks in the disk group into CDS disks.

disk — Specifies a single disk for conversion. You would use this option under the following circumstances:

- If a disk in the non-CDS disk group has cross-platform exposure, you may want other VxVM nodes to recognize the disk, but not to assume that it is available for initialization.
- If the native Logical Volume Manager (LVM) that is provided by the operating system needs to recognize CDS disks, but it is not required to initialize or manage these disks.
- Your intention is to move the disk into an existing CDS disk group.

The conversion involves evacuating objects from the disk, reinitializing the disk, and relocating objects back to disk. You can specify the -o novolstop option to perform the conversion on-line (that is, while access to the disk continues). If the `-o novolstop` option is not specified, stop any applications that are accessing the disks, and perform the conversion off-line.

Warning: Specifying the `-o novolstop` option can greatly increase the amount of time that is required to perform conversion.

Before you use the `vxcdsconvert` command, make sure you understand its options, attributes, and keywords.

See the `vxcdsconvert`(1M) manual page.

Converting a non-CDS disk group to a CDS disk group

To convert a non-CDS disk group to a CDS disk group

1. If the disk group contains one or more disks that you do not want to convert to CDS disks, use the `vxdg move` or `vxdg split` command to move the disks out of the disk group.
2. The disk group to be converted must have the following characteristics:
   - No dissociated or disabled objects.
   - No sparse plexes.
   - No volumes requiring recovery.
   - No volumes with pending snapshot operations.
   - No objects in an error state.

   To verify whether a non-CDS disk group can be converted to a CDS disk group, type the following command:

   ```
   # vxcdsconvert -g diskgroup -A group
   ```

3. If the disk group does not have a CDS-compatible disk group alignment, the objects in the disk group must be relayed out with a CDS-compatible alignment.
4. If the conversion is not going to performed on-line (that is, while access to the disk group continues), stop any applications that are accessing the disks.
5. Type one of the following forms of the CDS conversion utility (`vxcdsconvert`) to convert a non-CDS disk group to a CDS disk group.

   ```
   # vxcdsconvert -g diskgroup [-A] [-d defaults_file] \
     [-o novolstop] alignment [attribute=value] ...
   ```

```
# vxcdsconvert -g diskgroup [-A] [-d defaults_file] \
  [-o novolstop] group [attribute=value] ...
```

The `alignment` and `group` keywords have the following effect:

`alignment` Specifies alignment conversion where disks are not converted, and an object relayout is performed on the disk group. A successful completion results in an 8K-aligned disk group. You might consider this option, rather than converting the entire disk group, if you want to reduce the amount of work to be done for a later full conversion to CDS disk group.

`group` Specifies group conversion of all non-CDS disks in the disk group before realying out objects in the disk group.

The conversion involves evacuating objects from the disk, reinitializing the disk, and relocating objects back to disk. You can specify the `-o novolstop` option to perform the conversion on-line (that is, while access to the disk group continues). If the `-o novolstop` option is not specified, stop any applications that are accessing the disks, and perform the conversion off-line.

Warning: Specifying the `-o novolstop` option can greatly increase the amount of time that is required to perform conversion.

Conversion has the following side effects:

- Non-CDS disk group are upgraded by using the `vxdg upgrade` command. If the disk group was originally created by the conversion of an LVM volume group (VG), rolling back to the original LVM VG is not possible. If you decide to go through with the conversion, the rollback records for the disk group will be removed, so that an accidental rollback to an LVM VG cannot be done.
- Stopped, but startable volumes, are started for the duration of the conversion .
- Any volumes or other objects in the disk group that were created with the `layout=diskalign` attribute specified can no longer be disk aligned.
- Encapsulated disks may lose the ability to be unencapsulated.
- Performance may be degraded because data may have migrated to different regions of a disk, or to different disks.

In the following example, the disk group, `mydg`, and all its disks are converted to CDS while keeping its volumes are still online:

```
# vxcdsconvert -g mydg -o novolstop group \
  move_subdisks_ok=yes evac_subdisks_ok=yes \
  evac_disk_list=disk11,disk12,disk13,disk14
```

The evac_disk_list attrinute specifies a list of disks (`disk11` through `disk14`) to which subdisks can be evacuated to disks if required.

Before you use the `vxcdsconvert` command, make sure you understand its options, attributes, and keywords.

See the `vxcdsconvert`(1M) manual page.

Side effects of conversion

Conversion has the following side effects:

- Non-CDS disk group are upgraded by using the `vxdg upgrade` command. If the disk group was originally created by the conversion of an LVM volume group (VG), rolling back to the original LVM VG is not possible. If you decide to go through with the conversion, the rollback records for the disk group will be removed, so that an accidental rollback to an LVM VG cannot be done.
- Stopped, but startable, volumes are started for the duration of the conversion.
- Any volumes or other objects in the disk group that were created with the `layout=diskalign` attribute specified can no longer be disk aligned.
- Encapsulated disks may lose the ability to be unencapsulated.
- Performance may be degraded because data may have migrated to different regions of a disk, or to different disks.

Verifying licensing

The ability to create or import a CDS disk group is controlled by a CDS license. CDS licenses are included as part of the Veritas Storage Foundation license.

To verify the CDS enabling license

- Type the following command:

  ```
  # vxlicrep
  ```

  Verify the following line in the output:

  ```
  Cross-platform Data Sharing = Enabled
  ```

Defaults files

The following system defaults files in the `/etc/default` directory are used to specify the alignment of VxVM objects, the initialization or encapsulation of VM disks, the conversion of LVM disks, and the conversion of disk groups and their disks to the CDS-compatible format

`vxassist`

Specifies default values for the following parameters to the `vxcdsconvert` command that have an effect on the alignment of VxVM objects: `dgalign_checking`, `diskalign`, and `nodiskalign`.

See “Object alignment during volume creation” on page 113.

See the `vxassist`(1M) manual page.

`vxcdsconvert`

Specifies default values for the following parameters to the `vxcdsconvert` command: `evac_disk_list`, `evac_subdisks_ok`, `min_split_size`, `move_subdisks_ok`, `privlen`, and `split_subdisks_ok`.

The following is a sample `vxcdsconvert` defaults file:

```
evac_subdisks_ok=no
min_split_size=64k
move_subdisks_ok=yes
privlen=2048
split_subdisks_ok=move
```

An alternate defaults file can be specified by using the `-d` option with the `vxcdsconvert` command.

See the `vxcdsconvert`(1M) manual page.

`vxdg`

Specifies default values for the `cds`, `default_activation_mode` and `enable_activation` parameters to the `vxdg` command. The `default_activation_mode` and `enable_activation` parameters are only used with shared disk groups in a cluster.

The following is a sample `vxdg` defaults file:

```
cds=on
```

See the `vxdg`(1M) manual page.

`vxdisk`

Specifies default values for the `format` and `privlen` parameters to the `vxdisk` and `vxdisksetup` commands. These commands are used when disks are initialized by VxVM for the first time.They are also called implicitly by the `vxdiskadm` command and the Storage Foundation Manager (SFM) GUI.

The following is a sample `vxdisk` defaults file:

```
format=cdsdisk
privlen=2048
```

See the `vxdisk`(1M) manual page.

See the `vxdisksetup`(1M) manual page.

`vxencap`

Specifies default values for the `format, privlen, privoffset` and `puboffset` parameters to the `vxencap` and `vxlvmencap` commands. These commands are used when disks with existing partitions or slices are encapsulated, or when LVM disks are converted to VM disks. It is also called implicitly by the `vxdiskadm`, `vxconvert` (on AIX) and `vxvmconvert` (on HP-UX) commands, and by the SFM.

The following is a sample `vxencap` defaults file:

```
format=sliced
privlen=4096
privoffset=0
puboffset=1
```

See the `vxencap`(1M) manual page.

See the `vxconvert`(1M) manual page.

See the `vxvmconvert`(1M) manual page.

In the defaults files, a line that is empty, or that begins with a “#” character in the first column, is treated as a comment, and is ignored.

Apart from comment lines, all other lines must define attributes and their values using the format `attribute=value`. Each line starts in the first column, and is terminated by the value. No white space is allowed around the = sign.

Maintaining your system

Disk tasks

The following disk tasks are supported:

- Changing the default disk format
- Restoring CDS disk labels

Changing the default disk format

When disks are put under VxVM control, they are formatted with the default `cdsdisk` layout. This happens during the following operations:

- Initialization of disks
- Encapsulation of disks with existing partitions or slices (Linux and Solaris systems)
- Conversion of LVM disks (AIX, HP-UX and Linux systems)

You can override this behavior by changing the settings in the system defaults files. For example, you can change the default format to `sliced` for disk initialization by modifying the definition of the `format` attribute in the `/etc/default/vxdisk` defaults file.

To change the default format for disk encapsulation or LVM disk conversion

- Edit the `/etc/default/vxencap` defaults file, and change the definition of the `format` attribute.
  See "Defaults files" on page 120.

Restoring CDS disk labels

CDS disks have the following labels:

- Platform block
- AIX coexistence label
- HP-UX coexistence or VxVM ID block

There are also backup copies of each. If any of the primary labels become corrupted, VxVM will not bring the disk online and user intervention is required.

If two labels are intact, the disk is still recognized as a `cdsdisk` (though in the error state) and `vxdisk flush` can be used to restore the CDS disk labels from their backup copies.

Primary labels are at sectors 0, 7, and 16; and a normal flush will not flush sectors 7 and 16. Also, the private area is not updated as the disk is not in a disk group. There is no means of finding a "good" private region to flush from. In this case, it is possible to restore the CDS disk labels from the existing backups on disk using the flush operation.

If a corruption happened after the labels were read and the disk is still online and part of a disk group, then a flush operation will also flush the private region.

Warning: Caution and knowledge must be employed because the damage could involve more than the CDS disk labels. If the damage is constrained to the first 128K, the disk flush would fix it. This could happen if another system on the fabric wrote a disk label to a disk that was actually a CDS disk in some disk group.

To rewrite the CDS ID information on a specific disk

- Type the following command:

  ```
  # vxdisk flush disk_access_name
  ```

  This rewrites all labels except sectors 7 and 16.

To rewrite all the disks in a CDS disk group

- Type the following command:

  ```
  # vxdg flush diskgroup
  ```

  This rewrites all labels except sectors 7 and 16.

To forcibly rewrite the AIX coexistence label in sector 7 and the HP-UX coexistence label or VxVM ID block in sector 16

- Type the following command:

  ```
  # vxdisk -f flush disk_access_name
  ```

  This command rewrites all labels if there exists a valid VxVM ID block that points to a valid private region. The `-f` option is required to rewrite sectors 7 and 16 when a disk is taken offline due to label corruption (possibly by a Windows system on the same fabric).

Disk group tasks

The following disk group tasks are supported:

- Changing the alignment of a disk group during disk encapsulation
- Changing the alignment of a non-CDS disk group
- Determining the setting of the CDS attribute on a disk group
- Splitting a CDS disk group
- Moving objects between CDS disk groups and non-CDS disk groups
- Moving objects between CDS disk groups

- Joining disk groups
- Changing the default CDS setting for disk group creation
- Creating non-CDS disk groups
- Upgrading an older version non-CDS disk group
- Replacing a disk in a CDS disk group
- Setting the maximum number of devices for CDS disk groups

Changing the alignment of a disk group during disk encapsulation

If you use the `vxdiskadm` command to encapsulate a disk into a disk group with an alignment of 8K, the disk group alignment must be reduced to 1.

If you use the `vxencap` command to perform the encapsulation, the alignment is carried out automatically without a confirmation prompt.

To change the alignment of a disk group during disk encapsulation

- Run the `vxdiskadm` command, and select the “`Add or initialize one or more disks`” item from the main menu. As part of the encapsulation process, you are asked to confirm that a reduction of the disk group alignment from 8K to 1 is acceptable.

Changing the alignment of a non-CDS disk group

The alignment value can only be changed for disk groups with version 110 or greater.

For a CDS disk group, `alignment` can only take a value of `8k`. Attempts to set the alignment of a CDS disk group to 1 fail unless you first change it to a non-CDS disk group.

Increasing the alignment may require `vxcdsconvert` to be run to change the layout of the objects in the disk group.

To display the current alignment value of a disk group, use the `vxprint` command.

See “Displaying the disk group alignment” on page 131.

To change the alignment value of a disk group

- Type the `vxdg set` command:

```
# vxdg -g diskgroup set align={1|8k}
```

The operation to increase the alignment to 8K fails if objects exist in the disk group that do not conform to the new alignment restrictions. In that case, use the `vxcdsconvert alignment` command to change the layout of the objects:

```
# vxcdsconvert -g diskgroup [-A] [-d defaults_file] \
  [-o novolstop] alignment [attribute=value] ...
```

This command increases the alignment value of a disk group and its objects to 8K, without converting the disks.

The sequence 8K to 1 to 8K is possible only using `vxdg set` as long as the configuration does not change after the 8K to 1 transition.

See "Converting a non-CDS disk group to a CDS disk group" on page 117.

Splitting a CDS disk group

You can use the `vxdg split` command to create a CDS disk group from an existing CDS disk group. The new (target) disk group preserves the setting of the CDS attribuet and alignment in the original (source) disk group.

To split a CDS disk group

- Use the `vxdg split` command to split CDS disk groups.
  See the *Veritas Volume Manager Adminstrator's Guide*.

Moving objects between CDS disk groups and non-CDS disk groups

The alignment of a source non-CDS disk group must be 8K to allow objects to be moved to a target CDS disk group. If objects are moved from a CDS disk group to a target non-CDS disk group with an alignment of 1, the alignment of the target disk group remains unchanged.

To move objects between a CDS disk group and a non-CDS disk group

- Use the `vxdg move` command to move objects between a CDS disk group and a non-CDS disk groups.
  See the *Veritas Volume Manager Adminstrator's Guide*.

Moving objects between CDS disk groups

The disk group alignment does not change as a result of moving objects between CDS disk groups.

To move objects between CDS disk groups

- Use the `vxdg move` command to move objects between CDS disk groups.
  See the *Veritas Volume Manager Adminstrator's Guide*.

Joining disk groups

Joining two CDS disk groups or joining two non-CDS disk groups is permitted, but you cannot join a CDS disk group to a non-CDS disk group. If two non-CDS disk groups have different alignment values, the alignment of the resulting joined disk group is set to 1, and an informational message is displayed.

To join two disk groups

- Use the `vxdg join` command to join two disk groups.
  See the *Veritas Volume Manager Adminstrator's Guide*.

Changing the default CDS setting for disk group creation

To change the default CDS setting for disk group creation

- Edit the `/etc/default/vxdg` file, and change the setting for the cds attribute.

Creating non-CDS disk groups

A disk group with a version lower than 110 is given an alignment value equal to 1 when it is imported. This is because the `dg_align` value is not stored in the configuration database for such disk groups.

To create a non-CDS disk group with a version lower than 110

- Type the following `vxdg` command:

```
# vxdg -T version init diskgroup disk_name=disk_access_name
```

Upgrading an older version non-CDS disk group

You may want to upgrade a non-CDS disk group with a version lower than 110 in order to use new features othe rthan CDS. After upgrading the disk group, the `cds` attribute is set to `off`, and the disk group has an alignment of 1.

Note: You must also perform a disk group conversion (using the `vxcdsconvert` utility) to use the CDS feature.

To upgrade the non-CDS pre-version 110 disk group

- Type the following `vxdg` command:

```
# vxdg upgrade diskgroup
```

Replacing a disk in a CDS disk group

Note: When replacing a disk in a CDS disk group, you cannot use a non-CDS disk as the replacement.

To replace a disk in a CDS disk group

- Type the following commands:

  ```
  # vxdg -g diskgroup -k rmdisk disk_name
  # vxdg -g diskgroup -k adddisk disk_name=disk_access_name
  ```

  The `-k` option retains and reuses the disk media record for the disk that is being replaced. The following example shows a disk device `disk21` being reassigned to disk `mydg01`.

  ```
  # vxdg -g diskgroup -k rmdisk mydg01
  # vxdg -g diskgroup -k adddisk mydg01=disk21
  ```

  For other operating systems, use the appropriate device name format.

Setting the maximum number of devices for CDS disk groups

To set the maximum number of devices that can be created in a CDS disk group

- Type the following `vxdg set` command:

  ```
  # vxdg -g diskgroup set maxdev=max-devices
  ```

  The `maxdev` attribute can take any positive integer value that is greater than the number of devices that are currently in the disk group.

Changing the DRL map and log size

If DRL is enabled on a newly-created volume without specifying a log or map size, default values are used. You can use the command line attributes `logmap_len` and `loglen` in conjunction with the `vxassist, vxvol`, and `vxmake` commands to set the DRL map and DRL log sizes. The attributes can be used independently, or they can be combined.

You can change the DRL map size and DRL log size only when the volume is disabled and the DRL maps are not in use. Changes can be made to the DRL map size only for volumes in a CDS disk group.

The `logmap_len` attribute specifies the required size, in bytes, for the DRL log. It cannot be greater than the number of bytes available in the map on the disk.

To change the DRL map and log size

- Use the following commands to remove and rebuild the logs:

```
# vxassist -g diskgroup remove log volume nlog=0
# vxassist -g diskgroup addlog volume nlog=nlogs \
  logtype=drl logmap_len=len-bytes [loglen=len-blocks]
```

Note the following restrictions

If only `logmap_len` is specified	The DRL log size is set to the default value (33 * disk group alignment).
If `logmap_len` is greater than `(DRL log size)/2`	The command fails, and you need to either provide a sufficiently large `loglen` value or reduce `logmap_len`.
For CDS disk groups	The DRL map and log sizes are set to a minimum of 2 * (disk group alignment).

Creating a volume with a DRL log

To create a volume with a traditional DRL log by using the vxassist command

- Type the following command:

```
# vxassist -g diskgroup make volume length mirror=2 \
  logtype=drl [loglen=len-blocks] [logmap_len=len-bytes]
```

This command creates log subdisks that are each equal to the size of the DRL log.
Note the following restrictions

If neither `logmap_len` nor `loglen` is specified	■ `loglen` is set to a default value that is based on disk group alignment. ■ `maplen` is set to a reasonable value.
If only `loglen` is specified	■ For pre-version 110 disk groups, `maplen` is set to zero. ■ For version 110 and greater disk groups, `maplen` is set to use all the bytes available in the on-disk map.

If only logmap_len is specified	■ For pre-version 110 disk groups, logmap_len is not applicable. ■ For version 110 and greater disk groups, maplen must be less than the number of available bytes in the on-disk map for the default log length.

Setting the DRL map length

To set a DRL map length

1 Stop the volume to make the DRL inactive.

2 Type the following command:

```
# vxvol -g diskgroup set [loglen=len-blocks] \
  [logmap_len=len-bytes] volume
```

This command does not change the existing DRL map size.

Note the following restrictions

If both logmap_len and loglen are specified	■ if logmap_len is greater than loglen/2, vxvol fails with an error message. Either increase loglen to a sufficiently large value, or decrease logmap_len to a sufficiently small value. ■ The value of logmap_len cannot exceed the number of bytes in the on-disk map.
If logmap_len is specified	■ The value is constrained by size of the log, and cannot exceed the size of the on-disk map.Thesize of the on-disk map in blocks can be calculated from the following formula: round(loglen/nmaps) - 24 where nmaps is 2 for a private disk group, or 33 for a shared disk group. ■ The value of logmap_len cannot exceed the number of bytes in the on-disk map.

If `loglen` is specified	■ Specifying a value that is less than twice the disk group alignment value results in an error message. ■ The value is constrained by size of the logging subdisk.

Displaying information

This section describes the following tasks:

- Determining the setting of the CDS attribute on a disk group
- Displaying the maximum number of devices in a CDS disk group
- Displaying map length and map alignment of traditional DRL logs
- Displaying the disk group alignment
- Displaying the log map length and alignment
- Displaying offset and length information in units of 512 bytes

Determining the setting of the CDS attribute on a disk group

To determine the setting of the CDS attribute on a disk group

- Use the `vxdg list` command or the `vxprint` command to determine the setting of the CDS attribute, as shown in the following examples:

```
# vxdg list

NAME            STATE           ID
dgTestSol2      enabled,cds     1063238039.206.vmesc1

# vxdg list dgTestSol2

Group:      dgTestSol2
dgid:       1063238039.206.vmesc1
import-id:  1024.205
flags:      cds
version:    110
alignment:  8192 (bytes)
  .
  .
  .
```

```
# vxprint -F %cds -G -g dgTestSol2
```

```
on
```

The disk group, `dgTestSol2`, is shown as having the CDS flag set.

Displaying the maximum number of devices in a CDS disk group

To display the maximum number of devices in a CDS disk group

- Type the following command:

```
# vxprint -g diskgroup -G -F %maxdev
```

Displaying map length and map alignment of traditional DRL logs

To display the map length and map alignment of traditional DRL logs

- Type the following commands

```
# vxprint -g diskgroup -vl volume
# vxprint -g diskgroup -vF '%name %logmap_len %logmap_align' \
  volume
```

Displaying the disk group alignment

To display the disk group alignment

- Type the following command:

```
# vxprint -g diskgroup -G -F %align
```

Utilities such as `vxprint` and `vxdg list` that print information about disk group records also output the disk group alignment.

Displaying the log map length and alignment

To display the log map length and alignment

- Type the following command:

```
# vxprint -g diskgroup -lv volume
```

For example, to print information for the volume `vol1` in disk group `dg1`:

```
# vxprint -g dg1 -lv vol1
```

The output is of the form:

```
logging: type=REGION loglen=0 serial=0/0 mapalign=0
maplen=0 (disabled)
```

This indicates a log map alignment (`logmap_align`) value of 0, and a log map length (`logmap_len`) value of 0.

If the log map is set and enabled, the command and results may be in the following form:

```
# vxprint -lv drlvol

Disk group: dgTestSol
Volume:   drlvol
info:     len=20480
type:     usetype=fsgen
state:    state=ACTIVE kernel=ENABLED cdsrecovery=0/0 (clean)
assoc:    plexes=drlvol-01,drlvol-02,drlvol-03
policies: read=SELECT (round-robin) exceptions=GEN_DET_SPARSE
flags:    closed writecopy writeback
logging:  type=REGION loglen=528 serial=0/0 mapalign=16
maplen=512 (enabled)
apprecov: seqno=0/0
recovery: mode=default
recov_id=0
device: minor=46000 bdev=212/46000 cdev=212/46000
path=/dev/vx/dsk/dgTestSol/drlvol
perms:   user=root group=root mode=0600
guid:  {d968de3e-1dd1-11b2-8fc1-080020d223e5}
```

Displaying offset and length information in units of 512 bytes

To display offset and length information in units of 512 bytes

- Specify the `-b` option to the `vxprint` and `vxdisk` commands, as shown in these examples:

```
# vxprint -bm
# vxdisk -b list
```

Specifying the `-b` option enables consistent output to be obtained on different platforms. Without the `-b` option, the information is output in units of sectors. The number of bytes per sector differs between platforms.

When the `vxprint -bm` or `vxdisk -b` list command is used, the output also contains the `b` suffix, so that the output can be fed back to `vxmake`.

Default activation mode of shared disk groups

The default activation mode of shared disk groups involves a local in-kernel policy that differs between platforms. This means that, regardless of the platform on which the disk group was created, the importing platform will have platform-specific behavior with respect to activation of shared disk groups. Specifically, with the exception of HP-UX, importing a shared disk group results in the volumes being active and enabled for shared-write. In the case of HP-UX, the shared volumes will be inactive and require other actions to activate them for shared-write operations.

Additional considerations when importing CDS disk groups

Before you attempt to use CDS to move disk groups between different operating systems, and if the configuration of the disks has changed since the target system was last rebooted, you should consider the following points

Does the target system know about the disks?	For example, the disks may not have been connected to the system either physically (not cabled) or logically (using FC zoning or LUN masking) when the system was booted up, but they have subsequently been connected without rebooting the system. This can happen when bringing new storage on-line, or when adding an additional DMP path to existing storage. On the target system, both the operating system and VxVM must be informed of the existence of the new storage. Issue the appropriate command to tell the operating system to look for the storage. (On Linux, depending on the supported capabilities of the host adapter, you may need to reboot the target system to achieve this.) Having done this, run either of the following commands on the target system to have VxVM recognize the storage: # **vxdctl enable** # **vxdisk scandisks**

Do the disks contain partitions or slices?

Both the Solaris and Linux operating systems maintain information about partitions or slices on disks. If you repartition a disk after the target system was booted, use the appropriate command to instruct the operating system to rescan the disk’s TOC or partition table. For example, on a target Linux system, use the following command:

```
# blockdev --rereadpt
```

Having done this, run either of the following commands on the target system to have VxVM recognize the storage:

```
# vxdctl enable
# vxdisk scandisks
```

Has the format of any of the disks changed since the target system was last booted?

For example, if you use the `vxdisksetup -i` command to format a disk for VxVM on one system, the `vxdisk list` command on the target system may still show the format as being auto:none. If so, use either of the following commands on the target system to instruct VxVM to rescan the format of the disks:

```
# vxdctl enable
# vxdisk scandisks
```

File system considerations

Considerations about data in the file system

Data within a file system might not be in the appropriate format to be accessed if moved between different types of systems. For example, files stored in proprietary binary formats often require conversion for use on the target platform. Files containing databases might not be in a standard format that allows their access when moving a file system between various systems, even if those systems use the same byte order. Oracle 10g's Cross-Platform Transportable Tablespace is a notable exception; if used, this feature provides a consistent format across many platforms.

Some data is inherently portable, such as plain ASCII files. Other data is designed to be portable and the applications that access such data are able to access it irrespective of the system on which it was created, such as Adobe PDF files.

Note that the CDS facilities do not convert the end user data. The data is uninterpreted by the file system. Only individual applications have knowledge of the data formats, and thus those applications and end users must deal with this issue. This issue is not CDS-specific, but is true whenever data is moved between different types of systems.

Even though a user might have a file system with data that cannot be readily interpreted or manipulated on a different type of system, there still are reasons for moving such data by using CDS mechanisms. For example, if the desire is to bring a file system off line from its primary use location for purposes of backing it up without placing that load on the server or because the system on which it will be backed up is the one that has the tape devices directly attached to it, then using CDS to move the file system is appropriate.

An example is a principal file server that has various file systems being served by it over the network. If a second file server system with a different operating system was purchased to reduce the load on the original server, CDS can migrate the file system instead of having to move the data to different physical storage over the network, even if the data could not be interpreted or used by either the original or new file server. This is a scenario that often occurs when the data is only accessible or understood by software running on PCs and the file server is UNIX or Linux-based.

File system migration

File system migration refers to the system management operations related to stopping access to a file system, and then restarting these operations to access the file system from a different computer system. File system migration might be required to be done once, such as when permanently migrating a file system to another system without any future desire to move the file system back to its original system or to other systems. This type of file system migration is referred to as one-time file system migration. When ongoing file system migration between multiple systems is desired, this is known as ongoing file system migration. Different actions are required depending on the kind of migration, as described in the following sections.

Specifying the migration target

Most of the operations performed by the CDS commands require the target to which the file system is to be migrated to be specified by target specifiers in the following format:

```
os_name=name[,os_rel=release][,arch=arch_name]
[,vxfs_version=version][,bits=nbits]
```

The CDS commands require the following target specifiers:

`os_name=`*`name`*
Specifies the name of the target operating system to which the file system is planned to be migrated. Possible values are `HP-UX, AIX, SunOS`, or `Linux`. The os_name field must be specified if the target is specified.

`os_rel=`*`release`*
Specifies the operating system release version of the target. For example, `11.31`.

`arch=`*`arch_name`*
Specifies the architecture of the target. For example, specify `ia` or `pa` for HP-UX.

`vxfs_version=`*`version`*
Specifies the VxFS release version that is in use at the target. For example, 5.1.

`bits=`*`nbits`*
Specifies the kernel bits of the target. *nbits* can have a value of `32` or `64` to indicate whether the target is running a 32-bit kernel or 64-bit kernel.

While `os_name` must be specified for all `fscdsadm` invocations that permit the target to be specified, all other target specifiers are optional and are available for the user to fine tune the migration target specification.

The CDS commands use the limits information available in the default CDS limits file, `/etc/vx/cdslimitstab`. If the values for the optional target specifiers are not specified, `fscdsadm` will choose the defaults for the specified target based on the information available in the limits file that best fits the specified target, and proceed with the CDS operation. The chosen defaults are displayed to the user before proceeding with the migration.

Note: The default CDS limits information file, `/etc/vx/cdslimitstab`, is installed as part of the VxFS package. The contents of this file are used by the VxFS CDS commands and should not be altered.

Examples of target specifications

The following are examples of target specifications:

`os_name=AIX`
Specifies the target operating system and use defaults for the remainder.

```
os_name=HP-UX,
os_rel=11.23,
arch=ia,
vxfs_version=5.0,
bits=64
```

Specifies the operating system, operating system release version, architecture, VxFS version, and kernel bits of the target.

`os_name=SunOS, arch=sparc`	Specifies the operating system and architecture of the target.
`os_name=Linux, bits=32`	Specifies the operating system and kernel bits of the target.

Using the fscdsadm command

The `fscdsadm` command can be used to perform the following CDS tasks:

- Checking that the metadata limits are not exceeded
- Maintaining the list of target operating systems
- Enforcing the established CDS limits on a file system
- Ignoring the established CDS limits on a file system
- Validating the operating system targets for a file system
- Displaying the CDS status of a file system

Checking that the metadata limits are not exceeded

To check that the metadata limits are not exceeded

- Type the following command to check whether there are any file system entities with metadata that exceed the limits for the specified target operating system:

```
# fscdsadm -v -t target mount_point
```

Maintaining the list of target operating systems

When a file system will be migrated on an ongoing basis between multiple systems, the types of operating systems that are involved in these migrations are maintained in a `target_list` file. Knowing what these targets are allows VxFS to determine file system limits that are appropriate to all of these targets. The file system limits that are enforced are file size, user ID, and group ID. The contents of the `target_list` file are manipulated by using the `fscdsadm` command.

Adding an entry to the list of target operating systems

To add an entry to the list of target operating systems

- Type the following command:

```
# fscdsadm -o add -t target mount_point
```

See “Specifying the migration target” on page 135.

Removing an entry from the list of target operating systems

To remove an entry from the list of target operating systems

- Type the following command:

```
# fscdsadm -o remove -t target mount_point
```

See “Specifying the migration target” on page 135.

Removing all entries from the list of target operating systems

To remove all entries from the list of target operating systems

- Type the following command:

```
# fscdsadm -o none mount_point
```

Displaying the list of target operating systems

To display a list of all target operating systems

- Type the following command:

```
# fscdsadm -o list mount_point
```

Enforcing the established CDS limits on a file system

By default, CDS ignores the limits that are implied by the operating system targets that are listed in the `target_list` file.

To enforce the established CDS limits on a file system

- Type the following command:

```
# fscdsadm -l enforce mount_point
```

Ignoring the established CDS limits on a file system

By default, CDS ignores the limits that are implied by the operating system targets that are listed in the `target_list` file.

To ignore the established CDS limits on a file system

- Type the following command:

```
# fscdsadm -l ignore mount_point
```

Validating the operating system targets for a file system

To validate the operating system targets for a file system

- Type the following command:

  ```
  # fscdsadm -v mount_point
  ```

Displaying the CDS status of a file system

The CDS status that is maintained for a file system includes the following information:

- the `target_list` file
- the limits implied by the `target_list` file
- whether the limits are being enforced or ignored
- whether all files are within the CDS limits for all operating system targets that are listed in the `target_list` file

To display the CDS status of a file system

- Type the following command:

  ```
  # fscdsadm -s mount_point
  ```

Migrating a file system one time

This example describes a one-time migration of data between two operating systems. Some of the following steps require a backup of the file system to be created. To simplify the process, you can create one backup before performing any of the steps instead of creating multiple backups as you go.

To perform a one-time migration

1. If the underlying Volume Manager storage is not contained in a CDS disk group, it must first be upgraded to be a CDS disk group, and all other physical considerations related to migrating the storage physically between systems must first be addressed.

   See "Converting a non-CDS disk group to a CDS disk group" on page 117.

2. If the file system is using a disk layout version prior to 7, upgrade the file system to Version 7.

   See the *Veritas Storage Foundation Installation Guide*.

3 Use the following command to ensure that there are no files in the file system that will be inaccessible after migrating the data due to large file size or to differences in user or group ID between platforms:

```
# fscdsadm -v -t target mount_point
```

If such files exist, move the files to another file system or reduce the size of the files.

4 Unmount the file system:

```
# umount mount_point
```

5 Use the `fscdsconv` command to convert the file system to the opposite endian.

See "Converting the byte order of a file system" on page 142.

6 Make the physical storage and Volume Manager logical storage accessible on the Linux system by exporting the disk group from the source system and importing the disk group on the target system after resolving any other physical storage attachment issues.

See "Disk tasks" on page 121.

7 Mount the file system on the target system.

Migrating a file system on an ongoing basis

This example describes how to migrate a file system between platforms on an ongoing basis. Some of the following steps require a backup of the file system to be created. To simplify the process, you can create one backup before performing any of the steps instead of creating multiple backups as you go.

To perform an ongoing migration

1 Use the following command to ensure that there are no files in the file system that will be inaccessible after migrating the data due to large file size or to differences in user or group ID between platforms:

```
# fscdsadm -v -t target mount_point
```

If such files exist, move the files to another file system or reduce the size of the files.

2 Add the platform on the `target_list` file:

- If migrating a file system between the Solaris and Linux, add `SunOS` and `Linux` to the `target_list` file:

```
# fscdsadm -o add -t os_name=SunOS /mnt1
# fscdsadm -o add -t os_name=Linux /mnt1
```

- If migrating a file system between the HP-UX and Linux, add `HP-UX` and `Linux` to the `target_list` file:

  ```
  # fscdsadm -o add -t os_name=HP-UX /mnt1
  # fscdsadm -o add -t os_name=Linux /mnt1
  ```

3. Enforce the limits:

   ```
   # fscdsadm -l enforce mount_point
   ```

   This is the last of the preparation steps. When the file system is to be migrated, it must be unmounted, and then the storage moved and mounted on the target system.

4. Unmount the file system:

   ```
   # umount mount_point
   ```

5. Make the file system suitable for use on the specified target.

   See "Converting the byte order of a file system" on page 142.

6. Make the physical storage and Volume Manager logical storage accessible on the target system by exporting the disk group from the source system and importing the disk group on the target system after resolving any other physical storage attachment issues.

   See "Disk tasks" on page 121.

7. Mount the file system on the target system.

Stopping ongoing migration

To stop performing ongoing migration

- ◆ Type the following commands:

  ```
  # fscdsadm -l ignore mount_point
  # fscdsadm -o none mount_point
  ```

  The file system is left on the current system.

When to convert a file system

When moving a file system between two systems, it is essential to run the `fscdsconv` command to perform all of the file system migration tasks. The `fscdsconv` command validates the file system to ensure that it does not exceed any of the established CDS limits on the target, and converts the byte order of the file system if the byte order of the target is opposite to that of the current system.

Warning: Prior to VxFS 4.0 and disk layout Version 6, VxFS did not officially support moving file systems between different platforms, although in many cases a user may have successfully done so. Do not move file systems between platforms when using versions of VxFS prior to Version 4, or when using disk layouts earlier than Version 6. Instead, upgrade to VxFS 4.0 or higher, and disk layout Version 6 or later. Failure to upgrade before performing cross-platform movement can result in data loss or data corruption.

Converting the byte order of a file system

Use the `fscdsconv` command to migrate a file system from one system to another.

To convert the byte order of a file system

1 Determine the disk layout version of the file system that you will migrate:

```
# fstyp -v /dev/vx/rdsk/diskgroup/volume | grep version

magic a501fcf5 version 7 ctime Thu Jun 1 16:16:53 2006
```

Only file systems with Version 6 or later disk layout can be converted. If the file system has an earlier disk layout version, convert the file system to Version 6 or Version 7 disk layout before proceeding.

See the `vxfsconvert`(1M) manual page.

See the `vxupgrade`(1M) manual page.

2 Perform a full file system back up. Failure to do so could result in data loss or data corruption under some failure scenarios in which restoring from the backup is required.

3. Designate a file system with free space where `fscdsconv` may create a file that will contain recovery information for usage in the event of a failed conversion.

   Depending on the nature of the file system to be converted, for example if it is mirrored, you may wish to designate the recovery file to reside in a file system with the same level of failure tolerance. Having the same level of failure tolerance reduces the number of failure scenarios that would require trestoration from the backup.

4. Unmount the file system to be converted:

   ```
   # umount mount_point
   ```

5 Use the fscdsconv command to export the file system to the required target:

```
# fscdsconv -f recovery_file -t target -e special_device
```

target specifies the system to which you are migrating the file system.

See “Specifying the migration target” on page 135.

recovery_file is the name of the recovery file to be created by the fscdsconv command. special_device is the raw device or volume that contains the file system to be converted.

Include the file system that you chose in 3 when designating the recovery file.

For example, if the file system chosen to contain the recovery file is mounted on /data/fs3, the recovery file could be specified as /data/fs3/jan04recovery. If there is not enough disk space on the chosen file system for the recovery file to be created, the conversion aborts and the file system to be converted is left intact.

The recovery file is not only used for recovery purposes after a failure, but is also used to perform the conversion. The directory that will contain the recovery file should not allow non-system administrator users to remove or replace the file, as this could lead to data loss or security breaches. The file should be located in a directory that is not subject to system or local scripts will remove the file after a system reboot, such as that which occurs with the /tmp and /var/tmp directories on the Solaris operating system.

The recovery file is almost always a sparse file. The disk utilization of this file can best be determined by using the following command:

```
# du -sk filename
```

The recovery file is used only when the byte order of the file system must be converted to suit the specified migration target.

6 If you are converting multiple file systems at the same time, which requires the use of one recovery file per file system, record the names of the recovery files and their corresponding file systems being converted in the event that recovery from failures is required at a later time.

7 Based on the information provided regarding the migration target, fscdsconv constructs and displays the complete migration target and prompts the use to verify all details of the target. If the migration target must be changed, enter n to exit fscdsconv without modifying the file system. At this point in the process, fscdsconv has not used the specified recovery file.

8 If the byte order of the file system must be converted to migrate the file system to the specified target, fscdsconv prompts you to confirm the migration. Enter y to convert the byte order of the file system. If the byte order does not need to be converted, a message displays indicating this fact.

9 The fscdsconv command indicates if any files are violating the maximum file size, maximum UID, or maximum GID limits on the specified target and prompts you if it should continue. If you must take corrective action to ensure that no files violate the limits on the migration target, enter n to exit fscdsconv. At this point in the process, fscdsconv has not used the specified recovery file.

If the migration converted the byte order of the file system, fscdsconv created a recovery file. The recovery file is not removed after the migration completes, and can be used to restore the file system to its original state if required at a later time.

10 If a failure occurs during the conversion, the failure could be one of the following cases:

- System failure.
- fscdsconv failure due to program defect or abnormal termination resulting from user actions.

In such cases, the file system being converted is no longer in a state in which it can be mounted or accessed by normal means through other VxFS utilities. To recover the file system, invoke the fscdsconv command with the recovery flag, -r:

```
# fscdsconv -r -f recovery_file special_device
```

When the -r flag is specified, fscdsconv expects the recovery file to exist and that the file system being converted is the same file system specified in this second invocation of fscdsconv.

11 After invoking fscdsconv with the -r flag, the conversion process will restart and complete, given no subsequent failures.

In the event of another failure, repeat 10.

Under some circumstances, you will be required to restore the file system from the backup, such as if the disk fails that contains the recovery file. Failure to have created a backup would then result in total data loss in the file system. I/O errors on the device that holds the file system would also require a backup to be restored after the physical device problems are addressed. There may be other causes of failure that would require the use of the backup.

Importing and mounting a file system from another system

The fscdsconv command can be used to import and mount a file system that was previously used on another system.

To import and mount a file system from another system

- Convert the file system:

```
# fscdsconv -f recovery_file -i special_device
```

If the byte order of the file system needs to be converted	Enter y to convert the byte order of the file system when prompted by fscdsconv. If the migration converted the byte order of the file system, fscdsconv creates a recovery file that persists after the migration completes. If required, you can use this file to restore the file system to its original state at a later time.
If the byte order of the file system does not need to be converted	A message displays that the byte order of the file system does not need to be converted.

Alignment value and block size

On the AIX, Linux and Solaris operating systems, an alignment value of 1 is equivalent to a block size of 512 bytes. On the HP-UX operating system, it is equivalent to a block size of 1024 bytes.

The block size on HP-UX is different from that on other supported platforms. Output from commands such as vxdisk and vxprint looks different on HP-UX for the same disk group if the -b option is not specified.

Default activation mode of shared disk groups

The default activation mode of shared disk groups is a local in-kernel policy that differs between platforms. Regardless of the platform on which the disk group was created, the importing platform will have platform-specific behavior with respect to activation of shared disk groups. Specifically, with the exception of HP-UX, importing a shared disk group will result in the volumes being active and enabled for shared-write. In the case of HP-UX, the shared volumes will be inactive and require other actions to activate them for shared-write operations.

Disk group alignment and encapsulated disks

On the Solaris OS, all native file systems are cylinder aligned. Encapsulating such a disk results in subdisks that are also cylinder aligned. Such alignment will normally not be 8K aligned, but it will be 1K aligned. For the encapsulation process, there is no flexibility as to where on the disk the subdisks must be since the data location is predefined. If an alignment conflict occurs, user intervention is required. If the disk group alignment is 8K this operation will probably fail because this would require the cylinder to be an even number of 8K blocks in size.

Disk group import between Linux and non-Linux machines

A disk group created on a non-Linux system typically has device numbers greater than 1000. When that disk group is imported on a Linux machine with a pre-2.6 kernel, the devices are reassigned minor numbers less than 256.

If a disk group on a Linux system is imported to a non-Linux system, all device numbers will be less than 256. If those devices are available (that is, they do not conflict with devices in an imported boot disk group) they will be used. Otherwise new device numbers will be reassigned.

A single disk group could contain a number of devices exceeding the maximum number of devices for a given platform. In this case, the disk group cannot be imported on that platform because import would exhaust available minor devices for the VxVM driver. Although the case of minor number exhaustion is possible in a homogeneous environment, it will be more pronounced between platforms with different values for the maximum number of devices supported, such as Linux with a pre-2.6 kernel. This difference will render platforms with low maximum devices supported values less useful as heterogeneous disk group failover or recovery candidates.

Note: Using the disk group `maxdev` attribute may reduce the likelihood that a CDS disk group import on Linux with a per-2.6 kernel will exceed the maximum number of devices.

Migrating a snapshot volume

This example demonstrates how to migrate a snapshot volume containing a VxFS file system from a Solaris SPARC system (big endian) to a Linux system (little endian) or HP-UX system (big endian) to a Linux system (little endian).

To migrate a snapshot volume

1. Create the instant snapshot volume, `snapvol`, from an existing plex in the volume, `vol`, in the CDS disk group, `datadg`:

   ```
   # vxsnap -g datadg make source=vol/newvol=snapvol/nmirror=1
   ```

2. Quiesce any applications that are accessing the volume. For example, suspend updates to the volume that contains the database tables. The database may have a hot backup mode that allows you to do this by temporarily suspending writes to its tables.

3. Refresh the plexes of the snapshot volume using the following command:

   ```
   # vxsnap -g datadg refresh snapvol source=yes syncing=yes
   ```

4. The applications can now be unquiesced.

   If you temporarily suspended updates to the volume by a database in 2, release all the tables from hot backup mode.

5. Use the `vxsnap syncwait` command to wait for the synchronization to complete:

   ```
   # vxsnap -g datadg syncwait snapvol
   ```

6. Check the integrity of the file system, and then mount it on a suitable mount point:

   ```
   # fsck -F vxfs /dev/vx/rdsk/datadg/snapvol
   # mount -F vxfs /dev/vx/dsk/datadg/snapvol /mnt
   ```

7. Confirm whether the file system can be converted to the target operating system:

   ```
   # fscdstask validate Linux /mnt
   ```

8. Unmount the snapshot:

   ```
   # umount /mnt
   ```

9. Convert the file system to the opposite endian:

   ```
   # fscdsconv -f /tmp/fs_recov/recov.file /dev/vx/dsk/datadg/snapvol
   ```

   This step is only required if the source and target systems have the opposite endian configuration.

10 Split the snapshot volume into a new disk group, `migdg`, and deport that disk group:

```
# vxdg split datadg migdg snapvol
# vxdg deport migdg
```

11 Import the disk group, `migdg`, on the Linux system:

```
# vxdg import migdg
```

It may be necessary to reboot the Linux system so that it can detect the disks.

12 Use the following commands to recover and restart the snapshot volume:

```
# vxrecover -g migdg -m snapvol
```

13 Check the integrity of the file system, and then mount it on a suitable mount point:

```
# fsck -t vxfs /dev/vx/dsk/migdg/snapvol
# mount -t vxfs /dev/vx/dsk/migdg/snapvol /mnt
```

Chapter 6

Migrating from Solaris Volume Manager to VxVM

This chapter includes the following topics:

About migration from Solaris Volume Manager

Migrating from Solaris Volume Manager to VxVM is not supported on Solaris 10 x64.

Veritas Volume Manager (VxVM) provides utilities that you can use to convert objects, file systems, and device partitions controlled by the Sun Microsystems Solaris Volume Manager software (also known as Solstice Disk Suite™) software to VxVM control. The tools convert all Solaris Volume Manager objects into VxVM objects and encapsulate file systems and swap device partitions used by Solaris Volume Manager software. The conversion happens in place. You do not need to copy data into temporary storage. After the conversion, the Solaris Volume Manager software is disabled and can be removed.

The conversion utilities have the following characteristics:

- The utilities are provided in the `VRTSvxvm` package.
- All objects under Solaris Volume Manager control are converted. The conversion commands have no partial conversion option.
- You must use the recommended root disk conversion procedure if you want to move only a root disk from under Solaris Volume Manager control to VxVM.
  See "Converting a root disk" on page 167.

  Warning: You cannot use the conversion utilities in a mixed environment. Even if you move only the root disk from Solaris Volume Manager software to VxVM, you cannot later use the conversion utilities to move other Solaris Volume Manager objects to VxVM.

- The conversion cannot be done in a mixed environment, which is a system that has both Solaris Volume Manager metadevices and VxVM objects (disk groups, un-used plexes, etc).
- All file systems that are on a Solaris Volume Manager metadevice and that are listed in `/etc/vfstab` are put under VxVM control.
- A conversion retains the on-disk layouts of Solaris Volume Manager volumes and does not improve layouts as supported by VxVM.
- The conversion utilities do not allow a conversion to be reversed. Once the conversion process starts, the best way to return to the Solaris Volume Manager configuration is to restore it from backups.
- After the conversion process starts on RAID-5 volumes, the data in these volumes is reorganized. Returning RAID-5 volumes to Solaris Volume Manager control can be done only by restoring the Solaris Volume Manager configuration and the volume data from backups.
- If you are very familiar with Solaris Volume Manager software and VxVM, you may be able to reverse a conversion without performing a full Solaris Volume Manager restoration from backups. However, RAID-5 volumes must always be restored from backups because the conversion utilities always reorganize RAID-5 parity. To manually reverse a conversion, preserve all Solaris Volume Manager configuration information that is required for its restoration. This includes output from `metastat -p` and `metadb` for every diskset, and `prtvtoc` output for all disks.

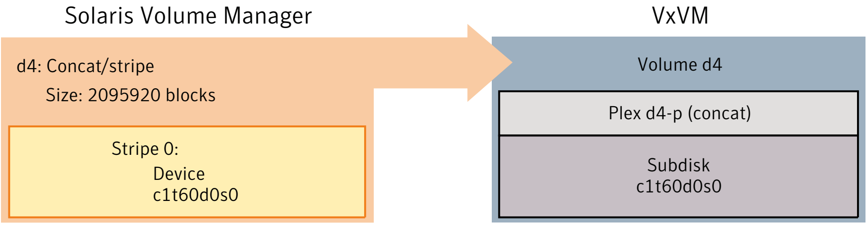
- Any partition that is on a disk with Solaris Volume Manager metadevices and that is not listed as a file system in `/etc/vfstab` is lost during the conversion. The partition is considered free space when the disk is converted to VxVM control. Even if the partitions are in use, such as for Oracle raw tablespace data, they are not encapsulated during the conversion. The raw partitions are lost when the disk layout changes. Either encapsulate the partition under Solaris Volume Manager control and allow the conversion to convert the resulting metadevice, or back up the partition, plan to create a new volume for it, and restore the partition after the conversion.

How Solaris Volume Manager objects are mapped to VxVM objects

The following illustrations give examples of how the conversion process maps Solaris Volume Manager concat/stripe objects to VxVM objects. RAID-5 volumes and mirrors come through the conversion intact.

Figure 6-1 shows a single Solaris Volume Manager partition encapsulated in a concat/stripe Solaris Volume Manager object.

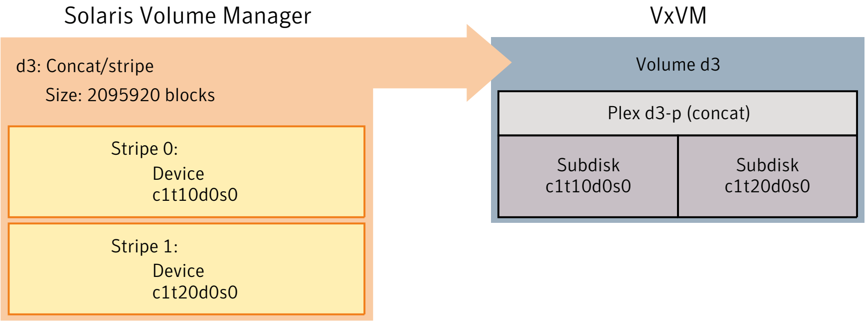
Figure 6-1 Single Solaris Volume Manager partition encapsulated in a concat/stripe Solaris Volume Manager object



The partition becomes a simple volume under VxVM.

Figure 6-2 shows a number of one-partition Solaris Volume Manager stripes in a concat/stripe Solaris Volume Manager object.

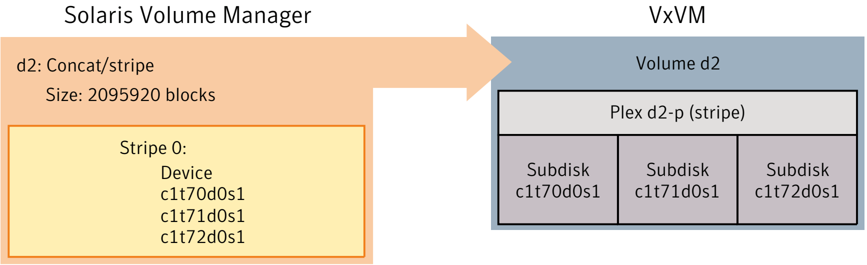
Figure 6-2 A number of one-partition Solaris Volume Manager stripes in a concat/stripe Solaris Volume Manager object



The stripes become a concat plex under VxVM.

Figure 6-3 shows a single Solaris Volume Manager stripe with an arbitrary number of partitions in a concat/stripe Solaris Volume Manager object.

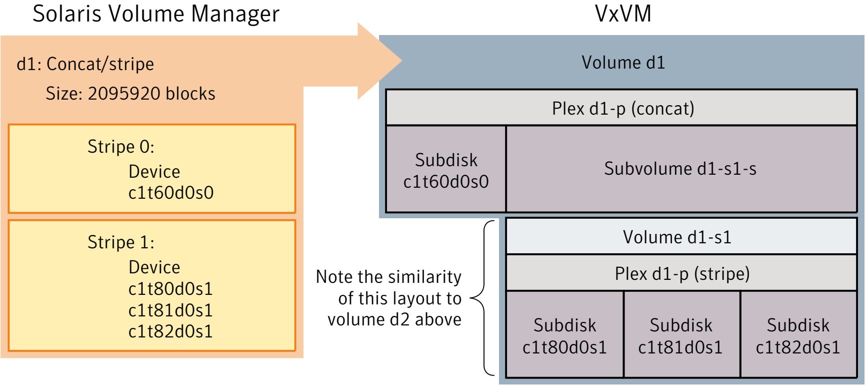
Figure 6-3 A single Solaris Volume Manager stripe with any number of partitions in a concat/stripe Solaris Volume Manager object



The partitions become a striped plex under VxVM.

Figure 6-4 shows a Solaris Volume Manager concat/stripe object with a mix of stripes.

Figure 6-4 A Solaris Volume Manager concat/stripe object with a mix of stripes



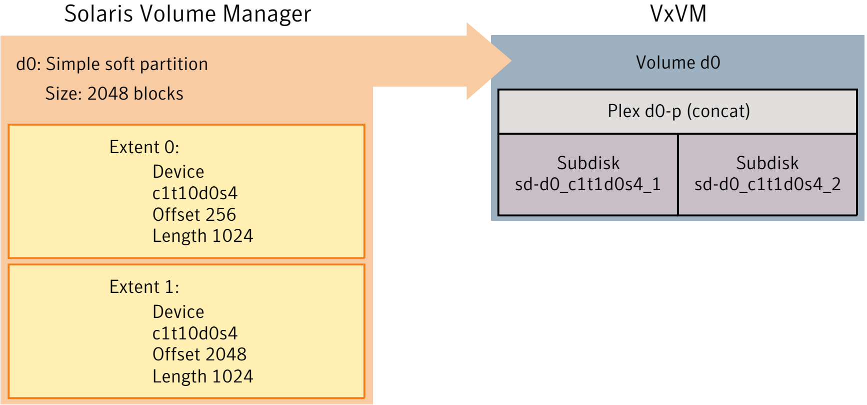
Under VxVM, the stripes are concatenated and may require a layered volume.

Conversion of soft partitions

The provided utilities support the conversion of the following soft partition-based layouts:

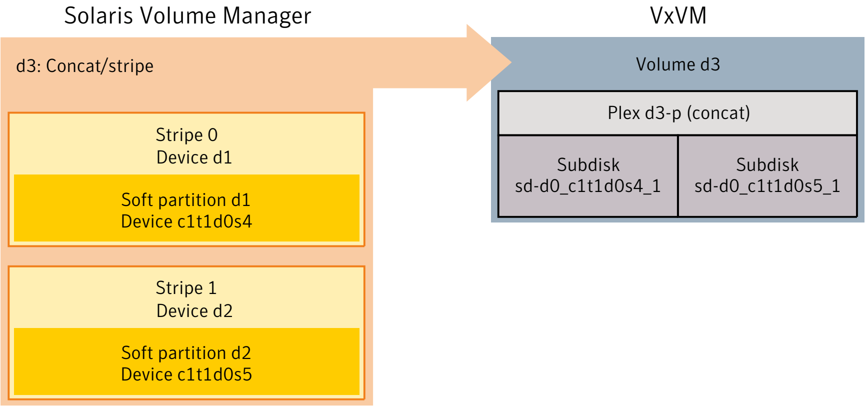
- A simple soft partition on a disk slice with one or more extents.
  Figure 6-5 shows the conversion of such a layout.

Figure 6-5 Simple soft partition on a disk slice with two extents



- Concat/Stripe on a soft partition on a disk slice with a single extent.
  Figure 6-6 shows the conversion of such a layout.

Figure 6-6 Concat/stripe Solaris Volume Manager object on a soft partition on a disk slice with a single extent



- RAID-5 on a soft partition on a disk slice with a single extent.
- Concat/Stripe on a soft partition on a disk slice with multiple extents.

The utilities do not support the conversion of the following soft partition-based layouts:

- RAID-5 on soft partitions on a disk with multiple extents.
- Soft partition on a logical volume.

More information about soft partitions may be found in the Solaris documentation.

See “Check metadevices” on page 160.

Overview of the conversion process

The conversion utilities allow you to do most of the preparation before starting the conversion, and before disrupting your system. To take advantage of this feature, read this entire manual and be familiar with the required tasks before starting the conversion from the Solaris Volume Manager configuration to VxVM.

The conversion utilities perform the following tasks:

- Planning and preparing for conversion.
  See “Plan and prepare for the conversion” on page 157.
- Set up the conversion.
  See “Set up the conversion” on page 157.
- Convert the volumes.
  See “Set up the conversion” on page 157.

- Perform post conversion cleanup tasks.
  See "Perform the conversion" on page 158.

Note: For a successful conversion, follow the steps in the given order.

Plan and prepare for the conversion

The tasks in this category can be done well before starting the actual conversion.

Plan the process and collect data for the conversion. The conversion may require changes to some Solaris Volume Manager metadevices.

See "Check metadevices" on page 160.

See "Planning the conversion" on page 158.

See "Preparing a Solaris Volume Manager configuration for conversion" on page 161.

Install the basic Veritas Volume Manager (without creating disk groups or objects and without initializing disks). The conversion utilities complete the VxVM setup with objects converted from Solaris Volume Manager software.

Set up the conversion

The tasks in this category should be done or updated just before starting the conversion. Information from the tasks in this category must be current when the conversion starts.

Use the `preconvert` utility to analyze the Solaris Volume Manager configuration and build a description for the VxVM conversion. You can run `preconvert` multiple times as it makes no changes to the Solaris Volume Manager configuration or file systems.

Run the `showconvert` utility to interpret `preconvert` output into a readable format. The `showconvert` output flags where changes must be made in the Solaris Volume Manager configuration for the conversion process to succeed. After making changes, run `preconvert` and `showconvert` again before starting the conversion.

Make backups of everything needed to restore the Solaris Volume Manager configuration.

See "Setting up a Solaris Volume Manager configuration for conversion" on page 162.

Perform the conversion

The first task in this category starts the conversion from Solaris Volume Manager software to VxVM. You have no easy way to return to a Solaris Volume Manager configuration once the conversion starts.

Run the `doconvert` utility to make the conversion. `Doconvert` relies on information from the `preconvert` utility.

Reboot the system. The reboot activates system configuration changes.

Change volume references in applications that use the volumes. VxVM uses volume names that are different than the names used by Solaris Volume Manager software. Every application that uses the Solaris Volume Manager volumes must be updated to use the new names.

See "Converting from the Solaris Volume Manager software to VxVM" on page 165.

Perform post-conversion tasks

After the `doconvert` utility performs a reboot of the system, the converted volumes and file systems are under the control of Veritas Volume Manager.

Use VxVM tools such as `vxassist` or the graphical user interface to improve the layout of volumes.

Remove or re-initialize the Solaris Volume Manager software for use on non-VxVM disks. After the conversion, the Solaris Volume Manager software is disabled and does not control any data. You can either remove the Solaris Volume Manager software or initialize the Solaris Volume Manager software with disks that are not used by VxVM. Both VxVM and the Solaris Volume Manager software can run on the same system.

See "Post conversion tasks" on page 166.

Planning the conversion

Before using any of the conversion utilities, you should consider how to schedule the steps in the process.

Scheduling considerations

The following preparation tasks can be done at any time before the conversion:

- Schedule downtime.
- Check and reconfigure Solaris Volume Manager metadevices.

- Identify references from applications to Solaris Volume Manager volumes.
- Install VxVM.

The setup tasks should be done just before starting the conversion. If you make changes to the Solaris Volume Manager configuration after doing the setup tasks, perform the following setup tasks again to update information used during the conversion:

- Run the `preconvert` utility.
- Run the `showconvert` utility.
- Make Solaris Volume Manager configuration changes if `showconvert` shows error messages.
- Re-run `preconvert` and `showconvert` if needed.
- Make Solaris Volume Manager backups.

The conversion tasks mark the point where you first make changes that cannot be reversed easily. Previous tasks of gathering information and reconfiguring the Solaris Volume Manager software (if needed) all allow you to continue using the Solaris Volume Manager software with no disruption. Do the following conversion tasks as soon as possible one after the other:

- Run the `doconvert` utility.
- Reboot the system.
- Change volume references in applications.

After the conversion completes successfully, the following post-conversion tasks are optional and can be done at any time:

- Improve volume layouts.
- Remove or reinitialize the Solaris Volume Manager software.

Schedule downtime

A conversion includes downtime for applications using Solaris Volume Manager volumes during the `doconvert`, reboot, and change volume references tasks and downtime for the system during the reboot. A second reboot happens automatically if a root device hosts Solaris Volume Manager objects or is a Solaris Volume Manager metadevice.

The doconvert utility takes about a minute unless the conversion includes RAID-5 volumes. RAID-5 conversions take about a minute for each megabyte of RAID-5 data. The total downtime to expect for a conversion is the sum of the following times:

- Approximately one minute to run `doconvert`.
- The time needed for one or two system reboots.
- Approximately one minute for each megabyte of RAID-5 data if RAID-5 volumes are included.
- The time needed for changing volume references in applications.

Check metadevices

Solaris Volume Manager metadevices may need changes before starting the conversion.

- Extended RAID-5 metadevices cannot be converted. A RAID-5 metadevice becomes extended when you add space after the device is first initialized.

  To find extended RAID-5 metadevices, use the `metastat -p` command of the Solaris Volume Manager software for each diskset. Look for RAID-5 devices in the output that have a `-o` option showing more original partitions than current partitions. The following example of output from `metastat -p` shows an extended metadevice:

  ```
  d100 -r clt65d0s3 clt66d0s3 clt67d0s3 clt99d0s3 -k -i 32b -0 3
  ```

  The example lists four partitions: 65, 66, 67, and 99. The `-o` entry shows the number of original partitions, three in this case.

  Use either of the following options to deal with extended RAID-5 devices:

  - Use the Solaris Volume Manager commands or the Solaris Volume Manager user interface to create a new metadevice large enough to contain all the data. Copy the data to the new metadevice.
  - Dump the data from the metadevice to tape and restore it to a VxVM volume after the conversion process finishes.

- Metadevices using a hot spare cannot be converted. Use the `metastat` command of the Solaris Volume Manager software to identify metadevices that currently use hot spares. The following example of output from `metastat` shows a metadevice using a hot spare:

  ```
  d3:Submirror of d51
     State: Okay
     Hot spare pool: hsp000
     Size 5922252 blocks
     Stripe 0:
     Device       Start Block     Dbase     State     Hot Spare
     clt75d0s1         0          No        Okay      clt66d0s6
  ```

With no hot spare in use, the “Hot Spare” column at the far right is empty.

Use either of the following options to deal with hot spares:

- Use the `metareplace` command of the Solaris Volume Manager software to move data from the hot spare onto the original partition (if there is room).
- Use the `metareplace` and `metahs` commands of the Solaris Volume Manager software to move the hot spare partition from the hot spare pool into the metadevice.

Identify references by applications

Find and list all references to Solaris Volume Manager metadevices from applications. The conversion utilities update `/etc/vfstab` with a new path for each metadevice that is converted to a VxVM volume. However, you must manually change references to Solaris Volume Manager metadevices in every application that uses a Solaris Volume Manager metadevice.

See “Run convertname” on page 163.

Preparing a Solaris Volume Manager configuration for conversion

The following preparation tasks can be done at any time before starting the conversion.

Installing VxVM

You must install Veritas Volume Manager before running the conversion utilities. Obtain the VxVM software and a full VxVM license key.

During the VxVM installation, do not run `vxinstall`. If required, you can use the `vxlicinst` command to install any license keys.

After installation, do not initialize any disks, and do not create any disk groups or VxVM objects.

The conversion process completes the VxVM installation and configuration.

Setting up a Solaris Volume Manager configuration for conversion

Do the setup tasks just before converting from the Solaris Volume Manager software to VxVM. If you make any changes between the setup tasks and the actual conversion, update the setup tasks.

Run preconvert

The `preconvert` utility analyzes the current Solaris Volume Manager configuration and builds a description for the new VxVM configuration. `Preconvert` does not make any changes to the Solaris Volume Manager configuration or to the host system. The description created by `preconvert` must be current with the state of the Solaris Volume Manager software when the conversion process starts or the process will fail. Always run `preconvert` again if you make any changes to the Solaris Volume Manager configuration.

Preconvert has no options and uses no arguments:

```
# preconvert
```

Messages from `preconvert` appear on the screen and in a file that is used by the `showconvert` utility.

`WARNING` errors are for your information only and require no changes to the Solaris Volume Manager configuration.

`FATAL` errors mean that the conversion process will fail unless you make changes to the Solaris Volume Manager configuration. A `FATAL` error does not stop `preconvert` from examining the rest of the Solaris Volume Manager configuration. `FATAL` errors that do not make sense may be related to earlier `FATAL` errors. Correct the earlier `FATAL` errors and run `preconvert` again to see if the later errors still exist.

The following are possible causes of FATAL errors:

- A metadevice is being initialized or resynchronized.
- A disk is offline.
- `The /etc/vfstab` file contains incorrect entries.
- An extended RAID-5 metadevice exists.
- A metadevice is using a hot spare.

Run showconvert

The `showconvert` utility displays the `preconvert` conversion plan in a readable format. The command `showconvert help` gives a list of command options.

```
# showconvert keyword [arg ...]
```

The following keywords and arguments are defined for the `showconvert` command:

`log`

Displays error messages generated by the `preconvert` utility. All `FATAL` messages must be resolved and `preconvert` run again before the conversion process can run successfully.

`list [diskgroup ...]`

Lists all the VxVM disk groups that will be created from Solaris Volume Manager disk sets and gives information about each disk group.

These VxVM disk groups are not created to be compatible with CDS. Additionally, the disks in these disk groups are set up in `slice` and not `cds` format. To make the disk groups compatible with CDS, use the `vxcdsconvert` command.

See the `vxcdsconvert`(1M) manual page.

`print diskgroup [vxprint_args]`

Uses standard arguments from the VxVM `vxprint` utility and displays the VxVM objects that the conversion will create.

The conversion plan displayed by `showconvert` describes the layouts that the conversion utilities create. The layouts will have the same data on the same disks as the Solaris Volume Manager configuration. If you want different layouts, make changes to the Solaris Volume Manager configuration and run `preconvert` and `showconvert` again.

Run convertname

The `convertname` utility takes Solaris Volume Manager device paths as arguments (metadevice paths or raw disk paths) and returns the VxVM volume path for the device as it will show after the conversion. Use the command to make a list of path names for replacing references in applications. One or more device paths as arguments returns one line for each argument with the proposed VxVM path. Use absolute path names. For example:

```
# convertname /dev/md/dsk/d2 /dev/md/rdsk/d3
```

Table 6-1 shows examples of how names are translated by the conversion process:

Table 6-1 Examples of the translation of device path names

Metadevice path	VxVM volume path
`/dev/md/dsk/d2`	`/dev/vx/dsk/rootdg/d2`
`/dev/md/rdsk/d3`	`/dev/vx/rdsk/rootdg/d3`
`/dev/md/setname/dsk/d10`	`/dev/vx/dsk/setname/d10`
`/dev/md/setname/rdsk/d12`	`/dev/vx/rdsk/setname/d12`

The following metadevices do not retain their dN names or are converted from raw partitions that did not have dN names:

- The volume for the root file system is always named `rootvol` as a VxVM convention.
- A trans metadevice has its data device converted using the master device name. The log device is lost.
- A file system encapsulated from a raw partition is named for the last component of its mount point.
- A swap device on a disk that will include VxVM volumes after the conversion receives a name starting with `swapvol`.
- The `/usr`, `/opt`, and `/var` file systems when converted from being Solaris Volume Manager metadevices or when encapsulated retain the names `/usr`, `/opt`, and `/var`.

Warning: Any partition that is on a disk with Solaris Volume Manager metadevices and that is not listed as a file system in `/etc/vfstab` is lost during the conversion. The partition is considered free space when the disk is converted to VxVM control. Even if the partitions are in use, such as for Oracle raw tablespace data, they are not encapsulated during the conversion. The raw partitions are lost when the disk layout changes. Either encapsulate the partition under the Solaris Volume Manager software and allow the conversion to convert the resulting metadevice, or back up the partition, plan to create a new volume for it, and restore the partition after the conversion.

Make backups

Make backups of all Solaris Volume Manager software-related files and information before running the conversion utility. If you need to return to the Solaris Volume Manager configuration, using backups is the recommended way.

Converting from the Solaris Volume Manager software to VxVM

Warning: This is the first step that makes changes, and that is not easily reversed. Be sure that you are ready for the conversion before performing this step. This step also includes downtime for volume management and leads to a reboot for the system.

Run the `doconvert` utility to start the conversion from the Solaris Volume Manager software to VxVM. A manual reboot of the system completes the conversion by updating configuration changes that were made by `doconvert`.

```
# doconvert [force]
```

The `doconvert` utility checks that the VxVM licenses are correct, makes configuration changes, and then prompts for a system reboot. You must reboot the system immediately to complete the conversion process. If you run doconvert with the `force` option, the system is automatically rebooted without prompting you.

Warning: Do not make system configuration changes after running `doconvert` and before rebooting. Such changes can lead to loss of data.

If you choose not to reboot, the location of a script is given that allows you to reverse the changes to the system files made by `doconvert`. If you run this script, the system files are reinstated and the conversion is prevented from completing at the next reboot.

If doconvert fails, the system remains unchanged. You can then return to the preparation steps to correct problems. The following situations cause doconvert to fail:

- The output of `preconvert` does not exist or is stale.
- `Preconvert` terminated with an error.
- VxVM licenses are not sufficient for the planned VxVM configuration.

- A VxVM configuration already exists.

During the conversion, `doconvert` displays locations for some system file backups.

Reboot the system

Reboot the system immediately after doconvert finishes. During the reboot:

- If the system root device hosts Solaris Volume Manager objects or is a Solaris Volume Manager metadevice, the root device becomes a VxVM volume during the reboot. The process of making the root device a VxVM volume automatically starts a second reboot.
- If the system goes down, from a power loss or other problem, the conversion process continues when the system boots again. Messages such as “Already exists...” appear for processes that completed before the system went down. The messages are harmless.

Change volume references

After the reboot, you must manually change references in applications that use Solaris Volume Manager metadevices. The references need to point to the new VxVM volumes.

See “Identify references by applications” on page 161.

See “Run convertname” on page 163.

Post conversion tasks

Do the following tasks at any time after a successful conversion.

Improve volume layouts

Start VxVM and use tools such as `vxassist` or the VxVM user interface to relayout volumes online. New layouts may require additional disk space.

You may want to create logs for converted RAID-5 volumes. RAID-5 logging from the Solaris Volume Manager configuration is not suitable for conversion to use by VxVM. Logs are not necessary for normal operations, but are useful for recovery. Use the following command to create a log for a RAID-5 volume:

```
# vxassist [-g diskgroup] addlog volume
```

Remove the Solaris Volume Manager software

After the conversion, the Solaris Volume Manager software controls no data and is disabled. You can safely remove the Solaris Volume Manager software or you can re-initialize the Solaris Volume Manager software for use on disks not controlled by VxVM. The Solaris Volume Manager software and VxVM can run at the same time on the same machine.

Converting a root disk

The following steps allow you to move only a root disk from Solaris Volume Manager control to VxVM control. Use this procedure if you want to move only the root disk from the Solaris Volume Manager software to VxVM.

Warning: You cannot use the conversion utilities in a mixed environment. Even if you move only the root disk from the Solaris Volume Manager software to VxVM, you cannot later use the conversion utilities to move other Solaris Volume Manager objects to Veritas Volume Manager.

To convert a root disk

1. Use the Solaris Volume Manager unencapsulation procedure to remove the root disk from Solaris Volume Manager control. See the Solaris Volume Manager documentation for details.
2. Create VxVM volumes from the partitions on the root disk by encapsulating the root disk.

Chapter 7

Thin Provisioning and SmartMove

This chapter includes the following topics:

About Thin Storage

Thin Storage is an array vendor solution for allocating storage to applications only when the storage is truly needed, from a pool of free storage. Thin Storage attempts to solve the problem of under utilization of available array capacity.

Thin Storage Reclamation-capable arrays and LUNs allow the administrators to release once-used storage to the pool of free storage. Storage is allocated from the free pool when files are created and written to in the file system. However, this storage is not released to the free pool when files get deleted; the administrator must perform the operation of reclaiming this storage for the free pool.

Veritas File System supports reclamation of the free blocks in the file system on Veritas Volume Manager-backed file systems. The operation of reclamation can be done on a disk, LUN, full file system, or part of a file system using the `vxdisk` and `fsadm` commands, and the `vxfs_ts_reclaim` API.

Identifying thin and thin reclamation LUNs

This section describes how to identify thin and thin reclamation LUNs

To identify LUNs

◆ To identify LUNs that are `thin` or `thin_relcm` type the following command:

```
# vxdisk -o thin list
```

Reclamation of storage on thin reclamation arrays

This section describes reclamation of storage on thin reclamation arrays.

How reclamation on a deleted volume works

Storage that is no longer in use, needs to be reclaimed by the array. The process of reclaiming storage on an array can be intense on the array. To avoid any impact on regular I/O's to the array, the reclaim operation is made asynchronous. When a volume is deleted the space previously used by the volume is tracked for later asynchronous reclamation. This asynchronous reclamation is handled by `vxrelocd` (or recovery) daemon.

To perform the reclaim operation during less critical time of the system can be controlled by the following two tunables `reclaim_on_delete_wait_period` and`reclaim_on_delete_start_time`.

The default value for these tunables are:

```
reclaim_on_delete_wait_period=1
reclaim_on_delete_start_time=22:00
```

`reclaim_on_delete_wait_period`

The storage space used by the deleted volume is reclaimed after `reclaim_on_delete_wait_period` days. The value of the tunable can be anything between `-1` to `367`. The default is set to `1`, that means the volume is deleted the next day. The storage is reclaimed immediately if the value is `-1`. The storage space is not reclaimed automatically, if the value is greater than `366`. It can only be reclaimed manually using `vxdisk reclaim` command.

`reclaim_on_delete_start_time`

This tunable specifies the time of the day, the reclaim on the deleted volume is performed. The default time is set to `22:00`. This can be changed to any time of the day.

By default, the `vxrelocd` daemon runs everyday at 22:00 hours and reclaims storage on the deleted volume that are a one day old.

The tunables can be changed using the `vxdefault` command.

Thin Reclamation of a disk, a disk group, or an enclosure

Use the `vxdisk reclaim` command to trigger online Thin Reclamation on one or more disks, disk groups, or enclosures. The following examples triggers reclamation on LUNs disk1 and disk2, disk group oradg, and Enclosure=EMC_Clariion0 respectively.

For disks:

```
# vxdisk reclaim disk1 disk2
```

For disk group:

```
# vxdisk reclaim oradg
```

For enclosure:

```
# vxdisk reclaim  EMC_CLARiiON0
```

You can only perform Thin Reclamation on LUNS which exhibit thin_rclm attribute/flag. VxVM automatically discovers LUNs that support Thin Reclamation from capable storage arrays. To list devices that are known to be thin or thin_rclm on a host, use the `vxdisk -o thin list` command. For example:

```
# vxdisk -o thin list
DEVICE          SIZE(mb)     PHYS_ALLOC(mb)  GROUP          TYPE
tagmastore-usp0_065a 10000        84         -              thinrclm
tagmastore-usp0_065b 10000        84         -              thinrclm
tagmastore-usp0_065c 10000        84         -              thinrclm
tagmastore-usp0_065d 10000        84         -              thinrclm
tagmastore-usp0_065e 10000        84         -              thinrclm
tagmastore-usp0_065f 10000        84         -              thinrclm
tagmastore-usp0_0652 10000        84         -              thinrclm
tagmastore-usp0_0653 10000        84         -              thinrclm
tagmastore-usp0_0654 10000        84         -              thinrclm
tagmastore-usp0_0655 10000        84         -              thinrclm
```

```
tagmastore-usp0_0656 10000        84         -              thinrclm
tagmastore-usp0_0657 10000        84         -              thinrclm
tagmastore-usp0_0658 10000        84         -              thinrclm
tagmastore-usp0_0659 10000        84         -              thinrclm
tagmastore-usp0_0660 10000        672        thindiskgroup  thinrclm
```

You can only perform Thin Reclamation if the VxVM volume is on a “mounted” VxFS file system.

For more information on how to trigger Thin Reclamation on a VxFS file system, see the *Veritas File System Administrator's Guide*.

Thin Reclamation takes considerable amount of time when you reclaim thin storage on a large number of LUNs or an enclosure or disk group.

About SmartMove

With Storage Foundation’s SmartMove feature, Veritas File System (VxFS) lets Veritas Volume Manager (VxVM) know which blocks have data. VxVM, which is the copy engine for migration, copies only the used blocks and avoids the empty spaces. This behavior helps optimize the thin storage utilization.

Setting up SmartMove

This section describes how to set up SmartMove.

Displaying the SmartMove configuration

This section describes how to display the SmartMove configuration.

To display the current SmartMove value type

◆ To display the current SmartMove value type the following command:

```
# vxdefault list
KEYWORD                           CURRENT-VALUE   DEFAULT-VALUE
usefssmartmove                    all             thinonly
...
```

Changing the SmartMove configuration

SmartMove has three different values where SmartMove can be applied or not. The three values are:

Value	Meaning
none	Do not use SmartMove at all.
thinonly	Use SmartMove for thin awares LUNs only (the default value)
all	Use SmartMove for all operations

To set the SmartMove value

- To set the SmartMove to apply to all volumes type the following command:

  ```
  # vxdefault set userfssmartmove <value>
  ```

  where *value* is either none, thinonly, or all.

Migrating to thin provisioning

The SmartMove™ feature enables migration from traditional LUNs to thinly provisioned LUNs, removing unused space in the process.

See the *Vertias Volume Manager Administrator's Guide* for more information on SmartMove and Thin Provisioning.

To migrate to thin provisioning

1. Check if the SmartMove Feature is enabled:

   See "Displaying the SmartMove configuration" on page 172.

   See "Changing the SmartMove configuration" on page 172.

2. Add the new, thin LUNs to the existing disk group:

   ```
   # vxdisksetup -i <da_name>
   # vxdg -g datadg adddisk <da_name>
   ```

   where <*da_name*> is the disk access name in VxVM.

   To enable enclosure-based naming (EBN), enter:

   ```
   # vxddladm set namingscheme=ebn
   ```

3. To identify thin or thinrclm LUNs, enter:

   ```
   # vxdisk -o thin list
   ```

4. Add the new, thin LUNs as a new plex to the volume.

NOTE: The VxFS file system must be mounted to get the benefits of the SmartMove feature.

The following methods are available to add the LUNs:

- Use the default settings:

  ```
  # vxassist -g datadg mirror datavol da_name
  ```

- Use the options for fast completion. The following command has more I/O affect.

  ```
  # vxassist -b -oiosize=1m -t thinmig -g datadg mirror \
    datavol da_name
  # vxtask monitor thinmig
  ```

- Use the options for minimal affect. The following command takes longer to complete:

  ```
  # vxassist -oslow -g datadg mirror datavol da_name
  ```

5 Optionally, test the performance of the new LUNs before removing the old LUNs.

To test the performance, use the following steps:

- Determine which plex corresponds to the thin LUNs:

  ```
  # vxprint -g datadg
  ```

```
TY NAME               ASSOC           KSTATE   LENGTH     PLOFFS  STATE     TUTIL0 PUTIL0
dg datadg             datadg          -        -          -       -         -        -
dm THINARRAY0_02      THINARRAY0_02   -        83886080   -       -         -        -
dm STDARRAY1_01       STDARRAY1_01    -        41943040   -       -OHOTUSE  -        -

v  datavol            fsgen           ENABLED  41943040   -       ACTIVE    -        -
pl datavol-01         datavol         ENABLED  41943040   -       ACTIVE    -        -
sd STDARRAY1_01-01    datavol-01      ENABLED  41943040   0       -         -        -
pl datavol-02         datavol         ENABLED  41943040   -       ACTIVE    -        -
sd THINARRAY0_02-01   datavol-02      ENABLED  41943040   0       -         -        -
```

  The above output indicates that the thin LUNs corresponds to plex datavol-02.

- Direct all reads to come from those LUNs:

  ```
  # vxvol -g datadg rdpol prefer datavol datavol-02
  ```

6 Remove the original non-thin LUNs.

Note: The ! character is a special character in some shells. This example shows how to escape it in a bash shell.

```
# vxassist -g datadg remove mirror datavol \!STDARRAY1_01
# vxdg -g datadg rmdisk STDARRAY1_01
# vxdisk rm STDARRAY1_01
```

7 Grow the file system and volume to use all of the larger thin LUN:

```
# vxresize -g datadg -x datavol 40g da_name
```

SmartMove and Thin array limitations

This section lists the SmartMove and Thin array limitations.

SmartMove, thin arrays, and mirrors

The SmartMove and Thin Storage take advantage of the VxFS file systems knowledge of what space on the disk is in use. If a volume contains a file system yet it is not mounted, then SmartMove cannot use the file system to determine what sections of the disk are in use and what sections are not. This means that care must be taken when mirroring a volume of thin Storage LUNs.

Here are some examples that assume volumes and snapshots are mirrored:

- `vxassist` mirrored volume creation SmartMove
- mirrored volume of mounted file system SmartMove
- snapshot refresh of mounted file system volume SmartMove

Here are some examples which assume volumes and snapshots are mirrored and recovery needs to occur on the volumes following a Thin Storage LUN or array failure:

- mirrored volume of mounted file system SmartMove
- mirrored volume of unmounted file system Full Resync
- mirrored snapshot volume recovery (mounted) SmartMove
- mirrored snapshot volume recovery (unmounted) Full Resync

Note: Extra recover steps using the `vxdisk reclaim` command may be necessary following array or LUN failures of Thin Storage arrays when the volumes are mirrored and unmounted during volume recovery.

SmartMove overhead

The SmartMove feature has an overhead associated with it. To avoid copying blocks that are unused, SmartMove must query the file system and determine if the blocks are in use. For very large and empty file systems this takes time, however as the file system fills up the overhead of checking that unused blocks reduces.

Chapter 8

Dynamic Storage Tiering

This chapter includes the following topics:

About Dynamic Storage Tiering

VxFS uses multi-tier online storage by way of the Dynamic Storage Tiering (DST) feature, which functions on top of multi-volume file systems. Multi-volume file systems are file systems that occupy two or more virtual volumes. The collection of volumes is known as a volume set. A volume set is made up of disks or disk array LUNs belonging to a single Veritas Volume Manager (VxVM) disk group. A multi-volume file system presents a single name space, making the existence of multiple volumes transparent to users and applications. Each volume retains a separate identity for administrative purposes, making it possible to control the locations to which individual files are directed.

Note: Some of the commands have changed or been removed between the 4.1 release and the 5.1 release to make placement policy management more user-friendly. The following commands have been removed: `fsrpadm`, `fsmove`, and `fssweep`. The output of the `queryfile`, `queryfs`, and `list` options of the `fsapadm` command now print the allocation order by name instead of number.

DST allows administrators of multi-volume VxFS file systems to manage the placement of files on individual volumes in a volume set by defining placement policies. Placement policies control both initial file location and the circumstances under which existing files are relocated. These placement policies cause the files to which they apply to be created and extended on specific subsets of a file system's volume set, known as placement classes. The files are relocated to volumes in other placement classes when they meet the specified naming, timing, access rate, and storage capacity-related conditions.

You make a VxVM volume part of a placement class by associating a volume tag with it. For file placement purposes, VxFS treats all of the volumes in a placement class as equivalent, and balances space allocation across them. A volume may have more than one tag associated with it. If a volume has multiple tags, the volume belongs to multiple placement classes and is subject to allocation and relocation policies that relate to any of the placement classes. Multiple tagging should be used carefully.

See "Placement classes" on page 179.

VxFS imposes no capacity, performance, availability, or other constraints on placement classes. Any volume may be added to any placement class, no matter what type the volume has nor what types other volumes in the class have. However, a good practice is to place volumes of similar I/O performance and availability in the same placement class.

Note: Dynamic Storage Tiering is a licensed feature. You must purchase a separate license key for DST to operate. See the *Veritas Storage Foundation Release Notes*.

The *Using Dynamic Storage Tiering* Symantec Yellow Book provides additional information regarding the Dynamic Storage Tiering feature, including the value of DST and best practices for using DST. You can download *Using Dynamic Storage Tiering* from the following Web page:

http://www.symantec.com/enterprise/yellowbooks/index.jsp

Supported Dynamic Storage Tiering document type definitions

Table 8-1 describes which releases of Veritas File System (VxFS) support specific Dynamic Storage Tiering (DST) document type definitions (DTDs).

Table 8-1 Supported Dynamic Storage Tiering document type definitions

VxFS Version	DTD Version	
	1.0	1.1
5.0	Supported	Not supported
5.1	Supported	Supported

Placement classes

A placement class is a Dynamic Storage Tiering attribute of a given volume in a volume set of a multi-volume file system. This attribute is a character string, and is known as a volume tag. A volume can have different tags, one of which can be the placement class. The placement class tag makes a volume distinguishable by DST.

Volume tags are organized as hierarchical name spaces in which periods separate the levels of the hierarchy . By convention, the uppermost level in the volume tag hierarchy denotes the Storage Foundation component or application that uses a tag, and the second level denotes the tag's purpose. DST recognizes volume tags of the form `vxfs.placement_class.`*class_name*. The prefix `vxfs` identifies a tag as being associated with VxFS. The `placement_class` string identifies the tag as a file placement class that DST uses. The *class_name* string represents the name of the file placement class to which the tagged volume belongs. For example, a volume with the tag **vxfs.placement_class.tier1** belongs to placement class `tier1`. Administrators use the `vxassist` command to associate tags with volumes.

See the `vxassist`(1M) manual page.

VxFS policy rules specify file placement in terms of placement classes rather than in terms of individual volumes. All volumes that belong to a particular placement class are interchangeable with respect to file creation and relocation operations. Specifying file placement in terms of placement classes rather than in terms of specific volumes simplifies the administration of multi-tier storage.

The administration of multi-tier storage is simplified in the following ways:

- Adding or removing volumes does not require a file placement policy change. If a volume with a tag value of `vxfs.placement_class.tier2` is added to a file system's volume set, all policies that refer to `tier2` immediately apply to the newly added volume with no administrative action. Similarly, volumes can be evacuated, that is, have data removed from them, and be removed from a file system without a policy change. The active policy continues to apply to the file system's remaining volumes.
- File placement policies are not specific to individual file systems. A file placement policy can be assigned to any file system whose volume set includes volumes tagged with the tag values (placement classes) named in the policy. This property makes it possible for data centers with large numbers of servers to define standard placement policies and apply them uniformly to all servers with a single administrative action.

Tagging volumes as placement classes

The following example tags the `vsavola` volume as placement class `tier1`, `vsavolb` as placement class `tier2`, `vsavolc` as placement class `tier3`, and `vsavold` as placement class `tier4` using the `vxadm` command.

To tag volumes

◆ Tag the volumes as placement classes:

```
# vxassist -g cfsdg settag vsavola vxfs.placement_class.tier1
# vxassist -g cfsdg settag vsavolb vxfs.placement_class.tier2
# vxassist -g cfsdg settag vsavolc vxfs.placement_class.tier3
# vxassist -g cfsdg settag vsavold vxfs.placement_class.tier4
```

Listing placement classes

Placement classes are listed using the `vxassist listtagvxassist listtag` command.

See the `vxassist`(1M) manual page.

The following example lists all volume tags, including placment classes, set on a volume `vsavola` in the diskgroup `cfsdg`.

To list placement classes

◆ List the volume tags, including placement classes:

```
# vxassist -g cfsdg listtag vsavola
```

Administering placement policies

A VxFS file placement policy document contains rules by which VxFS creates, relocates, and deletes files, but the placement policy does not refer to specific file systems or volumes. You can create a file system's active file placement policy by assigning a placement policy document to the file system via the `fsppadm` command or the GUI.

See the `fsppadm`(1M) manual page.

At most, one file placement policy can be assigned to a VxFS file system at any time. A file system may have no file placement policy assigned to it, in which case VxFS allocates space for new files according to its own internal algorithms.

In systems with Storage Foundation Management Server (SFMS) software installed, file placement policy information is stored in the SFMS database. The SFMS database contains both XML policy documents and lists of hosts and file systems for which each document is the current active policy. When a policy document is updated, SFMS can assign the updated document to all file systems whose current active policies are based on that document. By default, SFMS does not update file system active policies that have been created or modified locally, that is by the hosts that control the placement policies' file systems. If a SFMS administrator forces assignment of a placement policy to a file system, the file system's active placement policy is overwritten and any local changes that had been made to the placement policy are lost.

Assigning a placement policy

The following example uses the `fsppadm assign` command to assign the file placement policy represented in the XML policy document `/tmp/policy1.xml` for the file system at mount point `/mnt1`.

To assign a placement policy

◆ Assign a placement policy to a file system:

```
# fsppadm assign /mnt1 /tmp/policy1.xml
```

Unassigning a placement policy

The following example uses the `fsppadm unassign` command to unassign the active file placement policy from the file system at mount point `/mnt1`.

To unassign a placement policy

- Unassign the placement policy from a file system:

```
# fsppadm unassign /mnt1
```

Analyzing the space impact of enforcing a placement policy

The following example uses the `fsppadm analyze` command to analyze the impact if the enforce operation is performed on the file placement policy represented in the XML policy document `/tmp/policy1.xml` for the mount point `/mnt1`. The command builds the I/O temperature database if necessary.

To analyze the space impact of enforcing a placement policy

- Analyze the impact of enforcing the file placement policy represented in the XML policy document `/tmp/policy1.xml` for the mount point `/mnt1`:

```
# fsppadm analyze -F /tmp/policy1.xml -i /mnt1
```

Querying which files will be affected by enforcing a placement policy

The following example uses the `fsppadm query` command to generate a list of files that will be affected by enforcing a placement policy. The command provides details about where the files currently reside, to where the files will be relocated, and which rule in the placement policy applies to the files.

To query which files will be affected by enforcing a placement policy

- Query the files that will be affected:

```
# fsppadm query /mnt1/dir1/dir2 /mnt2 /mnt1/dir3
```

Enforcing a placement policy

Enforcing a placement policy for a file system requires that the policy be assigned to the file system. You must assign a placement policy before it can be enforced.

See "Assigning a placement policy" on page 181.

Enforce operations are logged in a hidden file, `.__fsppadm_enforce.log`, in the `lost+found` directory of the mount point. This log file contains details such as files' previous locations, the files' new locations, and the reasons for the files' relocations. The enforce operation creates the `.__fsppadm_enforce.log` file if the file does not exist. The enforce operation appends the file if the file already exists. The .__fsppadm_enforce.log file can be backed up or removed as with a normal file.

You can specify the -F option to specify a placement policy other than the existing active placement policy. This option can be used to enforce the rules given in the specified placement policy for maintenance purposes, such as for reclaiming a LUN from the file system.

You can specify the -p option to specify the number of concurrent threads to be used to perform the fsppadm operation. You specify the *io_nice* parameter as an integer between 1 and 100, with 50 being the default value. A value of 1 specifies 1 slave and 1 master thread per mount. A value of 50 specifies 16 slaves and 1 master thread per mount. A value of 100 specifies 32 slaves and 1 master thread per mount.

You can specify the -C option so that the fsppadm command processes only those files that have some activity stats logged in the File Change Log (FCL) file during the period specified in the placement policy. You can use the -C option only if the policy's ACCESSTEMP or IOTEMP elements use the Prefer criteria.

You can specify the -T option to specify the placement classes that contain files for the fsppadm command to sweep and relocate selectively. You can specify the -T option only if the policy uses the Prefer criteria forIOTEMP.

See the fsppadm(1M) manual page.

The following example uses the fsppadm enforce command to enforce the file placement policy for the file system at mount point /mnt1, and includes the access time, modification time, and file size of the specified paths in the report, /tmp/report.

To enforce a placement policy

◆ Enforce a placement policy for a file system:

```
# fsppadm enforce -a -r /tmp/report /mnt1
Current Current  Relocated Relocated
Class   Volume   Class     Volume    Rule    File
tier3   dstvole  tier3     dstvole   a_to_z  /mnt1/mds1/d1/file1
tier3   dstvole  tier3     dstvole   a_to_z  /mnt1/mds1/d1/file2
tier3   dstvole  tier3     dstvole   a_to_z  /mnt1/mds1/d1/d2/file3
tier3   dstvolf  tier3     dstvolf   a_to_z  /mnt1/mds1/d1/d2/file4
.
.
.
Sweep path    : /mnt1
Files moved   : 42
KB moved      : 1267


Tier Name         Size (KB)   Free Before (KB)    Free After (KB)
tier4                524288             524256             524256
tier3                524288             522968             522968
tier2                524288             524256             524256
tier1                524288             502188             501227
```

Validating a placement policy

The following example uses the `fsppadm validate` command to validate the placement policy `policy.xml` against all mounted file systems.

To validate a placement policy against all mounted file systems

◆ Validate the placement policy:

```
# fsppadm validate /tmp/policy.xml
```

File placement policy grammar

VxFS allocates and relocates files within a multi-volume file system based on properties in the file system metadata that pertains to the files. Placement decisions may be based on file name, directory of residence, time of last access, access frequency, file size, and ownership. An individual file system's criteria for allocating and relocating files are expressed in the file system's file placement policy.

A VxFS file placement policy defines the desired placement of sets of files on the volumes of a VxFS multi-volume file system. A file placement policy specifies the placement classes of volumes on which files should be created, and where and under what conditions the files should be relocated to volumes in alternate placement classes or deleted. You can create file placement policy documents, which are XML text files, using an XML editor, a text editor, or Veritas Storage Foundation Manager (SFM).

See the `/opt/VRTSvxfs/etc/placement_policy.dtd` file for the overall structure of a placement policy.

File placement policy rules

A VxFS file placement policy consists of one or more rules. Each rule applies to one or more files. The files to which a rule applies are designated in one or more `SELECT` statements. A `SELECT` statement designates files according to one or more of four properties: their names or naming patterns, the directories in which they reside, their owners' user names, and their owners' group names.

A file may be designated by more than one rule. For example, if one rule designates files in directory `/dir`, and another designates files owned by `user1`, a file in `/dir` that is owned by `user1` is designated by both rules. Only the rule that appears first in the placement policy applies to the file; subsequent rules are ignored.

You can define placement policies that do not encompass the entire file system name space. When a file that is not designated by any rule in its file system's active placement policy is created, VxFS places the file according to its own internal algorithms. To maintain full control over file placement, include a catchall rule at the end of each placement policy document with a `SELECT` statement that designates files by the naming pattern `*`. Such a rule designates all files that have not been designated by the rules appearing earlier in the placement policy document.

Two types of rules exist: `data` and `ckpt`. The `data` rule type allows DST to relocate normal data files. The `ckpt` rule type allows DST to relocate Storage Checkpoints. You specify the rule type by setting the `Flags` attribute for the rule.

SELECT statement

The VxFS placement policy rule `SELECT` statement designates the collection of files to which a rule applies.

The following XML snippet illustrates the general form of the `SELECT` statement:

```
<SELECT>
  <DIRECTORY Flags="directory_flag_value"> value
  </DIRECTORY>
  <PATTERN Flags="pattern_flag_value"> value </PATTERN>
  <USER> value </USER>
  <GROUP> value </GROUP>
</SELECT>
```

A `SELECT` statement may designate files by using the following selection criteria:

`<DIRECTORY>` A full path name relative to the file system mount point. The `Flags="directory_flag_value"` XML attribute must have a value of `nonrecursive`, denoting that only files in the specified directory are designated, or a value of `recursive`, denoting that files in all subdirectories of the specified directory are designated. The `Flags` attribute is mandatory.

The `<DIRECTORY>` criterion is optional, and may be specified more than once.

`<PATTERN>`

Either an exact file name or a pattern using a single wildcard character (*). For example, the pattern “abc*" denotes all files whose names begin with “abc". The pattern “abc.*" denotes all files whose names are exactly "abc" followed by a period and any extension. The pattern “*abc" denotes all files whose names end in “abc", even if the name is all or part of an extension. The pattern “*.abc" denotes files of any name whose name extension (following the period) is “abc". The pattern “ab*c" denotes all files whose names start with “ab" and end with “c". The first "*" character is treated as a wildcard, while any subsequent "*" characters are treated as literal text. The pattern cannot contain "/".

The wildcard character matches any character, including "`.`", "`?`", and "`[`", unlike using the wildcard in a shell.

The `Flags="pattern_flag_value"` XML attribute is optional, and if specified can only have a value of `recursive`. Specify `Flags="recursive"` only if the pattern is a directory. If `Flags` is not specified, the default attribute value is nonrecursive. If `Flags="recursive"` is specified, the enclosing selection criteria selects all files in any component directory that is anywhere below the directory specified by `<DIRECTORY>` if the component directory matches the pattern and either of the following is true:

- `<DIRECTORY>` is specified and has the recursive flag.
- `<DIRECTORY>` is not specified and the directory is anywhere in the file system.

If the pattern contains the wildcard character (*), wildcard character matching is performed.

The `<PATTERN>` criterion is optional, and may be specified more than once. Only one value can be specified per `<PATTERN>` element.

`<USER>`

User name of the file's owner. The user number cannot be specified in place of the name.

The `<USER>` criterion is optional, and may be specified more than once.

`<GROUP>`

Group name of the file's owner. The group number cannot be specified in place of the group name.

The `<GROUP>` criterion is optional, and may be specified more than once.

One or more instances of any or all of the file selection criteria may be specified within a single `SELECT` statement. If two or more selection criteria of different types are specified in a single statement, a file must satisfy one criterion of each type to be selected.

In the following example, only files that reside in either the ora/db or the crash/dump directory, and whose owner is either user1 or user2 are selected for possible action:

```
<SELECT>
  <DIRECTORY Flags="nonrecursive">ora/db</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">crash/dump</DIRECTORY>
  <USER>user1</USER>
  <USER>user2</USER>
</SELECT>
```

A rule may include multiple SELECT statements. If a file satisfies the selection criteria of one of the SELECT statements, it is eligible for action.

In the following example, any files owned by either user1 or user2, no matter in which directories they reside, as well as all files in the ora/db or crash/dump directories, no matter which users own them, are eligible for action:

```
<SELECT>
  <DIRECTORY Flags="nonrecursive">ora/db</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">crash/dump</DIRECTORY>
</SELECT>
<SELECT>
  <USER>user1</USER>
  <USER>user2</USER>
</SELECT>
```

When VxFS creates new files, VxFS applies active placement policy rules in the order of appearance in the active placement policy's XML source file. The first rule in which a SELECT statement designates the file to be created determines the file's placement; no later rules apply. Similarly, VxFS scans the active policy rules on behalf of each file when relocating files, stopping the rules scan when it reaches the first rule containing a SELECT statement that designates the file. This behavior holds true even if the applicable rule results in no action. Take for example a policy rule that indicates that .dat files inactive for 30 days should be relocated, and a later rule indicates that .dat files larger than 10 megabytes should be relocated. A 20 megabyte .dat file that has been inactive for 10 days will not be relocated because the earlier rule applied. The later rule is never scanned.

A placement policy rule's action statements apply to all files designated by any of the rule's SELECT statements. If an existing file is not designated by a SELECT statement in any rule of a file system's active placement policy, then DST does not relocate or delete the file. If an application creates a file that is not designated by a SELECT statement in a rule of the file system's active policy, then VxFS places the file according to its own internal algorithms. If this behavior is inappropriate, the last rule in the policy document on which the file system's active placement policy is based should specify <PATTERN>*</PATTERN> as the only selection criterion in its SELECT statement, and a CREATE statement naming the desired placement class for files not selected by other rules.

CREATE statement

A CREATE statement in a file placement policy rule specifies one or more placement classes of volumes on which VxFS should allocate space for new files to which the rule applies at the time the files are created. You can specify only placement classes, not individual volume names, in a CREATE statement.

A file placement policy rule may contain at most one CREATE statement. If a rule does not contain a CREATE statement, VxFS places files designated by the rule's SELECT statements according to its internal algorithms. However, rules without CREATE statements can be used to relocate or delete existing files that the rules' SELECT statements designate.

The following XML snippet illustrates the general form of the CREATE statement:

```
<CREATE>
  <ON Flags="flag_value">
    <DESTINATION>
      <CLASS> placement_class_name </CLASS>
      <BALANCE_SIZE Units="units_specifier"> chunk_size
      </BALANCE_SIZE>
    </DESTINATION>
    <DESTINATION> additional_placement_class_specifications
    </DESTINATION>
  </ON>
</CREATE>
```

A CREATE statement includes a single <ON> clause, in which one or more <DESTINATION> XML elements specify placement classes for initial file allocation in order of decreasing preference. VxFS allocates space for new files to which a rule applies on a volume in the first class specified, if available space permits. If space cannot be allocated on any volume in the first class, VxFS allocates space on a volume in the second class specified if available space permits, and so forth.

If space cannot be allocated on any volume in any of the placement classes specified, file creation fails with an ENOSPC error, even if adequate space is available elsewhere in the file system's volume set. This situation can be circumvented by specifying a Flags attribute with a value of "any" in the <ON> clause. If <ON Flags="any"> is specified in a CREATE statement, VxFS first attempts to allocate space for new files to which the rule applies on the specified placement classes. Failing that, VxFS resorts to its internal space allocation algorithms, so file allocation does not fail unless there is no available space any-where in the file system's volume set.

The `Flags="any"` attribute differs from the catchall rule in that this attribute applies only to files designated by the `SELECT` statement in the rule, which may be less inclusive than the `<PATTERN>*</PATTERN>` file selection specification of the catchall rule.

In addition to the placement class name specified in the `<CLASS>` sub-element, a `<DESTINATION>` XML element may contain a `<BALANCE_SIZE>` sub-element. Presence of a `<BALANCE_SIZE>` element indicates that space allocation should be distributed across the volumes of the placement class in chunks of the indicated size. For example, if a balance size of one megabyte is specified for a placement class containing three volumes, VxFS allocates the first megabyte of space for a new or extending file on the first (lowest indexed) volume in the class, the second megabyte on the second volume, the third megabyte on the third volume, the fourth megabyte on the first volume, and so forth. Using the `Units` attribute in the `<BALANCE_SIZE>` XML tag, the balance size value may be specified in the following units:

`bytes`	Bytes
`KB`	Kilobytes
`MB`	Megabytes
`GB`	Gigabytes

The `<BALANCE_SIZE>` element distributes the allocation of database files across the volumes in a placement class. In principle, distributing the data in each file across multiple volumes distributes the I/O load across the volumes as well.

The `CREATE` statement in the following example specifies that files to which the rule applies should be created on the `tier1` volume if space is available, and on one of the `tier2` volumes if not. If space allocation on `tier1` and `tier2` volumes is not possible, file creation fails, even if space is available on `tier3` volumes.

```
<CREATE>
  <ON>
    <DESTINATION>
      <CLASS>tier1</CLASS>
    </DESTINATION>
    <DESTINATION>
```

```
      <CLASS>tier2</CLASS>
      <BALANCE_SIZE Units="MB">1</BALANCE_SIZE>
    </DESTINATION>
  </ON>
</CREATE>
```

The <BALANCE_SIZE> element with a value of one megabyte is specified for allocations on tier2 volumes. For files allocated on tier2 volumes, the first megabyte would be allocated on the first volume, the second on the second volume, and so forth.

RELOCATE statement

The RELOCATE action statement of file placement policy rules specifies an action that VxFS takes on designated files during periodic scans of the file system, and the circumstances under which the actions should be taken. The fsppadm enforce command is used to scan all or part of a file system for files that should be relocated based on rules in the active placement policy at the time of the scan.

See the fsppadm(1M) manual page.

The fsppadm enforce command scans file systems in path name order. For each file, VxFS identifies the first applicable rule in the active placement policy, as determined by the rules' SELECT statements. If the file resides on a volume specified in the <FROM> clause of one of the rule's RELOCATE statements, and if the file meets the criteria for relocation specified in the statement's <WHEN> clause, the file is scheduled for relocation to a volume in the first placement class listed in the <TO> clause that has space available for the file. The scan that results from issuing the fsppadm enforce command runs to completion before any files are relocated.

The following XML snippet illustrates the general form of the RELOCATE statement:

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS> placement_class_name </CLASS>
    </SOURCE>
    <SOURCE> additional_placement_class_specifications
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS> placement_class_name </CLASS>
      <BALANCE_SIZE Units="units_specifier">
        chunk_size
      </BALANCE_SIZE>
    </DESTINATION>
    <DESTINATION>
      additional_placement_class_specifications
    </DESTINATION>
  </TO>
  <WHEN> relocation_conditions </WHEN>
</RELOCATE>
```

A RELOCATE statement contains the following clauses:

<FROM>

An optional clause that contains a list of placement classes from whose volumes designated files should be relocated if the files meet the conditions specified in the <WHEN> clause. No priority is associated with the ordering of placement classes listed in a <FROM> clause. If a file to which the rule applies is located on a volume in any specified placement class, the file is considered for relocation.

If a RELOCATE statement contains a <FROM> clause, VxFS only considers files that reside on volumes in placement classes specified in the clause for relocation. If no <FROM> clause is present, qualifying files are relocated regardless of where the files reside.

<TO>

Indicates the placement classes to which qualifying files should be relocated. Unlike the source placement class list in a FROM clause, placement classes in a <TO> clause are specified in priority order. Files are relocated to volumes in the first specified placement class if possible, to the second if not, and so forth.

The <TO> clause of the RELOCATE statement contains a list of <DESTINATION> XML elements specifying placement classes to whose volumes VxFS relocates qualifying files. Placement classes are specified in priority order. VxFS relocates qualifying files to volumes in the first placement class specified as long as space is available. A <DESTINATION> element may contain an optional <BALANCE_SIZE> modifier sub-element. The <BALANCE_SIZE>modifier indicates that relocated files should be distributed across the volumes of the destination placement class in chunks of the indicated size. For example, if a balance size of one megabyte is specified for a placement class containing three volumes, VxFS relocates the first megabyte the file to the first (lowest indexed) volume in the class, the second megabyte to the second volume, the third megabyte to the third volume, the fourth megabyte to the first volume, and so forth. Using the Units attribute in the <BALANCE_SIZE> XML tag, the chunk value may be specified in the balance size value may be specified in bytes (Units="bytes"), kilobytes (Units="KB"), megabytes (Units="MB"), or gigabytes (Units="GB").

The <BALANCE_SIZE> element distributes the allocation of database files across the volumes in a placement class. In principle, distributing the data in each file across multiple volumes distributes the I/O load across the volumes as well.

<WHEN>

An optional clause that indicates the conditions under which files to which the rule applies should be relocated. Files that have been unaccessed or unmodified for a specified period, reached a certain size, or reached a specific I/O temperature or access temperature level may be relocated. If a RELOCATE statement does not contain a <WHEN> clause, files to which the rule applies are relocated unconditionally.

A <WHEN> clause may be included in a RELOCATE statement to specify that files should be relocated only if any or all of four types of criteria are met. Files can be specified for relocation if they satisfy one or more criteria.

The following are the criteria that can be specified for the <WHEN> clause:

<ACCAGE>

This criterion is met when files are inactive for a designated period or during a designated period relative to the time at which the fsppadm enforce command was issued.

`<MODAGE>`	This criterion is met when files are unmodified for a designated period or during a designated period relative to the time at which the `fsppadm enforce` command was issued.
`<SIZE>`	This criterion is met when files exceed or drop below a designated size or fall within a designated size range.
`<IOTEMP>`	This criterion is met when files exceed or drop below a designated I/O temperature, or fall within a designated I/O temperature range. A file's I/O temperature is a measure of the I/O activity against it during the period designated by the `<PERIOD>`element prior to the time at which the `fsppadm enforce` command was issued. See "Calculating I/O temperature and access temperature" on page 207.
`<ACCESSTEMP>`	This criterion is met when files exceed or drop below a specified average access temperature, or fall within a specified access temperature range. A file's access temperature is similar to its I/O temperature, except that access temperature is computed using the number of I/O requests to the file, rather than the number of bytes transferred.

Note: The use of `<IOTEMP>` and `<ACCESSTEMP>` for data placement on VxFS servers that are used as NFS servers may not be very effective due to NFS caching. NFS client side caching and the way that NFS works can result in I/O initiated from an NFS client not producing NFS server side I/O. As such, any temperature measurements in place on the server side will not correctly reflect the I/O behavior that is specified by the placement policy.

If the server is solely used as an NFS server, this problem can potentially be mitigated by suitably adjusting or lowering the temperature thresholds. However, adjusting the thresholds may not always create the desired effect. In addition, if the same mount point is used both as an NFS export as well as a local mount, the temperature-based placement decisions will not be very effective due to the NFS cache skew.

The following XML snippet illustrates the general form of the `<WHEN>` clause in a `RELOCATE` statement:

```
<WHEN>
  <ACCAGE Units="units_value">
    <MIN Flags="comparison_operator">
      min_access_age</MIN>
    <MAX Flags="comparison_operator">
```

```
      max_access_age</MAX>
  </ACCAGE>
  <MODAGE Units="units_value">
    <MIN Flags="comparison_operator">
      min_modification_age</MIN>
    <MAX Flags="comparison_operator">
      max_modification_age</MAX>
  </MODAGE>
  <SIZE " Units="units_value">
    <MIN Flags="comparison_operator">
      min_size</MIN>
    <MAX Flags="comparison_operator">
      max_size</MAX>
  </SIZE>
  <IOTEMP Type="read_write_preference" Prefer="temperature_preference">
    <MIN Flags="comparison_operator">
      min_I/O_temperature</MIN>
    <MAX Flags="comparison_operator">
      max_I/O_temperature</MAX>
    <PERIOD Units="days_or_hours"> days_or_hours_of_interest </PERIOD>
  </IOTEMP>
  <ACCESSTEMP Type="read_write_preference"
   Prefer="temperature_preference">
    <MIN Flags="comparison_operator">
      min_access_temperature</MIN>
    <MAX Flags="comparison_operator">
      max_access_temperature</MAX>
    <PERIOD Units="days_or_hours"> days_or_hours_of_interest </PERIOD>
  </ACCESSTEMP>
</WHEN>
```

The access age (`<ACCAGE>`) element refers to the amount of time since a file was last accessed. VxFS computes access age by subtracting a file's time of last access, atime, from the time when the `fsppadm enforce` command was issued. The `<MIN>` and `<MAX>` XML elements in an `<ACCAGE>` clause, denote the minimum and maximum access age thresholds for relocation, respectively. These elements are optional, but at least one must be included. Using the `Units` XML attribute, the `<MIN>` and `<MAX>` elements may be specified in the following units:

`hours`	Hours
`days`	Days. A day is considered to be 24 hours prior to the time that the `fsppadm enforce` command was issued.

Both the <MIN> and <MAX> elements require Flags attributes to direct their operation.

For <MIN>, the following Flags attributes values may be specified:

gt	The time of last access must be greater than the specified interval.
eq	The time of last access must be equal to the specified interval.
gteq	The time of last access must be greater than or equal to the specified interval.

For <MAX>, the following Flags attributes values may be specified.

lt	The time of last access must be less than the specified interval.
lteq	The time of last access must be less than or equal to the specified interval.

Including a <MIN> element in a <WHEN> clause causes VxFS to relocate files to which the rule applies that have been inactive for longer than the specified interval. Such a rule would typically be used to relocate inactive files to less expensive storage tiers. Conversely, including <MAX> causes files accessed within the specified interval to be relocated. It would typically be used to move inactive files against which activity had recommenced to higher performance or more reliable storage. Including both <MIN> and <MAX> causes VxFS to relocate files whose access age lies between the two.

The modification age relocation criterion, <MODAGE>, is similar to access age, except that files' POSIX mtime values are used in computations. You would typically specify the <MODAGE> criterion to cause relocation of recently modified files to higher performance or more reliable storage tiers in anticipation that the files would be accessed recurrently in the near future.

The file size relocation criterion, <SIZE>, causes files to be relocated if the files are larger or smaller than the values specified in the <MIN> and <MAX> relocation criteria, respectively, at the time that the fsppadm enforce command was issued. Specifying both criteria causes VxFS to schedule relocation for files whose sizes lie between the two. Using the Units attribute, threshold file sizes may be specified in the following units:

bytes	Bytes
KB	Kilobytes
MB	Megabytes

GB	Gigabytes

Specifying the I/O temperature relocation criterion

The I/O temperature relocation criterion, <IOTEMP>, causes files to be relocated if their I/O temperatures rise above or drop below specified values over a specified period immediately prior to the time at which the fsppadm enforce command was issued. A file's I/O temperature is a measure of the read, write, or total I/O activity against it normalized to the file's size. Higher I/O temperatures indicate higher levels of application activity; lower temperatures indicate lower levels. VxFS computes a file's I/O temperature by dividing the number of bytes transferred to or from it (read, written, or both) during the specified period by its size at the time that the fsppadm enforce command was issued.

See "Calculating I/O temperature and access temperature" on page 207.

As with the other file relocation criteria, <IOTEMP> may be specified with a lower threshold by using the <MIN> element, an upper threshold by using the <MAX> element, or as a range by using both. However, I/O temperature is dimensionless and therefore has no specification for units.

VxFS computes files' I/O temperatures over the period between the time when the fsppadm enforce command was issued and the number of days or hours in the past specified in the <PERIOD> element, where a day is a 24 hour period. The default unit of time is days. You can specify hours as the time unit by setting the Units attribute of the <PERIOD> element to hours. Symantec recommends that you specify hours only if you are using solid state disks (SSDs).

See "Frequent scans" on page 218.

For example, if you issued the fsppadm enforce command at 2 PM on Wednesday and you want VxFS to look at file I/O activity for the period between 2 PM on Monday and 2 PM on Wednesday, which is a period of 2 days, you would specify the following <PERIOD> element:

```
<PERIOD> 2 </PERIOD>
```

If you instead want VxFS to look at file I/O activity between 3 hours prior to running the fsppadm enforce command and the time that you ran the command, you specify the following <PERIOD> element:

```
<PERIOD Units="hours"> 3 </PERIOD>
```

The amount of time specified in the <PERIOD> element should not exceed one or two weeks due to the disk space used by the File Change Log (FCL) file.

I/O temperature is a softer measure of I/O activity than access age. With access age, a single access to a file resets the file's atime to the current time. In contrast, a file's I/O temperature decreases gradually as time passes without the file being accessed, and increases gradually as the file is accessed periodically. For example, if a new 10 megabyte file is read completely five times on Monday and `fsppadm enforce` runs at midnight, the file's two-day I/O temperature will be five and its access age in days will be zero. If the file is read once on Tuesday, the file's access age in days at midnight will be zero, and its two-day I/O temperature will have dropped to three. If the file is read once on Wednesday, the file's access age at midnight will still be zero, but its two-day I/O temperature will have dropped to one, as the influence of Monday's I/O will have disappeared.

If the intention of a file placement policy is to keep files in place, such as on top-tier storage devices, as long as the files are being accessed at all, then access age is the more appropriate relocation criterion. However, if the intention is to relocate files as the I/O load on them decreases, then I/O temperature is more appropriate.

The case for upward relocation is similar. If files that have been relocated to lower-tier storage devices due to infrequent access experience renewed application activity, then it may be appropriate to relocate those files to top-tier devices. A policy rule that uses access age with a low `<MAX>` value, that is, the interval between `fsppadm enforce` runs, as a relocation criterion will cause files to be relocated that have been accessed even once during the interval. Conversely, a policy that uses I/O temperature with a `<MIN>` value will only relocate files that have experienced a sustained level of activity over the period of interest.

Prefer attribute

You can specify a value for the `Prefer` attribute for the `<IOTEMP>` and `<ACCESSTEMP>` criteria, which gives preference to relocating files. The `Prefer` attribute can take two values: `low` or `high`. If you specify `low`, Veritas File System (VxFS) relocates the files with the lower I/O temperature before relocating the files with the higher I/O temperature. If you specify `high`, VxFS relocates the files with the higher I/O temperature before relocating the files with the lower I/O temperature. Symantec recommends that you specify a `Prefer` attribute value only if you are using solid state disks (SSDs).

See “Prefer mechanism” on page 217.

Different `<PERIOD>` elements may be used in the `<IOTEMP>` and `<ACCESSTEMP>` criteria of different `RELOCATE` statements within the same policy.

The following placement policy snippet gives an example of the `Prefer` criteria:

```
<RELOCATE>
    ...
```

```
      <WHEN>
         <IOTEMP Type="nrbytes" Prefer="high">
            <MIN Flags="gteq"> 3.4 </MIN>
            <PERIOD Units="hours"> 6 </PERIOD>
         </IOTEMP>
      </WHEN>
   </RELOCATE>
```

If there are a number of files whose I/O temperature is greater than the given minimum value, the files with the higher temperature are first subject to the RELOCATE operation before the files with the lower temperature.

Average I/O activity

The Average criteria allows you to specify the value of the I/O temperature as a ratio of per-file activity that occurs over the time specified by the <PERIOD> element compared to the overall file system activity that occurs over a longer period of time. The <PERIOD> element in the RELOCATE criteria specifies the a number of hours or days immediately before the time of the scan. During that time, the I/O statistics that are collected are used to process the files that are being scanned. Since I/O activity can change over time, collect the average I/O activity over a longer duration than the <PERIOD> value itself, which is by default 24 hours. Doing so lets you compute an average temperature of the whole file system. Symantec recommends that you specify an Average attribute value only if you are using solid state disks (SSDs).

See "Average I/O activity" on page 218.

The following placement policy snippet gives an example of the Average criteria:

```
   <RELOCATE>
      ...
      <WHEN>
         <IOTEMP Type="nrbytes" Prefer="high" Average="*">
            <MIN Flags="gteq"> 1.5 </MIN>
            <PERIOD Units="hours"> 6 </PERIOD>
         </IOTEMP>
      </WHEN>
   </RELOCATE>
```

In the snippet, VxFS relocates any file whose read IOTEMP over the last 6 hours is 1.5 times that of all the active files in the whole file system over the last 24 hours. This Average criteria is more intuitive and easier to specify than the absolute values.

The following formula computes the read IOTEMP of a given file:

```
IOTEMP = (bytes of the file that are read in the PERIOD) /
         (PERIOD in hours * size of the file in bytes)
```

The write and read/write IOTEMP are also computed accordingly.

The following formula computes the average read IOTEMP:

```
Average IOTEMP = (bytes read of all active files in the last h hours) /
                 (h * size of all the active files in bytes)
```

h is 24 hours by default. The average write and read/write IOTEMP are also computed accordingly.

In the example snippet, the value 1.5 is the multiple of average read IOTEMP over the last 24 hours across the whole file system, or rather across all of the active inodes whose activity is still available in the File Change Log (FCL) file at the time of the scan. Thus, the files' read IOTEMP activity over the last 6 hours is compared against 1.5 times that of the last 24 hours average activity to make the relocation decision. Using this method eliminates the need to give a specific number for the `<IOTEMP>` or `<ACCESSTEMP>` criteria, and instead lets you specify a multiple of the Average temperature. Keeping this averaging period longer than the specified `<PERIOD>` value normalizes the effects of any spikes and lulls in the file activity.

You can also use the `Average` criteria with the `<ACCESSTEMP>` criteria. The purpose and usage are the same.

You determine the type of the average by whether you specify the `Average` criteria with the `<IOTEMP>` or with the `<ACCESSTEMP>` criteria. The `Average` criteria can be any of the following types, depending on the criteria used:

- read Average IOTEMP
- write Average IOTEMP
- rw Average IOTEMP
- read Average ACCESSTEMP
- write Average ACCESSTEMP
- rw Average ACCESSTEMP

The default `Average` is a 24 hour average temperature, which is the total of all of the temperatures available up to the last 24 hours in the FCL file, divided by the number of files for which such I/O statistics still exist in the FCL file. You can override the number of hours by specifying the `AveragePeriod` attribute in the `<PLACEMENT_POLICY>` element. Symantec recommends that you specify an `AveragePeriod` attribute value only if you are using solid state disks (SSDs).

The following example statement causes the average file system activity be collected and computed over a period of 30 hours instead of the default 24 hours:

```
<PLACEMENT_POLICY Name="Policy1" Version="5.1" AveragePeriod="30">
```

RELOCATE statement examples

The following example illustrates an unconditional relocation statement, which is the simplest form of the RELOCATE policy rule statement:

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier1</CLASS>
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
  </TO>
</RELOCATE>
```

The files designated by the rule's SELECT statement that reside on volumes in placement class tier1 at the time the fsppadm enforce command executes would be unconditionally relocated to volumes in placement class tier2 as long as space permitted. This type of rule might be used, for example, with applications that create and access new files but seldom access existing files once they have been processed. A CREATE statement would specify creation on tier1 volumes, which are presumably high performance or high availability, or both. Each instantiation of fsppadm enforce would relocate files created since the last run to tier2 volumes.

The following example illustrates a more comprehensive form of the RELOCATE statement that uses access age as the criterion for relocating files from tier1 volumes to tier2 volumes. This rule is designed to maintain free space on tier1 volumes by relocating inactive files to tier2 volumes:

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier1</CLASS>
    </SOURCE>
  </FROM>
  <TO>
```

```
    <DESTINATION>
      <CLASS>tier2</CLASS>
     </DESTINATION>
  </TO>
  <WHEN>
    <SIZE Units="MB">
      <MIN Flags="gt">1</MIN>
      <MAX Flags="lt">1000</MAX>
    </SIZE>
    <ACCAGE Units="days">
      <MIN Flags="gt">30</MIN>
    </ACCAGE>
  </WHEN>
</RELOCATE>
```

Files designated by the rule's `SELECT` statement are relocated from `tier1` volumes to `tier2` volumes if they are between 1 MB and 1000 MB in size and have not been accessed for 30 days. VxFS relocates qualifying files in the order in which it encounters them as it scans the file system's directory tree. VxFS stops scheduling qualifying files for relocation when when it calculates that already-scheduled relocations would result in `tier2` volumes being fully occupied.

The following example illustrates a possible companion rule that relocates files from `tier2` volumes to `tier1` ones based on their I/O temperatures. This rule might be used to return files that had been relocated to `tier2` volumes due to inactivity to `tier1` volumes when application activity against them increases. Using I/O temperature rather than access age as the relocation criterion reduces the chance of relocating files that are not actually being used frequently by applications. This rule does not cause files to be relocated unless there is sustained activity against them over the most recent two-day period.

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier2</CLASS>
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS>tier1</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <IOTEMP Type="nrbytes">
```

```
      <MIN Flags="gt">5</MIN>
      <PERIOD>2</PERIOD>
    </IOTEMP>
  </WHEN>
</RELOCATE>
```

This rule relocates files that reside on `tier2` volumes to `tier1` volumes if their I/O temperatures are above 5 for the two day period immediately preceding the issuing of the `fsppadm enforce` command. VxFS relocates qualifying files in the order in which it encounters them during its file system directory tree scan. When `tier1` volumes are fully occupied, VxFS stops scheduling qualifying files for relocation.

VxFS file placement policies are able to control file placement across any number of placement classes. The following example illustrates a rule for relocating files with low I/O temperatures from `tier1` volumes to `tier2` volumes, and to `tier3` volumes when `tier2` volumes are fully occupied:

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier1</CLASS>
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
    <DESTINATION>
      <CLASS>tier3</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <IOTEMP Type="nrbytes">
      <MAX Flags="lt">4</MAX>
      <PERIOD>3</PERIOD>
    </IOTEMP>
  </WHEN>
</RELOCATE>
```

This rule relocates files whose 3-day I/O temperatures are less than 4 and which reside on `tier1` volumes. When VxFS calculates that already-relocated files would result in `tier2` volumes being fully occupied, VxFS relocates qualifying files to tier3 volumes instead. VxFS relocates qualifying files as it encounters them in its scan of the file system directory tree.

The <FROM> clause in the RELOCATE statement is optional. If the clause is not present, VxFS evaluates files designated by the rule's SELECT statement for relocation no matter which volumes they reside on when the fsppadm enforce command is issued. The following example illustrates a fragment of a policy rule that relocates files according to their sizes, no matter where they reside when the fsppadm enforce command is issued:

```
<RELOCATE>
  <TO>
    <DESTINATION>
      <CLASS>tier1</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <SIZE Units="MB">
      <MAX Flags="lt">10</MAX>
    </SIZE>
  </WHEN>
</RELOCATE>
<RELOCATE>
  <TO>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <SIZE Units="MB">
      <MIN Flags="gteq">10</MIN>
      <MAX Flags="lt">100</MAX>
    </SIZE>
  </WHEN>
</RELOCATE>
<RELOCATE>
  <TO>
    <DESTINATION>
      <CLASS>tier3</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <SIZE Units="MB">
      <MIN Flags="gteq">100</MIN>
    </SIZE>
  </WHEN>
</RELOCATE>
```

This rule relocates files smaller than 10 megabytes to tier1 volumes, files between 10 and 100 megabytes to tier2 volumes, and files larger than 100 megabytes to tier3 volumes. VxFS relocates all qualifying files that do not already reside on volumes in their DESTINATION placement classes when the fsppadm enforce command is issued.

DELETE statement

The DELETE file placement policy rule statement is very similar to the RELOCATE statement in both form and function, lacking only the <TO> clause. File placement policy-based deletion may be thought of as relocation with a fixed destination.

Note: Use DELETE statements with caution.

The following XML snippet illustrates the general form of the DELETE statement:

```
<DELETE>
  <FROM>
    <SOURCE>
      <CLASS> placement_class_name </CLASS>
    </SOURCE>
    <SOURCE>
      additional_placement_class_specifications
    </SOURCE>
  </FROM>
  <WHEN> relocation_conditions </WHEN>
</DELETE>
```

A DELETE statement contains the following clauses:

<FROM> An optional clause that contains a list of placement classes from whose volumes designated files should be deleted if the files meet the conditions specified in the <WHEN> clause. No priority is associated with the ordering of placement classes in a <FROM> clause. If a file to which the rule applies is located on a volume in any specified placement class, the file is deleted. If a DELETE statement does not contain a <FROM> clause, VxFS deletes qualifying files no matter on which of a file system's volumes the files reside.

<WHEN> An optional clause specifying the conditions under which files to which the rule applies should be deleted. The form of the <WHEN> clause in a DELETE statement is identical to that of the <WHEN> clause in a RELOCATE statement. If a DELETE statement does not contain a <WHEN> clause, files designated by the rule's SELECT statement, and the <FROM> clause if it is present, are deleted unconditionally.

DELETE statement examples

The following example illustrates the use of the DELETE statement:

```
<DELETE>
  <FROM>
    <SOURCE>
      <CLASS>tier3</CLASS>
    </SOURCE>
  </FROM>
</DELETE>
<DELETE>
  <FROM>
    <SOURCE>
      <CLASS>tier2</CLASS>
    </SOURCE>
  </FROM>
  <WHEN>
    <ACCAGE Units="days">
      <MIN Flags="gt">120</MIN>
    </ACCAGE>
  </WHEN>
</DELETE>
```

The first DELETE statement unconditionally deletes files designated by the rule's SELECT statement that reside on tier3 volumes when the fsppadm enforce command is issued. The absence of a <WHEN> clause in the DELETE statement indicates that deletion of designated files is unconditional.

The second DELETE statement deletes files to which the rule applies that reside on tier2 volumes when the fsppadm enforce command is issued and that have not been accessed for the past 120 days.

Calculating I/O temperature and access temperature

An important application of VxFS Dynamic Storage Tiering is automating the relocation of inactive files to lower cost storage. If a file has not been accessed for the period of time specified in the `<ACCAGE>` element, a scan of the file system should schedule the file for relocation to a lower tier of storage. But, time since last access is inadequate as the only criterion for activity-based relocation.

Why time since last access is inadequate as the only criterion for activity-based relocation:

- Access age is a binary measure. The time since last access of a file is computed by subtracting the time at which the `fsppadm enforce` command is issued from the POSIX `atime` in the file's metadata. If a file is opened the day before the `fsppadm enforce` command, its time since last access is one day, even though it may have been inactive for the month preceding. If the intent of a policy rule is to relocate inactive files to lower tier volumes, it will perform badly against files that happen to be accessed, however casually, within the interval defined by the value of the `<ACCAGE>` pa-rameter.
- Access age is a poor indicator of resumption of significant activity. Using `ACCAGE`, the time since last access, as a criterion for relocating inactive files to lower tier volumes may fail to schedule some relocations that should be performed, but at least this method results in less relocat--ion activity than necessary. Using `ACCAGE` as a criterion for relocating previously inactive files that have become active is worse, because this method is likely to schedule relocation activity that is not warranted. If a policy rule's intent is to cause files that have experienced I/O activity in the recent past to be relocated to higher performing, perhaps more failure tolerant storage, `ACCAGE` is too coarse a filter. For example, in a rule specifying that files on `tier2` volumes that have been accessed within the last three days should be relocated to `tier1` volumes, no distinction is made between a file that was browsed by a single user and a file that actually was used intensively by applications.

DST implements the concept of I/O temperature and access temperature to overcome these deficiencies. A file's I/O temperature is equal to the number of bytes transferred to or from it over a specified period of time divided by the size of the file. For example, if a file occupies one megabyte of storage at the time of an `fsppadm enforce` operation and the data in the file has been completely read or written 15 times within the last three days, VxFS calculates its 3-day average I/O temperature to be 5 (15 MB of I/O ÷ 1 MB file size ÷ 3 days).

Similarly, a file's average access temperature is the number of read or write requests made to it over a specified number of 24-hour periods divided by the number of periods. Unlike I/O temperature, access temperature is unrelated to file size. A large file to which 20 I/O requests are made over a 2-day period has the same average access temperature as a small file accessed 20 times over a 2-day period.

If a file system's active placement policy includes any `<IOTEMP>` or `<ACCESSTEMP>` clauses, VxFS begins policy enforcement by using information in the file system's FCL file to calculate average I/O activity against all files in the file system during the longest `<PERIOD>` specified in the policy. Shorter specified periods are ignored. VxFS uses these calculations to qualify files for I/O temperature-based relocation and deletion.

Note: If FCL is turned off, I/O temperature-based relocation will not be accurate. When you invoke the `fsppadm enforce` command, the command displays a warning if the FCL is turned off.

As its name implies, the File Change Log records information about changes made to files in a VxFS file system. In addition to recording creations, deletions, extensions, the FCL periodically captures the cumulative amount of I/O activity (number of bytes read and written) on a file-by-file basis. File I/O activity is recorded in the FCL each time a file is opened or closed, as well as at timed intervals to capture information about files that remain open for long periods.

If a file system's active file placement policy contains `<IOTEMP>` clauses, execution of the `fsppadm enforce` command begins with a scan of the FCL to extract I/O activity information over the period of interest for the policy. The period of interest is the interval between the time at which the `fsppadm enforce` command was issued and that time minus the largest interval value specified in any `<PERIOD>` element in the active policy.

For files with I/O activity during the largest interval, VxFS computes an approximation of the amount of read, write, and total data transfer (the sum of the two) activity by subtracting the I/O levels in the oldest FCL record that pertains to the file from those in the newest. It then computes each file's I/O temperature by dividing its I/O activity by its size at `Tscan`. Dividing by file size is an implicit acknowledgement that relocating larger files consumes more I/O resources than relocating smaller ones. Using this algorithm requires that larger files must have more activity against them in order to reach a given I/O temperature, and thereby justify the resource cost of relocation.

While this computation is an approximation in several ways, it represents an easy to compute, and more importantly, unbiased estimate of relative recent I/O activity upon which reasonable relocation decisions can be based.

File relocation and deletion decisions can be based on read, write, or total I/O activity.

The following XML snippet illustrates the use of IOTEMP in a policy rule to specify relocation of low activity files from tier1 volumes to tier2 volumes:

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier1</CLASS>
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <IOTEMP Type="nrbytes">
      <MAX Flags="lt">3</MAX>
      <PERIOD Units="days">4</PERIOD>
    </IOTEMP>
  </WHEN>
</RELOCATE>
```

This snippet specifies that files to which the rule applies should be relocated from tier1 volumes to tier2 volumes if their I/O temperatures fall below 3 over a period of 4 days. The Type="nrbytes" XML attribute specifies that total data transfer activity, which is the the sum of bytes read and bytes written, should be used in the computation. For example, a 50 megabyte file that experienced less than 150 megabytes of data transfer over the 4-day period immediately preceding the fsppadm enforce scan would be a candidate for relocation. VxFS considers files that experience no activity over the period of interest to have an I/O temperature of zero. VxFS relocates qualifying files in the order in which it encounters the files in its scan of the file system directory tree.

Using I/O temperature or access temperature rather than a binary indication of activity, such as the POSIX atime or mtime, minimizes the chance of not relocating files that were only accessed occasionally during the period of interest. A large file that has had only a few bytes transferred to or from it would have a low I/O temperature, and would therefore be a candidate for relocation to tier2 volumes, even if the activity was very recent.

But, the greater value of I/O temperature or access temperature as a file relocation criterion lies in upward relocation: detecting increasing levels of I/O activity against files that had previously been relocated to lower tiers in a storage hierarchy due to inactivity or low temperatures, and relocating them to higher tiers in the storage hierarchy.

The following XML snippet illustrates relocating files from tier2 volumes to tier1 when the activity level against them increases.

```
<RELOCATE>
  <FROM>
    <SOURCE>
      <CLASS>tier2</CLASS>
    </SOURCE>
  </FROM>
  <TO>
    <DESTINATION>
      <CLASS>tier1</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <IOTEMP Type="nrbytes">
      <MAX Flags="gt">5</MAX>
      <PERIOD Units="days">2</PERIOD>
    </IOTEMP>
  </WHEN>
</RELOCATE>
```

The <RELOCATE> statement specifies that files on tier2 volumes whose I/O temperature as calculated using the number of bytes read is above 5 over a 2-day period are to be relocated to tier1 volumes. Bytes written to the file during the period of interest are not part of this calculation.

Using I/O temperature rather than a binary indicator of activity as a criterion for file relocation gives administrators a granular level of control over automated file relocation that can be used to attune policies to application requirements. For example, specifying a large value in the <PERIOD> element of an upward relocation statement prevents files from being relocated unless I/O activity against them is sustained. Alternatively, specifying a high temperature and a short period tends to relocate files based on short-term intensity of I/O activity against them.

I/O temperature and access temperature utilize the sqlite3 database for building a temporary table indexed on an inode. This temporary table is used to filter files based on I/O temperature and access temperature. The temporary table is stored in the database file .__fsppadm_fcliotemp.db, which resides in the lost+found directory of the mount point.

Multiple criteria in file placement policy rule statements

In certain cases, file placement policy rule statements may contain multiple clauses that affect their behavior. In general, when a rule statement contains multiple clauses of a given type, all clauses must be satisfied in order for the statement to be effective. There are four cases of note in which multiple clauses may be used.

Multiple file selection criteria in SELECT statement clauses

Within a single `SELECT` statement, all the selection criteria clauses of a single type are treated as a selection list. A file need only satisfy a single criterion of a given type to be designated.

In the following example, files in any of the `db/datafiles`, `db/indexes`, and `db/logs` directories, all relative to the file system mount point, would be selected:

```
<SELECT>
  <DIRECTORY Flags="nonrecursive">db/datafiles</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/indexes</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/logs</DIRECTORY>
</SELECT>
```

This example is in direct contrast to the treatment of selection criteria clauses of different types. When a `SELECT` statement includes multiple types of file selection criteria, a file must satisfy one criterion of each type in order for the rule's action statements to apply.

In the following example, a file must reside in one of `db/datafiles`, `db/indexes`, or `db/logs` and be owned by one of `DBA_Manager`, `MFG_DBA`, or `HR_DBA` to be designated for possible action:

```
<SELECT>
  <DIRECTORY Flags="nonrecursive">db/datafiles</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/indexes</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/logs</DIRECTORY>
  <USER>DBA_Manager</USER>
  <USER>MFG_DBA</USER>
  <USER>HR_DBA</USER>
</SELECT>
```

If a rule includes multiple `SELECT` statements, a file need only satisfy one of them to be selected for action. This property can be used to specify alternative conditions for file selection.

In the following example, a file need only reside in one of db/datafiles, db/indexes, or db/logs or be owned by one of DBA_Manager, MFG_DBA, or HR_DBA to be designated for possible action:

```
<SELECT>
  <DIRECTORY Flags="nonrecursive">db/datafiles</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/indexes</DIRECTORY>
  <DIRECTORY Flags="nonrecursive">db/logs</DIRECTORY>
</SELECT>
<SELECT>
  <USER>DBA_Manager</USER>
  <USER>MFG_DBA</USER>
  <USER>HR_DBA</USER>
</SELECT>
```

Multiple placement classes in <ON> clauses of CREATE statements and in <TO> clauses of RELOCATE statements

Both the <ON> clause of the CREATE statement and the <TO> clause of the RELOCATE statement can specify priority ordered lists of placement classes using multiple <DESTINATION> XML elements. VxFS uses a volume in the first placement class in a list for the designated purpose of file creation or relocation, if possible. If no volume in the first listed class has sufficient free space or if the file system's volume set does not contain any volumes with that placement class, VxFS uses a volume in the second listed class if possible. If no volume in the second listed class can be used, a volume in the third listed class is used if possible, and so forth.

The following example illustrates of three placement classes specified in the <ON> clause of a CREATE statement:

```
<CREATE>
  <ON>
    <DESTINATION>
      <CLASS>tier1</CLASS>
    </DESTINATION>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
    <DESTINATION>
      <CLASS>tier3</CLASS>
    </DESTINATION>
  </ON>
</CREATE>
```

In this statement, VxFS would allocate space for newly created files designated by the rule's SELECT statement on tier1 volumes if space was available. If no tier1 volume had sufficient free space, VxFS would attempt to allocate space on a tier2 volume. If no tier2 volume had sufficient free space, VxFS would attempt allocation on a tier3 volume. If sufficient space could not be allocated on a volume in any of the three specified placement classes, allocation would fail with an ENOSPC error, even if the file system's volume set included volumes in other placement classes that did have sufficient space.

The <TO> clause in the RELOCATE statement behaves similarly. VxFS relocates qualifying files to volumes in the first placement class specified if possible, to volumes in the second specified class if not, and so forth. If none of the destination criteria can be met, such as if all specified classes are fully occupied, qualifying files are not relocated, but no error is signaled in this case.

Multiple placement classes in <FROM> clauses of RELOCATE and DELETE statements

The <FROM> clause in RELOCATE and DELETE statements can include multiple source placement classes. However, unlike the <ON> and <TO> clauses, no order or priority is implied in <FROM> clauses. If a qualifying file resides on a volume in any of the placement classes specified in a <FROM> clause, it is relocated or deleted regardless of the position of its placement class in the <FROM> clause list of classes.

Multiple conditions in <WHEN> clauses of RELOCATE and DELETE statements

The <WHEN> clause in RELOCATE and DELETE statements may include multiple relocation criteria. Any or all of <ACCAGE>, <MODAGE>, <SIZE>, and <IOTEMP> can be specified. When multiple conditions are specified, all must be satisfied in order for a selected file to qualify for relocation or deletion.

In the following example, a selected file would have to be both inactive, that is, not accessed, for more than 30 days and larger than 100 megabytes to be eligible for relocation or deletion:

```
<WHEN>
  <ACCAGE Units="days">
    <MIN Flags="gt">30</MIN>
  </ACCAGE>
  <SIZE Units="MB">
    <MIN Flags="gt">100</MIN>
  </SIZE>
</WHEN>
```

You cannot write rules to relocate or delete a single designated set of files if the files meet one of two or more relocation or deletion criteria.

File placement policy rule and statement ordering

You can use the Dynamic Storage Tiering graphical user interface (GUI) to create any of four types of file placement policy documents. Alternatively, you can use a text editor or XML editor to create XML policy documents directly. The GUI places policy rule statements in the correct order to achieve the desired behavior. If you use a text editor, it is your responsibility to order policy rules and the statements in them so that the desired behavior results.

The rules that comprise a placement policy may occur in any order, but during both file allocation and `fsppadm enforce` relocation scans, the first rule in which a file is designated by a `SELECT` statement is the only rule against which that file is evaluated. Thus, rules whose purpose is to supersede a generally applicable behavior for a special class of files should precede the general rules in a file placement policy document.

The following XML snippet illustrates faulty rule placement with potentially unintended consequences:

```
<?xml version="1.0"?>
<!DOCTYPE FILE_PLACEMENT_POLICY SYSTEM "placement.dtd">
<FILE_PLACEMENT_POLICY Version="5.0">
  <RULE Name="GeneralRule">
    <SELECT>
      <PATTERN>*</PATTERN>
    </SELECT>
    <CREATE>
      <ON>
        <DESTINATION>
          <CLASS>tier2</CLASS>
        </DESTINATION>
      </ON>
    </CREATE>
    other_statements
  </RULE>
  <RULE Name="DatabaseRule">
    <SELECT>
      <PATTERN>*.db</PATTERN>
    </SELECT>
    <CREATE>
      <ON>
        <DESTINATION>
          <CLASS>tier1</CLASS>
        </DESTINATION>
      </ON>
    </CREATE>
    other_statements
  </RULE>
</FILE_PLACEMENT_POLICY>
```

The GeneralRule rule specifies that all files created in the file system, designated by `<PATTERN>*</PATTERN>`, should be created on `tier2` volumes. The `DatabaseRule` rule specifies that files whose names include an extension of `.db` should be created on `tier1` volumes. The `GeneralRule` rule applies to any file created in the file system, including those with a naming pattern of `*.db`, so the `DatabaseRule` rule will never apply to any file. This fault can be remedied by exchanging the order of the two rules. If the `DatabaseRule` rule occurs first in the policy document, VxFS encounters it first when determining where to new place files whose names follow the pattern `*.db`, and correctly allocates space for them on `tier1` volumes. For files to which the `DatabaseRule` rule does not apply, VxFS continues scanning the policy and allocates space according to the specification in the `CREATE` statement of the `GeneralRule` rule.

A similar consideration applies to statements within a placement policy rule. VxFS processes these statements in order, and stops processing on behalf of a file when it encounters a statement that pertains to the file. This can result in unintended behavior.

The following XML snippet illustrates a `RELOCATE` statement and a `DELETE` statement in a rule that is intended to relocate if the files have not been accessed in 30 days, and delete the files if they have not been accessed in 90 days:

```
<RELOCATE>
  <TO>
    <DESTINATION>
      <CLASS>tier2</CLASS>
    </DESTINATION>
  </TO>
  <WHEN>
    <ACCAGE Units="days">
      <MIN Flags="gt">30</MIN>
    </ACCAGE>
  </WHEN>
</RELOCATE>
<DELETE>
  <WHEN>
    <ACCAGE Units="days">
      <MIN Flags="gt">90</MIN>
    </ACCAGE>
  </WHEN>
</DELETE>
```

As written with the RELOCATE statement preceding the DELETE statement, files will never be deleted, because the <WHEN> clause in the RELOCATE statement applies to all selected files that have not been accessed for at least 30 days. This includes those that have not been accessed for 90 days. VxFS ceases to process a file against a placement policy when it identifies a statement that applies to that file, so the DELETE statement would never occur. This example illustrates the general point that RELOCATE and DELETE statements that specify less inclusive criteria should precede statements that specify more inclusive criteria in a file placement policy document. The GUI automatically produce the correct statement order for the policies it creates.

File placement policies and extending files

In a VxFS file system with an active file placement policy, the placement class on whose volume a file resides is part of its metadata, and is attached when it is created and updated when it is relocated. When an application extends a file, VxFS allocates the incremental space on the volume occupied by the file if possible. If not possible, VxFS allocates the space on another volume in the same placement class. For example, if a file is created on a tier1 volume and later relocated to a tier2 volume, extensions to the file that occur before the relocation have space allocated on a tier1 volume, while those occurring after to the relocation have their space allocated on tier2 volumes. When a file is relocated, all of its allocated space, including the space acquired by extension, is relocated to tier2 volumes in this case.

Using Dynamic Storage Tiering with solid state disks

The Dynamic Storage Tiering (DST) feature has been enhanced with the following placement policy features to support SSD-based tiers:

- Allowance of fine grained temperatures, such as allowing hours as units for the <IOTEMP> and <ACCESSTEMP> criteria

- Support of the Prefer attribute for the <IOTEMP> and <ACCESSTEMP> criteria
- Provision of a mechanism to relocate based on average I/O activity
- Reduction of the intensity and duration of scans to minimize the impact on resources, such as memory, CPU, and I/O bandwidth
- Quick identification of cold files

To gain these benefits, you must modify the existing placement policy as per the latest version of the DTD and assign the policy again. However, existing placement policies continue to function as before. You do not need to update the placement policies if you do not use the new features.

Fine grain temperatures

Before the solid state disk (SSD) enhancements, the Dynamic Storage Tiering (DST) feature computed temperature values on a day granularity. Day granularity is the I/O activity per day over at least one day. As such, the <PERIOD> element had to be in days for the <IOTEMP> and <ACCESSTEMP> criteria. With SSDs, relocation decisions might need to happen within the day itself, based on I/O activity that Veritas File System (VxFS) measured over a shorter duration. As such, you can now specify "hours" for the Units attribute value for the <IOTEMP> and <ACCESSTEMP> criteria.

See "Specifying the I/O temperature relocation criterion" on page 197.

The following placement policy snippet gives an example of specifying 4 hours as the period of time:

```
<RELOCATE>
   ...
   <WHEN>
      <IOTEMP Type="nwbytes">
         <MIN Flags="gteq"> 2 </MIN>
            <PERIOD Units="hours"> 4 </PERIOD>
      </IOTEMP>
   </WHEN>
</RELOCATE>
```

Prefer mechanism

You can now specify a value for the Prefer attribute for the <IOTEMP> and <ACCESSTEMP> criteria, which gives preference to relocating files.

See "Prefer attribute" on page 198.

In case of a solid state disk (SSD)-based tier, you might want to relocate a file to an SSD as soon as there is a marked increase in the I/O activity. However, once Veritas File System (VxFS) has relocated the file to an SSD, it may be beneficial to keep the file on the SSD as long as the activity remains high to avoid frequent thrashing. You want to watch the activity for some time longer than the time that you watched the activity when you relocated the file to the SSD before you decide to move the file off of the SSD.

The following placement policy snippet gives an example of the `Prefer` criteria:

```
<RELOCATE>
   ...
   <WHEN>
      <IOTEMP Type="nrbytes" Prefer="high">
         <MIN Flags="gteq"> 3.4 </MIN>
         <PERIOD Units="hours"> 6 </PERIOD>
      </IOTEMP>
   </WHEN>
</RELOCATE>
```

If there are a number of files whose I/O temperature is greater than the given minimum value, the files with the higher temperature are first subject to the `RELOCATE` operation before the files with the lower temperature. This is particularly useful in case of SSDs, which are limited in size and are expensive. As such, you generally want to use SSDs for the most active files.

Average I/O activity

Before the solid state disk (SSD) enhancements, you were required to specify an absolute value of the temperature when you used the `ACCESSTEMP` criteria and `IOTEMP` criteria in the Dynamic Storage Tiering (DST) placement policies. However, arriving at such absolute numbers is difficult and requires you to experiment and observe data access patterns over a period of time. Moreover, over a period of time, you might have to change this value due to changing access patterns. As such, you might need to repeat the experiment. To ease constructing `ACCESSTEMP` and `IOTEMP`-based policies, a new criteria has been introduced: `Average`.

See “Average I/O activity” on page 199.

Frequent scans

You now can specify "hours" for the `Units` attribute value, and as such the I/O stats collection PERIOD can be much shorter than in previous releases. When not using solid state disks (SSDs), you can only specify "days" for the `Units` attribute value, which might be sufficient for your needs. However, a PERIOD shorter than a day is required in the context of using SSDs since the candidate files and their activity levels can change during the day. As a result, DST must scan more frequently, which leads to a higher scan load on the host systems.

You must satisfy the following conflicting requirements simultaneously:

- Bring down the temperature collection windows to hourly levels.
- Reduce the impact of more frequent scans on resources, such as CPU, I/O, and memory.

The following scheme is an example of one way to reduce the impact of frequent scans:

- Confine the scan to only active files during the PERIOD by focusing only on the files that showed any activity in the File Change Log (FCL) by running the `fsppadm` command with the `-C` option.
  See "Quick identification of cold files" on page 219.
- Scan frequently, such as every few hours. Frequent scans potentially reduce the number of inodes that VxFS touches and logs in the File Change Log (FCL) file, thereby limiting the duration of each scan. As such, the changes that VxFS collects in the FCL file since the last scan provide details on fewer active files.
- Use the `<IOTEMP>` and `<ACCESSTEMP>` criteria to promote files to SSDs more aggressively, which leaves cold files sitting in SSDs.

Quick identification of cold files

The placement mechanism generally leaves the cold files in solid state disks (SSDs) if the files continue to remain inactive. This results in a lack of room for active files if the active files need to be moved into SSDs, and thus results in ineffective use of storage. An SSD enhancement for identifying cold files quickly solves this problem.

The enhancement is a method for quickly identifying files on a particular tier of the Dynamic Storage Tiering (DST) file system so that the files can be relocated if necessary. The method consists of a map that associates storage devices with the inodes of files residing on the storage devices.

Veritas File System (VxFS) updates the file location map during the following times:

- DST's own file relocations
- On examination of the file system's File Change Log (FCL) for changes that are made outside of DST's scope.

Both of these updates occur during DST's relocation scans, which are typically scheduled to occur periodically. But, you can also update the file location map anytime by running the fsppadm command with the -T option.

The -C option is useful to process active files before any other files. For best results, specify the -T option in conjunction with the -C option. Specifying both the -T option and -C option causes the fsppadm command to evacuate any cold files first to create room in the SSD tier to accommodate any active files that will be moved into the SSD tier via the -C option. Specifying -C in conjunction with -T confines the scope of the scan, which consumes less time and resources, and thus allows frequent scans to meet the dynamic needs of data placement.

See "Enforcing a placement policy" on page 182.

See the fsppadm(1M) manual page.

With the help of the map, instead of scanning the full file system, you can confine the scan to only the files on the SSD tiers in addition to the active files that VxFS recorded in the FCL. This scheme potentially achieves the dual purpose of reducing the temperature time granularity and at the same time reducing the scan load.

Example placement policy when using solid state disks

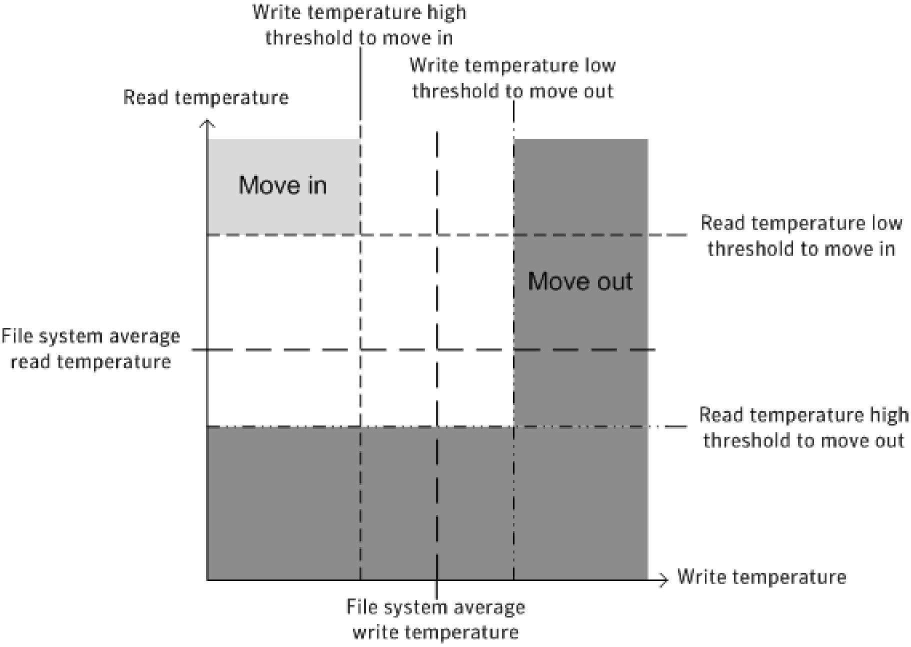
The following snippet is one possible placement policy for use with solid state disk (SSD)-based tiers.

```
<?xml version="1.0"?>
<!DOCTYPE PLACEMENT_POLICY SYSTEM "/opt/VRTSvxfs/etc/placement_policy.dtd">
<PLACEMENT_POLICY Version="5.0" Name="SSD_policy">
  <RULE Flags="data" Name="all_files">
    <COMMENT>
      The first two RELOCATEs will do the evacuation
      out of SSDs to create room for any relocations
      into the SSDs by the third RELOCATE. The parameters
      that can be tuned are basically values for PERIOD and
      the values of MIN and/or MAX as the per the case.
      The values for MIN and MAX are treated as multiples of
      average activity over past 24 hour period.
    </COMMENT>
    <SELECT>
      <PATTERN> * </PATTERN>
    </SELECT>

    <CREATE>
      <COMMENT>
        create files on ssdtier, failing which
        create them on other tiers
      </COMMENT>
      <ON>
        <DESTINATION Flags="any">
          <CLASS> ssdtier </CLASS>
        </DESTINATION>
      </ON>
    </CREATE>

    <RELOCATE>
      <COMMENT>
        Move the files out of SSD if thier last 3 hour
        write IOTEMP is more than 1.5 times the last
        24 hour average write IOTEMP. The PERIOD is
        purposely shorter than the other RELOCATEs
        because we want to move it out as soon as
        write activity starts peaking. This criteria
        could be used to reduce SSD wear outs.
      </COMMENT>
      <FROM>
        <SOURCE>
          <CLASS> ssdtier </CLASS>
        </SOURCE>
      </FROM>
      <TO>
        <DESTINATION>
          <CLASS> nonssd_tier </CLASS>
        </DESTINATION>
      </TO>
      <WHEN>
        <IOTEMP Type="nwbytes" Average="*">
          <MIN Flags="gt"> 1.5 </MIN>
          <PERIOD Units="hours"> 3 </PERIOD>
        </IOTEMP>
      </WHEN>
    </RELOCATE>

    <RELOCATE>
      <COMMENT>
        OR move the files out of SSD if their last 6 hour
        read IOTEMP is less than half the last 24 hour
        average read IOTEMP. The PERIOD is longer,
        we may want to observe longer periods
        having brought the file in. This avoids quickly
        sending the file out of SSDs once in.
      </COMMENT>
      <FROM>
        <SOURCE>
          <CLASS> ssdtier </CLASS>
        </SOURCE>
      </FROM>
      <TO>
        <DESTINATION>
          <CLASS> nonssd_tier </CLASS>
        </DESTINATION>
      </TO>
      <WHEN>
        <IOTEMP Type="nrbytes" Average="*">
          <MAX Flags="lt"> 0.5 </MAX>
          <PERIOD Units="hours"> 6 </PERIOD>
        </IOTEMP>
      </WHEN>
    </RELOCATE>

    <RELOCATE>
      <COMMENT>
        OR move the files into SSD if their last 3 hour
        read IOTEMP is more than or equal to 1.5 times
        the last 24 hour average read IOTEMP AND
        their last 6 hour write IOTEMP is less than
        half of the last 24 hour average write IOTEMP
      </COMMENT>
      <TO>
        <DESTINATION>
          <CLASS> ssd_tier </CLASS>
        </DESTINATION>
      </TO>
      <WHEN>
        <IOTEMP Type="nrbytes" Prefer="high" Average="*">
          <MIN Flags="gteq"> 1.5 </MIN>
          <PERIOD Units="hours"> 3 </PERIOD>
        </IOTEMP>
        <IOTEMP Type="nwbytes" Average="*">
          <MAX Flags="lt"> 0.5 </MAX>
          <PERIOD Units="hours"> 3 </PERIOD>
```

```
            </IOTEMP>
          </WHEN>
        </RELOCATE>
      </RULE>
    </PLACEMENT_POLICY>
```

In this placement policy, new files are created on the SSD tiers if space is available, or elsewhere if space is not available. When enforce is performed, the files that are currently in SSDs whose write activity is increased above a threshold or whose read activity fell below a threshold over a given period are moved out of the SSDs. The first two RELOCATEs capture this intent. However, the files whose read activity intensified above a threshold and whose write activity does not exceed a threshold over the given period are moved into SSDs, while giving preference to files with higher read activity.

The following figure illustrates the behavior of the example placement policy:


Write temperature high threshold to move in
Write temperature low threshold to move out
Read temperature
Move in
Read temperature low threshold to move in
Move out
File system average read temperature
Read temperature high threshold to move out
Write temperature
File system average write temperature

The files whose I/O activity falls in the light gray area are good candidates for moving in to SSD storage. These files have less write activity such that they have less impact on wear leveling, and the slower write times to SSDs is less of a factor. These files have intense read activity, which also makes the files ideal for placement on SSDs since read activity does not cause any wear leveling side effects, and reads are faster from SSDs. In contrast, the files whose I/O activity falls in the dark gray area are good candidates to be moved out of SSD storage, since they have more write activity or less read activity. Greater write activity leads to greater wear leveling of the SSDs, and your file system's performance suffers from the slower write times of SSDs. Lesser read activity means that you are not benefitting from the faster read times of SSDs with these files.

Recovering from CDS errors

This appendix includes the following topics:

- CDS error codes and recovery actions

CDS error codes and recovery actions

Table A-1 lists the CDS error codes and the action that is required.

Table A-1 Error codes and required actions

Error number	Message	Action
329	Cannot join a non-CDS disk group and a CDS disk group	Change the non-CDS disk group into a CDS disk group (or vice versa), then retry the join operation.
330	Disk group is for a different platform	Import the disk group on the correct platform. It cannot be imported on this platform.
331	Volume has a log which is not CDS compatible	To get a log which is CDS compatible, you need to stop the volume, if currently active, then start the volume. After the volume has been successfully started, retry setting the CDS attribute for the disk group.
332	License has expired, or is not available for CDS	Obtain a license from Symantec that enables the usage of CDS disk groups.

Table A-1 Error codes and required actions *(continued)*

Error number	Message	Action
333	Non-CDS disk cannot be placed in a CDS disk group	Do one of the following: ■ Add the disk to another disk group that is a non-CDS disk group. ■ Re-initialize the disk as a CDS disk so that it can be added to the CDS disk group. ■ Change the CDS disk group into a non-CDS disk group and then add the disk.
334	Disk group alignment not CDS compatible	Change the alignment of the disk group to 8K and then retry setting the CDS attribute for the disk group.
335	Subdisk length violates disk group alignment	Ensure that sub-disk length value is a multiple of 8K.
336	Subdisk offset violates disk group alignment	Ensure that sub-disk offset value is a multiple of 8K.
337	Subdisk plex offset violates disk group alignment	Ensure that sub-disk plex offset value is a multiple of 8K.
338	Plex stripe width violates disk group alignment	Ensure that plex stripe width value is a multiple of 8K.
339	Volume or log length violates disk group alignment	Ensure that thelength of the volume is a multiple of 8K. For a log, set the value of the `dgalign_checking` attribute to `round`. This ensurs e that the length of the log is silently rounded to a valid value.
340	Last disk media offset violates disk group alignment	Reassociate the DM record prior to upgrading.

Table A-1 Error codes and required actions *(continued)*

Error number	Message	Action
341	Too many device nodes in disk group	Increase the number of device nodes allowed in the disk group, if not already at the maximum. Otherwise, you need to remove volumes from the disk group, possibly by splitting the disk group.
342	Map length too large for current log length	Use a smaller map length for the DRL/DCM log, or increase the log length and retry.
343	Volume log map alignment violates disk group alignment	Remove the DRL/DCM log, then add it back after changing the alignment of the disk group.
345	Disk group contains an old-style RVG which cannot be imported on this platform	Import the disk group on the platform that created the RVG. To import the disk group on this platform, first remove the RVG on the creating platform.
346	Cache object autogrow by max_autogrow violates disk group alignment	Ensure that cache attribute value is a multiple of 8K.
347	User transactions are disabled for the disk group	Retry the command as it was temporarily disallowed by the `vxcdsconvert` command executing at the same time.
348	Disk is in use	Contact Technical Support.

Appendix B

Files and scripts for sample scenarios

This appendix includes the following topics:

About files and scripts for sample scenarios

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

Note: These scripts are not supported by Symantec, and are provided for informational use only. You can purchase customization of the environment through Veritas Vpro Consulting Services.

Table B-1 list the files and scripts.

Table B-1 Files and scripts for sample scenarios

File or script	Used for...
Script to initiate online off-host backup of an Oracle database	■ Online off-host backup. See "Making an off-host backup of an online database" on page 46.
Script to put an Oracle database into hot backup mode, Script to quiesce a Sybase ASE database or Script to suspend I/O for a DB2 database	■ Online backup. See About online database backup ■ Decision support. See "About decision support" on page 59.
Script to end Oracle database hot backup mode, Script to release a Sybase ASE database from quiesce mode or Script to resume I/O for a DB2 database	■ Online backup. See "About online database backup" on page 41. ■ Decision support. See "About decision support" on page 59.
Script to perform off-host backup	■ Online off-host backup. See "Making an off-host backup of an online database" on page 46.
Script to create an off-host replica Oracle database	■ Decision support. See "Creating an off-host replica database" on page 65.
Script to complete, recover and start a replica Oracle database or Script to start a replica Sybase ASE database	■ Decision support. See "Creating an off-host replica database" on page 65.

Script to initiate online off-host backup of an Oracle database

Use this script to initiate online off-host backup of an Oracle database.

```
#!/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
newvollist=”snap_dbase_vol source=dbase_vol/newvol=snap_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 make $newvollist

# 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).

# Move the snapshot volumes into a separate disk group.

vxdg split $dbasedg $snapdg $snapvollist

# 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.
# Note: Replace “rsh” with “remsh” on HP-UX systems.

rsh $dbnode -c “do_backup.sh $snapvollist”

# 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.

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

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

vxsnap -g $dbasedg reattach $volsnaplist
```

Script to put an Oracle database into hot backup mode

Use this script to put an 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 a Sybase ASE database

Use this script to quiesce a 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

Use this 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

Use this 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
#           archives the contents of the redo logs recorded during the back
```

Script to release a Sybase ASE database from quiesce mode

Use this script to release a 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

Use this 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

Use this 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

Use this 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.
# Note: Replace “rsh” with “remsh” on HP-UX systems.

rsh $dbnode -c “startdb.sh $vollist”
```

Script to complete, recover and start a replica Oracle database

Use this script to complete, recover and start a 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 a replica Sybase ASE database

Use this script to start a 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

This appendix includes the following topics:

About preparing a replica Oracle database

This appendix describes how to set up a replica off-host Oracle database to be used for decision support.

See “Creating an off-host replica database” on page 65.

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 246.

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:

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

  ```
  GROUP N '/dbase_vol/logN' SIZE 52428288,
  ```

  so that they read:

  ```
  GROUP N '/rep/dbase_vol/logN' SIZE 52428288,
  ```

- 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',
  ```

- 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 a control file” on page 248.

8 Copy the Oracle initialization file for the original database to a new initialization file for the replica database.

For example, the Oracle initialization file for the original database is shown as `initdbase.ora`.

See “Initialization file for original production database” on page 248.

For example, the initialization file for the replica database is shown as `initREP1.ora`.

See “Initialization file for replica Oracle database” on page 250.

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

The following example shows the text control file for the 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 a control file

The following example shows the SQL script to create a 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

The following example shows the initialization file for the 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,s2
4,s25,s26,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

The following example shows the initialization file for the 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,s2
4,s25,s26,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
```

Glossary

address-length pair
Identifies the starting block address and the length of an extent (in file system or logical blocks).

archived log mode
Used to retrieve information on transactions that occur during a hot backup.

asynchronous I/O
A format of I/O that performs non-blocking reads and writes. This enables the system to handle multiple I/O requests simultaneously.

atomic operation
An operation that either succeeds completely or fails and leaves everything as it was before the operation was started. If the operation succeeds, all aspects of the operation take effect at once and the intermediate states of change are invisible. If any aspect of the operation fails, then the operation aborts without leaving partial changes.

autoextend
An Oracle feature that automatically grows a database file by a prespecified size, up to a prespecified maximum size.

backup mode
A state of the Oracle tablespace that lets you perform online backup.

BLI (block-level incremental) backup
A method used to back up only changed data blocks, not changed files, since the last backup.

block map
A file system is divided into fixed-size blocks when it is created. As data is written to a file, unused blocks are allocated in ranges of blocks, called extents. The extents are listed or pointed to from the inode. The term used for the data that represents how to translate an offset in a file to a file system block is the “block map” for the file.

boot disk
A disk used for booting an operating system.

buffered I/O
A mode of I/O operation (where I/O is any operation, program, or device that transfers data to or from a computer) that first transfers data into the Operating System buffer cache.

cache
Any memory used to reduce the time required to respond to an I/O request. The read cache holds data in anticipation that it will be requested by a client. The write cache holds data written until it can be safely stored on non-volatile storage media.

Cached Quick I/O
Cached Quick I/O allows databases to make more efficient use of large system memory while still maintaining the performance benefits of Quick I/O. Cached Quick I/O provides an efficient, selective buffering mechanism to complement asynchronous I/O.

cluster A set of hosts that share a set of disks.

cluster-shareable disk group A disk group in which the disks are shared between more than one host.

cold backup The process of backing up a database that is not in active use.

command launcher A graphical user interface (GUI) window that displays a list of tasks that can be performed by Veritas Volume Manager or other objects. Each task is listed with the object type, task (action), and a description of the task. A task is launched by clicking on the task in the Command Launcher. concatenation A Veritas Volume Manager layout style characterized by subdisks that are arranged sequentially and contiguously.

concurrent I/O A form of Direct I/O that does not require file-level write locks when writing to a file. Concurrent I/O allows the relational database management system (RDBMS) to write to a given file concurrently.

configuration database A set of records containing detailed information on existing Veritas Volume Manager objects (such as disk and volume attributes). A single copy of a configuration database is called a configuration copy.

container A physical storage device that can be identified by a directory name, a device name, or a file name.

control file An Oracle control file specifies the physical structure of an Oracle database, including such things as the database name, names and locations of the datafiles and redo log files, and the timestamp of when the database was created. When you start an Oracle database, the control file is used to identify the database instance name redo log files that must be opened for transaction recording and recovery and datafiles where data is stored.

copy-on-write A technique for preserving the original of some data. As data is modified by a write operation, the original copy of data is copied.

Applicable to Storage Checkpoint technology, where original data, at the time of the Storage Checkpoint, must be copied from the file system to the Storage Checkpoint when it is to be overwritten. This preserves the frozen image of the file system in the Storage Checkpoint.

data block A logical database data storage unit. Blocks contain the actual data. When a database is created, a data block size is specified. The database then uses and allocates database space in data blocks.

DCO (data change object) A Veritas Volume Manager object used to manage FastResync map information in the DCO log volume. A DCO and a DCO log volume must be associated with a volume in order to implement persistent FastResync on that volume.

Data Storage Checkpoint A Storage Checkpoint that is a complete image of a file system. For more information, see "Storage Checkpoint."

database A database is a collection of information that is organized in a structured fashion. Two examples of databases are Relational Databases (such as Oracle, Sybase, or DB2), where data is stored in tables and generally accessed by one or more keys and Flat File Databases, where data is not generally broken up into tables and relationships. Databases generally provide tools and/or interfaces to retrieve data.

datafile A physical database attribute that contains database data. An Oracle datafile can only be associated with a single database. One or more datafiles form a logical database storage unit called a tablespace.

data change object See DCO.

DCO log volume A volume used to hold persistent FastResync change maps.

DSS (Decision Support Systems) Computer-based systems used to model, identify, and solve problems, and make decisions.

defragmentation The act of reorganizing data to reduce fragmentation. Data in file systems become fragmented over time.

device file A block- or character-special file located in the /dev directory representing a device.

device name The name of a device file, which represents a device. AIX syntax is Disk_#; HP-UX syntax is c#t#d#; Linux syntax is sda, where "a" could be any alphabetical letter; Solaris syntax is c#t#d#s#.

direct I/O An unbuffered form of I/O that bypasses the kernel’s buffering of data. With direct I/O, data is transferred directly between the disk and the user application.

Dirty Region Logging The procedure by which the Veritas Volume Manager monitors and logs modifications to a plex. A bitmap of changed regions is kept in an associated subdisk called a log subdisk.

disk access name The name used to access a physical disk, such as Disk_1 on an AIX system, c1t1d1 on an HP-UX system, sda on a Linux system, or c0t0d0s0 on a Solaris system. The term device name can also be used to refer to the disk access name.

disk array A collection of disks logically and physically arranged into an object. Arrays provide benefits including data redundancy and improved performance.

disk cache A section of RAM that provides a cache between the disk and the application. Disk cache enables the computer to operate faster. Because retrieving data from hard disk can be slow, a disk caching program helps solve this problem by placing recently accessed data in the disk cache. Next time that data is needed, it may already be available in the disk cache; otherwise a time-consuming operation to the hard disk is necessary.

disk group A collection of disks that share a common configuration.

A disk group configuration is a set of records containing detailed information on existing Veritas Volume Manager objects (such as disk and volume attributes) and their relationships. Each disk group has an administrator-assigned name and an internally defined unique ID. The root disk group (rootdg) is a special private disk group.

disk name A Veritas Volume Manager logical or administrative name chosen for the disk, such as disk03. The term disk media name is also used to refer to the disk name.

Dynamic Multipathing See DMP.

Decision Support Systems See DSS.

DMP (Dynamic Multipathing) A Veritas Volume Manager feature that allows the use of multiple paths to the same storage device for load balancing and redundancy.

error handling Routines in a program that respond to errors. The measurement of quality in error handling is based on how the system informs the user of such conditions and what alternatives it provides for dealing with them.

evacuate Moving subdisks from the source disks to target disks.

exabyte A measure of memory or storage. An exabyte is approximately 1,000,000,000,000,000,000 bytes (technically 2 to the 60th power, or 1,152,921,504,606,846,976 bytes). Also EB.

extent A logical database attribute that defines a group of contiguous file system data blocks that are treated as a unit. An extent is defined by a starting block and a length.

extent attributes An extent allocation policy that is associated with a file and/or file system. See also address-length pair.

failover The act of moving a service from a failure state back to a running/available state. Services are generally applications running on machines and failover is the process of restarting these applications on a second system when the first has suffered a failure.

file system A collection of files organized together into a structure. File systems are based on a hierarchical structure consisting of directories and files.

file system block The fundamental minimum size of allocation in a file system.

fileset A collection of files within a file system.

fixed extent size An extent attribute associated with overriding the default allocation policy of the file system.

fragmentation Storage of data in non-contiguous areas on disk. As files are updated, new data is stored in available free space, which may not be contiguous. Fragmented files cause extra read/write head movement, slowing disk accesses.

gigabyte A measure of memory or storage. A gigabyte is approximately 1,000,000,000 bytes (technically, 2 to the 30th power, or 1,073,741,824 bytes). Also GB, Gbyte, and G-byte.

HA (high availability) The ability of a system to perform its function continuously (without significant interruption) for a significantly longer period of time than the combined reliabilities of its individual components. High availability is most often achieved through failure tolerance and inclusion of redundancy; from redundant disk to systems, networks, and entire sites.

hot backup The process of backing up a database that is online and in active use.

hot pluggable To pull a component out of a system and plug in a new one while the power is still on and the unit is still operating. Redundant systems can be designed to swap disk drives, circuit boards, power supplies, CPUs, or virtually anything else that is duplexed within the computer. Also known as hot swappable.

hot-relocation A Veritas Volume Manager technique of automatically restoring redundancy and access to mirrored and RAID-5 volumes when a disk fails. This is done by relocating the affected subdisks to disks designated as spares and/or free space in the same disk group.

inode list An inode is an on-disk data structure in the file system that defines everything about the file, except its name. Inodes contain information such as user and group ownership, access mode (permissions), access time, file size, file type, and the block map for the data contents of the file. Each inode is identified by a unique inode number in the file system where it resides. The inode number is used to find the inode in the inode list for the file system. The inode list is a series of inodes. There is one inode in the list for every file in the file system.

instance When you start a database, a system global area (SGA) is allocated and the Oracle processes are started. The SGA is the area of memory used for database information shared by all database users. The Oracle processes and the SGA create what is called an Oracle instance.

intent logging A logging scheme that records pending changes to a file system structure. These changes are recorded in an intent log.

interrupt key A way to end or break out of any operation and return to the system prompt by pressing Ctrl-C.

kernel asynchronous I/O A form of I/O that performs non-blocking system level reads and writes. This enables the system to handle multiple I/O requests simultaneously.

kilobyte A measure of memory or storage. A kilobyte is approximately a thousand bytes (technically, 2 to the 10th power, or 1,024 bytes). Also KB, Kbyte, kbyte, and K-byte.

large file A file more than two gigabytes in size. An operating system that uses a 32-bit signed integer to address file contents will not support large files; however, the Version 4 disk layout feature of VxFS supports file sizes of up to two terabytes.

large file system A file system more than two gigabytes in size. VxFS, in conjunction with VxVM, supports large file systems.

latency The amount of time it takes for a given piece of work to be completed. For file systems, this typically refers to the amount of time it takes a given file system operation to return to the user. Also commonly used to describe disk seek times.

load balancing The tuning of a computer system, network tuning, or disk subsystem in order to more evenly distribute the data and/or processing across available resources. For example, in clustering, load balancing might distribute the incoming transactions evenly to all servers, or it might redirect them to the next available server.

load sharing The division of a task among several components without any attempt to equalize each component's share of the load. When several components are load sharing, it is possible for some of the shared components to be operating at full capacity and limiting performance, while others components are under utilized.

LUN (logical unit number) A method of expanding the number of SCSI devices that can be placed on one SCSI bus. Logical Unit Numbers address up to seven devices at each SCSI ID on an 8-bit bus or up to 15 devices at each ID on a 16-bit bus.

logical volume See volume.

logical unit number See LUN.

master node A computer which controls another computer or a peripheral.

megabyte A measure of memory or storage. A megabyte is approximately 1,000,000 bytes (technically, 2 to the 20th power, or 1,048,576 bytes). Also MB, Mbyte, mbyte, and K-byte.

metadata Data that describes other data. Data dictionaries and repositories are examples of metadata. The term may also refer to any file or database that holds information about another database's structure, attributes, processing, or changes.

mirror A duplicate copy of a volume and the data therein (in the form of an ordered collection of subdisks). Each mirror is one copy of the volume with which the mirror is associated. The terms mirror and plex can be used synonymously.

mirroring A layout technique that mirrors the contents of a volume onto multiple plexes. Each plex duplicates the data stored on the volume, but the plexes themselves may have different layouts.

mount point The directory path name at which a file system attaches to the file system hierarchy.

multithreaded Having multiple concurrent or pseudo-concurrent execution sequences. Used to describe processes in computer systems. Multithreaded processes are one means by which I/O request-intensive applications can use independent access to volumes and disk arrays to increase I/O performance.

NBU See Veritas NetBackup (NBU).

node One of the hosts in a cluster.

object (VxVM) An entity that is defined to and recognized internally by the Veritas Volume Manager. The VxVM objects include volumes, plexes, subdisks, disks, and disk groups. There are two types of VxVM disk objects—one for the physical aspect of the disk and the other for the logical aspect of the disk.

Online Transaction Processing See OLTP.

online administration An administrative feature that allows configuration changes without system or database down time.

OLTP (Online Transaction Processing) A type of system designed to support transaction-oriented applications. OLTP systems are designed to respond immediately to user requests and each request is considered to be a single transaction. Requests can involve adding, retrieving, updating or removing data.

paging The transfer of program segments (pages) into and out of memory. Although paging is the primary mechanism for virtual memory, excessive paging is not desirable.

parity A calculated value that can be used to reconstruct data after a failure. While data is being written to a RAID-5 volume, parity is also calculated by performing an exclusive OR (XOR) procedure on data. The resulting parity is then written to the volume. If a portion of a RAID-5 volume fails, the data that was on that portion of the failed volume can be recreated from the remaining data and the parity.

partition The logical areas into which a disk is divided.

persistence Information or state that will survive a system reboot or crash.

petabyte A measure of memory or storage. A petabyte is approximately 1,000 terabytes (technically, 2 to the 50th power).

plex A duplicate copy of a volume and its data (in the form of an ordered collection of subdisks). Each plex is one copy of a volume with which the plex is associated. The terms mirror and plex can be used synonymously.

preallocation Prespecifying space for a file so that disk blocks will physically be part of a file before they are needed. Enabling an application to preallocate space for a file guarantees that a specified amount of space will be available for that file, even if the file system is otherwise out of space.

Quick I/O Quick I/O presents a regular Veritas File System file to an application as a raw character device. This allows Quick I/O files to take advantage of asynchronous I/O and direct I/O to and from the disk device, as well as bypassing the UNIX single-writer lock behavior for most file system files.

Quick I/O file A regular UNIX file that is accessed using the Quick I/O naming extension (::cdev:vxfs:).

RAID A Redundant Array of Independent Disks (RAID) is a disk array set up with part of the combined storage capacity used for storing duplicate information about the data stored in that array. This makes it possible to regenerate the data if a disk failure occurs.

recovery mode The process of recovering the file systems once the file systems have been restored. In a media failure scenario, the backups must be restored before the database can be recovered.

redo log files Redo log files record transactions pending on a database. If a failure prevents data from being permanently written to datafiles, changes can be obtained from the redo log files. Every Oracle database has a set of two or more redo log files.

repository A repository holds the name, type, range of values, source, and authorization for access for each data element in a database. The database maintains a repository for administrative and reporting use.

Pertinent information, needed to display configuration information and interact with the database, is stored in the repository.

root disk The disk containing the root file system.

root disk group A special private disk group on the system. The root disk group is named rootdg. However, starting with the 4.1 release of Veritas Volume Manager, the root disk group is no longer needed.

root file system The initial file system mounted as part of the UNIX kernel startup sequence.

script A file, containing one or more commands that can be run to perform processing.

shared disk group A disk group in which the disks are shared by multiple hosts (also referred to as a cluster-shareable disk group).

sector A minimal unit of the disk partitioning. The size of a sector can vary between systems.

A sector is commonly 512 bytes.

segment Any partition, reserved area, partial component, or piece of a larger structure.

System Global Area See SGA.

single threading The processing of one transaction to completion before starting the next.

slave node A node that is not designated as a master node.

slice The standard division of a logical disk device. The terms partition and slice can be used synonymously.

snapped file system A file system whose exact image has been used to create a snapshot file system.

snapped volume A volume whose exact image has been used to create a snapshot volume.

snapshot A point-in-time image of a volume or file system that can be used as a backup.

snapshot file system An exact copy of a mounted file system, at a specific point in time, that is used for online backup. A snapshot file system is not persistent and it will not survive a crash or reboot of the system.

snapshot volume An exact copy of a volume, at a specific point in time. The snapshot is created based on disk mirroring and is used for online backup purposes.

spanning A layout technique that permits a volume (and its file system or database) too large to fit on a single disk to distribute its data across multiple disks or volumes.

Storage Checkpoint An efficient snapshot technology for creating a point-in-time image of a currently mounted VxFS file system. A Storage Checkpoint presents a consistent, point-in-time view of the file system by identifying and maintaining modified file system blocks.

storage class Set of volumes with the same volume tag.

Storage Rollback On-disk restore capability for faster recovery from logical errors, such as accidentally deleting a file. Because each Storage Checkpoint is a point-in-time image of a file system, Storage Rollback simply restores or rolls back a file or entire file system to a Storage Checkpoint.

stripe A set of stripe units that occupy the same positions across a series of columns in a multi-disk layout.

stripe unit Equally sized areas that are allocated alternately on the subdisks (within columns) of each striped plex. In an array, this is a set of logically contiguous blocks that exist on each disk before allocations are made from the next disk in the array.

stripe unit size The size of each stripe unit. The default stripe unit size for VxVM is 32 sectors (16K). For RAID 0 stripping, the stripe unit size is 128 sectors (64K). For VxVM RAID 5, the stripe unit size is 32 sectors (16K). A stripe unit size has also historically been referred to as a stripe width.

striping A layout technique that spreads data across several physical disks using stripes. The data is allocated alternately to the stripes within the subdisks of each plex.

subdisk A consecutive set of contiguous disk blocks that form a logical disk segment. Subdisks can be associated with plexes to form volumes.

superuser A user with unlimited access privileges who can perform any and all operations on a computer. In UNIX, this user may also be referred to as the “root” user. On Windows/NT, it is the “Administrator.”

SGA (System Global Area) The area of memory used for database information shared by all database users. Each SGA contains the data and control information for a single Oracle instance.

tablespace A tablespace is a storage structure (containing tables, indexes, large objects, and long data) that allows you to assign the location of database and table data directly onto containers. Tablespaces reside in database partition groups.

In an Oracle database, an allocation of space used to hold schema objects (triggers, stored procedures, tables, etc.). A tablespace is associated with one or more datafiles.

terabyte A measure of memory or storage. A terabyte is approximately 1,000,000,000,000 bytes (technically, 2 to the 40th power, or 1,000 GB). Also TB.

throughput A measure of work accomplished in a given amount of time. For file systems, this typically refers to the number of I/O operations in a given period of time.

UFS The Solaris name for a file system type derived from the 4.2 Berkeley Fast File System.

unbuffered I/O I/O that bypasses the file system cache for the purpose of increasing I/O performance (also known as direct I/O).

Veritas Enterprise Administrator Application that is required to access graphical user interface (GUI) functionality.

Veritas Extension for Oracle Disk Manager A feature of Veritas Storage Foundation for Oracle that lets Oracle create and manage database storage, as well as performing I/Os in a file system without the performance degradation typically associated with running databases on file systems.

NBU (Veritas NetBackup) A product that lets you back up, archive, and restore files, directories, or raw partitions that reside on your client system.

VVR (Veritas Volume Replicator) A feature of Veritas Volume Manager, VVR is a data replication tool designed to contribute to an effective disaster recovery plan.

Version Checkpoint In a DB2 UDB EEE or ESE Data Partitioning Feature (DPF) environment, a Version Checkpoint is a set of checkpoints that spans multiple partitions. A Version Checkpoint contains one Storage Checkpoint per partition. A Storage Checkpoint is an efficient snapshot technology for creating a point-in-time image of a currently mounted VxFS file system. A Storage Checkpoint presents a consistent, point-in-time view of the file system by identifying and maintaining modified file system blocks.

volume A logical disk device that appears to applications, databases, and file systems as a physical disk partition. A logical disk can encompass multiple or one to many physical volumes.

volume layout A variety of layouts that allows you to configure your database to meet performance and availability requirements. This includes spanning, striping (RAID-0), mirroring (RAID-1), mirrored stripe volumes (RAID-0+1), striped mirror volumes (RAID-1+0), and RAID 5.

volume manager objects Volumes and their virtual components. See object (VxVM).

VVR See “Veritas Volume Replicator (VVR).”

vxfs or VxFS The acronym for Veritas File System.

vxvm or VxVM The acronym for Veritas Volume Manager.

Index