Veritas InfoScale™ 7.4.1 Solutions Guide - Linux

VERITAS™

Last updated: 2019-07-18

Legal Notice

Copyright © 2019 Veritas Technologies LLC. All rights reserved.

Veritas and the Veritas Logo are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

This product may contain third-party software for which Veritas is required to provide attribution to the third-party (“Third-Party Programs”). Some of the Third-Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Refer to the third-party legal notices document accompanying this Veritas product or available at:

https://www.veritas.com/about/legal/license-agreements

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 Veritas Technologies LLC 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. VERITAS TECHNOLOGIES LLC 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, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. 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.

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

http://www.veritas.com

Technical Support

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

https://www.veritas.com/support

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

https://my.veritas.com

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

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

Documentation

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

https://sort.veritas.com/documents

Documentation feedback

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

infoscaledocs@veritas.com

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

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

Veritas Services and Operations Readiness Tools (SORT)

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

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

Contents

Section 1 Introducing Veritas InfoScale 12

Chapter 1 Introducing Veritas InfoScale 13

About the Veritas InfoScale product suite 13
Components of the Veritas InfoScale product suite 13

Section 2 Solutions for Veritas InfoScale products 15

Chapter 2 Solutions for Veritas InfoScale products 16

Use cases for Veritas InfoScale products 16
Feature support across Veritas InfoScale 7.4.1 products 21
Using SmartMove and Thin Provisioning with Sybase databases 24
Running multiple parallel applications within a single cluster using the application isolation feature 25
Scaling FSS storage capacity with dedicated storage nodes using application isolation feature 35
Finding Veritas InfoScale product use cases information 43

Section 3 Stack-level migration to IPv6 or dual stack 45

Chapter 3 Stack-level migration to IPv6 or dual stack 46

Migrating Veritas InfoScale products to support IPv6/dual-stack 46

Section 4 Improving database performance 49

Chapter 4 Overview of database accelerators 50

About Veritas InfoScale product components database accelerators 50

Chapter 5 Improving database performance with Veritas Concurrent I/O 52

About Concurrent I/O 52
How Concurrent I/O works 52
Tasks for enabling and disabling Concurrent I/O 53
Enabling Concurrent I/O for Sybase 53
Disabling Concurrent I/O for Sybase 54

Chapter 6 Improving database performance with atomic write I/O 55

About the atomic write I/O 55
Requirements for atomic write I/O 56
Restrictions on atomic write I/O functionality 56
How the atomic write I/O feature of Storage Foundation helps MySQL databases 57
VxVM and VxFS exported IOCTLs 57
Configuring atomic write I/O support for MySQL on VxVM raw volumes 58
Configuring atomic write I/O support for MySQL on VxFS file systems 60
Dynamically growing the atomic write capable file system 62
Disabling atomic write I/O support 62

Section 5 Using point-in-time copies 63

Chapter 7 Understanding point-in-time copy methods 64

About point-in-time copies 64
Implementing point-in time copy solutions on a primary host 65
Implementing off-host point-in-time copy solutions 66
When to use point-in-time copies 72
About Storage Foundation point-in-time copy technologies 73
Volume-level snapshots 74
Storage Checkpoints 75

Chapter 8 Backing up and recovering 77

Storage Foundation and High Availability Solutions backup and recovery methods 77
Preserving multiple point-in-time copies 78
Setting up multiple point-in-time copies 78
Refreshing point-in-time copies 80

Recovering from logical corruption 81
Off-host processing using refreshed snapshot images 83
Online database backups 83
Making a backup of an online database on the same host 84
Making an off-host backup of an online database 93
Backing up on an off-host cluster file system 101
Mounting a file system for shared access 103
Preparing a snapshot of a mounted file system with shared access 103
Backing up a snapshot of a mounted file system with shared access 105
Resynchronizing a volume from its snapshot volume 108
Reattaching snapshot plexes 109
Database recovery using Storage Checkpoints 110
Creating Storage Checkpoints 110
Rolling back a database 111

Chapter 9 Backing up and recovering in a NetBackup environment 113

About Veritas NetBackup 113
About using NetBackup for backup and restore for Sybase 114
Using NetBackup in an SFHA Solutions product environment 114
Clustering a NetBackup Master Server 114
Backing up and recovering a VxVM volume using NetBackup 115
Recovering a VxVM volume using NetBackup 117

Chapter 10 Off-host processing 118

Veritas InfoScale Storage Foundation off-host processing methods 118
Using a replica database for decision support 119
Creating a replica database on the same host 120
Creating an off-host replica database 132
What is off-host processing? 145
About using VVR for off-host processing 145

Chapter 11 Creating and refreshing test environments 146

About test environments 146
Creating a test environment 146
Refreshing a test environment 147

Chapter 12 Creating point-in-time copies of files 150

Using FileSnaps to create point-in-time copies of files 150
Using FileSnaps to provision virtual desktops 150
Using FileSnaps to optimize write intensive applications for virtual machines 151
Using FileSnaps to create multiple copies of data instantly 151

Section 6 Maximizing storage utilization 152

Chapter 13 Optimizing storage tiering with SmartTier 153

About SmartTier 153
About VxFS multi-volume file systems 155
About VxVM volume sets 156
About volume tags 156
SmartTier use cases for Sybase 157
Setting up a filesystem for storage tiering with SmartTier 157
Relocating old archive logs to tier two storage using SmartTier 160
Relocating inactive tablespaces or segments to tier two storage 162
Relocating active indexes to premium storage 165
Relocating all indexes to premium storage 167

Chapter 14 Optimizing storage with Flexible Storage Sharing 171

About Flexible Storage Sharing 171
Limitations of Flexible Storage Sharing 172
About use cases for optimizing storage with Flexible Storage Sharing 173
Setting up an SFRAC clustered environment with shared nothing storage 173
Implementing the SmartTier feature with hybrid storage 174
Configuring a campus cluster without shared storage 174

Section 7 Migrating data 175

Chapter 15 Understanding data migration 176

Types of data migration 176

Chapter 16 Offline migration from LVM to VxVM 178

About migration from LVM 178
Converting unused LVM physical volumes to VxVM disks 179
LVM volume group to VxVM disk group conversion 180
Volume group conversion limitations 181
Converting LVM volume groups to VxVM disk groups 183
Examples of second stage failure analysis 194
LVM volume group restoration 196
Restoring an LVM volume group 196

Chapter 17 Offline conversion of native file system to VxFS 198

About the offline conversion of native file system to VxFS 198
Requirements for offline conversion of a native file system to VxFS 199
Converting the native file system to VxFS 199

Chapter 18 Online migration of a native file system to the VxFS file system 201

About online migration of a native file system to the VxFS file system 201
Administrative interface for online migration of a native file system to the VxFS file system 202
Migrating a native file system to the VxFS file system 203
Backing out an online migration of a native file system to the VxFS file system 206
VxFS features not available during online migration 207
Limitations of online migration 208

Chapter 19 Migrating storage arrays 209

Array migration for storage using Linux 209
Overview of storage mirroring for migration 210
Allocating new storage 211
Initializing the new disk 213
Checking the current VxVM information 214
Adding a new disk to the disk group 215
Mirroring 216
Monitoring 217
Mirror completion 218
Removing old storage 218

Post-mirroring steps 219

Chapter 20 Migrating data between platforms 220

Overview of the Cross-Platform Data Sharing (CDS) feature 220
Shared data across platforms 221
Disk drive sector size 222
Block size issues 222
Operating system data 222
CDS disk format and disk groups 222
CDS disk access and format 223
Non-CDS disk groups 226
Disk group alignment 226
Setting up your system to use Cross-platform Data Sharing (CDS) 228
Creating CDS disks from uninitialized disks 229
Creating CDS disks from initialized VxVM disks 230
Creating CDS disk groups 231
Converting non-CDS disks to CDS disks 232
Converting a non-CDS disk group to a CDS disk group 233
Verifying licensing 235
Defaults files 235
Maintaining your system 237
Disk tasks 238
Disk group tasks 240
Displaying information 246
Default activation mode of shared disk groups 249
Additional considerations when importing CDS disk groups 249
File system considerations 250
Considerations about data in the file system 251
File system migration 251
Specifying the migration target 252
Using the fscdsadm command 253
Migrating a file system one time 255
Migrating a file system on an ongoing basis 256
When to convert a file system 258
Converting the byte order of a file system 258
Alignment value and block size 262
Migrating a snapshot volume 262

Chapter 21 Migrating from Oracle ASM to Veritas File System 265

About the migration 265
Pre-requisites for migration 270
Preparing to migrate 270
Migrating Oracle databases from Oracle ASM to VxFS 272

Section 8 Just in time availability solution for vSphere 274

Chapter 22 Just in time availability solution for vSphere 275

About Just In Time Availability 275
Getting started with Just In Time Availability 281
Prerequisites 283
Supported operating systems and configurations 285
Setting up a plan 286
Managing a plan 288
Deleting a plan 290
Viewing the properties 290
Viewing the history tab 291
Limitations of Just In Time Availability 291

Section 9 Veritas InfoScale 4K sector device support solution 292

Chapter 23 Veritas InfoScale 4k sector device support solution 293

About 4K sector size technology 293
Veritas InfoScale unsupported configurations 294
Migrating VxFS file system from 512-bytes sector size devices to 4K sector size devices 295

Section 10 Reference 297

Appendix A Veritas AppProtect logs and operation states 298

Log files 298
Plan states 299

Appendix B Troubleshooting Veritas AppProtect 301

Troubleshooting Just In Time Availability 301

Section 1

Introducing Veritas InfoScale

- Chapter 1. Introducing Veritas InfoScale

Chapter 1

Introducing Veritas InfoScale

This chapter includes the following topics:

- About the Veritas InfoScale product suite
- Components of the Veritas InfoScale product suite

About the Veritas InfoScale product suite

The Veritas InfoScale product suite addresses enterprise IT service continuity needs. They provide resiliency and software defined storage for critical services across a data center in physical, virtual, and cloud environments. The clustering solution provides high availability and disaster recovery for applications across geographies.

The Veritas InfoScale product suite offers the following products:

- Veritas InfoScale Foundation
- Veritas InfoScale Storage
- Veritas InfoScale Availability
- Veritas InfoScale Enterprise

Components of the Veritas InfoScale product suite

Each new InfoScale product consists of one or more components. Each component within a product offers a unique capability that you can configure for use in your environment.

Table 1-1 lists the components of each Veritas InfoScale product.

Table 1-1 Veritas InfoScale product suite

Product	Description	Components
Veritas InfoScale™ Foundation	Veritas InfoScale™ Foundation delivers a comprehensive solution for heterogeneous online storage management while increasing storage utilization and enhancing storage I/O path availability.	Storage Foundation (SF) Standard (entry-level features)
Veritas InfoScale™ Storage	Veritas InfoScale™ Storage enables organizations to provision and manage storage independently of hardware types or locations while delivering predictable Quality-of-Service, higher performance, and better Return-on-Investment.	Storage Foundation (SF) Enterprise including Replication Storage Foundation Cluster File System (SFCFS)
Veritas InfoScale™ Availability	Veritas InfoScale™ Availability helps keep an organization's information and critical business services up and running on premise and across globally dispersed data centers.	Cluster Server (VCS) including HA/DR
Veritas InfoScale™ Enterprise	Veritas InfoScale™ Enterprise addresses enterprise IT service continuity needs. It provides resiliency and software defined storage for critical services across your datacenter infrastructure.	Cluster Server (VCS) including HA/DR Storage Foundation (SF) Enterprise including Replication Storage Foundation and High Availability (SFHA) Storage Foundation Cluster File System High Availability (SFCFSHA) Storage Foundation for Oracle RAC (SF Oracle RAC) Storage Foundation for Sybase ASE CE (SFSYBASECE)

Section 2

Solutions for Veritas InfoScale products

- Chapter 2. Solutions for Veritas InfoScale products

Chapter 2

Solutions for Veritas InfoScale products

This chapter includes the following topics:

- Use cases for Veritas InfoScale products
- Feature support across Veritas InfoScale 7.4.1 products
- Using SmartMove and Thin Provisioning with Sybase databases
- Running multiple parallel applications within a single cluster using the application isolation feature
- Scaling FSS storage capacity with dedicated storage nodes using application isolation feature
- Finding Veritas InfoScale product use cases information

Use cases for Veritas InfoScale products

Veritas InfoScale Storage Foundation and High Availability (SFHA) Solutions product components and features can be used individually and in concert to improve performance, resilience and ease of management for your storage and applications. This guide documents key use cases for the management features of SFHA Solutions products.

Note: The commands used for the Red Hat Enterprise Linux (RHEL) operating system in this document also apply to supported RHEL-compatible distributions.

Table 2-1 Key use cases for SFHA Solutions products

Use case	Veritas InfoScale feature
Improve database performance using SFHA Solutions database accelerators to enable your database to achieve the speed of raw disk while retaining the management features and convenience of a file system. See "About Veritas InfoScale product components database accelerators" on page 50.	Concurrent I/O See "About Concurrent I/O" on page 52. Veritas Extension for Oracle Disk Manager Veritas Extension for Cached Oracle Disk Manager **Note:** For ODM amd Cached ODM information, see *Storage Foundation: Storage and Availability Managment for Oracle Databases*.
Protect your data using SFHA Solutions Flashsnap, Storage Checkpoints, and NetBackup point-in-time copy methods to back up and recover your data. See "Storage Foundation and High Availability Solutions backup and recovery methods" on page 77. See "About point-in-time copies" on page 64.	FlashSnap See "Preserving multiple point-in-time copies" on page 78. See "Online database backups" on page 83. See "Backing up on an off-host cluster file system" on page 101. See "Storage Foundation and High Availability Solutions backup and recovery methods" on page 77. Storage Checkpoints See "Database recovery using Storage Checkpoints" on page 110. NetBackup with SFHA Solutions See "About Veritas NetBackup" on page 113.
Process your data off-host to avoid performance loss to your production hosts by using SFHA Solutions volume snapshots. See "Veritas InfoScale Storage Foundation off-host processing methods" on page 118.	FlashSnap See "Using a replica database for decision support" on page 119.
Optimize copies of your production database for test, decision modeling, and development purposes by using SFHA Solutions point-in-time copy methods. See "About test environments" on page 146.	FlashSnap See "Creating a test environment" on page 146.

Table 2-1 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Make file level point-in-time snapshots using SFHA Solutions space-optimized FileSnap when you need finer granualarity for your point-in-time copies than file systems or volumes. You can use FileSnap for cloning virtual machines. See " Using FileSnaps to create point-in-time copies of files" on page 150.	FileSnap See "Using FileSnaps to provision virtual desktops" on page 150.
Maximize your storage utilization using SFHA Solutions SmartTier to move data to storage tiers based on age, priority, and access rate criteria. See "About SmartTier" on page 153.	SmartTier See "Setting up a filesystem for storage tiering with SmartTier" on page 157.
Maximize storage utilization for data redundancy, high availability, and disaster recovery, without physically shared storage. See "About Flexible Storage Sharing" on page 171.	Flexible Storage Sharing See "Setting up an SFRAC clustered environment with shared nothing storage" on page 173. See "Implementing the SmartTier feature with hybrid storage" on page 174. See "Configuring a campus cluster without shared storage" on page 174.
Improve your data efficiency on solid state drives (SSDs) through I/O caching using advanced, customizable hueristics to determine which data to cache and how that data gets removed from the cache.	SmartIO read caching for applications running on VxVM volumes SmartIO read caching for applications running on VxFS file systems SmartIO write caching for applications running on VxFS file systems SmartIO caching for databases on VxFS file systems SmartIO caching for databases on VxVM volumes SmartIO write-back caching for databases is not supported on SFRAC See the *Veritas InfoScale 7.4.1 SmartIO for Solid-State Drives Solutions Guide*.

Table 2-1 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Convert your data from native OS file system and volumes to VxFS and VxVM using SFHA Solutions conversion utilities. See "Types of data migration" on page 176.	Offline conversion utility See "Types of data migration" on page 176. Online migration utility
Convert your data from raw disk to VxFS: use SFHA Solutions. See "Types of data migration" on page 176.	Offline conversion utility See "Types of data migration" on page 176.
Migrate your data from one platform to another (server migration) using SFHA Solutions. See "Overview of the Cross-Platform Data Sharing (CDS) feature" on page 220.	Portable Data Containers See "Overview of the Cross-Platform Data Sharing (CDS) feature" on page 220.
Migrate your data across arrays using SFHA Solutions Portable Data Containers. See "Array migration for storage using Linux" on page 209.	Volume mirroring See "Overview of storage mirroring for migration" on page 210.
Plan a maintenance of virtual machines in a vSphere environment for a planned failover and recovery of application during unplanned failure using the Just In Time Availability solution.	Just In Time Availability solution See "About Just In Time Availability" on page 275.
Improve the native and optimized format of your storage devices using the Veritas InfoScale solution which provides support with the advanced format or 4K (4096 bytes) sector devices (formatted with 4KB) in storage environments.	Veritas InfoScale 4K sector device support solution See "About 4K sector size technology" on page 293. See "Veritas InfoScale unsupported configurations" on page 294. See "Migrating VxFS file system from 512-bytes sector size devices to 4K sector size devices" on page 295.

Table 2-1 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Multiple parallel applications in a data warehouse that require flexible sharing of data such as ETL pipeline, where output of one stage becomes input for the next stage. (for example, accounting system needs to combine data from different applications such as sales, payroll and purchasing)	Verita InfoScale application isolation See "Running multiple parallel applications within a single cluster using the application isolation feature" on page 25. More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.
Relax complete zoning requirement of SAN storage to all CVM nodes. This enables merging of independent clusters for better manageability.	Verita InfoScale application isolation See "Running multiple parallel applications within a single cluster using the application isolation feature" on page 25. More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Table 2-1 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Enabling multiple independent clustered applications to use a commonly shared pool of scalable DAS storage. This facilitates adding of storage-only nodes to cluster for growing storage capacity and compute nodes for dedicated application use.	Verita InfoScale application isolation See "Scaling FSS storage capacity with dedicated storage nodes using application isolation feature" on page 35. More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Feature support across Veritas InfoScale 7.4.1 products

Veritas InfoScale solutions and use cases for Oracle are based on the shared management features of Veritas InfoScale Storage Foundation and High Availability (SFHA) Solutions products. Clustering features are available separately through Cluster Server (VCS) as well as through the SFHA Solutions products.

Table 2-2 lists the features supported across SFHA Solutions products.

Table 2-2 Storage management features in Veritas InfoScale products

Storage management feature	Veritas InfoScale Foundation	Veritas InfoScale Storage	Veritas InfoScale Availability	Veritas InfoScale Enterprise
Veritas Extension for Oracle Disk Manager	N	Y	N	Y

Table 2-2 Storage management features in Veritas InfoScale products *(continued)*

Storage management feature	Veritas InfoScale Foundation	Veritas InfoScale Storage	Veritas InfoScale Availability	Veritas InfoScale Enterprise
Veritas Extension for Cached Oracle Disk Manager **Note:** Not supported for Oracle RAC.	N	Y	N	Y
Quick I/O **Note:** Not supported in Linux.	N	Y	N	Y
Cached Quick I/O **Note:** Not supported in Linux.	N	Y	N	Y
Compression	N	Y	N	Y
Deduplication	N	Y	N	Y
Flexible Storage Sharing	N	Y	N	Y
SmartIO **Note:** SFRAC does not support Writeback caching.	N	Y	N	Y
SmartMove	N	Y	N	Y
SmartTier	N	Y	N	Y
Thin Reclamation	N	Y	N	Y
Portable Data Containers	N	Y	N	Y
Database FlashSnap	N	Y	N	Y
Database Storage Checkpoints	N	Y	N	Y
FileSnap	N	Y	N	Y

Table 2-2 Storage management features in Veritas InfoScale products *(continued)*

Storage management feature	Veritas InfoScale Foundation	Veritas InfoScale Storage	Veritas InfoScale Availability	Veritas InfoScale Enterprise
Volume replication	N	Y	N	Y
File replication **Note:** Supported on Linux only.	N	Y	N	Y
Advanced support for virtual storage	Y	Y	Y	Y
Clustering features for high availability (HA)	N	N	Y	N
Disaster recovery features (HA/DR)	N	N	Y	N
Dynamic Multi-pathing	Y	N	Y	Y

Table 2-3 lists the high availability and disaster recovery features available in VCS.

Table 2-3 Availability management features in Veritas InfoScale SFHA solutions products

Availability management feature	VCS HA/DR
Clustering for high availability (HA)	Y
Database and application/ISV agents	Y
Advanced failover logic	Y
Data integrity protection with I/O fencing	Y
Advanced virtual machines support	Y
Virtual Business Services	Y
Replication agents	Y
Replicated Data Cluster	Y
Campus (stretch) cluster	Y
Global clustering (GCO)	Y

Table 2-3 Availability management features in Veritas InfoScale SFHA solutions products *(continued)*

Availability management feature	VCS HA/DR
Fire Drill	Y

- O=Feature is not included in your license but may be licensed separately.
- N=Feature is not supported with your license.

Notes:

- SmartTier is an expanded and renamed version of Dynamic Storage Tiering (DST).
- All features listed in Table 2-2 and Table 2-3 are supported on Linux except as noted. Consult specific product documentation for information on supported operating systems.
- Most features listed in Table 2-2 and Table 2-3 are supported on Linux virtual environments. For specific details, see the *Veritas InfoScale 7.4.1 Virtualization Guide Linux*.

Using SmartMove and Thin Provisioning with Sybase databases

You can use SmartMove and Thin Provisioning with Storage Foundation and High Availability products and Sybase databases.

When data files are deleted, you can reclaim the storage space used by these files if the underlying devices are thin reclaimable LUNs.

For information about the Storage Foundation Thin Reclamation feature, see the *Storage Foundation Administrator's Guide*.

Running multiple parallel applications within a single cluster using the application isolation feature

Customer scenario

Multiple parallel applications that require flexible sharing of data in a data warehouse are currently deployed on separate clusters. Access across clusters is provided by NFS or other distributed file system technologies. You want to deploy multiple parallel applications that require flexible sharing of data within a single cluster.

In a data center, multiple clusters exist with their dedicated fail over nodes.

There is a need to optimize the deployment of these disjoint clusters as a single large cluster.

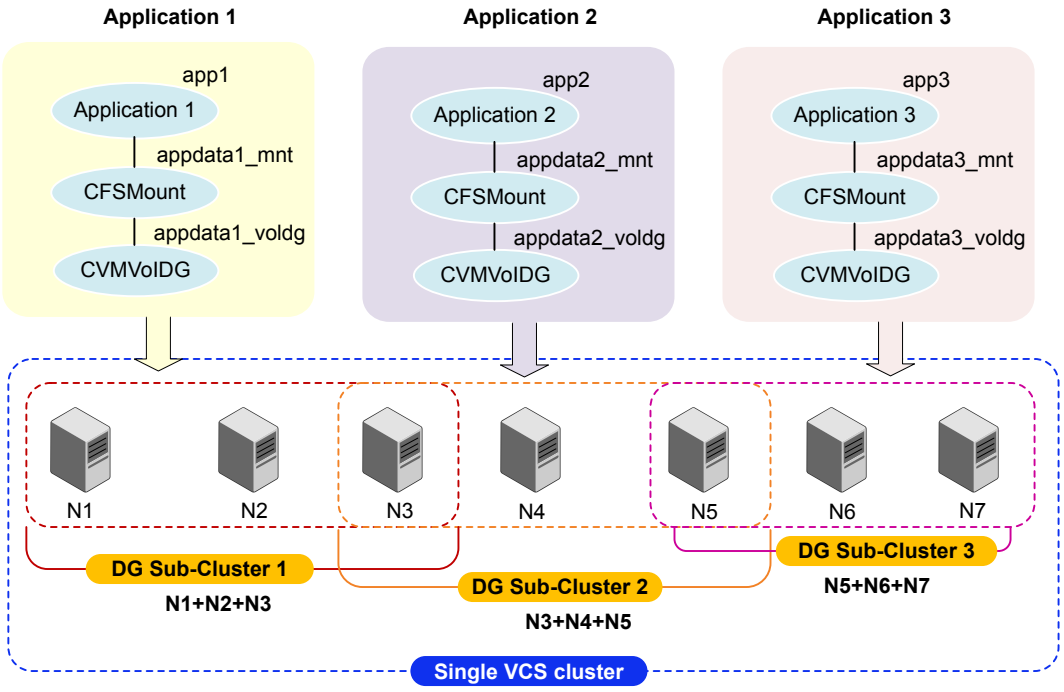
Configuration overview

Business critical applications require dedicated hardware to avoid the impact of configuration changes of one application on other applications. For example, when a node leaves or joins the cluster, it affects the cluster and the applications running on it. If multiple applications are configured on a large cluster, configuration changes have the potential to cause application downtime.

With the application isolation feature, Veritas InfoScale provides logical isolation between applications at the disk group boundary. This is very helpful when applications require occasional sharing of data. Data can be copied efficiently between applications by using Veritas Volume Manager snapshots and disk group split, join, or move operations. Updates to data can be optimally shared by copying only the changed data. Thus, existing configurations that have multiple applications on a large cluster can be made more resilient and scalable with the application isolation feature.

Visibility of disk groups can be limited only to the required nodes. Making disk group configurations available to a smaller set of nodes improves performance and scalability of Veritas Volume Manager configuration operations.

The following figure illustrates a scenario where three applications are logically isolated to operate from a specific set of nodes within a single large VCS cluster. This configuration can be deployed to serve any of the above mentioned scenarios.



Supported configuration

- Veritas InfoScale 7.2 and later
- Red Hat Enterprise Linux (RHEL) and supported RHEL-compatible distributions, SUSE Linux Enterprise Server (SLES) versions supported in this release

Reference documents

Storage Foundation Cluster File System High Availability Administrator's Guide

Storage Foundation for Oracle RAC Configuration and Upgrade Guide.

Solution

See "To run multiple parallel applications within a single Veritas InfoScale cluster using the application isolation feature" on page 27.

To run multiple parallel applications within a single Veritas InfoScale cluster using the application isolation feature

1. Install and configure Veritas InfoScale Enterprise 7.2 or later on the nodes.
2. Enable the application isolation feature in the cluster.

   Enabling the feature changes the import and deport behaviour. As a result, you must manually add the shared disk groups to the VCS configuration.

   See the topic "Enabling the application isolation feature in CVM environments" in the *Storage Foundation Cluster File System High Availability Administrator's Guide*.
3. Identify the shared disk groups on which you want to configure the applications.
4. Initialize the disk groups and create the volumes and file systems you want to use for your applications.

   Run the commands from any one of the nodes in the disk group sub-cluster. For example, if node1, node2, node3 belong to the sub-cluster `DGSubCluster1`, run the following commands from any one of the nodes: node1, node2, node3.

   Disk group sub-cluster 1:

   ```
   # vxdg -s init appdg1 disk1 disk2 disk3
   # vxassist -g appdg1 make appvol1 100g nmirror=2
   # mkfs -t vxfs /dev/vx/rdsk/appdg1/appvol1
   ```

   Disk group sub-cluster 2:

   ```
   # vxdg -s init appdg2 disk4 disk5 disk6
   # vxassist -g appdg2 make appvol2 100g nmirror=2
   # mkfs -t vxfs /dev/vx/rdsk/appdg2/appvol2
   ```

   Disk group sub-cluster 3:

   ```
   # vxdg -s init appdg3 disk7 disk8 disk9
   # vxassist -g appdg3 make appvol3 100g nmirror=2
   # mkfs -t vxfs /dev/vx/rdsk/appdg3/appvol3
   ```

5 Configure the OCR, voting disk, and CSSD resources on all nodes in cluster. It is recommended to have a mirror of the OCR and voting disk on each node in the cluster.

For instructions, see the Section "Installation and upgrade of Oracle RAC" in the *Storage Foundation for Oracle RAC Configuration and Upgrade Guide.*

6 Configure application `app1` on node1, node2 and node3.

The following commands add the application app1 to the VCS configuration.

```
# hagrp -add app1
# hagrp -modify app1 SystemList  node1 0 node2 1 node3 2
# hagrp -modify app1 AutoFailOver 0
# hagrp -modify app1 Parallel 1
# hagrp -modify app1 AutoStartList  node1 node2 node3
```

Add disk group resources to the VCS configuration.

```
# hares -add appdg1_voldg CVMVolDg app1
# hares -modify appdg1_voldg Critical 0
# hares -modify appdg1_voldg CVMDiskGroup appdg1
# hares -modify appdg1_voldg CVMVolume  appvol1
```

Change the activation mode of the shared disk group to shared-write.

```
# hares -local appdg1_voldg CVMActivation
# hares -modify appdg1_voldg NodeList  node1 node2 node3
# hares -modify appdg1_voldg CVMActivation sw
# hares -modify appdg1_voldg Enabled 1
```

Add the CFS mount resources for the application to the VCS configuration.

```
# hares -add appdata1_mnt CFSMount app1
# hares -modify appdata1_mnt Critical 0
# hares -modify appdata1_mnt MountPoint "/appdata1_mnt"
# hares -modify appdata1_mnt BlockDevice "/dev/vx/dsk/appdg1/appvol1"
# hares -local appdata1_mnt MountOpt
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node1
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node2
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node3
# hares -modify appdata1_mnt NodeList  node1 node2 node3
# hares -modify appdata1_mnt Enabled 1
```

Add the application's oracle database to the VCS configuration.

```
# hares -add ora_app1 Oracle app1
# hares -modify ora_app1 Critical 0
# hares -local ora_app1 Sid
# hares -modify ora_app1 Sid app1_db1 -sys node1
# hares -modify ora_app1 Sid app1_db2 -sys node2
# hares -modify ora_app1 Sid app1_db3 -sys node3
# hares -modify ora_app1 Owner oracle
```

```
# hares -modify ora_app1 Home "/u02/app/oracle/dbhome"
# hares -modify ora_app1 StartUpOpt SRVCTLSTART
# hares -modify ora_app1 ShutDownOpt SRVCTLSTOP
# hares -modify ora_app1 DBName app1_db
```

7 Configure application `app2` on node3, node4 and node5.

. The following commands add the application app2 to the VCS configuration.

```
# hagrp -add app2
# hagrp -modify app2 SystemList  node3 0 node4 1 node5 2
# hagrp -modify app2 AutoFailOver 0
# hagrp -modify app2 Parallel 1
# hagrp -modify app2 AutoStartList  node3 node4 node5
```

Add disk group resources to the VCS configuration.

```
# hares -add appdg2_voldg CVMVolDg app2
# hares -modify appdg2_voldg Critical 0
# hares -modify appdg2_voldg CVMDiskGroup appdg2
# hares -modify appdg2_voldg CVMVolume  appvol2
```

Change the activation mode of the shared disk group to shared-write.

```
# hares -local appdg2_voldg CVMActivation
# hares -modify appdg2_voldg NodeList  node3 node4 node5
# hares -modify appdg2_voldg CVMActivation sw
# hares -modify appdg2_voldg Enabled 1
```

Add the CFS mount resources for the application to the VCS configuration.

```
# hares -add appdata2_mnt CFSMount app2
# hares -modify appdata2_mnt Critical 0
# hares -modify appdata2_mnt MountPoint "/appdata2_mnt"
# hares -modify appdata2_mnt BlockDevice "/dev/vx/dsk/appdg2/appvol2"
# hares -local appdata2_mnt MountOpt
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node3
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node4
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node5
# hares -modify appdata2_mnt NodeList  node3 node4 node5
# hares -modify appdata2_mnt Enabled 1
```

Add the application's oracle database to the VCS configuration.

```
# hares -add ora_app2 Oracle app2
# hares -modify ora_app2 Critical 0
# hares -local ora_app2 Sid
# hares -modify ora_app2 Sid app2_db1 -sys node3
# hares -modify ora_app2 Sid app2_db2 -sys node4
# hares -modify ora_app2 Sid app2_db3 -sys node5
# hares -modify ora_app2 Owner oracle
```

```
# hares -modify ora_app2 Home "/u02/app/oracle/dbhome"
# hares -modify ora_app2 StartUpOpt SRVCTLSTART
```

```
# hares -modify ora_app2 ShutDownOpt SRVCTLSTOP
# hares -modify ora_app2 DBName app2_db
```

8 Configure application `app3` on node5, node6 and node7.

. The following commands add the application app3 to the VCS configuration.

```
# hagrp -add app3
# hagrp -modify app3 SystemList  node5 0 node6 1 node7 2
# hagrp -modify app3 AutoFailOver 0
# hagrp -modify app3 Parallel 1
# hagrp -modify app3 AutoStartList  node5 node6 node7
```

Add disk group resources to the VCS configuration.

```
# hares -add appdg3_voldg CVMVolDg app3
# hares -modify appdg3_voldg Critical 0
# hares -modify appdg3_voldg CVMDiskGroup appdg3
# hares -modify appdg3_voldg CVMVolume  appvol3
```

Change the activation mode of the shared disk group to shared-write.

```
# hares -local appdg3_voldg CVMActivation
# hares -modify appdg3_voldg NodeList  node5 node6 node7
# hares -modify appdg3_voldg CVMActivation sw
# hares -modify appdg3_voldg Enabled 1
```

Add the CFS mount resources for the application to the VCS configuration.

```
# hares -add appdata3_mnt CFSMount app3
# hares -modify appdata3_mnt Critical 0
# hares -modify appdata3_mnt MountPoint "/appdata3_mnt"
# hares -modify appdata3_mnt BlockDevice "/dev/vx/dsk/appdg3/appvol3"
# hares -local appdata3_mnt MountOpt
# hares -modify appdata3_mnt MountOpt "rw,cluster" -sys node5
# hares -modify appdata3_mnt MountOpt "rw,cluster" -sys node6
# hares -modify appdata3_mnt MountOpt "rw,cluster" -sys node7
# hares -modify appdata3_mnt NodeList  node5 node6 node7
# hares -modify appdata3_mnt Enabled 1
```

Add the application's oracle database to the VCS configuration.

```
# hares -add ora_app3 Oracle app3
# hares -modify ora_app3 Critical 0
# hares -local ora_app3 Sid
# hares -modify ora_app3 Sid app3_db1 -sys node5
# hares -modify ora_app3 Sid app3_db2 -sys node6
# hares -modify ora_app3 Sid app3_db3 -sys node7
# hares -modify ora_app3 Owner oracle
```

```
# hares -modify ora_app3 Home "/u02/app/oracle/dbhome"
# hares -modify ora_app3 StartUpOpt SRVCTLSTART
# hares -modify ora_app3 ShutDownOpt SRVCTLSTOP
# hares -modify ora_app3 DBName app3_db
```

Scaling FSS storage capacity with dedicated storage nodes using application isolation feature

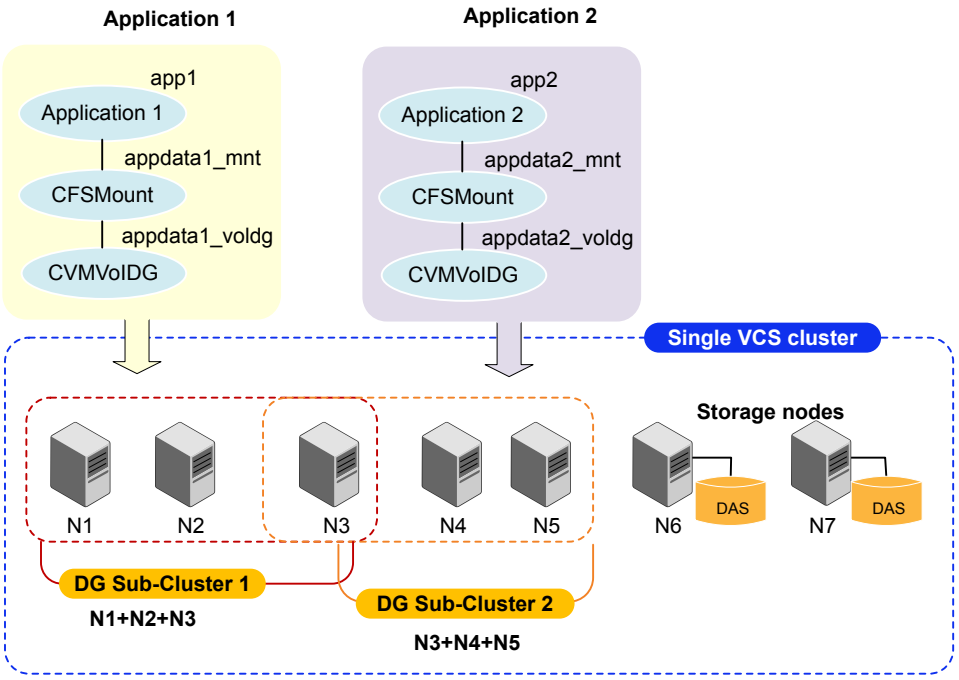
Customer scenario

Shared-nothing architectures rely on network infrastructure instead of Storage Area Networks (SAN) to provide access to shared data. With the Flexible Storage Sharing feature of Veritas InfoScale, high performance clustered applications can get rid of the complexity and cost of SAN storage while still providing access to the shared name space requirement of clustered applications.

Configuration overview

In the traditional clustered volume manager (CVM) environment, the shared disk groups are imported on all cluster nodes. As a result, it was difficult to increase storage capacity by adding more storage nodes without scaling the application. With application isolation and flexible storage sharing (FSS), it is now possible to add nodes and create a pool of storage to use them across multiple clustered applications. This completely eliminates the need for SAN storage in data centers allowing ease of use in addition to significant cost reductions.

The following figure illustrates a scenario where two applications are configured on a specific set of nodes in the cluster. Two storage nodes are contributing their DAS storage to the applications.



Supported configuration

- Veritas InfoScale 7.2 and later
- Red Hat Enterprise Linux (RHEL) and supported RHEL-compatible distributions, SUSE Linux Enterprise Server (SLES) versions supported in this release

Reference documents

Storage Foundation Cluster File System High Availability Administrator's Guide

Storage Foundation for Oracle RAC Configuration and Upgrade Guide.

Solution

See "To scale FSS storage capacity with dedicated storage nodes using application isolation feature" on page 37.

The commands in the procedure assume the use of clustered application Oracle RAC. Other supported clustered applications can be similarly configured.

To scale FSS storage capacity with dedicated storage nodes using application isolation feature

1. Install and configure Veritas InfoScale Enterprise 7.2 or later on the nodes.
2. Enable the application isolation feature in the cluster.

   Enabling the feature changes the import and deport behaviour. As a result, you must manually add the shared disk groups to the VCS configuration.

   See the topic "Enabling the application isolation feature in CVM environments" in the *Storage Foundation Cluster File System High Availability Administrator's Guide*.
3. Export the DAS storage from each storage node. Run the command on the node from which you are exporting the disk.

   Veritas InfoScale Storage supports auto-mapping of storage. You can obtain disks directly from the output of `vxdisk -o cluster list` command instead of exporting disks from each storage node.

   ```
   # vxdisk export node6_disk1 node6_disk2 \
   node6_disk3 node6_disk4
   # vxdisk export node7_disk1 node7_disk2 \
   node7_disk3 node7_disk4
   ```
4. Identify the shared disk groups on which you want to configure the applications.
5. Initialize the disk groups and create the volumes and file systems you want to use for your applications.

   Run the following commands from any one of the nodes in the disk group sub-cluster. For example, if node1 and node2 belong to the sub-cluster `DGSubCluster1`, run the following commands from any one of the nodes: node1 or node2.

   Disk group sub-cluster 1:

   ```
   # vxdg -o fss -s init appdg1 node6_disk1 \
   node6_disk2 node7_disk1 node7_disk2
   # vxassist -g appdg1 make appvol1 100g nmirror=2
   # mkfs -t vxfs /dev/vx/rdsk/appdg1/appvol1
   ```

   Disk group sub-cluster 2:

   ```
   # vxdg -o fss -s init appdg2 node6_disk3 \
   node6_disk4 node7_disk3 node7_disk4
   # vxassist -g appdg2 make appvol2 100g nmirror=2
   # mkfs -t vxfs /dev/vx/rdsk/appdg2/appvol2
   ```

6 Configure the OCR, voting disk, and CSSD resources on all nodes in cluster. It is recommended to have a mirror of the OCR and voting disk on each node in the cluster.

For instructions, see the Section "Installation and upgrade of Oracle RAC" in the *Storage Foundation for Oracle RAC Configuration and Upgrade Guide*..

7 Configure application `app1` on node1, node2 and node3.

. The following commands add the application app1 to the VCS configuration.

```
# hagrp -add app1
# hagrp -modify app1 SystemList  node1 0 node2 1 node3 2
# hagrp -modify app1 AutoFailOver 0
# hagrp -modify app1 Parallel 1
# hagrp -modify app1 AutoStartList  node1 node2 node3
```

Add disk group resources to the VCS configuration.

```
# hares -add appdg1_voldg CVMVolDg app1
# hares -modify appdg1_voldg Critical 0
# hares -modify appdg1_voldg CVMDiskGroup appdg1
# hares -modify appdg1_voldg CVMVolume  appvol1
```

Change the activation mode of the shared disk group to shared-write.

```
# hares -local appdg1_voldg CVMActivation
# hares -modify appdg1_voldg NodeList  node1 node2 node3
# hares -modify appdg1_voldg CVMActivation sw
# hares -modify appdg1_voldg Enabled 1
```

Add the CFS mount resources for the application to the VCS configuration.

```
# hares -add appdata1_mnt CFSMount app1
# hares -modify appdata1_mnt Critical 0
# hares -modify appdata1_mnt MountPoint "/appdata1_mnt"
# hares -modify appdata1_mnt BlockDevice "/dev/vx/dsk/appdg1/appvol1"
# hares -local appdata1_mnt MountOpt
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node1
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node2
# hares -modify appdata1_mnt MountOpt "rw,cluster" -sys node3
# hares -modify appdata1_mnt NodeList  node1 node2 node3
# hares -modify appdata1_mnt Enabled 1
```

Add the application's oracle database to the VCS configuration.

```
# hares -add ora_app1 Oracle app1
# hares -modify ora_app1 Critical 0
# hares -local ora_app1 Sid
# hares -modify ora_app1 Sid app1_db1 -sys node1
# hares -modify ora_app1 Sid app1_db2 -sys node2
# hares -modify ora_app1 Sid app1_db3 -sys node3
# hares -modify ora_app1 Owner oracle
```

```
# hares -modify ora_app1 Home "/u02/app/oracle/dbhome"
# hares -modify ora_app1 StartUpOpt SRVCTLSTART
```

```
# hares -modify ora_app1 ShutDownOpt SRVCTLSTOP
# hares -modify ora_app1 DBName app1_db
```

8 Configure application `app2` on node3, node4 and node5.

. The following commands add the application app2 to the VCS configuration.

```
# hagrp -add app2
# hagrp -modify app2 SystemList  node3 0 node4 1 node5 2
# hagrp -modify app2 AutoFailOver 0
# hagrp -modify app2 Parallel 1
# hagrp -modify app2 AutoStartList  node3 node4 node5
```

Add disk group resources to the VCS configuration.

```
# hares -add appdg2_voldg CVMVolDg app2
# hares -modify appdg2_voldg Critical 0
# hares -modify appdg2_voldg CVMDiskGroup appdg2
# hares -modify appdg2_voldg CVMVolume  appvol2
```

Change the activation mode of the shared disk group to shared-write.

```
# hares -local appdg2_voldg CVMActivation
# hares -modify appdg2_voldg NodeList  node3 node4 node5
# hares -modify appdg2_voldg CVMActivation sw
# hares -modify appdg2_voldg Enabled 1
```

Add the CFS mount resources for the application to the VCS configuration.

```
# hares -add appdata2_mnt CFSMount app2
# hares -modify appdata2_mnt Critical 0
# hares -modify appdata2_mnt MountPoint "/appdata2_mnt"
# hares -modify appdata2_mnt BlockDevice "/dev/vx/dsk/appdg2/appvol2"
# hares -local appdata2_mnt MountOpt
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node3
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node4
# hares -modify appdata2_mnt MountOpt "rw,cluster" -sys node5
# hares -modify appdata2_mnt NodeList  node3 node4 node5
# hares -modify appdata2_mnt Enabled 1
```

Add the application's oracle database to the VCS configuration.

```
# hares -add ora_app2 Oracle app2
# hares -modify ora_app2 Critical 0
# hares -local ora_app2 Sid
# hares -modify ora_app2 Sid app2_db1 -sys node3
# hares -modify ora_app2 Sid app2_db2 -sys node4
# hares -modify ora_app2 Sid app2_db3 -sys node5
# hares -modify ora_app2 Owner oracle
```

```
# hares -modify ora_app2 Home "/u02/app/oracle/dbhome"
# hares -modify ora_app2 StartUpOpt SRVCTLSTART
# hares -modify ora_app2 ShutDownOpt SRVCTLSTOP
# hares -modify ora_app2 DBName app2_db
```

Finding Veritas InfoScale product use cases information

The following Storage Foundation and High Availability Solutions management features are illustrated with use case examples in this guide:

- Improving database performance
- Backing up and recovering your data
- Processing data off-host
- Optimizing test and development environments
- Maximizing storage utilization
- Converting your data from native OS to VxFS
- Converting your data from raw disk to VxFS
- Migrating your data from one platform to another (server migration)
- Migrating your data across arrays

For Storage Foundation and High Availability Solutions management features concept and administrative information, see the following guides:

- *Storage Foundation Administrator's Guide*.
- *Storage Foundation Cluster File System High Availability Administrator's Guide*.
- *Storage Foundation for Oracle RAC Administrator's Guide*.
- *Storage Foundation for Sybase ASE CE Administrator's Guide*.
- *Veritas InfoScale SmartIO for Solid-State Drives Solutions Guide*.

For Information on using Storage Foundation and High Availability Solutions management features with Oracle databases, see *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

For Information on using Storage Foundation and High Availability Solutions management features with DB2 databases, see: *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

For Information on using Storage Foundation and High Availability Solutions replication features, see *Veritas InfoScale Replication Administrator's Guide*.

Section 3

Stack-level migration to IPv6 or dual stack

- Chapter 3. Stack-level migration to IPv6 or dual stack

Chapter 3

Stack-level migration to IPv6 or dual stack

This chapter includes the following topics:

- Migrating Veritas InfoScale products to support IPv6/dual-stack

Migrating Veritas InfoScale products to support IPv6/dual-stack

Each new InfoScale product consists of one or more components. Each component within a product offers a unique capability that you can configure for use in your environment.

Most of the components that make up the InfoScale products are dependent on each other for various functions. Therefore, when you migrate those components to support IPv6 or dual-stack, you must follow a certain sequence.

InfoScale products can be installed and configured in several ways depending on the requirements of your IT environment. You can choose to deploy only those components that provide the features that you need to use. The following topics describe how to migrate all the components that the InfoScale product suite comprises. You can focus on the migration tasks for the components that are configured in your environment and ignore the rest.

For example, the following figure depicts the basic implementation workflow for InfoScale Enterprise. In addition to these components, your environment may also have components like Oracle RAC, ASM, and so on.

Figure 3-1 Basic implementation workflow for InfoScale Enterprise

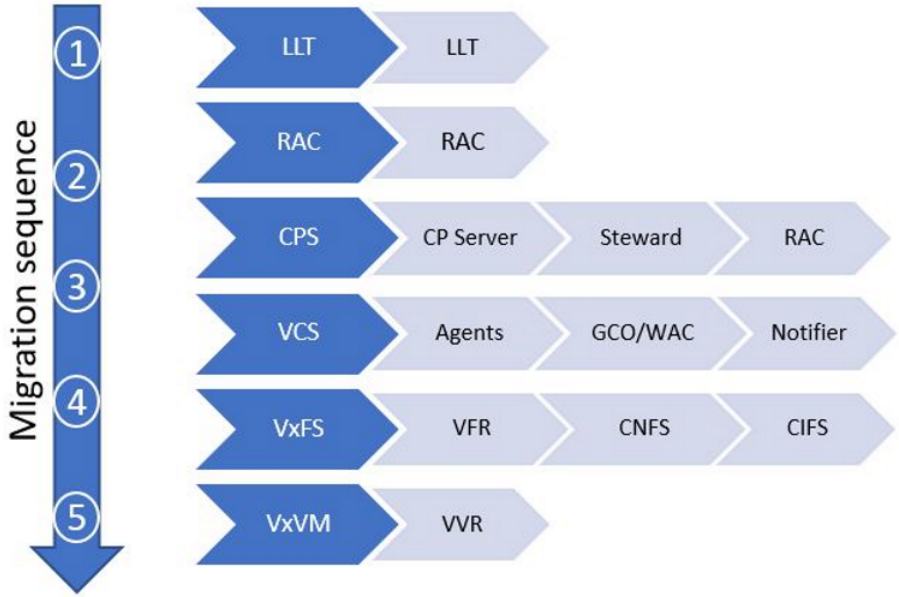


Note: The sequence in which you migrate the InfoScale components is different than the sequence in which they are implemented.

To migrate InfoScale products to support IPv6/dual-stack

Follow this sequence when you migrate the InfoScale components so that their dependencies are addressed successfully. If some of the components listed here are not present in your deployment, proceed to migrate the next component in the sequence.

Figure 3-2 Sequence for migrating the InfoScale components to Ipv6/dual-stack



The following table lists the documents that you can refer to for details on how to migrate each component to IPv6/dual-stack:

Table 3-1 Reference documentation

Migration priority	Module	Feature	Documentation reference
1	LLT	LLT	For details, see the *Cluster Server Configuration and Upgrade Guide*
2	CPS	**1** CP Server **2** Steward **3** RAC	For details, see the following guides: CP Server: *Cluster Server Administrator's Guide* Steward: *Veritas InfoScale Disaster Recovery Implementation Guide*
3	VCS	**1** Agent **2** GCO/WAC **3** Notifier	For details, see the following guides: Agents: *Cluster Server Bundled Agent's Reference Guide* GCO/WAC: *Cluster Server Administrator's Guide*
4	VxFS	**1** VFR **2** CNFS **3** CIFS	For details, see the following guides: VFR: *Veritas InfoScale Replication Administrator's Guide* CNFS/CIFS: *Storage Foundation Cluster File System High Availability Administrator's Guide*
5	VxVM	VVR	For details, see the *Veritas InfoScale Replication Administrator's Guide*

Section 4

Improving database performance

- Chapter 4. Overview of database accelerators
- Chapter 5. Improving database performance with Veritas Concurrent I/O
- Chapter 6. Improving database performance with atomic write I/O

Chapter 4

Overview of database accelerators

This chapter includes the following topics:

- About Veritas InfoScale product components database accelerators

About Veritas InfoScale product components database accelerators

The major concern in any environment is maintaining respectable performance or meeting performance service level agreements (SLAs). Veritas InfoScale product components improve the overall performance of database environments in a variety of ways.

Table 4-1 Veritas InfoScale product components database accelerators

Veritas InfoScale database accelerator	Supported databases	Use cases and considerations
Oracle Disk Manager (ODM)	Oracle	■ To improve Oracle performance and manage system bandwidth through an improved Application Programming Interface (API) that contains advanced kernel support for file I/O. ■ To use Oracle Resilvering and turn off Veritas Volume Manager Dirty Region Logging (DRL) to increase performance, use ODM. ■ To reduce the time required to restore consistency, freeing more I/O bandwidth for business-critical applications, use SmartSync recovery accelerator.
Cached Oracle Disk Manager (Cached ODM)	Oracle	To enable selected I/O to use caching to improve ODM I/O performance, use Cached ODM.
Concurrent I/O	DB2 Sybase	Concurrent I/O (CIO) is optimized for DB2 and Sybase environments To achieve improved performance for databases run on VxFS file systems without restrictions on increasing file size, use Veritas InfoScale Concurrent I/O.

These database accelerator technologies enable database performance equal to raw disk partitions, but with the manageability benefits of a file system. With the Dynamic Multi-pathing (DMP) feature of Storage Foundation, performance is maximized by load-balancing I/O activity across all available paths from server to array. DMP supports all major hardware RAID vendors, hence there is no need for third-party multi-pathing software, reducing the total cost of ownership.

Veritas InfoScale database accelerators enable you to manage performance for your database with more precision.

For details about using ODM and Cached ODM for Oracle, see *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

For details about using Concurrent I/O for DB2, see *Veritas InfoScale Storage and Availability Management for DB2 Databases*.

Chapter 5

Improving database performance with Veritas Concurrent I/O

This chapter includes the following topics:

- About Concurrent I/O
- Tasks for enabling and disabling Concurrent I/O

About Concurrent I/O

Concurrent I/O improves the performance of regular files on a VxFS file system. This simplifies administrative tasks and allows databases, which do not have a sequential read/write requirement, to access files concurrently. This chapter describes how to use the Concurrent I/O feature.

How Concurrent I/O works

Traditionally, Linux semantics require that read and write operations on a file occur in a serialized order. Because of this, a file system must enforce strict ordering of overlapping read and write operations. However, databases do not usually require this level of control and implement concurrency control internally, without using a file system for order enforcement.

The Concurrent I/O feature removes these semantics from the read and write operations for databases and other applications that do not require serialization.

The benefits of using Concurrent I/O are:

- Concurrency between a single writer and multiple readers

- Concurrency among multiple writers
- Minimalization of serialization for extending writes
- All I/Os are direct and do not use file system caching
- I/O requests are sent directly to file systems
- Inode locking is avoided

Tasks for enabling and disabling Concurrent I/O

Concurrent I/O is not turned on by default and must be enabled manually. You will also have to manually disable Concurrent I/O if you choose not to use it in the future.

You can perform the following tasks:

- Enable Concurrent I/O
- Disable Concurrent I/O

Enabling Concurrent I/O for Sybase

Because you do not need to extend name spaces and present the files as devices, you can enable Concurrent I/O on regular files.

Before enabling Concurrent I/O, review the following:

Prerequisites

- To use the Concurrent I/O feature, the file system must be a VxFS file system.
- Make sure the mount point on which you plan to mount the file system exists.
- Make sure the DBA can access the mount point.

To enable Concurrent I/O on a file system using mount with the -o cio option

- Mount the file system using the `mount` command as follows:

```
# /usr/sbin/mount -t vxfs -o cio special /mount_point
```

where:

- *special* is a block special device.
- /*mount_point* is the directory where the file system will be mounted.

For example for Sybase, to mount a file system named `/datavol`on a mount point named `/sybasedata`:

```
# /usr/sbin/mount -t vxfs -o cio /dev/vx/dsk/sybasedg/datavol \
/sybasedata
```

The following is an example of mounting a directory (where the new SMS containers are located) to use Concurrent I/O.

To mount an SMS container named `/container1` on a mount point named `/mysms`:

```
# /usr/sbin/mount -Vt namefs -o cio /datavol/mysms/container1 /mysms
```

Disabling Concurrent I/O for Sybase

If you need to disable Concurrent I/O, unmount the VxFS file system and mount it again without the mount option.

To disable Concurrent I/O on a file system using the mount command

1. Shutdown the Sybase instance.
2. Unmount the file sytem using the `umount` command.
3. Mount the file system again using the `mount` command without using the `-o cio` option.

Chapter 6

Improving database performance with atomic write I/O

This chapter includes the following topics:

- About the atomic write I/O
- Requirements for atomic write I/O
- Restrictions on atomic write I/O functionality
- How the atomic write I/O feature of Storage Foundation helps MySQL databases
- VxVM and VxFS exported IOCTLs
- Configuring atomic write I/O support for MySQL on VxVM raw volumes
- Configuring atomic write I/O support for MySQL on VxFS file systems
- Dynamically growing the atomic write capable file system
- Disabling atomic write I/O support

About the atomic write I/O

Standard block devices provide atomicity of the device sector size. The Fusion ioMemory card support atomic write I/O which provides atomicity for an I/O operation, even if it spans sectors of the device. Atomic write I/O ensures that all the blocks that are mentioned in the operation are written successfully on the device, or none of the blocks are written. Veritas leverages this capability of Fusion ioMemory card for Veritas file systems and volumes.

Requirements for atomic write I/O

Atomic write I/O is only supported for RHEL 6.X Linux distributions on which atomic-write supported firmware and ioMemory-VSL stack is available from SanDisk.

Creating an atomic write capable volume requires the disk group version 200 or later.

In addition, the following requirements apply:

- Fusion ioMemory card with Firmware and VSL stack version 3.3.3 or later.
- Atomic write I/O support must be enabled on the hardware side. The supported hardware listed in the ioMemory-VSL-3.3.3 release notes are expected to work for this feature.

Restrictions on atomic write I/O functionality

This section describes the limitations of the atomic write I/O feature.

When atomic write I/O support is configured for VxVM raw volumes, the following limitations apply:

- This functionality is not supported in CVM, FSS, VVR, or SmartIO environment.
- Atomic write I/O is supported on concatenated volume layouts only.
- Write I/O spanning across the atomic write I/O boundary is not supported.
- Vector atomic write I/O is not supported.
- Snapshot and mirroring of atomic write capable volume is not supported.

When atomic write I/O support is configured for VxFS file systems, the above limitations apply along with the following additional limitations:

- FileSnap is not supported on an atomic capable volume.
- Vector atomic write I/O is not supported.
- Atomic writes are not supported on writeable clones. Promotion of writeable clones to primary is not supported when the file system resides on an atomic write enabled volume.
- The “contig” option to setext is not honored. Similarly, extent size and reservation sizes that are not a multiple of the atomic write size are not honored.
- dd copy of a file-system from a non-atomic capable volume to an atomic-capable volume is not supported.
- Writes will return the error code ENOTSUP in the following cases:

- The starting file offset is not aligned to a 512-byte boundary.
- The ending file offset is not aligned to a 512-byte boundary, or the length is not a multiple of 512 bytes.
- The memory buffer does not start on a 512-byte boundary.
- The I/O straddles an atomic write (typically 16K) boundary. To determine the atomic write size, use the following command:

  ```
  # vxprint -g diskgroup -m volume
  ```

  An example of an atomic write that straddles a 16K boundary is one with offset 15K and length 2K.
- The length exceeds the atomic write size typically 16K.

How the atomic write I/O feature of Storage Foundation helps MySQL databases

Database applications are required to maintain Atomicity, Consistency, Isolation, Durability (ACID) properties for data integrity. The InnoDB storage engine of MySQL writes twice to achieve atomicity: once to the double write buffer and once to the actual tablespace. With an atomic write I/O, the writes to the double write buffer can be avoided, resulting in better performance and longer lifetime of the SSD.

Storage Foundation supports atomic write I/O in the following situations:

- directly on raw VxVM volumes
- on VxFS file systems on top of VxVM volumes
  This scenario supports the MySQL capability of auto-extending the configured databases dynamically. If the database files consume all of the space on the file system, then you can grow the underlying file system and volume dynamically.
  See "Dynamically growing the atomic write capable file system" on page 62.

VxVM and VxFS exported IOCTLs

Veritas Volume Manager (VxVM) and Veritas File System (VxFS) export the following IOCTLs for controlling atomic write capability on volumes and VxFS file systems. Applications can use the following IOCTLs:

- DFS_IOCTL_ATOMIC_WRITE_SET:
  A MySQL-specific IOCTL for VxVM volumes, which instructs VxVM that all further write IO on this volume should be treated as atomic writes.
- VOL_SET_ATOMIC_WRITE:

An IOCTL exported by VxVM, which behaves the same as DFS_IOCTL_ATOMIC_WRITE_SET.

- VOL_GET_ATOMIC_WRITE:
  An IOCTL that reports if the volume supports atomic write or not.
- VX_ATM_WR
  A cache advisory added to VxFS. This advisory requires the file to be opened with O_DIRECT, or the VX_DIRECT or VX_CONCURRENT advisory to be set or the file system to be mounted with the concurrent I/O (CIO) option. This advisory returns EINVAL if none of the constraints are met.

Configuring atomic write I/O support for MySQL on VxVM raw volumes

This section describes installing and configuring steps to use MySQL with atomic write support on raw VxVM volumes.

Enabling the atomic write I/O support for MySQL on VxVM raw volumes

1. Install the Fusion ioMemory card and enable atomic write support on the SSD.

   For information, see the SanDisk documentation.

2. Bring the SanDisk devices under VxVM control, as follows:

   - Discover the devices:

     ```
     # vxdisk scandisks
     ```

   - Display the devices that are available for VxVM use:

     ```
     # vxdisk list
     ```

     For example:

     ```
     # vxdisk list
     DEVICE      TYPE      DISK  GROUP STATUS
     fiodrive0_1 auto:none -     -     online invalid ssdtrim atomic-write
     ```

   - Initialize the disks:

     ```
     # /etc/vx/bin/vxdisksetup -i fio_device
     ```

   - Verify that the disks are under VxVM control and have atomic write support:

     ```
     # vxdisk list
     ```

For example:

```
# vxdisk list
DEVICE      TYPE          DISK  GROUP STATUS
fiodrive0_1 auto:cdsdisk  -     -     online ssdtrim atomic-write
```

3. Add the device to a disk group. The disk group can include both SSDs and HDDs.
   - If you do not have a disk group, create the disk group:

     ```
     # vxdg init diskgroup dev1=fiodrive0_1
     ```

   - If you already have a disk group, add the device to the disk group:

     ```
     # vxdg -g diskgroup adddisk fiodrive0_1
     ```

4. Create the atomic write capable volume:

   ```
   # vxassist -A -g diskgroup make volume length mediatype:ssd
   ```

   Where:

   the -A option creates an atomic write capable volume of concatenated layout, on the atomic write capable disks.

5. Verify that the volume is atomic write capable:

   ```
   # vxprint -g diskgroup -m volume \
   | grep atomic
   atomic_wr_capable=on
   atomic_wr_iosize=16
   ```

   Where:

   `atomic_wr_capable` attribute indicates whether or not the volume supports atomic writes

   `atomic_wr_iosize` indicates the supported size of the atomic write I/O.

6. Configure the MySQL application with atomic write I/O support.
7. Configure the MySQL application to place the database on the atomic write capable volume.
8. Start the MySQL application.

Configuring atomic write I/O support for MySQL on VxFS file systems

This section describes installing and configuring steps to use MySQL with atomic write support for VxFS file systems on VxVM volumes.

Enabling the atomic write I/O support for MySQL for VxFS file systems on VxVM volumes

1. Install the Fusion ioMemory card and enable atomic write support on the SSD.

   For information, see the SanDisk documentation.

2. Bring the SanDisk devices under VxVM control, as follows:

   - Discover the devices:

     ```
     # vxdisk scandisks
     ```

   - Display the devices that are available for VxVM use:

     ```
     # vxdisk list
     ```

     For example:

     ```
     # vxdisk list
     DEVICE      TYPE      DISK GROUP STATUS
     fiodrive0_1 auto:none -    -     online invalid ssdtrim atomic-write
     ```

   - Initialize the disks:

     ```
     # /etc/vx/bin/vxdisksetup -i SanDisk_device
     ```

   - Verify that the disks are under VxVM control and have atomic write support:

     ```
     # vxdisk list
     ```

     For example:

     ```
     # vxdisk list
     DEVICE      TYPE          DISK  GROUP STATUS
     fiodrive0_1 auto:cdsdisk  -     -     online ssdtrim atomic-write
     ```

3. Add the device to a disk group. The disk group can include both SSDs and HDDs.

   - If you do not have a disk group, create the disk group:

```
# vxdg init diskgroup dev1=fiodrive0_1
```

- If you already have a disk group, add the device to the disk group:

```
# vxdg -g diskgroup adddisk fiodrive0_1
```

4. Create the atomic write capable volume:

```
# vxassist -A -g diskgroup make volume length mediatype:ssd
```

Where:

the -A option creates an atomic write capable volume of concatenated layout, on the atomic write capable disks.

5. Verify that the volume is atomic write capable:

```
# vxprint -g diskgroup -m volume | grep atomic
atomic_wr_capable=on
atomic_wr_iosize=16
```

Where:

`atomic_wr_capable` attribute indicates whether or not the volume supports atomic writes

`atomic_wr_iosize` indicates the supported size of the atomic write I/O.

6. Create a VxFS file system over the atomic write capable volume.

```
# mkfs.vxfs /dev/vx/rdsk/diskgroup/volume
```

7. Mount the file system at an appropriate location:

```
# mount.vxfs /dev/vx/dsk/diskgroup/volume /mnt1
```

8. Configure the MySQL application with atomic write I/O support.
9. Configure the MySQL application to place the data file on the VxFS mount point.

10 Start the MySQL server.

11 Verify that MySQL is running with atomic write support using the following query:

```
# mysql MariaDB [(none)]> select @@innodb_use_atomic_writes ;
+----------------------------+
| @@innodb_use_atomic_writes|
+----------------------------+
|                          1|
+----------------------------+
1 row in set (0.00 sec)
```

Dynamically growing the atomic write capable file system

If the file system hosting the MySQL database files runs out of space, you can dynamically grow the atomic write capable volume with the VxFS file system.

To dynamically grow the atomic write capable volume with the VxFS file system

1 Add atomic write capable disks to the disk group.

2 Resize the atomic write capable volume together with the VxFS file system.

```
# /etc/vx/bin/vxresize -F vxfs -g diskgroup  volume
   newlength mediatype:ssd
```

Disabling atomic write I/O support

You do not have to disable atomic write support at the Veritas Volume Manager volume or Veritas File System level. Disable atomic write I/O from the MySQL application.

The volume remains ready to be used for atomic write I/O, whenever atomic write I/O is enabled again from the MySQL application.

For information about configuring MySQL server and atomic write I/O support in MySQL, see the MySQL documentation.

Section 5

Using point-in-time copies

- Chapter 7. Understanding point-in-time copy methods
- Chapter 8. Backing up and recovering
- Chapter 9. Backing up and recovering in a NetBackup environment
- Chapter 10. Off-host processing
- Chapter 11. Creating and refreshing test environments
- Chapter 12. Creating point-in-time copies of files

Chapter 7

Understanding point-in-time copy methods

This chapter includes the following topics:

- About point-in-time copies
- When to use point-in-time copies
- About Storage Foundation point-in-time copy technologies

About point-in-time copies

Storage Foundation offers a flexible and efficient means of managing business-critical data. Storage Foundation lets you capture an online image of an actively changing database at a given instant, called a point-in-time copy.

More and more, the expectation is that the data must be continuously available (24x7) for transaction processing, decision making, intellectual property creation, and so forth. Protecting the data from loss or destruction is also increasingly important. Formerly, data was taken out of service so that the data did not change while data backups occured; however, this option does not meet the need for minimal down time.

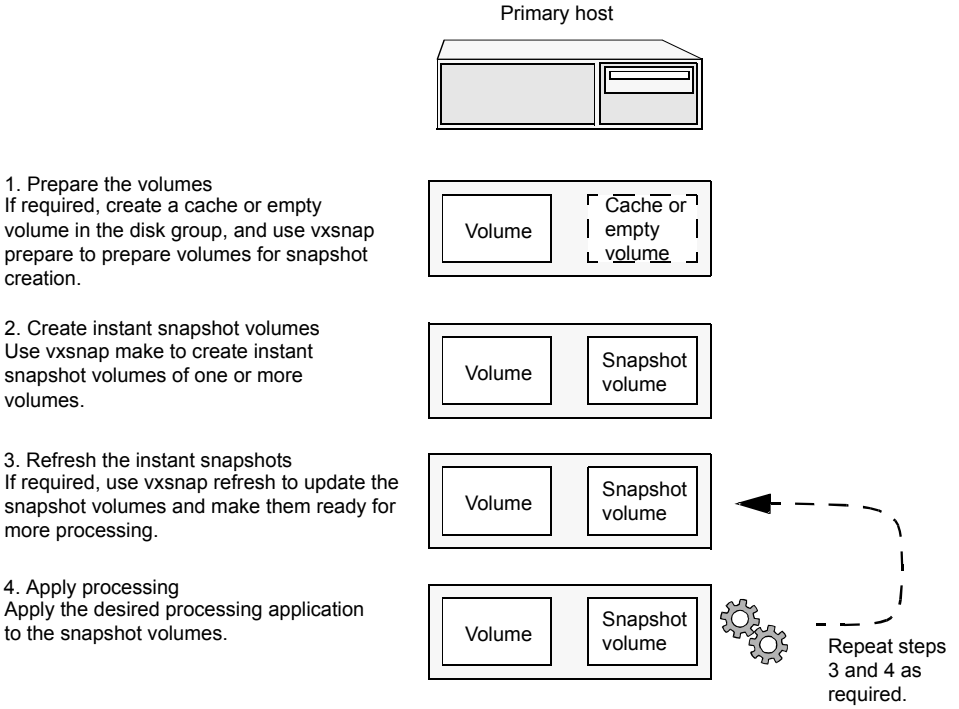
A point-in-time copy enables you to maximize the online availability of the data. You can perform system backup, upgrade, or perform other maintenance tasks on the point-in-time copies. The point-in-time copies can be processed on the same host as the active data, or a different host. If required, you can offload processing of the point-in-time copies onto another host to avoid contention for system resources on your production server. This method is called off-host processing. If implemented

correctly, off-host processing solutions have almost no impact on the performance of the primary production system.

Implementing point-in time copy solutions on a primary host

Figure 7-1 illustrates the steps that are needed to set up the processing solution on the primary host.

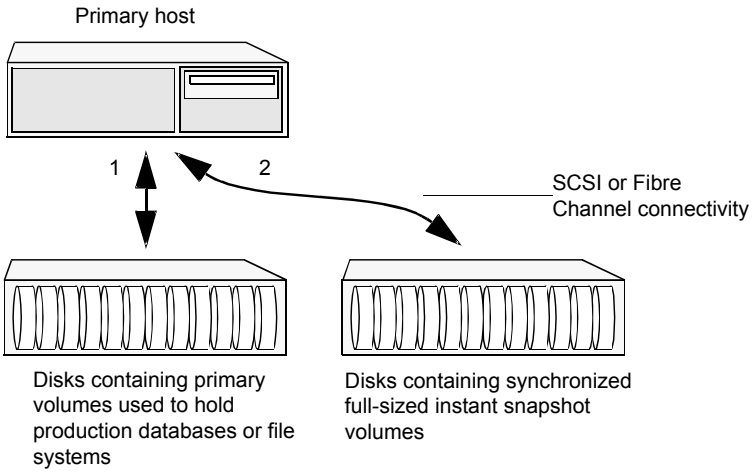
Figure 7-1 Using snapshots and FastResync to implement point-in-time copy solutions on a primary host



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 7-2 shows the suggested arrangement for implementing solutions where the primary host is used and disk contention is to be avoided.

Figure 7-2 Example point-in-time copy solution on a primary host



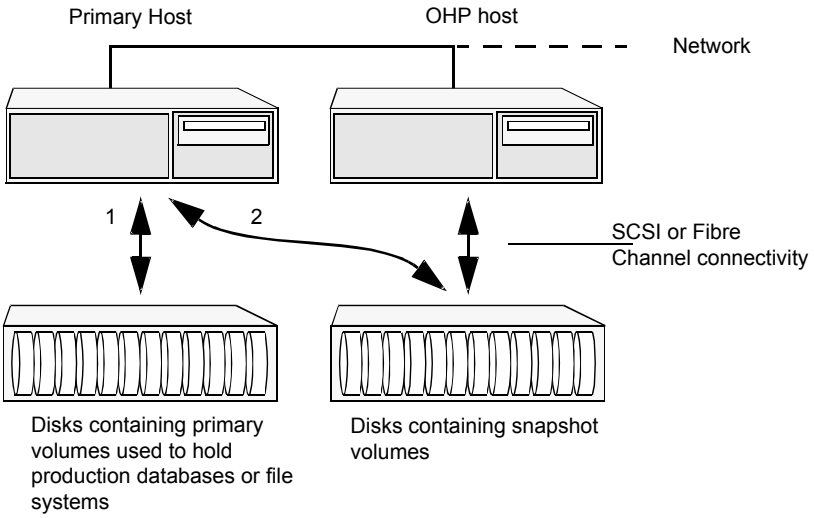
In this setup, it is recommended that separate paths (shown as 1 and 2) from separate controllers be configured to the disks containing the primary volumes and the snapshot volumes. This avoids contention for disk access, but the primary host's CPU, memory and I/O resources are more heavily utilized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

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

Implementing off-host point-in-time copy solutions

Figure 7-3 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).

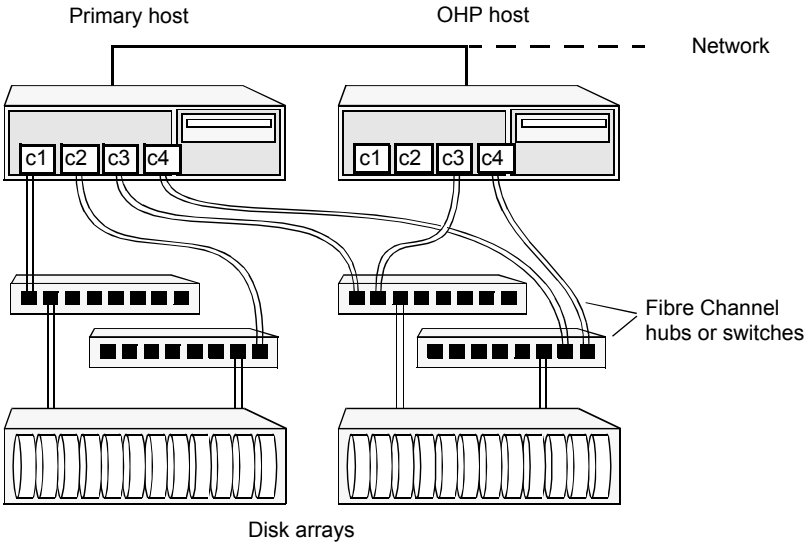
Figure 7-3 Example implementation of an off-host point-in-time copy solution



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 7-3 should be connected to different controllers.

Figure 7-4 shows an example of how you might achieve such connectivity using Fibre Channel technology with 4 Fibre Channel controllers in the primary host.

Figure 7-4 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 7-5 shows how off-host processing might be implemented in a cluster by configuring one of the cluster nodes as the OHP node.

Figure 7-5 Example implementation of an off-host point-in-time copy solution using a cluster node

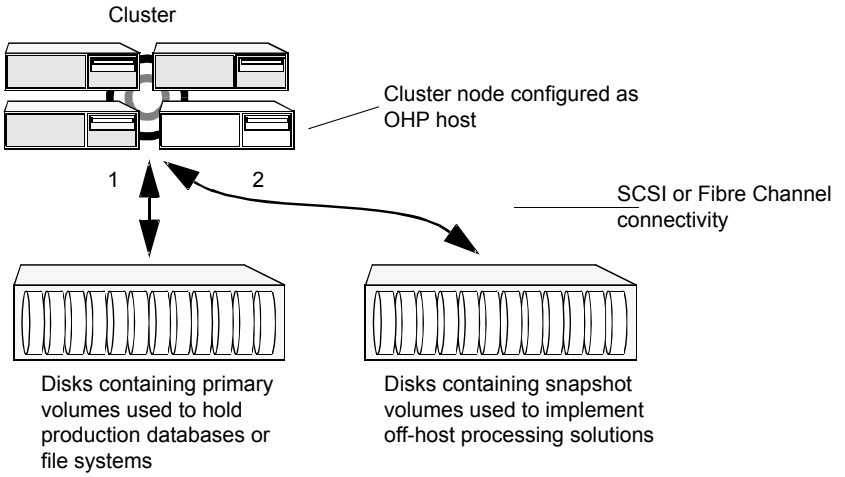
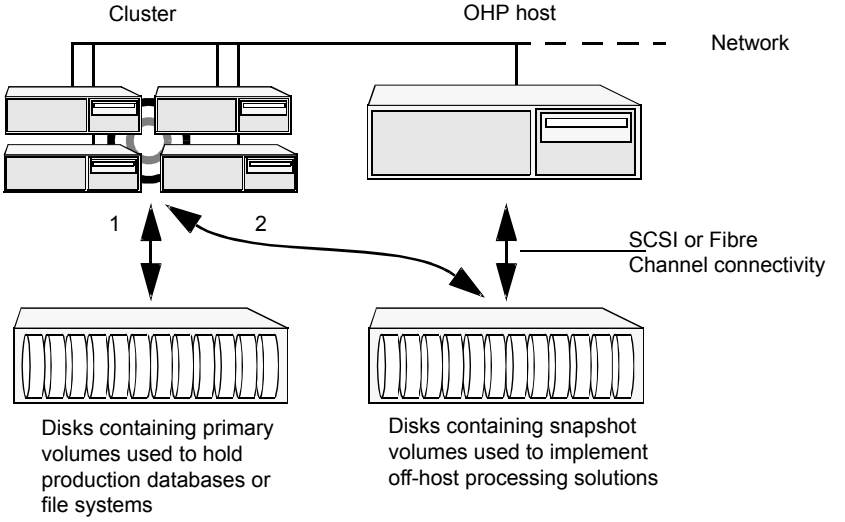


Figure 7-6 shows an alternative arrangement, where the OHP node could be a separate system that has a network connection to the cluster, but which is not a cluster node and is not connected to the cluster's private network.

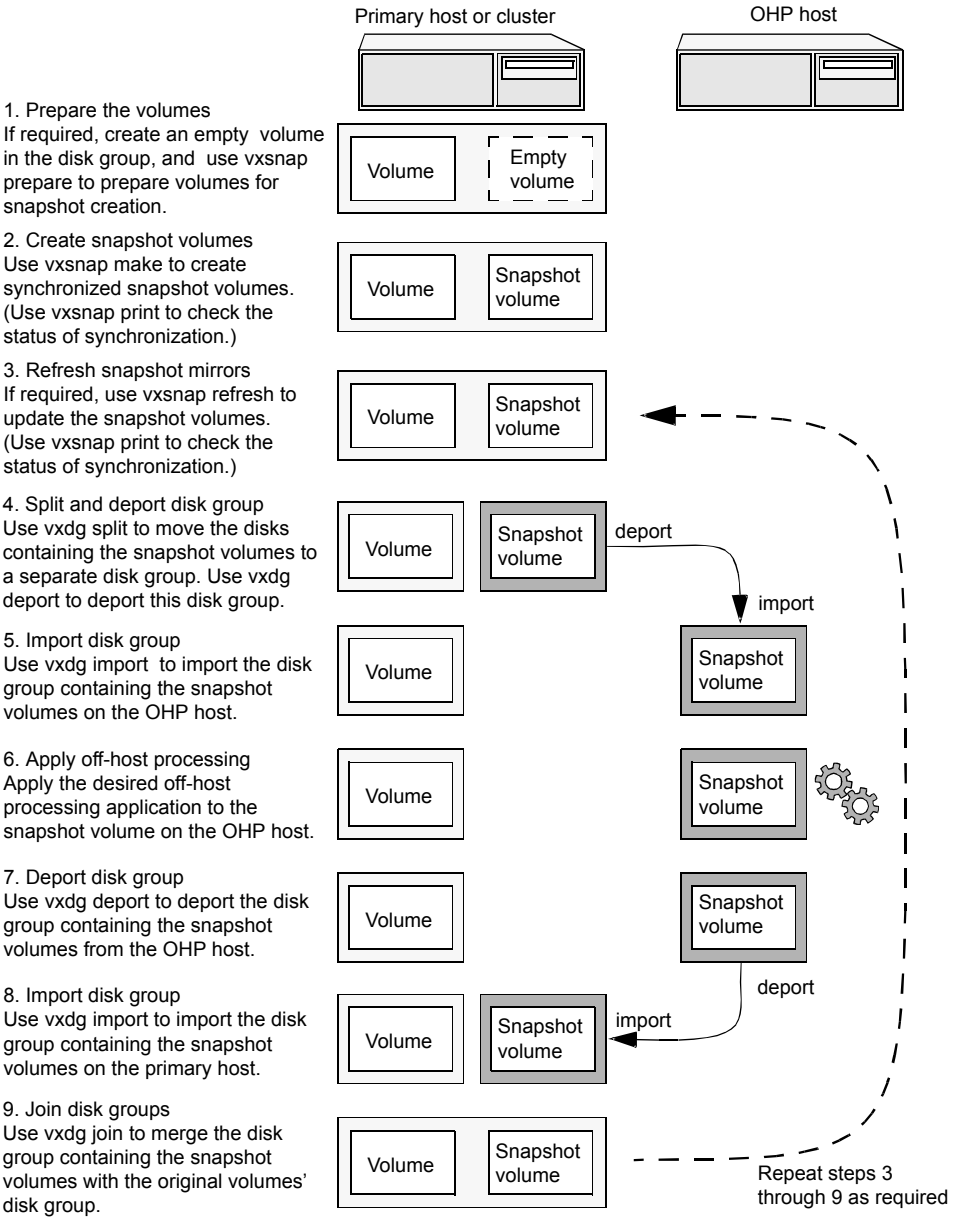
Figure 7-6 Example implementation of an off-host point-in-time copy solution using a separate OHP host



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

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

Figure 7-7 Implementing off-host processing solutions



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.

When to use point-in-time copies

The following typical activities are suitable for point-in-time copy solutions implemented using Veritas InfoScale 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.
- Providing data continuity —To provide continuity of service in the event of primary storage failure, you can use point-in-time copy solutions to recover application data. In the event of server failure, you can use point-in-time copy solutions in conjunction with the high availability cluster functionality of SFCFSHA or SFHA.
- 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.
  Use Storage Checkpoints to quickly roll back a database instance to an earlier point in time.
- Cloning data—You can clone your file system or application data. This functionality enable you to quickly and efficiently provision virtual desktops.

All of the snapshot solutions mentioned above are also available on the disaster recovery site, in conjunction with Volume Replicator.

For more information about snapshots with replication, see the *Veritas InfoScale 7.4.1 Replication Administrator's Guide*.

Storage Foundation provides several point-in-time copy solutions that support your needs, including the following use cases:

- Creating a replica database for decision support.
  See "Using a replica database for decision support" on page 119.
- Backing up and recovering a database with snapshots.
  See "Online database backups" on page 83.
- Backing up and recovering an off-host cluster file system
  See "Backing up on an off-host cluster file system" on page 101.
- Backing up and recovering an online database.
  See "Database recovery using Storage Checkpoints" on page 110.

About Storage Foundation point-in-time copy technologies

This topic introduces the point-in-time copy solutions that you can implement using the Veritas FlashSnap™ technology. Veritas FlashSnap technology requires a Veritas InfoScale Enterprise or Storage licenses.

Veritas InfoScale 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. There are several types of volume-level snapshots. These features are suitable for solutions where separate storage is desirable to create the snapshot. For example, lower-tier storage. Some of these techniques provided exceptional offhost processing capabilities.
- 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.

See "Storage Checkpoints" on page 75.

- File level snapshots.
  The FileSnap feature provides snapshots at the level of individual files.

Volume-level snapshots

A volume snapshot is an image of a Veritas Volume Manager (VxVM) volume at a given point in time. You can also take a snapshot of a volume set.

Volume snapshots allow you to make backup copies of your volumes online with minimal interruption to users. You can then use the backup copies to restore data that has been lost due to disk failure, software errors or human mistakes, or to create replica volumes for the purposes of report generation, application development, or testing.

Volume snapshots can also be used to implement off-host online backup.

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.

Persistent FastResync of volume snapshots

If persistent FastResync is enabled on a volume, VxVM uses a FastResync map to keep track of which blocks are updated in the volume and in the snapshot.

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

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

Data integrity in volume snapshots

A volume snapshot captures 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 in memory by the overlying file system, or by applications such as databases that have files open in the file system. Snapshots are always crash consistent, that is, the snapshot can be put to use by letting the application perform its recovery. This is similar to how the application recovery occurs after a server crash. If the `fsgen` volume usage type is set on a volume that contains a mounted Veritas File System (VxFS), VxVM coordinates with VxFS to flush data that is in the cache to the volume. Therefore, these snapshots are always VxFS consistent and require no VxFS recovery while mounting.

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. However, in all scenarios where application coordinate, snapshots are crash-recoverable.

Storage Checkpoints

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

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

Storage Checkpoints and the Storage Rollback feature of Storage Foundation for Databases enable rapid recovery of databases from logical errors such as database corruption, missing files and dropped table spaces. You can mount successive Storage Checkpoints of a database to locate the error, and then roll back the database to a Storage Checkpoint before the problem occurred.

See "Database recovery using Storage Checkpoints" on page 110.

Chapter 8

Backing up and recovering

This chapter includes the following topics:

- Storage Foundation and High Availability Solutions backup and recovery methods
- Preserving multiple point-in-time copies
- Online database backups
- Backing up on an off-host cluster file system
- Database recovery using Storage Checkpoints

Storage Foundation and High Availability Solutions backup and recovery methods

Storage Foundation and High Availability Solutions (SFHA Solutions) provide point-in-time copy methods which can be applied to multiple database backup use cases.

Examples are provided for the following use cases:

- Creating and maintaining a full image snapshot and incremental point-in-time copies
- Off-host database backup
- Online database backup
- Database recovery with Storage Checkpoints
- Backing up and restoring with NetBackup

For basic backup and recovery configuration information, see the *Storage Foundation Administrator's Guide*.

Preserving multiple point-in-time copies

On-disk snapshots are efficient when it comes to recovering a logically corrupted data. Storage Foundation and High Availability Solutions (SFHA Solutions) provide a cost effective and very efficient mechanism to manage multiple copies of production data at different points in time. With FlashSnap, you can create a solution to manage the whole lifecycle of snapshots for recovery from logical data corruption. You can create a series of point-in-time copies and preserve them for a specified time or a certain number of copies. You can use the preserved snapshot image itself for business continuity in case of primary storage failure or for off-host processing.

The following example procedures illustrate how to create a full image snapshot and periodic point-in-time copies for recovery. With multiple point-in-time copies to choose from, you can select the point-in-time to which you want to recover with relative precision.

Setting up multiple point-in-time copies

To set up the initial configuration for multiple point-in-time copies, set up storage for the point-in-time copies that will be configured over time.

In the example procedures, *disk1*, *disk2*, …, *diskN* are the LUNs configured on tier 1 storage for application data. A subset of these LUNs *logdisk1*, *logdisk2*, …, *logdiskN*, will be used to configure DCO. Disks *sdisk1*, *sdisk2*, …, *sdiskN* are disks from tier 2 storage.

Note: If you have an enclosure or disk array with storage that is backed by write cache, Veritas recommends that you use the same set of LUNs for the DCO and for the data volume.

If no logdisks are specified by default, Veritas Volume Manager (VxVM) tries to allocate the DCO from the same LUNs used for the data volumes.

See Figure 7-4 on page 68.

You will need to make sure your cache is big enough for the multiple copies with multiple changes. The following guidelines may be useful for estimating your requirements.

To determine your storage requirements, use the following:

Table 8-1 Storage requirements

S_p	Represents the storage requirement for the primary volume

Table 8-1 Storage requirements *(continued)*

S_b	Represents the storage requirement for the primary break-off snapshot.
N_c	Represents the number of point-in-time copies to be maintained.
S_c	Represents the average size of the changes that occur in an interval before the snapshot is taken.
S_t	Represents the total storage requirement.

The total storage requirement for management of multiple point-in-time copies can be roughly calculated as:

$S_b = S_p$

$S_t = S_b + N_c * S_c$

To determine the size of the cache volume, use the following:

Table 8-2 Cache volume requirements

N_c	Represents the number of point-in-time copies to be maintained.
S_c	Represents the average size of the changes that occur in an interval .
R_c	Represents the region-size for cache-object.
S_t	Represents the total storage requirement.

The size of cache-volume to be configured can be calculated as:

Nc * Sc * Rc

This equation assumes that the application IO size granularity is smaller than cache-object region-size by factor of at most R_c

To configure the initial setup for multiple point-in-time copies

1. If the primary application storage is already configured for snapshots, that is, the DCO is already attached for the primary volume, go to step 2.

   If not, configure the primary volumes and prepare them for snapshots.

   For example:

   ```
   # vxassist -g appdg make appvol 10T <disk1 disk2 ... diskN >
   # vxsnap -g appdg prepare appvol
   ```

2. Configure a snapshot volume to use as the primary, full-image snapshot of the primary volume. The snapshot volume can be allocated from tier 2 storage.

   ```
   # vxassist -g appdg make snap-appvol 10T <sdisk1 sdisk2 ... sdiskN >
   # vxsnap -g appdg prepare snap-appvol  \
   <alloc=slogdisk1, slogdisk2, ...slogdiskN>
   ```

3. Establish the relationship between the primary volume and the snapshot volume. Wait for synchronization of the snapshot to complete.

   ```
   # vxsnap -g appdg make source=appvol/snapvol=snap-appvol/sync=yes
   # vxsnap -g appdg syncwait snap-appvol
   ```

4. Create a volume in the disk group to use for the cache volume. The cache volume is used for space-optimized point-in-time copies created at regular intervals. The cache volume can be allocated from tier 2 storage.

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

5. Configure a shared cache object on the cache volume.

   ```
   # vxmake -g appdg cache snapcache cachevolname=cachevol
   ```

6. Start the cache object.

   ```
   # vxcache -g appdg start snapcache
   ```

   You now have an initial setup in place to create regular point-in-time copies.

Refreshing point-in-time copies

After configuring your volumes for snapshots, you can periodically invoke a script with steps similar to following to create point-in-time copies at regular intervals.

To identify snapshot age

- To find the oldest and the most recent snapshots, use the creation time of the snapshots. You can use either of the following commands:
  - Use the following command and find the SNAPDATE of snapshot volume.

    ```
    # vxsnap -g appdg list appvol
    ```

  - Use the following command:

    ```
    # vxprint -g appdg -m snapobject_name| grep creation_time
    ```

    where the *snapobject-name* is *appvol-snp*, *appvol-snp1* .... *appvol-snpN*.

To refresh the primary snapshot

- Refresh the primary snapshot from the primary volume.

  ```
  # vxsnap -g appdg refresh snap-appvol source=appvol
  ```

To create cascaded snapshot of the refreshed snapshot volume

- Create a cascaded snapshot of the refreshed snapshot volume.

  ```
  # vxsnap -g appdg make source=snap-appvol/new=sosnap-\
  appvol${NEW_SNAP_IDX}/cache=snapcache/infrontof=snap-appvol
  ```

To remove the oldest point-in-time copy

- If the limit on number of point-in-time copies is reached, remove the oldest point-in-time copy.

  ```
  # vxedit -g appdg -rf rm sosnap-appvol${ OLDEST_SNAP_IDX }
  ```

Recovering from logical corruption

You can use the preserved snapshot image in case of primary storage corruption. You must identify the most recent snapshot that is not affected by the logical corruption.

To identify the most recent valid snapshot

1 For each snapshot, starting from the most recent to the oldest, verify the snapshot image. Create a space-optimized snapshot of the point-in-time copy to generate a synthetic replica of the point-in-time image.

```
# vxsnap -g appdg make source=sosnapappvol${
CURIDX}/new=syn-appvol/cache=snapcache/sync=no
```

2 Mount the synthetic replica and verify the data.

If a synthetic replica is corrupted, proceed to 3.

When you identify a synthetic replica that is not corrupted, you can proceed to the recovery steps.

See "To recover from logical corruption" on page 82.

3 Unmount the synthetic replica, remove it and go back to verify the next most recent point-in-time copy. Use the following command to dissociate the synthetic replica and remove it:

```
# vxsnap -g appdg dis syn-appvol
# vxedit -g appdg -rf rm syn-appvol
```

When you find the most recent uncorrupted snapshot, use it to restore the primary volume.

To recover from logical corruption

1 If the application is running on the primary volume, stop the application.

2 Unmount the application volume.

3 Restore the primary volume from the synthetic replica.

```
# vxsnap -g appdg  restore appvol source=syn-appvol
```

4 Resume the application:

- Mount the primary volume.
- Verify the content of the primary volume.
- Restart the application.

Off-host processing using refreshed snapshot images

Preserved point-in-time images can also be used to perform off-host processing. Using preserved point-in-time images for this purpose requires that the storage used for creating the snapshots must be:

- Accessible from the application host
- Accessible from the off-host processing host
- Split into a separate disk group

To split the snapshot storage into a separate disk group

- Split the snapshot storage into a separate disk group.

```
# vxdg split appdg snapdg snap-appvol
```

The *snapdg* disk group can optionally be deported from the application host using the `vxdg deport` command and imported on another host using the `vxdg import` command to continue to perform off-host processing.

To refresh snapshot images for off-host processing

1. Deport the *snapdg* disk group from the off-host processing host.

```
# vxdg deport snapdg
```

2. Import the *snapdg* disk group on the application host.

```
# vxdg import snapdg
```

3. On the application host, join the *snapdg* disk group to *appdg*.

```
# vxdg join snapdg appdg
```

After this step, you can proceed with the steps for managing point-in-time copies.

See “Refreshing point-in-time copies” on page 80.

Online database backups

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:

- See “Making a backup of an online database on the same host” on page 84.

- See “Making an off-host backup of an online database” on page 93.

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

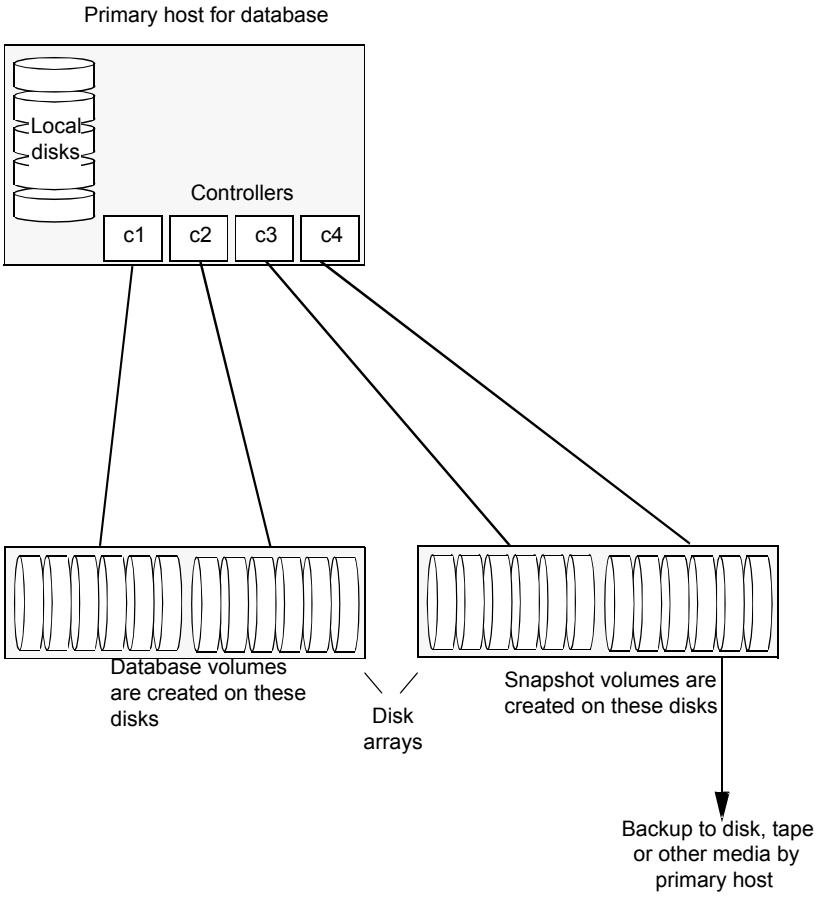
For more information about using snapshots to back up Oracle databases, see the *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

Note: The sample database scripts in the following procedures are not supported by Veritas, and are provided for informational use only. You can purchase customization of the environment through Veritas Vpro Consulting Services.

Making a backup of an online database on the same host

Figure 8-1 shows an example with two primary database volumes to be backed up, *database_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 8-1 Example system configuration for database backup on the primary host



Note: It is assumed that you have already prepared the volumes containing the file systems for the datafiles to be backed up as described in the example.

To make an online database backup

- Prepare the snapshot, either full-sized or space-optimized.
  Depending on the application, space-optimized snapshots typically require 10% of the disk space that is required for full-sized instant snapshots.
- Create snapshot mirrors for volumes containing VxFS file systems for database files to be backed up.

- Make the online database backup.

Some scenarios where full-instant snapshot works better are:

- Off host processing is planned for a databases backup.
- If a space-optimized snapshot is taken for longer duration and modified frequently then it is not much different than the full-snapshot. So, for performance reason full-snapshot will be preferred,

Preparing a full-sized instant snapshot for a backup

You can use a full-sized instant snapshot for your online or off-host database backup.

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.

To make a full-sized instant snapshot for 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 database_dg addmir database_vol [nmirror=N] \
  [alloc=storage_attributes]
# vxsnap -g database_dg make \
  source=database_vol/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.

   You can specify at least *N* number of disks if the specified number of mirrors is *N*.

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

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

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

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`. These steps are valid only for instant snap DCOs.

---

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

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

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

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

5 Use the `vxassist` command to create a volume, *snapvol*, of the required size and redundancy. You can use storage attributes to specify which disks should be used for the volume. The `init=active` attribute makes the volume available immediately.

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

6 Prepare the snapshot volume for instant snapshot operations as shown here:

```
# vxsnap [-g diskgroup] prepare snapvol [ndcomirs=number] \
  regionsz=$RSZ [storage_attributes]
```

It is recommended that you specify the same number of DCO mirrors (*ndcomirror*) as the number of mirrors in the volume (*nmirror*).

7 Use the following command to create the snapshot:

```
# vxsnap -g database_dg make source=database_vol/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 database_dg make source=database_vol1/snapvol=snapvol1 \
source=database_vol2/newvol=snapvol2 \
source=database_vol3/snapvol=snapvol3
```

When you are ready to make a backup, proceed to make a backup of an online database on the same host.

Preparing a space-optimized snapshot for a database backup

If a snapshot volume is to be used on the same host, and will not be moved to another host, 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.

To prepare a space-optimized snapshot for a backup of an online database

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, *database_dg*, on the disks *disk16* and *disk17*:

   ```
   # vxassist -g database_dg 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=cachevol [regionsize=size] [autogrow=on] \
  [highwatermark=hwmk] [autogrowby=agbvalue] \
  [maxautogrow=maxagbvalue]]
```

If you specify the region size, 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 not allowed to grow in size as required, specify `autogrow=off`. By default, the ability to automatically grow the cache is turned on.

In the following example, the cache object, *cache_object*, 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 database_dg cache cache_object 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, start the cache object *cache_object*:

```
# vxcache -g database_dg start cache_object
```

5 Create a space-optimized snapshot with your cache object.

```
# vxsnap -g database_dg make \
source=database_vol1/newvol=snapvol1/cache=cache_object
```

6 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 database_dg make \
  source=database_vol1/newvol=snapvol1/cache=cache_object \
```

```
source=database_vol2/newvol=snapvol2/cache=cache_object \
source=database_vol3/newvol=snapvol3/cache=cache_object
```

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

When you are ready to make a backup, proceed to make a backup of an online database on the same host

Backing up a Sybase database on the same host

You can make an online backup of your Sybase database.

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

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

```
#!/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
!
```

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

```
# vxsnap -g database_dg refresh snapvol1 source=database_vol1 \
  [snapvol2 source=database_vol2]...
```

For example, to refresh the snapshots *snapvol1*, *snapvol2* and *snapvol3*:

```
# vxsnap -g database_dg refresh snapvol1 source=database_vol1 \
  snapvol2 source=database_vol2 snapvol3 source=database_vol3
```

3 If you have temporarily suspended updates to volumes, release all the tablespaces or databases from quiesce mode.

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

```
#!/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
!
```

4 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 -t vxfs /dev/vx/rdsk/database_dg/snapvol
# mount -t vxfs /dev/vx/dsk/database_dg/snapvol
     mount_point
```

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

```
# umount mount_point
```

5 Repeat steps in this procedure each time that you need to back up the volume.

Resynchronizing a 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).

To resynchronize a volume from its snapshot volume

- Enter:

```
# vxsnap -g diskgroup restore database_vol 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 *database_vol* from its snapshot volume *snapvol* without removing the snapshot volume:

```
# vxsnap -g database_dg restore database_vol \
   source=snapvol destroy=no
```

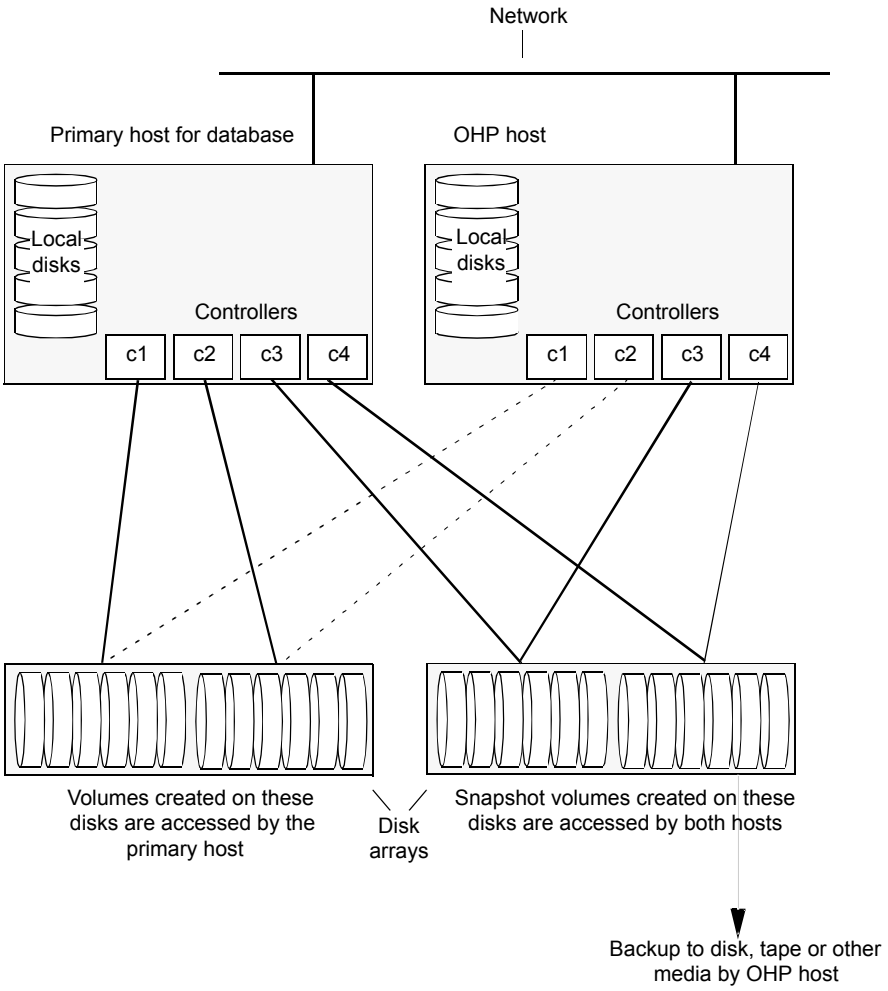
Note: You must shut down the database and unmount the file system that is configured on the original volume before attempting to resynchronize its contents from a snapshot.

Making an off-host backup of an online database

Figure 8-2 shows an example of two primary database volumes to be backed up, *database_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 off-host processing host to have access to the disks that contain the primary database volumes.

Figure 8-2 Example system configuration for off-host database backup



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. However, if the primary host is not also the master node, most Veritas Volume Manager (VxVM) operations on shared disk groups are best performed on the master node.

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

- Prepare the full-sized snapshot for backing up.
  See "Preparing a full-sized instant snapshot for a backup" on page 86.

- Make the off-host database backup of the database.
  See “Making an off-host backup of an online Sybase database” on page 95.

Making an off-host backup of an online Sybase database

The procedure for off-host database backup is designed to minimize copy-on-write operations that can impact system performance. You can use this procedure whether the database volumes are in a cluster-shareable disk group or a private disk group on a single host. If the disk group is cluster-shareable, you can use a node in the cluster for the off-host processing (OHP) host. In that case, you can omit the steps to split the disk group and deport it to the OHP host. The disk group is already accessible to the OHP host. Similarly, when you refresh the snapshot you do not need to reimport the snapshot and rejoin the snapshot disk group to the primary host.

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

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

```
# vxsnap -g database_dg addmir database_vol [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. As the Sybase database administrator, put the database in quiesce mode by using a script such as that shown in the example.

```
#!/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
!
```

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 database_dg make \
   source=database_vol/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 database_dg make source=database_vol1/snapvol=snapvol1 \
   source=database_vol/snapvol=snapvol2 \
   source=database_vol3/snapvol=snapvol3 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 quiesce mode. As the Sybase database administrator, release the database from quiesce mode using a script such as that shown in the example.

```
#!/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
!
```

5 If the primary host and the snapshot host are in the same cluster, and the disk group is shared, the snapshot volume is already accessable to the OHP host. Skip to step 9.

If the OHP host is not in the cluster, perform the following steps to make the snapshot volume accessible to the OHP host.

On the primary host, split the disks containing the snapshot volumes into a separate disk group, *snapvoldg*, from the original disk group, *database_dg* using the following command:

```
# vxdg split database_dg 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 VxVM will recover the volumes automatically after the disk group import unless it is set to not recover automatically. Check if the snapshot volume is initially disabled and not recovered following the split.

If a volume is in the DISABLED state, use the following command on the OHP host to recover and restart the snapshot volume:

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

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

```
# umount mount_point
```

10 If the primary host and the snapshot host are in the same cluster, and the disk group is shared, the snapshot volume is already accessible to the primary host. Skip to step 14.

If the OHP host is not in the cluster, perform the following steps to make the snapshot volume accessible to the primary host.

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 database_dg
```

13 VxVM will recover the volumes automatically after the join unless it is set not to recover automatically. Check if the snapshot volumes are initially disabled and not recovered following the join.

If a volume is in the DISABLED state, use the following command on the primary host to recover and restart the snapshot volume:

```
# vxrecover -g database_dg -m snapvol
```

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

```
# vxsnap -g database_dg reattach snapvol source=database_vol \
   [snapvol2 source=database_vol2]...
```

For example, to reattach the snapshot volumes *snapvol1*, *snapvol2* and *snapvol3*:

```
# vxsnap -g database_dg reattach snapvol1 source=database_vol1 \
   snapvol2 source=database_vol2 snapvol3 source=database_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.

Resynchronizing a 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).

To resynchronize a volume

- Use the following command syntax:

```
vxsnap -g database_dg restore database_vol 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 *database_vol* from its snapshot volume *snapvol* without removing the snapshot volume:

```
# vxsnap -g database_dg restore database_vol \
   source=snapvol 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.

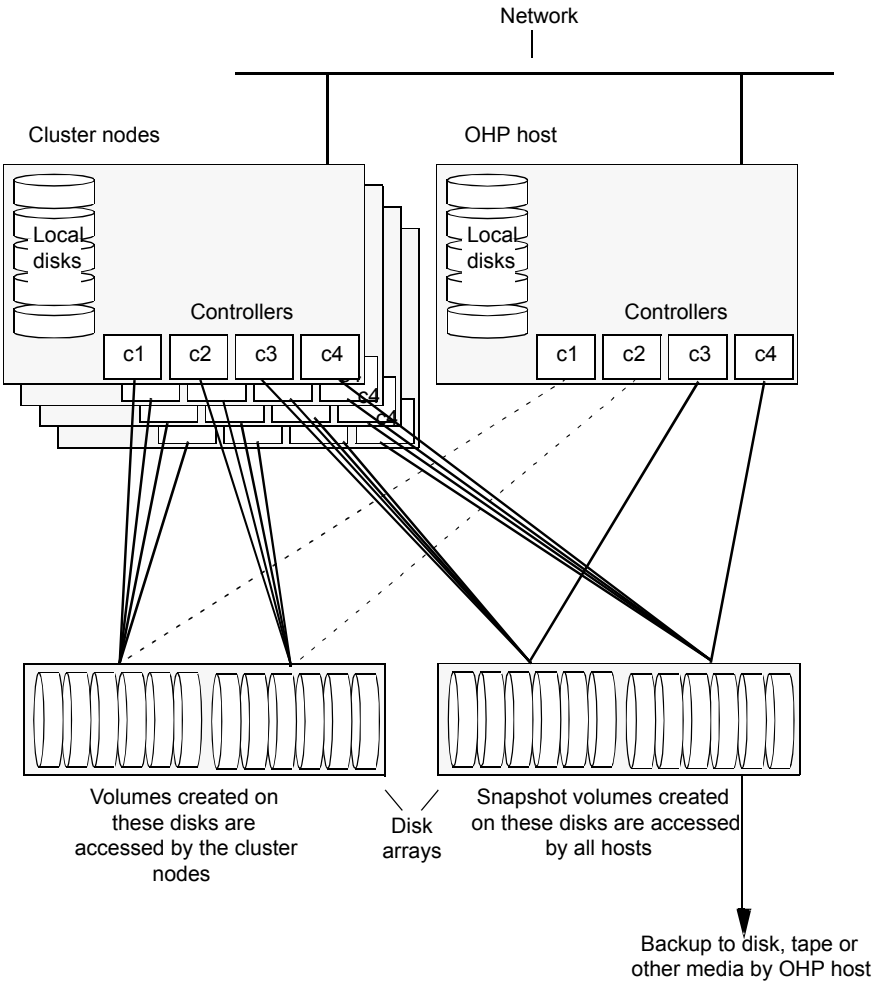
Backing up on an off-host cluster file system

Storage Foundation Cluster File System High Availability (SFCFSHA) allows cluster nodes to share access to the same file system. SFCFSHA 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 8-3 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*.

Figure 8-3 System configuration for off-host file system backup scenarios



To set up an off-host cluster file system backup:

- Mount a VxFS file system for shared access by the nodes of a cluster.
  See "Mounting a file system for shared access" on page 103.
- Prepare a snapshot of the mounted file system with shared access.
  See "Preparing a snapshot of a mounted file system with shared access" on page 103.
- Back up a snapshot of a mounted file system with shared access

See "Backing up a snapshot of a mounted file system with shared access" on page 105.

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

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 -t vxfs -o cluster /dev/vx/dsk/database_dg/database_vol
   mount_point
```

For example, to mount the volume *database_vol* in the disk group *database_dg* for shared access on the mount point, `/mnt_pnt`:

```
# mount -t vxfs -o cluster /dev/vx/dsk/database_dg/database_vol /mnt_pnt
```

Preparing a snapshot of a mounted file system with shared access

You must use a full-sized snapshot for your off-host backup.

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.

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

1. On the master node, verify that the volume has an instant snap data change object (DCO) and DCO volume, and that FastResync is enabled on the volume:

   ```
   # vxprint -g database_dg -F%instant database_vol

   # vxprint -g database_dg -F%fastresync database_vol
   ```

   If both commands return the value of ON, proceed to step 3. Otherwise, continue with step 2.

2. Use the following command to prepare a volume for instant snapshots:

   ```
   # vxsnap -g database_dg prepare database_vol [regionsize=size] \
   [ndcomirs=number] [alloc=storage_attributes]
   ```

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

```
# LEN=`vxprint [-g database_dg] -F%len database_vol`
```

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`. These steps are valid only for instant snap DCOs.

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

```
#  DCONAME=`vxprint [-g database_dg] -F%dco_name database_vol`
```

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

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

6 Use the `vxassist` command to create a volume, *snapvol*, of the required size and redundancy, together with an instant snap DCO volume with the correct region size:

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

7 Then use the following command to create the snapshot:

```
# vxsnap -g database_dg make source=database_dg/snapvol=snapvol
```

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

Backing up a snapshot of a mounted file system with shared access

While you can run the commands in the following steps from any node, Veritas recommends running them from the master node.

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

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

   ```
   # vxsnap -g database_dg refresh snapvol source=database_vol \
     [snapvol2 source=database_vol2]... syncing=yes
   ```

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

   For example, to refresh the snapshot *snapvol*:

   ```
   # vxsnap -g database_dg refresh snapvol source=database_vol \
     syncing=yes
   ```

   This command can be run every time you want to back up the data. The `vxsnap refresh` command will resync only those regions which have been modified since the last refresh.

2. On any node of the cluster, 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 database_dg syncwait snapvol
   ```

   For example, to wait for synchronization to finish for the snapshots *snapvol*:

   ```
   # vxsnap -g database_dg syncwait snapvol
   ```

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.

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

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

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

```
# vxdg split database_dg splitdg snapvol
```

4 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
```

5 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
```

6 VxVM will recover the volumes automatically after the disk group import unless it is set not to recover automatically. Check if the snapshot volume is initially disabled and not recovered following the split.

If a volume is in the DISABLED state, 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
```

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

```
# fsck -t vxfs /dev/vx/rdsk/database_dg/database_vol

# mount -t vxfs /dev/vx/dsk/database_dg/database_vol
   mount_point
```

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

```
# fsck -t vxfs /dev/vx/rdsk/splitdg/snapvol
# mount -t vxfs /dev/vx/dsk/splitdg/snapvol /bak/mnt_pnt
```

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

```
# umount mount_point
```

9 On the off-host processing host, use the following command to deport the snapshot volume's disk group:

```
# vxdg deport snapvoldg
```

For example, to deport *splitdg*:

```
# vxdg deport splitdg
```

10 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
```

11 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 database_dg
```

For example, to join disk group *splitdg* with *database_dg*:

```
# vxdg join splitdg database_dg
```

12 VxVM will recover the volumes automatically after the join unless it is set not to recover automatically. Check if the snapshot volumes are initially disabled and not recovered following the join.

If a volume is in the DISABLED state, use the following command on the primary host to recover and restart the snapshot volume:

```
# vxrecover -g database_dg -m snapvol
```

13 When the recover is complete, use the following command to refresh the snapshot volume, and make its contents refreshed from the primary volume:

```
# vxsnap -g database_dg refresh snapvol source=database_vol \
syncing=yes

# vxsnap -g database_dg syncwait snapvol
```

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

Repeat the entire procedure each time that you need to back up the volume.

Resynchronizing a volume from its snapshot 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).

To resynchronize a volume from its snapshot volume

- Enter:

```
vxsnap -g database_dg restore database_vol 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 *database_vol* from its snapshot volume *snapvol* without removing the snapshot volume:

```
# vxsnap -g database_dg restore database_vol source=snapvol 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 database_dg] reattach snapvol source=database_vol \
    [nmirror=number]
  ```

  For example the following command reattaches 1 plex from the snapshot volume, *snapvol*, to the volume, *database_vol*:

  ```
  # vxsnap -g database_dg reattach snapvol source=database_vol 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.

Database recovery using 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.

For information on how to administer Storage Checkpoints, see *Storage Foundation Administrator's Guide*.

For information on how to administer Database Storage Checkpoints for an Oracle database, see *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

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.

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 9

Backing up and recovering in a NetBackup environment

This chapter includes the following topics:

- About Veritas NetBackup
- About using NetBackup for backup and restore for Sybase
- Using NetBackup in an SFHA Solutions product environment

About Veritas NetBackup

Veritas NetBackup provides backup, archive, and restore capabilities for database files and directories contained on client systems in a client-server network. NetBackup server software resides on platforms that manage physical backup storage devices. The NetBackup server provides robotic control, media management, error handling, scheduling, and a repository of all client backup images.

Administrators can set up schedules for automatic, unattended full and incremental backups. These backups are managed entirely by the NetBackup server. The administrator can also manually back up clients. Client users can perform backups, archives, and restores from their client system, and once started, these operations also run under the control of the NetBackup server.

Veritas NetBackup can be configured for DB2 in an Extended Edition (EE) or Extended-Enterprise Edition (EEE) environment. For detailed information and instructions on configuring DB2 for EEE, see “Configuring for a DB2 EEE (DPF) Environment” in the *Veritas NetBackup for DB2 System Administrator's Guide for UNIX*.

Veritas NetBackup, while not a shipped component of Storage Foundation Enterprise products, can be purchased separately.

About using NetBackup for backup and restore for Sybase

Veritas NetBackup for Sybase is not included in the standard Veritas Database Edition. The information included here is for reference only.

Veritas NetBackup for Sybase integrates the database backup and recovery capabilities of Sybase Backup Server with the backup and recovery management capabilities of NetBackup.

Veritas NetBackup works with Sybase APIs to provide high-performance backup and restore for Sybase dataservers. With Veritas NetBackup, you can set up schedules for automatic, unattended backups for Sybase ASE dataservers (NetBackup clients) across the network. These backups can be full database dumps or incremental backups (transaction logs) and are managed by the NetBackup server. You can also manually backup dataservers. The Sybase `dump` and `load` commands are used to perform backups and restores.

Veritas NetBackup has both graphical and menu driven user interfaces to suit your needs.

For details, refer to *NetBackup System Administrator's Guide for UNIX*.

Using NetBackup in an SFHA Solutions product environment

You can enhance the ease of use and efficiency of your SFHA Solutions product and NetBackup by integrating them as follows:

- Clustering a NetBackup Master Server
- Backing up and recovering a VxVM volume using NetBackup

Clustering a NetBackup Master Server

To enable your NetBackup Master Server to be highly available in a cluster environment, use the following procedure.

To make a NetBackup Master Server, media, and processes highly available

1. Verify that your versions of NetBackup and Cluster Server are compatible. Detailed combination information is included in the NetBackup cluster compatibility list:
   - For NetBackup 7.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH126902
   - For NetBackup 6.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH43619
   - For NetBackup 5.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH29272
   - For more on NetBackup compatibility, see
     https://www.veritas.com/support/en_US/dpp.15145.html
2. The steps to cluster a Master Server are different for different versions of NetBackup. See the applicable NetBackup guide for directions.

   https://sort.veritas.com

To verify the robustness of the VCS resources and NetBackup processes

1. Verify that you can online the Netbackup master.
2. Verify that you can offline the Netbackup master.
3. Verify that you can monitor all the NetBackup resources.

Backing up and recovering a VxVM volume using NetBackup

To enable NetBackup to backup objects on a VxVM volume, use the following procedure. This procedure enables an Instant Recovery (IR) using a VxVM volume.

To back up objects in a VxVM volume using NetBackup

1. Create a VxVM disk group with six disks. The number of disks may vary depending on the volume size, disk size, volume layout, and snapshot method.

   If the system this test is running on is a clustered system, create a shared disk group using the `-s` option.

   ```
   # vxdg -s init database_dg disk1 disk2 disk3 \
   disk4 disk5 disk6
   ```

2. Create a "mirror-striped" VxVM volume with a size of 10 Gbytes or the maximum size of the disk, whichever is larger.

   ```
   # vxassist -g database_dg make vol_name 10G \
   layout=mirror-stripe init=active
   # vxvol -g database_dg set fastresync=on vol_name
   # vxassist -g database_dg snapstart nmirror=1 vol_name
   ```

   Note: There are three types of snapshot: mirror, full-size instant, and space-optimized instant shanpshots. The example uses an Instant Recovery (IR) snapshot. For snapshot creation details:

   See pages 104-107 of the *NetBackup Snapshot Client Administrator's Guide* for 7.6.

   See https://www.veritas.com/support/en_US/article.DOC6459

   ---

3. Make the file system on the volume.
4. Mount a VxFS file system on the volume.

   If the VxVM volume is a clustered volume, mount the VxFS file system with the "-o cluster" option.
5. Fill up the VxFS file system up to the desired level. For example, you can fill to 95% full, or to whatever level is appropriate for your file system.
6. Store the `cksum`(1) for these files.
7. Un-mount the VxFS file system.
8. Enable the following Advanced Client option:
   - Perform Snapshot Backup.
   - Set **Advanced Snapshot Options** to **vxvm**.

- Enable **Retain snapshots for instant recovery**.

9 Back up the VxVM volume with the NetBackup policy.

See *NetBackup Snapshot Client Administrator's Guide* for 7.6.

See https://www.veritas.com/support/en_US/article.DOC6459

Recovering a VxVM volume using NetBackup

To enable NetBackup to recover objects on a VxVM volume, use the following procedure. This procedure performs an Instant Recovery (IR) using a VxVM volume.

To recover objects in a VxVM volume using NetBackup

1 Initialize the VxVM volume to zeros.

2 Recover the VxVM volume to the newly initialized VxVM volume.

3 Mount the VxFS file system on the empty VxVM volume.

4 Verify the cksum(1) values against the files recovered.

Chapter 10

Off-host processing

This chapter includes the following topics:

- Veritas InfoScale Storage Foundation off-host processing methods
- Using a replica database for decision support
- What is off-host processing?
- About using VVR for off-host processing

Veritas InfoScale Storage Foundation off-host processing methods

While backup and recovery is an important use case for Veritas InfoScale point-in-time copy methods, they can also be used for:

- Periodic analysis (mining) of production data
- Predictive what-if analysis
- Software testing against real data
- Application or database problem diagnosis and resolution

Off-host processing use cases are similar to the backup use case in that they generally require consistent images of production data sets. They differ from backup in the three important respects:

- Access mode
  Whereas backup is a read-only activity, most off-host processing activities update the data they process. Thus, Snapshot File Systems are of limited utility for off-host processing uses.
- Multiple uses

Backup uses each source data image once, after which the snapshot can be discarded. With other use cases, it is often useful to perform several experiments on the same data set. It is possible to take snapshots of both Storage Checkpoints and Space-Optimized Instant Snapshots of production data. This facility provides multiple identical data images for exploratory applications at almost no incremental overhead. Rather than testing destructively against a snapshot containing the data set state of interest, tests can be run against snapshots of that snapshot. After each test, the snapshot used can be deleted, leaving the original snapshot containing the starting state intact. Any number of tests or analyses can start with the same data, providing comparable alternatives for evaluation. All such tests can be run while production applications simultaneously process live data.

- Scheduling
  Whereas backup is typically a regularly scheduled activity, allowing storage and I/O capacity needs to be planned, other applications of snapshots must run with little or no notice. Full-sized and space-optimized instant snapshots and Storage Checkpoints provide instantly accessible snapshots, and are therefore more suitable for these applications.

Veritas InfoScale examples for data analysis and off-host processing use cases:

- Decision support
- Active secondary use-case with VVR

Using a replica database for 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:

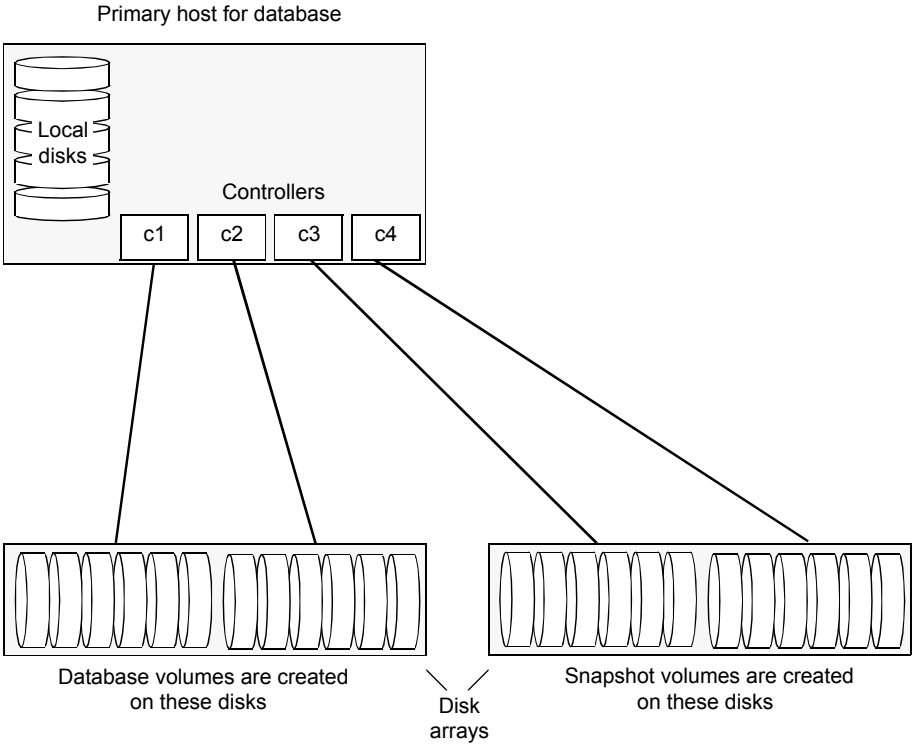
- See “Creating a replica database on the same host” on page 120.
- See “Creating an off-host replica database” on page 132.

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.

Creating a replica database on the same host

Figure 10-1 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 10-1 Example system configuration for decision support on the primary host



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

- Prepare the snapshot, either full-sized or space-optimized.
  See "Preparing a full-sized instant snapshot for a backup" on page 86.
- Create snapshot mirrors for volumes containing VxFS file systems for database files to be backed up.
- Make the database replica.
- All commands require superuser (`root`) or equivalent privileges.

Preparing for the replica database

To prepare a snapshot for a replica database 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. On the master node, verify that the volume has an instant snap data change object (DCO) and DCO volume, and FastResync is enabled on the volume:

   ```
   # vxprint -g database_dg -F%instant database_vol

   # vxprint -g database_dg -F%fastresync database_vol
   ```

   If both commands return the value as ON, proceed to step 3. Otherwise, continue with step 2.

2. Use the following command to prepare a volume for instant snapshots:

   ```
   # vxsnap -g database_dg prepare database_vol [regionsize=size] \
   [ndcomirs=number] [alloc=storage_attributes]
   ```

3. 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 database_dg 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.

4. 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`. These steps are valid only for an instant snap DCO.

---

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

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

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

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

7 Use the `vxassist` command to create a volume, snapvol, of the required size and redundancy. You can use storage attributes to specify which disks should be used for the volume. The `init=active` attribute makes the volume available immediately.

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

8 Prepare the snapshot volume for instant snapshot operations as shown here:

```
# vxsnap [-g diskgroup] prepare snapvol [ndcomirs=number] \
  regionsz=$RSZ [storage_attributes]
```

It is recommended that you specify the same number of DCO mirrors (`ndcomirror`) as the number of mirrors in the volume (`nmirror`).

9 To create the snapshot, use the following command:

```
# vxsnap -g database_dg 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 database_dg 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 database_dg 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. To create the cache object, follow step 10 through step 13.

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 database_dg make \
  source=vol1/newvol=svol1/cache=dbaseco \
  source=vol2/newvol=svol2/cache=dbaseco \
  source=vol3/newvol=svol3/cache=dbaseco
```

10 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.

11 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.

12 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 you specify the region size, 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 not allowed to grow in size as required, specify `autogrow=off`. By default, the ability to automatically grow the cache is turned on.

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
```

13 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
```

Note: This step sets up the snapshot volumes, and starts tracking changes to the original volumes.

Creating a replica database

After you prepare the snapshot, you are ready to create a replica of the database.

To create the replica database

1 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.

```
#!/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
!
```

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.

```
#!/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
!
```

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 143.

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

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

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

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

3 If you temporarily suspended updates to volumes in step 1, 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.

```
#!/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
!
```

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

```
#!/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
!
```

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 143.

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

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

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

```
# fsck -t vxfs /dev/vx/rdsk/database_dg/snap1_dbase_vol
# mount -t vxfs /dev/vx/dsk/database_dg/snap1_dbase_vol \
  /rep_dbase_vol
```

5 Copy any required log files from the primary database to the replica database.

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.

6 As the database administrator, start the new database:

- For a Sybase ASE database, use a script such as that shown in the example.

```
#!/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 -t vxfs /dev/vx/rdsk/$snapvoldg/snap_$i
   mount -t 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
!
```

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 1 to refresh the contents of the snapshot from the original volume.

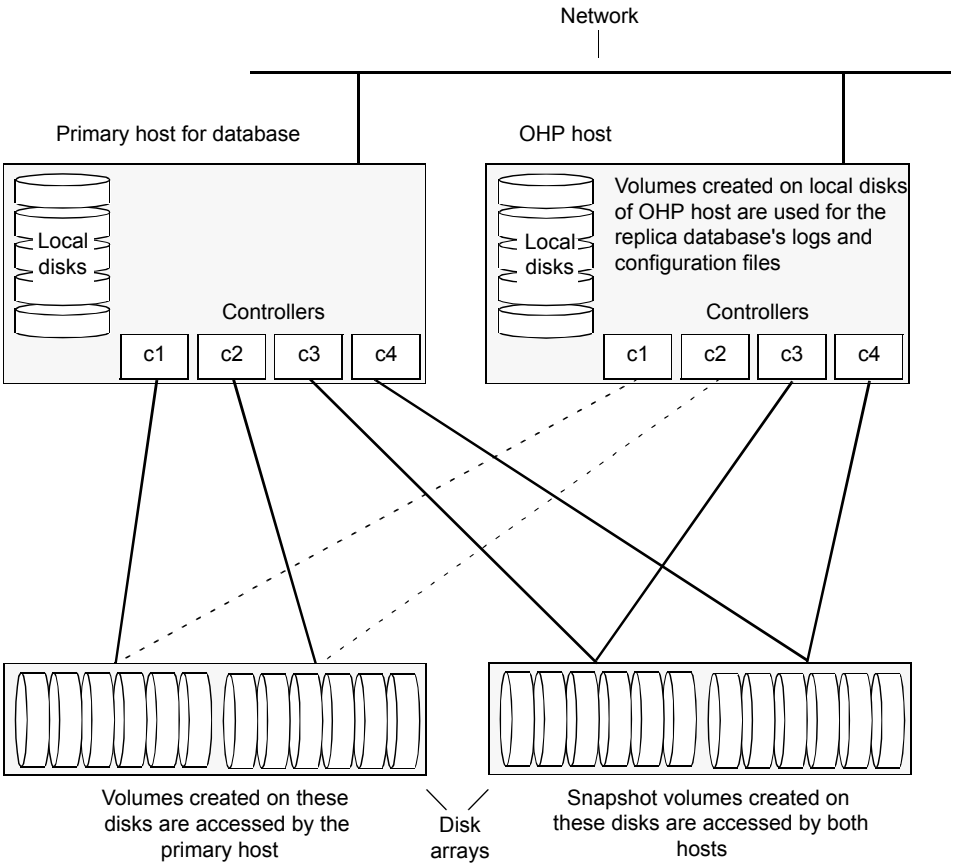
Creating an off-host replica database

Figure 10-2 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 off-host processing 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 10-2 Example system configuration for off-host decision support



To set up a replica database to be used for decision support on another host

- Prepare the full-sized snapshot.
  See "Preparing a space-optimized snapshot for a database backup" on page 88.
- Create snapshot mirrors for volumes containing VxFS file systems for database files to be backed up.
- Make the database replica.
- All commands require superuser (`root`) or equivalent privileges.

Setting up a replica database for off-host decision support

To set up a replica database for off-host decision support

1. If you have not already done so, prepare the off-host processing 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. 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 database_dg 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.

   Then use the following command to create the snapshot:

   ```
   # vxsnap -g database_dg 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 database_dg 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.

```
#!/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
!
```

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.

```
#!/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
!
```

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 143.

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

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

5 If you temporarily suspended updates to volumes in step 3 , 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.

```
#!/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
!
```

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

```
#!/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
!
```

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 database_dg 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 database_dg syncwait svol1
# vxsnap -g database_dg syncwait svol2
# vxsnap -g database_dg 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, *database_dg*:

```
# vxdg split database_dg snapvoldg snapvol ...
```

For example to split the snap volumes from *database_dg*:

```
# vxdg split database_dg 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 off-host processing 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 VxVM will recover the volumes automatically after the disk group import unless it is set to not recover automatically. Check if the snapshot volume is initially disabled and not recovered following the split.

If a volume is in the DISABLED state, use the following command on the off-host processing host to recover and restart the snapshot volume:

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

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

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

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

```
# fsck -t vxfs /dev/vx/rdsk/snapvoldg/snap1_dbase_vol
# mount -t 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 off-host processing host.

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 off-host processing 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 a Sybase ASE database, use a script such as that shown in the example.

```
#!/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 $*
do
   fsck -t vxfs /dev/vx/rdsk/$snapvoldg/snap_$i
   mount -t vxfs /dev/vx/dsk/$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
!
```

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 off-host processing host, shut down the replica database, and use the following command to unmount each of the snapshot volumes:

```
# umount mount_point
```

2 On the off-host processing 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 database_dg
```

5 VxVM will recover the volumes automatically after the join unless it is set to not recover automatically. Check if the snapshot volumes are initially disabled and not recovered following the join.

If a volume is in the DISABLED state, use the following command on the primary host to recover and restart the snapshot volume:

```
# vxrecover -g database_dg -m snapvol
```

6 Use the steps in Creating an off-host replica database to resynchronize the snapshot and make the snapshot available at off-host processing host again.

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 off-host processing host.

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

   ```
   load transaction from dump_device with standby_access
   ```

3. On the off-host processing 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.

What is off-host processing?

Off-host processing consists of performing operations on application data on a host other than the one where the application is running. Typical operations include Decision Support Systems (DSS) and backup. In a VVR environment, off-host processing operations can be performed on the Secondary of the Replicated Data Set. This reduces the load on the application server, the Primary.

The model for data access on the Secondary is that you break off a mirror from each data volume in the RVG, perform the operation on the mirror, and then reattach the mirror while replication is in progress.

About using VVR for off-host processing

This chapter explains how to use Volume Replicator (VVR) for off-host processing on the Secondary host. You can use the In-Band Control (IBC) Messaging feature with the FastResync (FMR) feature of Veritas Volume Manager (VxVM) and its integration with VVR to take application-consistent snapshots at the replicated volume group (RVG) level. This lets you perform off-host processing on the Secondary host.

This chapter explains how to perform off-host processing operations using the `vradmin ibc` command. You can also use the `vxibc` commands to perform off-host processing operations.

Chapter 11

Creating and refreshing test environments

This chapter includes the following topics:

- About test environments
- Creating a test environment
- Refreshing a test environment

About test environments

Sometimes, there is a need to do some testing or development on a copy of production data. In such scenarios, it is essential to provide isolation of these environments from the production environment. This is required so that there is no interference of testing or development environments with production environment. Storage Foundation can provide a very efficient and cost effective mechanism to create multiple test setups at the same time from a copy of production data. This is done without affecting performance of the production application, at the same time providing complete isolation.

Creating a test environment

Before you set up a test or development environment, you must have a production application volume already created in the application disk group.

To prepare for a test environment

- Prepare the application data volume(s) for snapshot operation

```
# vxsnap -g appdg prepare appvol
```

To create a test environment

1. Identify disks to create break-off snapshots. These disks need not be from the same array as the application volume. These disks must be visible to the host that will run test/dev environment.
2. Use these disks to create a mirror breakoff snapshot:
   - Add the mirror to create a breakoff snapshot. This step copies application volume data into the new mirror added to create the snapshot.

     ```
     # vxsnap -g appdg addmir appvol alloc=<sdisk1,sdisk2,...>
     ```

   - Create a snapshot.

     ```
     # vxsnap -g appdg make src=appvol/nmirror=1/new=snapvol
     ```

3. Split the diskgroup containing the mirror breakoff snapshot.

   ```
   # vxdg split appdg testdevdg snapvol
   ```

4. Deport the diskgroup from the production application host

   ```
   # vxdg deport testdevdg
   ```

5. Import the *testdev* disk group on the host that will run the test environment.

   ```
   # vxdg import testdevdg
   ```

   Once this step is done, the *snapvol* present in the *testdevdg* disk group is ready to be used for testing or development purposes. If required, it is also possible to create multiple copies of *snapvol* using Storage Foundation's Flashsnap feature by creating a snapshot of *snapvol* using method described above.

Refreshing a test environment

Periodically, it may be required to resynchronize the test or development environment with current production data. This can be efficiently achieved using the Flashsnap feature of Storage Foundation and High Availability Solutions products.

To refresh a test environment

1. Deport the *testdevdg* disk group from the test environment. This step requires stopping the usage of *snapvol* in the test environment.

   ```
   # vxdg deport testdevdg
   ```

2. Import *testdevdg* into the production environment.

   ```
   # vxdg import testdevdg
   ```

3. Reattach the *snapvol* to *appvol* in order to synchronize current production data. Note that this synchronization is very efficient since it copies only the changed data.

   ```
   # vxsnap -g appdg reattach snapvol source=appvol
   ```

4. When you need to setup the *testdevdg* environment again, recreate the break-off snapshot.

   ```
   # vxsnap -g appdg make src=appvol/nmirror=1/new=snapvol
   ```

5. Split the diskgroup containing the mirror breakoff snapshot.

   ```
   # vxdg split appdg testdevdg snapvol
   ```

6. Deport the diskgroup from the production application host

   ```
   # vxdg deport testdevdg
   ```

7. Import the *testdev* disk group on the host that will run the test environment.

   ```
   # vxdg import testdevdg
   ```

   Once this step is done, the *snapvol* present in *testdevdg* is ready to be used for testing or development purposes.

You can also create further snapshots of *snapvol* in order to create more test or development environments using the same snapshot. For this purpose, the following mechanisms can be used:

- Mirror breakoff snapshots
  See "Preparing a full-sized instant snapshot for a backup" on page 86.
- Space-optimized snapshot
  See "Preparing a space-optimized snapshot for a database backup" on page 88.
- Veritas File System Storage Checkpoints

See “Creating Storage Checkpoints” on page 110.

For more detailed information, see the *Storage Foundation™ Administrator's Guide*

Chapter 12

Creating point-in-time copies of files

This chapter includes the following topics:

- Using FileSnaps to create point-in-time copies of files

Using FileSnaps to create point-in-time copies of files

The key to obtaining maximum performance with FileSnaps is to minimize the copy-on-write overhead. You can achieved this by enabling lazy copy-on-write. Lazy copy-on-write is easy to enable and usually results in significantly better performance. If lazy copy-on-write is not a viable option for the use case under consideration, an efficient allocation of the source file can reduce the need of copy-on-write.

Using FileSnaps to provision virtual desktops

Virtual desktop infrastructure (VDI) operating system boot images are a good use case for FileSnaps. The parts of the boot images that can change are user profile, page files (or swap for UNIX/Linux) and application data. You should separate such data from boot images to minimize unsharing. You should allocate a single extent to the master boot image file.

The following example uses a 4 GB master boot image that has a single extent that will be shared by all snapshots.

```
# touch /vdi_images/master_image
# /opt/VRTS/bin/setext -r 4g -f chgsize /vdi_images/master_image
```

The `master_image` file can be presented as a disk device to the virtual machine for installing the operating system. Once the operating system is installed and configured, the file is ready for snapshots.

Using FileSnaps to optimize write intensive applications for virtual machines

When virtual machines are spawned to perform certain tasks that are write intensive, a significant amount of unsharing can take place. Veritas recommends that you optimize performance by enabling lazy copy-on-write. If the use case does not allow enabling lazy copy-on-write, with careful planning, you can reduce the occurrence of unsharing. The easiest way to reduce unsharing is to separate the application data to a file other than the boot image. If you cannot do this due to the nature of your applications, then you can take actions similar to the following example.

Assume that the disk space required for a boot image and the application data is 20 GB. Out of this, only 4 GB is used by the operating system and the remaining 16 GB is the space for applications to write. Any data or binaries that are required by each instance of the virtual machine can still be part of the first 4 GB of the shared extent. Since most of the writes are expected to take place on the 16 GB portion, you should allocate the master image in such a way that the 16 GB of space is not shared, as shown in the following commands:

```
  # touch /vdi_images/master_image
  # /opt/VRTS/bin/setext -r 4g -f chgsize /vdi_images/master_image
  # dd if=/dev/zero of=/vdi_images/master_image seek=20971520 \
bs=1024 count=1
```

The last command creates a 20 GB hole at the end of the file. Since holes do not have any extents allocated, the writes to hole do not need to be unshared.

Using FileSnaps to create multiple copies of data instantly

It is common to create one or more copies of production data for the purpose of generating reports, mining, and testing. These cases frequently update the copies of the data with the most current data, and one or more copies of the data always exists. FileSnaps can be used to create multiple copies instantly. The application that uses the original data can see a slight performance hit due to the unsharing of data that can take place during updates.

Section 6

Maximizing storage utilization

- Chapter 13. Optimizing storage tiering with SmartTier
- Chapter 14. Optimizing storage with Flexible Storage Sharing

Chapter 13

Optimizing storage tiering with SmartTier

This chapter includes the following topics:

- About SmartTier
- About VxFS multi-volume file systems
- About VxVM volume sets
- About volume tags
- SmartTier use cases for Sybase
- Setting up a filesystem for storage tiering with SmartTier
- Relocating old archive logs to tier two storage using SmartTier
- Relocating inactive tablespaces or segments to tier two storage
- Relocating active indexes to premium storage
- Relocating all indexes to premium storage

About SmartTier

SmartTier matches data storage with data usage requirements. After data matching, the data can then be relocated based upon data usage and other requirements determined by the storage or database administrator (DBA).

As more and more data is retained over a period of time, eventually, some of that data is needed less frequently. The data that is needed less frequently still requires a large amount of disk space. SmartTier enables the database administrator to manage data so that less frequently used data can be moved to slower, less

expensive disks. This also permits the frequently accessed data to be stored on faster disks for quicker retrieval.

Tiered storage is the assignment of different types of data to different storage types to improve performance and reduce costs. With SmartTier, storage classes are used to designate which disks make up a particular tier. There are two common ways of defining storage classes:

- Performance, or storage, cost class: The most-used class consists of fast, expensive disks. When data is no longer needed on a regular basis, the data can be moved to a different class that is made up of slower, less expensive disks.
- Resilience class: Each class consists of non-mirrored volumes, mirrored volumes, and n-way mirrored volumes.

  For example, a database is usually made up of data, an index, and logs. The data could be set up with a three-way mirror because data is critical. The index could be set up with a two-way mirror because the index is important, but can be recreated. The redo and archive logs are not required on a daily basis but are vital to database recovery and should also be mirrored.

SmartTier is a VxFS feature that enables you to allocate file storage space from different storage tiers according to rules you create. SmartTier provides a more flexible alternative compared to current approaches for tiered storage. Static storage tiering involves a manual one- time assignment of application files to a storage class, which is inflexible over a long term. Hierarchical Storage Management solutions typically require files to be migrated back into a file system name space before an application access request can be fulfilled, leading to latency and run-time overhead. In contrast, SmartTier allows organizations to:

- Optimize storage assets by dynamically moving a file to its optimal storage tier as the value of the file changes over time
- Automate the movement of data between storage tiers without changing the way users or applications access the files
- Migrate data automatically based on policies set up by administrators, eliminating operational requirements for tiered storage and downtime commonly associated with data movement

---

Note: SmartTier is the expanded and renamed feature previously known as Dynamic Storage Tiering (DST).

SmartTier policies control initial file location and the circumstances under which existing files are relocated. These policies cause the files to which they apply to be created and extended on specific subsets of a file systems's volume set, known as

placement classes. The files are relocated to volumes in other placement classes when they meet specified naming, timing, access rate, and storage capacity-related conditions.

In addition to preset policies, you can manually move files to faster or slower storage with SmartTier, when necessary. You can also run reports that list active policies, display file activity, display volume usage, or show file statistics.

SmartTier leverages two key technologies included with Veritas InfoScale products: support for multi-volume file systems and automatic policy-based placement of files within the storage managed by a file system. A multi-volume file system occupies two or more virtual storage volumes and thereby enables a single file system to span across multiple, possibly heterogeneous, physical storage devices. For example the first volume could reside on EMC Symmetrix DMX spindles, and the second volume could reside on EMC CLARiiON spindles. By presenting a single name space, multi-volumes are transparent to users and applications. This multi-volume file system remains aware of each volume's identity, making it possible to control the locations at which individual files are stored. When combined with the automatic policy-based placement of files, the multi-volume file system provides an ideal storage tiering facility, which moves data automatically without any downtime requirements for applications and users alike.

In a database environment, the access age rule can be applied to some files. However, some data files, for instance are updated every time they are accessed and hence access age rules cannot be used. SmartTier provides mechanisms to relocate portions of files as well as entire files to a secondary tier.

To use SmartTier, your storage must be managed using the following features:

- VxFS multi-volume file system
- VxVM volume set
- Volume tags
- SmartTier management at the file level
- SmartTier management at the sub-file level

About VxFS 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, and 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.

This feature is available only on file systems meeting the following requirements:

- The minimum disk group version is 140.
- The minimum file system layout version is 7 for file level SmartTier.
- The minimum file system layout version is 8 for sub-file level SmartTier.

To convert your existing VxFS system to a VxFS multi-volume file system, you must convert a single volume to a volume set.

The VxFS volume administration utility (fsvoladm utility) can be used to administer VxFS volumes. The fsvoladm utility performs administrative tasks, such as adding, removing, resizing, encapsulating volumes, and setting, clearing, or querying flags on volumes in a specified Veritas File System.

See the `fsvoladm` (1M) manual page for additional information about using this utility.

About VxVM volume sets

Volume sets allow several volumes to be represented by a single logical object. Volume sets cannot be empty. All I/O from and to the underlying volumes is directed via the I/O interfaces of the volume set. The volume set feature supports the multi-volume enhancement to Veritas File System (VxFS). This feature allows file systems to make best use of the different performance and availability characteristics of the underlying volumes. For example, file system metadata could be stored on volumes with higher redundancy, and user data on volumes with better performance.

About volume tags

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.

Warning: Multiple tagging should be used carefully.

A placement class is a SmartTier 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.

SmartTier use cases for Sybase

Veritas InfoScale products include SmartTier, a storage tiering feature which enables you to tier your data to achieve optimal use of your storage.

Example procedures illustrate the following use cases:

- Relocating archive logs older than 2 days to Tier-2 storage
- Relocating inactive tablespaces or segments to Tier-2 storage
- Relocating active indexes to Tier-0 storage
- Relocating all indexes to Tier-0 storage

Setting up a filesystem for storage tiering with SmartTier

In the use case examples, the following circumstances apply:

- The database containers are in the file system */DBdata*
- The database archived logs are in the file system */DBarch*

To create required filesystems for SmartTier

1 List the disks:

```
# vxdisk list

DEVICE         TYPE             DISK        GROUP      STATUS
fas30700_0   auto:cdsdisk    fas30700_0   ---      online thin
fas30700_1   auto:cdsdisk    fas30700_1   ---      online thin
fas30700_2   auto:cdsdisk    fas30700_2   ---      online thin
fas30700_3   auto:cdsdisk    fas30700_3   ---      online thin
fas30700_4   auto:cdsdisk    fas30700_4   ---      online thin
fas30700_5   auto:cdsdisk    fas30700_5   ---      online thin
fas30700_6   auto:cdsdisk    fas30700_6   ---      online thin
fas30700_7   auto:cdsdisk    fas30700_7   ---      online thin
fas30700_8   auto:cdsdisk    fas30700_8   ---      online thin
```

Assume there are 3 LUNs on each tier.

2 Create the disk group.

```
# vxdg init DBdg  fas30700_0   fas30700_1   fas30700_2  \
fas30700_3   fas30700_4   fas30700_5   fas30700_6   fas30700_7  \
fas30700_8
```

3 Create the volumes *datavol* and *archvol*.

```
# vxassist -g DBdg make datavol  200G alloc=fas30700_3,\
fas30700_4,fas30700_5
# vxassist -g DBdg make archvol    50G alloc= fas30700_3,\
fas30700_4,fas30700_5
```

Tag *datavol* and *archvol* as tier-1.

```
# vxassist -g DBdg settag datavol vxfs.placement_class.tier1
# vxassist -g DBdg settag archvol vxfs.placement_class.tier1
```

4 Create the Tier-0 volumes.

```
# vxassist -g DBdg  make tier0_vol1 50G alloc= fas30700_0,\
fas30700_1,fas30700_2
# vxassist -g DBdg  make tier0_vol2 50G alloc= fas30700_0,\
fas30700_1,fas30700_2
# vxassist -g DBdg settag tier0_vol1 vxfs.placement_class.tier0
# vxassist -g DBdg settag tier0_vol2 vxfs.placement_class.tier0
```

5 Create the Tier-2 volumes.

```
# vxassist -g DBdg  make tier2_vol1 50G alloc= fas30700_6,\
fas30700_7,fas30700_8
# vxassist -g DBdg  make tier2_vol2 50G alloc= fas30700_6,\
fas30700_7,fas30700_8
# vxassist -g DBdg settag tier2_vol1 vxfs.placement_class.tier2
# vxassist -g DBdg settag tier2_vol2 vxfs.placement_class.tier2
```

6 Convert *datavol* and *archvol* to a volume set.

```
# vxvset -g DBdg make datavol_mvfs datavol
# vxvset  -g DBdg  make archvol_mvfs archvol
```

7 Add the volumes *Tier-0* and *Tier-2* to *datavol_mvfs*.

```
# vxvset -g  DBdg addvol datavol_mvfs tier0_vol1
# vxvset -g  DBdg addvol datavol_mvfs  tier2_vol1
```

8 Add the volume *Tier-2* to *archvol_mvfs*

```
# vxvset -g DBdg  archvol_mvfs  tier2_vol2
```

9 Make the file system and mount *datavol_mvfs* and *archvol_mvfs*.

```
# mkfs -t vxfs /dev/vx/rdsk/DBdg/datavol_mvfs
```

10 Mount the *DBdata* file system

```
# mount -t vxfs /dev/vx/dsk/DBdg/datavol_mvfs /DBdata
```

11 Mount the *DBarch* filesystem

```
# mount -t  vxfs /dev/vx/dsk/DBdg/archvol_mvfs /DBarch
```

12 Migrate the database into the newly created, SmartTier-ready file system. You can migrate the database either by restoring from backup or copying appropriate files into respective filesystems.

See the database documentation for more information.

Relocating old archive logs to tier two storage using SmartTier

A busy database can generate few hundred gigabytes of archivelogs per day. Restoring these archive logs from tape backup is not ideal because it increases database recovery time. Regulatory requirements could mandate that these archive logs be preserved for several weeks.

To save storage costs, you can relocate archive logs older than two days (for example) into tier two storage. To achieve this you must create a policy file, for example, archive_policy.xml.

Note: The relocating archive logs use case applies for Sybase environments.

To relocate archive logs that are more than two days old to Tier-2

1 Create a policy file. A sample XML policy file is provided below.

```
<?xml version="1.0"?>
<!DOCTYPE PLACEMENT_POLICY SYSTEM "/opt/VRTSvxfs/etc\
   /placement_policy.dtd">
<PLACEMENT_POLICY Version="5.0" Name="access_age_based">
    <RULE Flags="data" Name="Key-Files-Rule">
         <COMMENT>
            This rule deals with key files such as archive logs.
         </COMMENT>

    <SELECT Flags="Data">
         <COMMENT>
             You want all files. So choose pattern as '*'
         </COMMENT>
         <PATTERN>  *  </PATTERN>
     </SELECT>

    <CREATE>
        <ON>
            <DESTINATION>
                <CLASS> tier1 </CLASS>
            </DESTINATION>
        </ON>
    </CREATE>

    <RELOCATE>
    <TO>
          <DESTINATION>
              <CLASS> tier2 </CLASS>
          </DESTINATION>
    </TO>
          <WHEN>
                  <ACCAGE Units="days">
                      <MIN Flags="gt">2</MIN>
                  </ACCAGE>
          </WHEN>
    </RELOCATE>

    </RULE>
</PLACEMENT_POLICY>
```

Notice the `ACCAGE` units in the `WHEN` clause.

2 To locate additional sample policy files, go to `/opt/VRTSvxfs/etc`.

The access age-based policy is appropriate for this use case. Pay attention to the `CREATE ON` and `RELOCATE TO` sections of the XML file.

To apply a policy file

1 As root, validate *archive_policy.xml*

```
# fsppadm validate /DBarch archive_policy.xml
```

2 If the validation process is not successful, correct the problem. Validate *archive_policy.xml* successfully before proceeding.

3 Assign the policy to /DBarch filesystem

```
# fsppadm assign /DBarch archive_policy.xml
```

4 Enforce the policy. The relocation of two day old archive logs happens when the enforcement step is performed. The policy enforcements must be done every day to relocate aged archive logs. This enforcement can be performed on demand as needed or by using a cron- like scheduler.

```
# fsppadm enforce /DBarch
```

Relocating inactive tablespaces or segments to tier two storage

It is general practice to use partitions in databases. Each partition maps to a unique tablespace. For example in a shopping goods database, the orders table can be portioned into orders of each quarter. Q1 orders can be organized into *Q1_order_tbs tablespace*, Q2 order can be organized into *Q2_order_tbs*.

As the quarters go by, the activity on older quarter data decreases. By relocating old quarter data into Tier-2, significant storage costs can be saved. The relocation of data can be done when the database is online.

For the following example use case, the steps illustrate how to relocate Q1 order data into Tier-2 in the beginning of Q3. The example steps assume that all the database data is in the */DBdata* filesystem.

To prepare to relocate Q1 order data into Tier-2 storage for DB2

1. Obtain a list of containers belonging to *Q1_order_tbs*.

```
$ db2inst1$  db2 list  tablespaces
```

2. Find the tablespace-id for the tablespace *Q1_order_tbs*.

```
$ db2inst1$ db2  list  tablespace containers for <tablespace-id>
```

3. Find the path names for the containers and store them in file *Q1_order_files.txt*.

```
#cat  Q1_order_files.txt
          NODE0000/Q1_order_file1.f
          NODE0000/Q1_order_file2.f
          ...
          NODE0000/Q1_order_fileN.f
```

To prepare to relocate Q1 order data into Tier-2 storage for Sybase

1. Obtain a list of datafiles belonging to segment *Q1_order_tbs*. System Procedures `sp_helpsegment` and *sp_helpdevice* can be used for this purpose.

```
sybsadmin$ sp_helpsegment Q1_order_tbss
```

Note: In Sybase terminology, a "tablespace" is same as a "segment."

2. Note down the device names for the segment *Q1_order_tbs*.
3. For each device name use the `sp_helpdevice` system procedure to get the physical path name of the datafile.

```
sybsadmin$ sp_helpdevice <device name>
```

4. Save all the datafile path names in *Q1_order_files.txt*

```
# cat  Q1_order_files.txt
          NODE0000/Q1_order_file1.f
          NODE0000/Q1_order_file2.f
          ...
          NODE0000/Q1_order_fileN.f
```

To relocate Q1 order data into Tier-2

1 Prepare a policy XML file. For the example, the policy file name is *Q1_order_policy.xml*. Below is a sample policy.

This is policy is for unconditional relocation and hence there is no `WHEN` clause. There are multiple `PATTERN` statements as part of the `SELECT` clause. Each `PATTERN` selects a different file.

```
<?xml version="1.0"?>
<!DOCTYPE PLACEMENT_POLICY SYSTEM "/opt/VRTSvxfs/etc/\
placement_policy.dtd">
<PLACEMENT_POLICY Version="5.0" Name="selected files">
    <RULE Flags="data" Name="Key-Files-Rule">
        <COMMENT>
            This rule deals with key important files.
        </COMMENT>

        <SELECT Flags="Data">
            <DIRECTORY Flags="nonrecursive" > NODE0000</DIRECTORY>
            <PATTERN> Q1_order_file1.f  </PATTERN>
            <PATTERN> Q1_order_file2.f  </PATTERN>
            <PATTERN> Q1_order_fileN.f  </PATTERN>
        </SELECT>

        <RELOCATE>
             <COMMENT>
                Note that there is no WHEN clause.
             </COMMENT>
             <TO>
                <DESTINATION>
                    <CLASS> tier2 </CLASS>
                </DESTINATION>
             </TO>
         </RELOCATE>

    </RULE>
</PLACEMENT_POLICY>
```

2 Validate the policy *Q1_order_policy.xml*.

```
# fsppadm validate /DBdata Q1_order_policy.xml
```

3 Assign the policy.

```
# fsppadm assign /DBdata Q1_order_policy.xml
```

4 Enforce the policy.

```
# fsppadm enforce /DBdata
```

Relocating active indexes to premium storage

The database transaction rate depends upon how fast indexes can be accessed. If Indexes reside on slow storage, the database transaction rate suffers. Tier-0 storage is generally too expensive to be practical to relocate the entire table data to Tier-0. Indexes are generally much smaller in size and are created to improve the database transaction rate, hence it is more practical to relocate active indexes to Tier-0 storage. Using SmartTier you can move active indexes to Tier-0 storage.

For the following telephone company database example procedure, assume the *call_details* table has an index *call_idx* on the column *customer_id*.

To prepare to relocate *call_idx* to Tier-0 storage for DB2

1 Find the tablespace where *call_idx* resides.

```
$ db2inst1$  db2 connect to PROD
$ db2inst1$  db2 select index_tbspace from syscat.tables \
where tabname='call_details'
```

2 In this example, the index is in tablespace *tbs_call_idx*. To get the tablespace id for *tbs_call_idx* and the list of containers:

```
$ db2inst1$  db2  list tablespaces
```

Note the tablespace id for *tbs_call_idx*.

3 List the containers and record the filenames in the tabelspace *tbs_call_idx*.

```
$ db2inst1$  db2 list tablespace containers for <tablespace-id>
```

4 Store the files in *index_files.txt*.

```
# cat  index_files.txt
/DB2data/NODE0000/IDX/call1.idx
/DB2data/NODE0000/IDX/call2.idx
/DB2data/NODE0000/IDX/call3.idx
```

To prepare to relocate *call_idx* to premium storage for Sybase

1. Obtain a list of datafiles for the *call_idx* segment.

```
$ sybsadmin$ sp_helpsegment call_idx
```

2. Note down the device names for the segment *call_idx*.
3. For each device name use the `sp_helpdevice` system procedure to get the physical pathname of the datafile.

```
sybsadmin$ sp_helpdevice <device name>
```

4. Save all the datafile path names in *index_files.txt*.

```
# cat  index_files.txt
/SYBdata/NODE0000/IDX/call1.idx
/SYBdata/NODE0000/IDX/call2.idx
/SYBdata/NODE0000/IDX/call3.idx
```

To relocate *call_idx* to Tier-0 storage

1 Prepare the policy *index_policy.xml*.

Example policy:

```
<?xml version="1.0"?>
<!DOCTYPE PLACEMENT_POLICY SYSTEM "/opt/VRTSvxfs/etc/\
placement_policy.dtd">
<PLACEMENT_POLICY Version="5.0" Name="selected files">
    <RULE Flags="data" Name="Key-Files-Rule">
        <COMMENT>
            This rule deals with key important files.
        </COMMENT>

        <SELECT Flags="Data">
            <DIRECTORY Flags="nonrecursive" > NODE0000</DIRECTORY>
            <PATTERN> call*.idx  </PATTERN>
        </SELECT>

        <RELOCATE>
            <COMMENT>
                Note that there is no WHEN clause.
         </COMMENT>
         <TO>
            <DESTINATION>
                <CLASS> tier0 </CLASS>
            </DESTINATION>
         </TO>
         </RELOCATE>

    </RULE>
</PLACEMENT_POLICY>
```

2 Assign and enforce the policy.

```
# fsppadm validate /DBdata index_policy.xml
# fsppadm assign /DBdata index_policy.xml
# fsppadm enforce /DBdata
```

Relocating all indexes to premium storage

It is a common practice for DBAs to name index files with some common extensions. For example, all index files are named with “.inx” extensions. If your Tier-0 storage

has enough capacity, you can relocate all indexes of the database to Tier-0 storage. You can also make sure all index containers created with this special extension are automatically created on Tier-0 storage by using the `CREATE` and `RELOCATE` clause of policy definition.

To relocate all indexes to Tier-0 storage

1 Create a policy such as the following example:

```
# cat index_policy.xml

<?xml version="1.0"?>
<!DOCTYPE PLACEMENT_POLICY SYSTEM "/opt/VRTSvxfs/etc/\
placement_policy.dtd">
<PLACEMENT_POLICY Version="5.0" Name="selected files">
    <RULE Flags="data" Name="Key-Files-Rule">
        <COMMENT>
            This rule deals with key important files.
        </COMMENT>

        <SELECT Flags="Data">
            <PATTERN> *.inx  </PATTERN>
        </SELECT>

        <CREATE>
            <COMMENT>
                Note that there  are two DESTINATION.
            </COMMENT>
            <ON>
                <DESTINATION>
                    <CLASS> tier0 </CLASS>
                </DESTINATION>
                <DESTINATION>
                    <CLASS> tier1</CLASS>
                </DESTINATION>
            </ON>
        </CREATE>

             <RELOCATE>
                 <COMMENT>
                     Note that there is no WHEN clause.
                 </COMMENT>
                 <TO>
                     <DESTINATION>
                         <CLASS> tier0 </CLASS>
                     </DESTINATION>
                 </TO>
             </RELOCATE>

    </RULE>
</PLACEMENT_POLICY>
```

2. To make sure file creation succeeds even if Tier-0 runs out of space, add two ON clauses as in the example policy in 1.
3. Assign and enforce the policy.

```
# fsppadm validate /DBdata index_policy.xml
# fsppadm assign /DBdata index_policy.xml
# fsppadm enforce /DBdata
```

Chapter 14

Optimizing storage with Flexible Storage Sharing

This chapter includes the following topics:

- About Flexible Storage Sharing
- About use cases for optimizing storage with Flexible Storage Sharing
- Setting up an SFRAC clustered environment with shared nothing storage
- Implementing the SmartTier feature with hybrid storage
- Configuring a campus cluster without shared storage

About Flexible Storage Sharing

Flexible Storage Sharing (FSS) enables network sharing of local storage, cluster wide. The local storage can be in the form of Direct Attached Storage (DAS) or internal disk drives. Network shared storage is enabled by using a network interconnect between the nodes of a cluster.

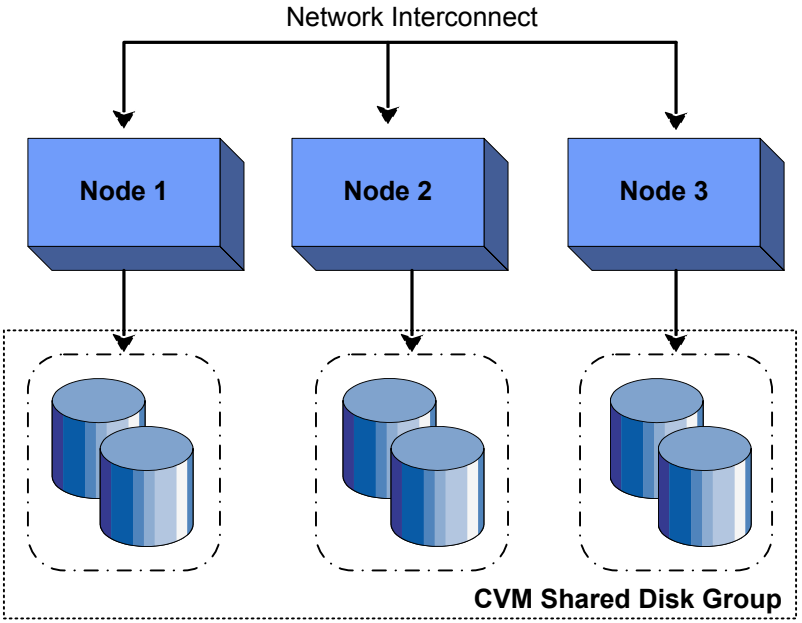
FSS allows network shared storage to co-exist with physically shared storage, and logical volumes can be created using both types of storage creating a common storage namespace. Logical volumes using network shared storage provide data redundancy, high availability, and disaster recovery capabilities, without requiring physically shared storage, transparently to file systems and applications.

FSS can be used with SmartIO technology for remote caching to service nodes that may not have local SSDs.

FSS is supported on clusters containing up to 64 nodes with CVM protocol versions 140 and above. For more details, refer to the *Veritas InfoScale Release Notes*.

Figure 14-1 shows a Flexible Storage Sharing environment.

Figure 14-1 Flexible Storage Sharing Environment



Limitations of Flexible Storage Sharing

Note the following limitations for using Flexible Storage Sharing (FSS):

- FSS is only supported on clusters of up to 64 nodes.
- Disk initialization operations should be performed only on nodes with local connectivity to the disk.
- FSS does not support the use of boot disks, opaque disks, and non-VxVM disks for network sharing.
- Hot-relocation is disabled on FSS disk groups.
- The VxVM cloned disks operations are not supported with FSS disk groups.
- FSS does not support non-SCSI3 disks connected to multiple hosts.
- Dynamic LUN Expansion (DLE) is not supported.
- FSS only supports instant data change object (DCO), created using the `vxsnap` operation or by specifying "logtype=dco dcoversion=20" attributes during volume creation.

- By default creating a mirror between SSD and HDD is not supported through `vxassist`, as the underlying mediatypes are different. To workaround this issue, you can create a volume with one mediatype, for instance the HDD, which is the default mediatype, and then later add a mirror on the SSD.
  For example:

  ```
  # vxassist -g diskgroup make volume size init=none

  # vxassist -g diskgroup mirror volume mediatype:ssd

  # vxvol -g diskgroup init active volume
  ```

  See the "Administering mirrored volumes using vxassist" section in the *Storage Foundation Cluster File System High Availability Administrator's Guide* or the *Storage Foundation for Oracle RAC Administrator's Guide*.

About use cases for optimizing storage with Flexible Storage Sharing

The following lists includes several use cases for which you would want to use the FSS feature:

- Setting up an SFRAC clustered environment with shared nothing storage
- Implementing the SmartTier feature with hybrid storage
- Configuring a campus cluster without shared storage

See the *Storage Foundation Cluster File System High Availability Administrator's Guide* or the *Storage Foundation for Oracle RAC Administrator's Guide* for more information on the FSS feature.

Setting up an SFRAC clustered environment with shared nothing storage

FSS lets you run parallel applications in an SFRAC clustered environment without Fibre Channel shared storage connectivity. The network interconnect between nodes provides low latency and high throughput network sharing of local storage. As a result, storage connectivity and topology become transparent to applications. This use case lets you quickly provision clusters for applications with parallel access without requiring complex SAN provisioning.

See the *Storage Foundation for Oracle RAC Administrator's guide* for more information on setting up an SFRAC clustered environment and administering FSS.

Implementing the SmartTier feature with hybrid storage

SmartTier lets you optimize storage tiering by matching data storage with data usage requirements. SmartTier policies relocate data based upon data usage and other predetermined requirements. Less frequently accessed data can be moved to slower disks, whereas frequently accessed data can be stored on faster disks for quicker retrieval.

FSS supports a combination of internal storage and SAN storage access to the cluster. Using SmartTier, you can map more than one volume to a single file system, and then configure policies that automatically relocate files from one volume to another to improve overall application performance. Implementing SmartTier with shared hybrid storage lets you augment overall storage with SAN storage in an online and transparent manner when local storage capacity is limited.

See the *Storage Foundation Cluster File System High Availability Administrator's Guide* for more information using SmartTier to maximize storage utilization and administering FSS.

See "About SmartTier" on page 153.

Configuring a campus cluster without shared storage

FSS lets you configure an Active/Active campus cluster configuration with nodes across the site. Network sharing of local storage and mirroring across sites provides a disaster recovery solution without requiring the cost and complexity of Fibre Channel connectivity across sites.

See the *Veritas InfoScale Disaster Recovery Implementation Guide* for more information on configuring a campus cluster.

See the *Storage Foundation Cluster File System High Availability Administrator's Guide* for more information on administering FSS.

Section 7

Migrating data

- Chapter 15. Understanding data migration
- Chapter 16. Offline migration from LVM to VxVM
- Chapter 17. Offline conversion of native file system to VxFS
- Chapter 18. Online migration of a native file system to the VxFS file system
- Chapter 19. Migrating storage arrays
- Chapter 20. Migrating data between platforms
- Chapter 21. Migrating from Oracle ASM to Veritas File System

Chapter 15

Understanding data migration

This chapter includes the following topics:

- Types of data migration

Types of data migration

This section describes the following types of data migration:

- Migrating data from LVM to Storage Foundation using offline migration
  When you install Storage Foundation, you may already have some volumes that are controlled by the Logical Volume Manager. You can preserve your data and convert these volumes to Veritas Volume Manager volumes.
  See "About migration from LVM" on page 178.
- Migrating data from native file system to VxFS using the offline conversion
  To meet your storage needs you may want to convert your native file system to VxFS. Veritas provides the vxfsconvert utility for offine conversion of the file system. The conversion is an in-place conversion where data is not copied or migrated, but only the bmaps are updated. The conversion does not allow you to revert back to the earlier file system.
  See "About the offline conversion of native file system to VxFS" on page 198.
- Migrating data between platforms using Cross-platform Data Sharing (CDS)
  Storage Foundation lets you create disks and volumes so that the data can be read by systems running different operating systems. CDS disks and volumes cannot be mounted and accessed from different operating systems at the same time. The CDS functionality provides an easy way to migrate data between one system and another system running a different operating system.
  See "Overview of the Cross-Platform Data Sharing (CDS) feature" on page 220.

- Migrating data between arrays
  Storage Foundation supports arrays from various vendors. If your storage needs change, you can move your data between arrays.
  See “Array migration for storage using Linux” on page 209.

  Note: The procedures are different if you plan to migrate to a thin array from a thick array.

Chapter 16

Offline migration from LVM to VxVM

This chapter includes the following topics:

- About migration from LVM
- Converting unused LVM physical volumes to VxVM disks
- LVM volume group to VxVM disk group conversion
- LVM volume group restoration

About migration from LVM

Veritas Volume Manager (VxVM) provides the `vxvmconvert` utility for converting Logical Volume Manager (LVM) volume groups and the objects that they contain to the equivalent VxVM disk groups and objects. Conversion of LVM2 volume groups is supported provided that the version of LVM2 is 2.00.33 or later.

Disks on your system that are managed by LVM can be of two types:

- Unused disks or disk partitions, which contain no user data, and are not used by any volume group, but which have LVM disk headers written by `pvcreate`.
  See "Converting unused LVM physical volumes to VxVM disks" on page 179.
- LVM disks or disk partitions in volume groups, and which contain logical volumes and volume groups.
- See "LVM volume group to VxVM disk group conversion" on page 180.

A converted VxVM disk group can also be reverted to an LVM volume group.

See "LVM volume group restoration" on page 196.

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

Converting unused LVM physical volumes to VxVM disks

LVM disks or disk partitions that are not part of any volume group, and which contain no user data, can be converted by removing the LVM disk headers.

Warning: Make sure that the disks to be converted are not in use in any LVM configuration. Any user data on these disks is destroyed during conversion.

To convert unused LVM physical volumes to VxVM disks

1 Use the `pvscan` command to make sure that the disk is not part of any volume group as shown in this example:

```
# pvscan
pvscan -- reading all physical volumes (this may take a while...)
pvscan -- inactive PV "/dev/sde1" is in no VG [8.48 GB]
pvscan -- ACTIVE PV "/dev/sdf" of VG "vg02" [8.47 GB / 8.47 GB free]
pvscan -- inactive PV "/dev/sdg" is in no VG [8.48 GB]
pvscan -- ACTIVE PV "/dev/sdh1" of VG "vg02" [8.47 GB / 8.47 GB free]
pvscan -- total: 4 [33.92 GB] / in use: 2 [16.96 GB] / in no
VG: 2 [16.96 GB]
```

This shows that the disk devices `sdf` and `sdh1` are associated with volume group, `vg02`, but `sde1` and `sdg` are not in any volume group.

2 Use the following commands to remove LVM header information from each disk:

```
# dd if=/dev/zero of=/dev/diskdev bs=1k count=3
# blockdev --rereadpt /dev/diskdev
```

Warning: When running `dd` on a disk partition, make sure that you specify the device for the disk partition rather than the disk name. Otherwise, you will overwrite information for other partitions on the disk.

3 After overwriting the LVM header, use the `fdisk` or `sfdisk` command to edit the partition table on the disk:

```
# fdisk -l /dev/diskdev
```

If the LVM disk was created on an entire disk, relabel it as a DOS or SUN partition.

If the LVM disk was created on a disk partition, change the partition type from "`Linux LVM`" to "`Linux`".

4 After writing the partition table to the disk, a disk or disk partition (where there is no other useful partition on the disk) may be initialized as a VM disk by running the `vxdiskadm` command and selecting item `1 Add or initialize one or more disks`, or by using the VEA GUI. For a disk partition that coexists with other partitions on a disk, initialize the partition as a simple disk.

LVM volume group to VxVM disk group conversion

Read these guidelines carefully before beginning any volume group conversion. The conversion process involves many steps. Although the tools provided help you with the conversion, some of the steps cannot be automated. Make sure that you understand how the conversion process works and what you need to do before trying to convert a volume group. Make sure that you have taken backups of the data on the volumes.

The conversion utility, `vxvmconvert`, is an interactive, menu-driven program that walks you through most of the steps for converting LVM volume groups. LVM volume groups are converted to VxVM disk groups in place. The public areas of the disks that contain user data, (file systems, databases, and so on) are not affected by the conversion. However, the conversion process overwrites the LVM configuration areas on the disks, and changes the names of the logical storage objects. For this reason, conversion is necessarily an off-line procedure. All applications must be shut down that would normally access the volume groups that are undergoing conversion.

During the conversion, the `vxvmconvert` utility tries to create space for the VxVM private region by using on-disk data migration. If a disk has enough available free space, no intervention is required. If there is insufficient space on the disk, the `vxvmconvert` utility displays a list of suitable disks in the same volume group to which the data can be migrated. After selecting a disk, the data is migrated to create space for the VxVM private region.

Volume group conversion limitations

Some LVM volume configurations cannot be converted to VxVM. The following are some reasons why a conversion might fail:

- Existing VxVM disks use enclosure-based naming (EBN). The `vxvmconvert` utility requires that the disks use operating system-based naming (OSN). If the system to be converted uses enclosure-based naming, change the disk naming scheme to OSN before conversion. After the conversion, you can change the naming scheme back to EBN.

  For more information about disk device naming in VxVM, see the *Storage Foundation Administrator's Guide*.
- The volume group has insufficient space for its configuration data. During conversion, the areas of the disks that used to store LVM configuration data are overwritten with VxVM configuration data. If the VxVM configuration data that needs to be written cannot fit into the space occupied by LVM configuration data, the volume group cannot be converted unless additional disks are specified.
- A volume group contains a root volume. The `vxvmconvert` utility does not currently support conversion to VxVM root volumes. The root disk can be converted to a VxVM volume if it is not an LVM volume.
- There is insufficient space on the root disk to save information about each physical disk. For large volume groups (for example, 200GB or more total storage on twenty or more 10GB drives), the required space may be as much as 30MB.
- An attempt is made to convert a volume which contains space-optimized snapshots. Such snapshots cannot be converted. Remove the snapshot and restart the conversion. After conversion, use the features available in VxVM to create new snapshots.
- Unsupported devices (for example, Linux metadevices or RAM disks) are in use as physical volumes.
- To create a VxVM private region, the `vxvmconvert` utility can use the LVM2 `pvmove` utility to move physical extents across a disk. This requires that the `dm_mirror` device mapper is loaded into the kernel. If extent movement is required for an LVM volume, you are instructed to use the `vgconvert` utility to convert the volume group to an LVM2 volume group.
- The volume group contains a volume which has an unrecognized partitioning scheme. Adding a disk device to VxVM control requires that VxVM recognize the disk partitioning scheme. If the Sun partitions are overwritten with LVM metadata, so that the disk has no VxVM recognized partition table, the conversion will fail.

- The volume group contains more than one physical extent on a specific disk device.

You can use the analyze option in `vxvmconvert` to help you in identifying which volume groups can be converted.

See "Examples of second stage failure analysis" on page 194.

Converting LVM volume groups to VxVM disk groups

To convert LVM volume groups to VxVM disk groups

1 Identify the LVM disks and volume groups that are to be converted. Use LVM administrative utilities such as `vgdisplay` to identify the candidate LVM volume groups and the disks that comprise them. You can also use the `listvg` operation in `vxvmconvert` to examine groups and their member disks, and the `list` operation to display the disks known to the system as shown here:

```
# vxvmconvert
   .
   .
   .
Select an operation to perform: list
   .
   .
   .
Enter disk device or "all"[<address>,all,q,?](default: all) all

DEVICE        DISK        GROUP       STATUS
cciss/c0d0    -           -           online invalid
cciss/c0d1    -           -           online
sda           -           -           online
sdb           disk01      rootdg      online
sdc           disk02      rootdg      online
sdd           disk03      rootdg      online
sde           -           -           error
sdf           -           -           error
sdg           -           -           error
sdh           -           -           error

Device to list in detail [<address>,none,q,?] (default: none)
```

The `DEVICE` column shows the disk access names of the physical disks. If a disk has a disk media name entry in the `DISK` column, it is under VM control, and the GROUP column indicates its membership of a disk group. The `STATUS` column shows the availability of the disk to VxVM. LVM disks are displayed in the `error` state as they are unusable by VxVM.

To list LVM volume group information, use the `listvg` operation:

```
Select an operation to perform: listvg
   .
   .
   .
Enter Volume Group (i.e.- vg04) or "all"
[<address>,all,q,?] (default: all) all
```

```
LVM VOLUME GROUP INFORMATION
Name         Type         Physical Volumes
vg02         Non-Root   /dev/sdf /dev/sdh1

Volume Group to list in detail
[<address>,none,q,?] (default: none) vg02
--- Volume group ---
VG Name                       vg02
VG Access                     read/write
VG Status                     available/resizable
VG #                          0
MAX LV                        256
Cur LV                        0
Open LV                       0
MAX LV Size                   255.99 GB
Max PV                        256
Cur PV                        2
Act PV                        2
VG Size                       16.95 GB
PE Size                       4 MB
Total PE                      4338
Alloc PE / Size               0 / 0
Free PE / Size                4338 / 16.95 GB
VG UUID                       IxlERp-poi2-GO2D-od2b-G7fd-3zjX-PYycMn

--- No logical volumes defined in "vg02" ---

--- Physical volumes ---
PV Name (#)                   /dev/sdf (2)
PV Status                     available / allocatable
Total PE / Free               PE 2169 / 2169
PV Name (#)                   /dev/sdh1 (1)
PV Status                     available / allocatable
Total PE / Free PE            2169 / 2169

List another LVM Volume Group? [y,n,q,?] (default: n)
```

2 Plan for the new VxVM logical volume names. Conversion changes the device names by which your system accesses data in the volumes. LVM creates device nodes for its logical volumes in `/dev` under directories named for the volume group. VxVM create device nodes in `/dev/vx/dsk/diskgroup` and

`/dev/vx/rdsk/diskgroup`. After conversion is complete, the LVM device nodes no longer exist on the system.

For file systems listed in `/etc/fstab`, `vxvmconvert` substitutes the new VxVM device names for the old LVM volume names, to prevent problems with fsck, mount, and other such utilities. However, other applications that refer to specific device node names may fail if the device no longer exists in the same place.

Examine the following types of application to see if they reference LVM device names, and are at risk:

- Databases that access raw logical devices.
- Backups that are performed on device nodes named in private files. Labeling of backups may also record device names.
- Scripts run by `cron`.
- Other administrative scripts.

3 Select item `1 Analyze LVM Volume Groups for Conversion` from the `vxvmconvert` main menu to see if conversion of each LVM volume group is possible.

This step is optional. Analysis can be run on a live system while users are accessing their data. This is useful when you have a large number of groups and disks for conversion to allow for the optimal planning and management of conversion downtime.

The following is sample output from the successful analysis of a volume group:

```
Select an operation to perform: 1
   .
   .
   .
Select Volume Groups to analyze:
[<pattern-list>,all,list,listvg,q,?] vg02

vg02

Analyze this Volume Group? [y,n,q,?] (default: y) y

Conversion Analysis of the following devices was successful.

/dev/sdf /dev/sdh1

Hit RETURN to continue.

Second Stage Conversion Analysis of vg02

Volume Group vg02 has been analyzed and prepared for conversion.

Volume Group Analysis Completed

Hit RETURN to continue.
```

If off-disk data migration is required because there is insufficient space for on-disk data migration, you are prompted to select additional disks that can be used.

The analysis may fail for one of a number of reasons.

See "Volume group conversion limitations" on page 181.

The messages output by `vxvmconvert` explain the type of failure, and detail actions that you can take before retrying the analysis.

See "Examples of second stage failure analysis" on page 194.

4 Back up your LVM configuration and user data before attempting the conversion to VxVM. Similarly, you should back up the LVM configuration itself.

Warning: During a conversion, any spurious reboots, power outages, hardware errors, or operating system bugs can have unpredictable consequences. You are advised to safeguard your data with a set of verified backups.

Before running vxvmconvert, you can use the `vgcfgbackup` utility to save a copy of the configuration of an LVM volume group, as shown here:

```
# vgcfgbackup volume_group_name
```

This creates a backup file, `/etc/lvmconf/volume_group_name.conf`. Save this file to another location (such as off-line on tape or some other medium) to prevent the conversion process from overwriting it. If necessary, the LVM configuration can be restored from the backup file.

The `vxvmconvert` utility also saves a snapshot of the LVM configuration data during conversion of each disk. This data is saved in a different format from that of `vgcfgbackup`, and it can only be used with the `vxvmconvert` program. With certain limitations, you can use the data to reinstate the LVM volumes after they have been converted to VxVM. Even though `vxvmconvert` provides this mechanism for backing up the LVM configuration, you are advised to use `vgcfgbackup` to save the LVM configuration information for each LVM volume group.

Before performing a backup of the user data, note that backup procedures may have dependencies on the volume names that are currently in use on your system. Conversion to VxVM changes the volume names. You need to understand the implications that such name changes have for restoring from any backups that you make.

5 Prevent access by applications to volumes in the volume groups to be converted. This may require you to stop databases, unmount file systems, and so on.

`vxvmconvert` attempts to unmount mounted file systems before starting conversion. However, it makes no attempt to stop applications that are using those file systems, nor does it attempt to deal with applications such as databases that are running on raw LVM volumes.

The LVM logical volumes to be converted must all be available to the `vxvmconvert` process. Do not deactivate the volume group or any logical volumes before running `vxvmconvert`.

You can use the following command to activate a volume group:

```
# vgchange -a y volume_group_name
```

6 Start the conversion of each volume group by selecting item `2 Convert LVM Volume Groups to VxVM` from the `vxvmconvert` main menu. The volume group is analyzed to ensure that conversion is possible. If the analysis is successful, you are asked whether you wish to perform the conversion.

Convert one volume group at a time to avoid errors during conversion.

The following is sample output from a successful conversion:

```
Select an operation to perform: 2
   .
   .
   .
Select Volume Groups to convert:
[<pattern-list>,all,list,listvg,q,? vg02

vg02

Convert this Volume Group? [y,n,q,?] (default: y) y

Conversion Analysis of the following devices was successful.

/dev/sdf /dev/sdh1

Hit RETURN to continue.

Second Stage Conversion Analysis of vg02

Volume Group vg02 has been analyzed and prepared for conversion.

Are you ready to commit to these changes?[y,n,q,?](default: y) y

vxlvmconv: making log directory /etc/vx/lvmconv/vg02.d/log.
vxlvmconv: starting conversion for VG "vg02" -
Thu Feb 26 09:08:57 IST 2004

vgchange -- volume group "vg02" successfully deactivated

vxlvmconv: checking disk connectivity
Starting Conversion of vg02 to VxVM
fdisk ..
disksetup ..
dginit ..
make .
```

```
volinit ..
vxlvmconv: Conversion complete.

Convert other LVM Volume Groups? [y,n,q,?] (default: n)
```

If off-disk data migration is required because there is insufficient space for on-disk data migration, you are prompted to select additional disks that can be used.

7 After converting the LVM volume groups, you can use the `list` operation in `vxvmconvert` to examine the status of the converted disks, as shown in this example:

```
Select an operation to perform: list
   .
   .
   .
Enter disk device or "all"[<address>,all,q,?](default: all) all

DEVICE       DISK         GROUP        STATUS
cciss/c0d0   -            -            online invalid
cciss/c0d1   -            -            online
sda          -            -            online
sdb          disk01       rootdg       online
sdc          disk02       rootdg       online
sdd          disk03       rootdg       online
sde1         vg0101       vg01         online
sdf          vg0201       vg02         online
sdg          vg0102       vg01         online
sdh1         vg0202       vg02         online

Device to list in detail [<address>,none,q,?] (default: none)
```

The LVM disks that were previously shown in the `error` state are now displayed as `online` to VxVM.

You can also use the `vxprint` command to display the details of the objects in the converted volumes (the `TUTIL0` and `PUTIL0` columns are omitted for clarity):

```
# vxprint

Disk group: rootdg

TY NAME         ASSOC        KSTATE    LENGTH    PLOFFS   STATE
dg rootdg       rootdg       -         -         -        -

dm disk01       sdb          -         17778528  -        -
dm disk02       sdc          -         17778528  -        -
dm disk03       sdd          -         17778528  -        -

Disk group: vg01
```

```
TY NAME         ASSOC        KSTATE    LENGTH    PLOFFS   STATE
dg vg01         vg01         -         -         -        -

dm vg0101       sde1         -         17774975  -        -
dm vg0102       sdg          -         17772544  -        -

v stripevol     gen          ENABLED   1638400   -        ACTIVE
pl stripevol-01stripevol     ENABLED   1638400   -        ACTIVE
sd vg0102-01    stripevol-01 ENABLED   819200    0        -
sd vg0101-01    stripevol-01 ENABLED   819200    0        -

Disk group: vg02

TY NAME         ASSOC        KSTATE    LENGTH    PLOFFS   STATE
dg vg02         vg02         -         -         -        -

dm vg0201       sdf          -         17772544  -        -
dm vg0202       sdh1         -         17774975  -        -

v concatvol     gen          ENABLED   163840    -        ACTIVE
pl concatvol-01concatvol     ENABLED   163840    -        ACTIVE
sd vg0202-02    concatvol-01 ENABLED   163840    0        -
v stripevol     gen          ENABLED   81920     -        ACTIVE
pl stripevol-01stripevol     ENABLED   81920     -        ACTIVE
sd vg0202-01    stripevol-01 ENABLED   40960     0        -
sd vg0201-01    stripevol-01 ENABLED   40960     0        -
```

8 Implement the changes to applications and configuration files that are required for the new VxVM volume names. (You prepared the information for this step in step 2.)

9 File systems can now be mounted on the new devices, and applications can be restarted. If you unmounted any file systems before running `vxvmconvert`, remount them using their new volume names. The `vxvmconvert` utility automatically remounts any file systems that were left mounted.

10 The disks in each new VxVM disk group are given VM disk media names that are based on the disk group name. For example, if a disk group is named `mydg`, its disks are assigned names such as `mydg01`, `mydg02`, and so on. Plexes within each VxVM volume are named `mydg01-01`, `mydg01-02`, and so on. If required, you can rename disks and plexes.

Only rename VxVM objects in the converted disk groups when you are fully satisfied with the configuration. Renaming VxVM objects prevents you from using `vxvmconvert` to restore the original LVM volume groups.

Examples of second stage failure analysis

Second stage failure analysis examines the existing LVM volume groups, and reports where manual intervention is required to correct a problem because the existing volume groups do not meet the conversion criteria.

See "Volume group conversion limitations" on page 181.

Snapshot in the volume group

The following sample output is from an analysis that has failed because of the presence of a snapshot in the volume group:

```
Snapshot conversion is not supported in this version. Please
remove this volume and restart the conversion process
if you want to continue.
```

The solution is to remove the snapshot volume from the volume group.

dm_mirror module not loaded in the kernel

The following sample output is from an analysis that has failed because the `dm_mirror` module (required by the LVM2 `pvmove` utility) is not loaded in the kernel:

```
Conversion requires some extent movement which cannot be done
without the dm_mirror target in the kernel. Please consider
installing the dm_mirror target in kernel and retry the
conversion.
```

The solution is to ensure that the `dm_mirror` module is loaded in the kernel.

Conversion requires extent movement on an LVM1 volume group

The following sample output is from an analysis that has failed because the LVM2 `pvmove` utility cannot be used to move extents on an LVM1 volume group:

```
Conversion requires some extent movement which cannot
be done on a LVM1 volume group. Please consider converting
the volume group to LVM2 and retry the conversion analysis again.
```

The solution is to use the LVM2 `vgconvert` command to convert the LVM1 volume group to an LVM2 volume group, before retrying the conversion.

Unrecognized partition in volume group

The following sample output is from an analysis that has failed because of the unrecognized partition in the volume group:

```
LVM VG(<VG name>) uses unrecognised partitioning, and cannot
be converted. Please remove the VG from the list of conversion candidates
and retry the conversion operation.
```

The solution is to use the `fdisk` utility to create a new empty DOS partition table on the device. For example:

```
# fdisk /dev/sdf
```

The following is the typical output from the `fdisk` utility:

```
Device contains neither a valid DOS partition table, nor Sun, SGI
or OSF disklabel

Building a new DOS disklabel. Changes will remain in memory only, until you
decide to write them. After that, of course, the previous content won't
be recoverable.

The number of cylinders for this disk is set to 17769.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:

1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs

(e.g., DOS FDISK, OS/2 FDISK)

Warning: invalid flag 0x0000 of partition table 4 will be corrected
```

```
by w(rite)

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

LVM volume group restoration

In some circumstances, you may want to restore an LVM volume group from a VxVM disk group, for example:

- An error occurred during conversion, such as a system crash or a disk failure, that caused the conversion to be corrupted.
- Conversion was only partially successful for a set of LVM volume groups.

The ability to restore the original LVM configuration using `vxvmconvert` depends on whether any changes have been made to the VxVM configuration since the original conversion was performed. Any of the following actions changes the VxVM configuration, and makes restoration by vxvmconvert impossible:

- Adding or removing disks to a converted disk group.
- Adding or removing converted disk groups.
- Changing the names of VxVM objects in a converted disk group.
- Resizing of volumes.

Complete restoration from backups of the LVM configuration and user data is required if the VxVM configuration has changed.

If a conversion is interrupted, you can complete it by running the command `/etc/vx/bin/vxlvmconv`, provided that you have not made any changes to the system that would make conversion impossible.

Restoring an LVM volume group

Provided that the configuration of the converted disk groups has not been changed, you can use the `vxvmconvert` command to convert the disk groups back to the original LVM volume groups. The information that was recorded during conversion about the LVM configuration and other configuration files such as `/etc/fstab` and LVM device files is used to restore the LVM volume groups. User data is not changed.

If the configuration has changed, an error is displayed. You must then use full restoration from backups instead.

Warning: A record of the LVM configuration information is stored in the `root` file system. However, this should not be taken as assurance that a full restoration from backups will not be needed.

To restore an LVM volume group

- Select item `3 Roll back from VxVM to LVM` from the main menu of the `vxvmconvert` command, as shown in this example:

```
Select an operation to perform: 3
   .
   .
   .
Select Volume Group(s) to rollback : [all,list,q,?] list

mddev
vg01
vg02

Select Volume Group(s) to rollback : [all,list,q,?] vg02
Rolling back LVM configuration records for Volume Group vg02
Starting Rollback for VG "vg02"
.......

Selected Volume Groups have been restored.

Hit any key to continue
```

Chapter 17

Offline conversion of native file system to VxFS

This chapter includes the following topics:

- About the offline conversion of native file system to VxFS
- Requirements for offline conversion of a native file system to VxFS
- Converting the native file system to VxFS

About the offline conversion of native file system to VxFS

Veritas InfoScale provides the `vxfsconvert` utility to support the offline conversion of your native file system (Ext2, Ext3, and Ext4) to VxFS. The conversion is an in-place conversion where data is not copied or migrated, but only the bmaps are updated. The applications must be offline during the conversion process.

Before converting the file system, the `vxfsconvert` utility performs the checks to determine the available free space and that the existing file system is not marked dirty. The free space is required to create the framework for VxFS by arranging the VxFS metadata and to convert the native file system metadata to that of VxFS metadata.

Once the `vxfsconvert` utility completes the conversion process, you cannot revert back to the earlier file system.

For more details about the `vxfsconvert` utility, refer to, `vxfsconvert`(1M) manual page.

Requirements for offline conversion of a native file system to VxFS

Before you run the `vxfsconvert` utility to begin the file system conversion, ensure that you meet the following requirements:

- The Logical Volume Manager (LVM) volume groups and objects are converted to the equivalent VxVM disk groups and objects.
  See "About migration from LVM" on page 178.
- Sufficient disk space is available to create the VxFS framework and to arrange the VxFS metadata.
- The native file system is not dirty.
- You have run the `vxanalyse` utility to verify if the file system conversion can be started.

  ```
  vxanalyse <mountpoint>
  ```

  Note: Support for vxanalyse is available only for Ext4.

  For more details about the `vxanalyse` utility, refer to, `vxanalyse`(1M) manual page.
- After ensuring the readiness for the conversion process, all the applications that are running on the system are brought offline.

Converting the native file system to VxFS

Perform the following steps to convert your native file system to VxFS, while the file system is not mounted (offline conversion).

Before you begin with the conversion, note that after the file system is converted to VxFS, the extended attributes of the existing file system are not preserved, and you cannot revert back to the earlier file system.

To convert the file system to VxFS

1. Install InfoScale Storage on the systems on which you want to convert the file system.

   See the `Veritas InfoScale Installation Guide`.
2. Convert the Logical Volume Manager (LVM) volume groups and objects to the equivalent VxVM disk groups and objects.

   See "About migration from LVM" on page 178.

3 Take the existing file system offline.

```
umount /mnt
```

4 Check for the amount of disk space required for conversion.

```
vxfsconvert -e /dev/vx/dsk/diskgroup_name/volume_name
UX:vxfs vxfsconvert: INFO: V-3-21783: Total of 729120K bytes
required to complete the conversion
```

5 Run the `vxfsconvert` utility to convert the file system.

```
vxfsconvert /dev/vx/dsk/testdg/vol2
Do you wish to commit to conversion? (ynq) y
UX:vxfs vxfsconvert: INFO: V-3-21852:  CONVERSION WAS SUCCESSFUL
```

6 Run VxFS Full File System Check and Repair Utility (fsck) to remove the host file system's metadata structures.

```
fsck -t vxfs -y /dev/vx/dsk/diskgroup_name/volume_name
```

7 Bring the converted (VxFS) online.

```
mount -t vxfs /dev/vx/dsk/diskgroup_name/volume_name /mnt1
```

Chapter 18

Online migration of a native file system to the VxFS file system

This chapter includes the following topics:

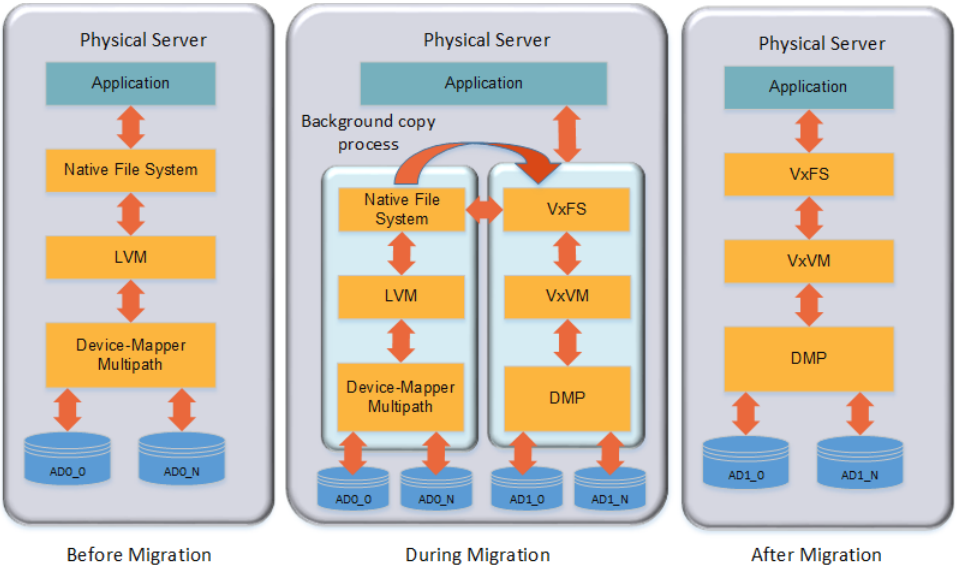
- About online migration of a native file system to the VxFS file system
- Administrative interface for online migration of a native file system to the VxFS file system
- Migrating a native file system to the VxFS file system
- Backing out an online migration of a native file system to the VxFS file system
- VxFS features not available during online migration

About online migration of a native file system to the VxFS file system

The online migration feature provides a method to migrate a native file system to the VxFS file system. The native file system is referred to as source file system and the VxFS file system is referred to as the target file system. The online migration takes minimum amounts of clearly bounded, easy to schedule downtime. Online migration is not an in-place conversion and requires a separate storage. During online migration the application remains online and the native file system data is copied over to the VxFS file system. Both of the file systems are kept in sync during migration. This makes online migration back-out and recovery seamless. The online migration tool also provides an option to throttle the background copy operation to speed up or slow down the migration based on your production needs.

Figure 18-1 illustrates the overall migration process.

Figure 18-1 Migration process



You can migrate EXT4 file system.

Administrative interface for online migration of a native file system to the VxFS file system

Online migration of a native file system to the VxFS file system can be started using the `fsmigadm` VxFS administrative command.

Table 18-1 describes the `fsmigadm` keywords.

Table 18-1

Keyword	Usage
`analyze`	Analyzes the source file system that is to be converted to VxFS and generates an analysis report.
`start`	Starts the migration.
`list`	Lists all ongoing migrations.

Table 18-1 *(continued)*

Keyword	Usage
`status`	Shows a detailed status of the migration, including the percentage of completion, for the specified file system, or for all file systems under migration.
`throttle`	Throttles the background copy operation.
`pause`	Pauses the background copy operation for one or more of migrations.
`resume`	Resumes the background copy operation if the operation was paused or the background copy operation was killed before the migration completed.
`commit`	Commits the migration.
`abort`	Aborts the migration.

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

Migrating a native file system to the VxFS file system

The following procedure migrates a native file system to the VxFS file system.

Note: You cannot unmount the target (VxFS) file system nor the source file system after you start the migration. Only the commit or abort operation can unmount the target file system. Do not force unmount the source file system; use the abort operation to stop the migration and unmount the source file system.

To migrate a native file system to the VxFS file system

1. Install Storage Foundation on the physical application host.

   See the *Veritas InfoScale Installation Guide*.

2. Add new storage to the physical application host on which you will configure Veritas Volume Manager (VxVM).

   See the *Storage Foundation Administrator's Guide*.

3 Create a VxVM volume according to the your desired configuration on the newly added storage. The volume size cannot be less than source file system size.

```
# vxdg init migdg disk_access_name
# vxassist -g migdg make vol1 size
```

See the *Storage Foundation Administrator's Guide*.

4 Mount the source file system if the file system is not mounted already.

```
# mount -t ext4 /dev/sdh /mnt1
```

5 (Optional) Run the `fsmigadm analyze` command and ensure that all checks pass:

```
# fsmigadm analyze /dev/vx/dsk/migdg/vol1 /mnt1
```

Here `/dev/vx/dsk/migdg/vol1` is the target device and `/mnt1` is the mounted source file system.

6 If the application is online, then shut down the application.

7 Start the migration by running `fsmigadm start`:

```
# fsmigadm start /dev/vx/dsk/migdg/vol1 /mnt1
```

The `fsmigadm` command performs the following tasks:

- Unmounts the source file system.
- Creates a VxFS file system using the `mkfs` command on the new storage provided, specifying the same block size (*bsize*) as the source file system. You can use the `-b` *`blocksize`* option with `fsmigadm start` to specify your desired supported VxFS block size.
- Mounts the target file system.
- Mounts the source file system inside the target file system, as `/mnt1/lost+found/srcfs`.

You can perform the following operations during the migration on the target VxFS file system:

- You can get the status of the migration using the `fsmigadm status` command:

```
# fsmigadm status /mnt1

/mnt1:
```

```
Source Device:                  /dev/sdh
Target Device:                  /dev/vx/dsk/migdg/vol1
Throttle rate:                  0 MB/s
Copy rate:                      0.00 MB/s
Total files copied:             9104
Total data copied:              585.01 MB
Migration Status:               Migration completed
```

- You can speed up or slow down the migration using the `fsmigadm throttle` command:

  ```
  # fsmigadm throttle 9g /mnt1
  ```

- You can pause the migration using `fsmigadm pause` command:

  ```
  # fsmigadm pause /mnt1
  ```

- You can resume the migration using the `fsmigadm resume` command:

  ```
  # fsmigadm resume /mnt1
  ```

The application can remain online throughout the entire migration operation. When the background copy operation completes, you are alerted via the system log.

Both the target and the source file systems are kept up-to-date until the migration is committed.

8 While the background copy operation proceeds, you can bring the application online.

9 After the background copy operation completes, if you brought the application online while the migration operation proceeded, then shut down the application again.

10 Commit the migration:

```
# fsmigadm commit /mnt1
```

The `fsmigadm` command unmounts the source file system, unmounts the target file system, and then remounts the migrated target VxFS file system on the same mount point.

Note: Make sure to commit the migration only after the background copy operation is completed.

---

11 Start the application on the Storage Foundation stack.

Backing out an online migration of a native file system to the VxFS file system

The following procedure backs out an online migration operation of a native file system to the VxFS file system.

Note: As both source and target file system are kept in sync during migration, the application sometimes experiences performance degradation.

In the case of a system failure, if the migration operation completed before the system crashed, then you are able to use the VxFS file system.

To back out an online migration operation of a native file system to the VxFS file system

1 Shut down the application

2 Abort the migration:

```
# fsmigadm abort /mnt1
```

The source file system is mounted again.

3 Bring the application online.

VxFS features not available during online migration

During the online migration process, the following VxFS features are not supported on a file system that you are migrating:

- Block clear (`blkclear`) `mount` option
- Cached Quick I/O
- Cross-platform data sharing (portable data containers)
- Data management application programming interface (DMAPI)
- File Change Log (FCL)
- File promotion (undelete)
- Fileset quotas
- Forced unmount
- Online resize
- Quick I/O
- Quotas
- Reverse name lookup
- SmartTier
- Snapshots
- Storage Checkpoints
- FileSnaps
- Compression
- SmartIO
- Storage Foundation Cluster File System High Availability (SFCFSHA)

During the online migration process, the following commands are not supported on a file system that you are migrating:

- `fiostat`
- `fsadm`
- `tar`
- `vxdump`
- `vxfreeze`

- `vxrestore`
- `vxupgrade`

All of the listed features and commands become available after you commit the migration.

Limitations of online migration

Consider the following limitations while performing online migration on VxFS:

- Online migration cannot be performed on a nested source mount point.
- Migration from a VxFS file system to a VxFS file system is not supported.
- Multiple mounts of source or target file system are not supported.
- Bind mount of a source or a target file system is not supported.
- Some source file attributes such as immutable, secured deletion, and append are lost during online migration. Only the VxFS supported extended attributes such as user, security, system.posix_acl_access, and system.posix_acl_default are preserved.
- Online migration is supported with only Oracle database workload.
- If an error is encountered during migration, migration is discontinued by disabling the target file system. The error messages are logged onto the console. After this, all file system operation by the application will fail. The user is expected to abort the migration manually. After the abort operation, the application needs to be brought online, on the source (native) file system.

Chapter 19

Migrating storage arrays

This chapter includes the following topics:

- Array migration for storage using Linux
- Overview of storage mirroring for migration
- Allocating new storage
- Initializing the new disk
- Checking the current VxVM information
- Adding a new disk to the disk group
- Mirroring
- Monitoring
- Mirror completion
- Removing old storage
- Post-mirroring steps

Array migration for storage using Linux

The array migration example documented for this use case uses a Linux system. The example details would be different for AIX, Solaris, or Windows systems.

Storage Foundation and High Availability Solutions (SFHA Solutions) products provide enterprise-class software tools which enable companies to achieve data management goals which would otherwise require more expensive hardware or time-consuming consultative solutions.

For many organizations, both large and small, storage arrays tend to serve as useful storage repositories for periods of 3-6 years. Companies are constantly evaluating

new storage solutions in their efforts to drive down costs, improve performance and increase capacity. The flexibility of Storage Foundation and High Availability Solutions enable efficient migration to new storage and improve the overall availability of data.

While there are several methods for accomplishing the migration, the most basic and traditional method is using a volume level mirror. The example procedures:

- Provide system administrators responsible for SFHA Solutions systems within their organization a demonstration of the steps required for performing an online storage array migration from one array to another.
- Illustrate the migration process using a Linux system which is connected to two different storage arrays through a SAN.
- Provide steps for starting with a file system with a single volume, mirroring the volume to a volume to another array, and then detaching the original storage.
- Are performed from the command prompt.
- Use Operating System Based Naming (OSN) for disk devices (sdb, sdc, etc).

There are two user interface options:

- The SFHA Solutions command line interface (CLI).
- The Veritas InfoScale Operations Manager graphical user interface (GUI) has a storage migration wizard.
  See the Veritas InfoScale Operations Manager documentation for details:
  https://sort.veritas.com/documents/doc_details/vom/7.0/Windows%20and%20UNIX/ProductGuides/

Note: Veritas NetBackup PureDisk comes bundled with the Storage Foundation and High Availability Solutions software for the purpose of enhanced storage management and high availability. Storage arrays used with PureDisk can also be migrated using the SFHA Solutions methodologies.

Overview of storage mirroring for migration

The migration example occurs between a Hitachi 9900 array, 350 GB disk/LUN and a NetApps 3050 Fibre-Attached 199GB disk/LUN.

To migrate storage using storage mirroring

1. Connect the new array to the SAN.
2. Zone the array controller ports(s) to the server HBA port(s).
3. Create or present the new LUN(s) on the array.
4. Map the new LUN(s) to the server HBA port(s).

5 Stop any processes that are running on this volume or file system.

6 Rescan hardware using `rescan-scsi-bus.sh` and `scsidev` commands, or reboot (optional).

7 Confirm that the operating system has access to the new target storage (*Array-B*).

8 Bring new disks under Veritas Volume Manager (VxVM) control.

9 Start the VxVM mirroring process to synchronize the plexes between the source and target array enclosures.

10 Monitor the mirroring process.

11 After mirroring is complete, logically remove disks from VxVM control.

Note: The example Linux system happens to be running as a Veritas NetBackup Puredisk server which includes the Storage Foundation software. Puredisk also supports this mode of storage array migration.

---

12 Disconnect the old storage array (enclosure).

Allocating new storage

The first step to migrating array storage is to allocate new storage to the server.

To allocate new storage as in the example

1 Create the LUN(s) on the new array.

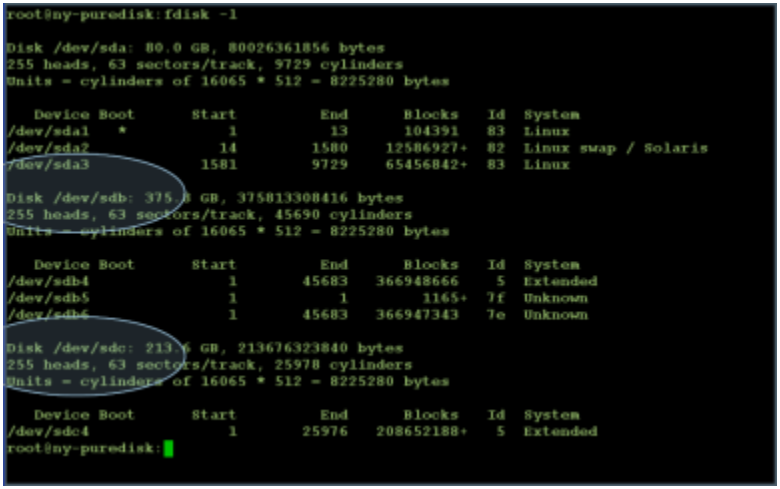
2 Map the new LUN(s) to the server.

3 Zone the new LUN(s) to the server.

4 Reboot or rescan using a native OS tool such as “fdisk” and the new external disk is now visible.

In the example, the original disk (*/dev/sdb*) has already been initialized by Veritas Volume Manager (VxVM).

Note that it has a partition layout already established. Note also the different disk sizes. It may turn out that you want to use smaller or larger LUNs. This is fine, but if you are going to mirror to a smaller LUN you will need to shrink the original volume so that it can fit onto the physical disk device or LUNs.



To shrink the original volume

- You can shrink the new volume size to *n* gigabytes:

  ```
  # vxassist -g diskgroup_name shrinkto volume_name ng
  ```

  Then resize the file system:

  ```
  # /opt/VRTS/bin/fsadm -t vxfs -b new_size_in_sectors /Storage
  ```

  Alternately, use the `vxresize` command to resize both the volume and the file system.

To grow the original volume

- You can increase the new volume size to *n* gigabytes:

  ```
  # vxassist -g diskgroup_name growto volume_name ng
  ```

  Then resize the file system:

  ```
  # /opt/VRTS/bin/fsadm -t vxfs -b new_size_in_sectors /Storage
  ```

  Alternately, use the `vxresize` command to resize both the volume and the file system.

Note: SmartMove enables you to migrate from thick array LUNs to thin array LUNs on those enclosures that support Thin Provisioning.

Initializing the new disk

Now that the operating system recognizes the new disk, the next step is to initialize it.

To initialize the disk as in the example

- Use the `vxdisksetup` command to establish the appropriate VxVM-friendly partition layout for Veritas Volume Manager.

Note below that the internal name *OLDDISK* is assigned to the old disk. The new disk is assigned a unique name later for the sake of clarity.

```
root@ny-puredisk:vxdisk list
DEVICE       TYPE            DISK         GROUP        STATUS
sda          auto:none       -            -            online invalid
sdb          auto:sliced     OLDDISK      PDDG         online
sdc          auto:none       -            -            online invalid
root@ny-puredisk:
root@ny-puredisk:
root@ny-puredisk:/usr/lib/vxvm/bin/vxdisksetup sdc
root@ny-puredisk:vxdisk list
DEVICE       TYPE            DISK         GROUP        STATUS
sda          auto:none       -            -            online invalid
sdb          auto:sliced     OLDDISK      PDDG         online
sdc          auto:none       -            -            online invalid
root@ny-puredisk:
```

The disk is now initialized under VxVM control. Note below, that the disk has a new partition table similar to the existing disk (*sdb*)and is ready to be joined to the Veritas disk group *PDDG* (name of the example disk group).

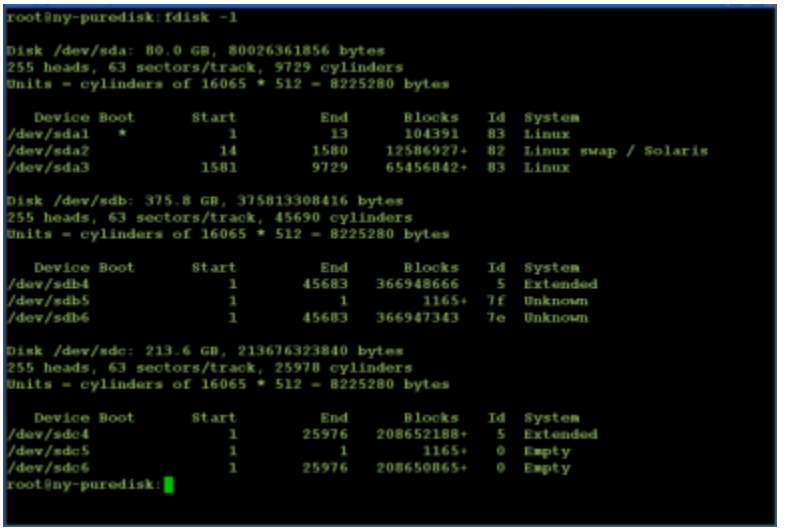
```
root@ny-puredisk:fdisk -l /dev/sdc

Disk /dev/sdc: 213.6 GB, 213676323840 bytes
255 heads, 63 sectors/track, 25978 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc4               1       25976   208652188+   5  Extended
/dev/sdc5               1           1        1165+  7f  Unknown
/dev/sdc6               1       25976   208650865+  7e  Unknown
root@ny-puredisk:
```

Checking the current VxVM information

Check the VxVM information after initializing the new disk. The screen shot below illustrates all the disks on the server along with their corresponding partition tables. Note that disks *sdb* and *sdc* are partitioned in the same manner since they were both set up with the `vxdisksetup` command.



The screen shot below shows the VxVM hierarchy for existing storage objects. Remember that we are working with a live and running server. We are using a logical disk group called *PDDG* which has other storage objects subordinate to it. The most important storage object here is the volume which is called *Storage*. The volume name can be any arbitrary name that you want, but for this example, the volume name is "*Storage*". The volume object is denoted by "v" in the output of the `vxprint` command. Other objects are subdisks (*sd*) which represents a single contiguous range of blocks on a single LUN. The other object here is a plex ("pl") which represents the virtual object or container to which the OS reads and writes. In `vxprint`, the length values are expressed in sectors, which in Linux are 512 bytes each. The raw volume size is 377487360 sectors in length, or when multiplied by 512 bytes (512*377487360) is 193273528320 bytes, or about 193 GB(2).

Notice that when the new disk was added it was 213GB yet the original existing *Storage* volume was 250GB. The *Storage* volume had to first be shrunk to a size equal the same (or smaller) number of sectors as the disk to which it would be mirrored.

```
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0
PUTIL0
dg PDDG         PDDG         -        -        -        -        -       -

dm OLDDISK      sdb          -        733894672 -       -        -       -
dm sdc          sdc          -        417301712 -       -        -       -

v  Storage      fsgen        ENABLED  377487360 -       ACTIVE   -       -
pl Storage-02   Storage      ENABLED  377487360 -       ACTIVE   -       -
sd OLDDISK-01   Storage-02   ENABLED  377487360 0       -        -       -
root@ny-puredisk:vxedit -g PDDG rename sdc NEWDISK
```

To shrink a volume as in the example *Storage* volume

- Use the `vxresize` command:

```
# vxresize -f -t my-shrinktask -g PDDG Storage 193g
```

The original physical disk (“dm”) that has been grouped into the *PDDG* diskgroup is called *sdb* but we have assigned the internal name *OLDDISK* for the purpose of this example. This can be done with the `vxedit` command using the `rename` operand. We also see the new disk (*sdc*) under VxVM control. It has been initialized but not yet assigned to any disk group.

```
root@ny-puredisk:vxdisk list
DEVICE       TYPE            DISK         GROUP        STATUS
sda          auto:none       -            -            online invalid
sdb          auto:sliced     OLDDISK      PDDG         online
sdc          auto:none       -            -            online invalid
root@ny-puredisk:
```

Adding a new disk to the disk group

The next step is adding the new disk into the *PDDG* disk group and assigning the name of *NEWDISK* to the disk.

To add a new disk to the example disk group

1. Initialize the disk.
2. Add the disk into the *PDDG* disk group

```
root@ny-puredisk:vxdisk init sdc
root@ny-puredisk:vxdg -g PDDG adddisk sdc

root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH    PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -         -        -        -       -
dm OLDDISK      sdb          -        733894672 -        -        -       -
dm sdc          sdc          -        417301712 -        -        -       -

v  Storage      fsgen        ENABLED  377487360 -        ACTIVE   -       -
pl Storage-02   Storage      ENABLED  377487360 -        ACTIVE   -       -
sd OLDDISK-01   Storage-02   ENABLED  377487360 0        -        -       -
```

The internal VxVM name of the new disk is changed from the default *disk* to *NEWDISK*.

```
root@ny-puredisk:vxedit -g PDDG rename sdc NEWDISK
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH    PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -         -        -        -       -

dm OLDDISK      sdb          -        733894672 -        -        -       -
dm NEWDISK      sdc          -        417301712 -        -        -       -

v  Storage      fsgen        ENABLED  377487360 -        ACTIVE   -       -
pl Storage-02   Storage      ENABLED  377487360 -        ACTIVE   -       -
sd OLDDISK-01   Storage-02   ENABLED  377487360 0        -        -       -
```

Mirroring

The next step is to start the mirroring process. We used the `vxassist` command to transform the *Storage* volume from a simple, concatenated volume into a mirrored volume. Optionally, a DRL (Dirty Region Log) can be added to the volume. If enabled, the DRL speeds recovery of mirrored volumes after a system crash. It requires an additional 1 Megabyte of extra disk space.

```
root@ny-puredisk: /usr/sbin/vxassist -b -g PDDG mirror Storage layout=nostripe alloc=NEWDISK
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH    PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -         -        -        -       -

dm OLDDISK      sdb          -        733894672 -        -        -       -
dm NEWDISK      sdc          -        417301712 -        -        -       -

v  Storage      fsgen        ENABLED  377487360 -        ACTIVE   ATT1    -
pl Storage-01   Storage      ENABLED  377487360 -        TEMPRMSD ATT     -
sd NEWDISK-01   Storage-01   ENABLED  377487360 0        -        -       -
pl Storage-02   Storage      ENABLED  377487360 -        ACTIVE   -       -
sd MDSDISK-01   Storage-02   ENABLED  377487360 0        -        -       -
```

To add a DRL as in the example *Storage* volume

- Use:

```
# vxassist -g PDDG addlog Storage logtype=drl
```

For more information about DRL logging,:

See the *Storage Foundation Administrator's Guide*

Monitoring

The mirroring process must complete before you can proceed. During this time there may be a heavy I/O load on the system as the mirroring process reads from one disk and writes to another.

To monitor the mirroring progress

- Use the `vxtask list` command.

Raw I/O statistics can also be monitored with the `vxstat` command. Mirroring should be done either during times of low demand on the server, or, optionally, to have the services stopped completely. While the initial synchronization is underway, the STATE of the new plex is still TEMPRMSD.

```
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -        -        -        -       -

dm OLDDISK      sdb          -        733894672 -       -        -       -
dm NEWDISK      sdc          -        417301712 -       -        -       -

v  Storage      fsgen        ENABLED  377487360 -       ACTIVE   ATT1    -
pl Storage-01   Storage      ENABLED  377487360 -       TEMPRMSD ATT     -
sd NEWDISK-01   Storage-01   ENABLED  377487360 0       -        -       -
pl Storage-02   Storage      ENABLED  377487360 -       ACTIVE   -       -
sd OLDDISK-01   Storage-02   ENABLED  377487360 0       -        -       -
```

To pause and resume mirroring

- Use the `vxtask` command.

To throttle the mirroring process and free up I/O if needed

- Use the `vxtask` command.

```
root@ny-puredisk:vxtask list
TASKID  PTID TYPE/STATE    PCT   PROGRESS
   228          ATCOPY/R 00.12% 0/377487360/449792 PLXATT Storage Storage-01 PDDG
```

The TEMPRMSD plex state is used by `vxassist` when attaching new data plexes to a volume. If the synchronization operation does not complete, the plex and its subdisks are removed.

See https://www.veritas.com/support/en_US/article.TECH19044

Mirror completion

When the mirroring completes, you can see that the output from `vxprint` shows the volume now has two active plexes associated with it. This is the mirrored volume comprised of two plexes, each plex residing on separate physical storage arrays.

```
root@ny-puredisk:vxtask list
                                    ← (note)
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -        -        -        -       -

dm OLDDISK      sdb          -        733894672 -       -        -       -
dm NEWDISK      sdc          -        417301712 -       -        -       -

v  Storage      fsgen        ENABLED  377487360 -       ACTIVE   -       -
pl Storage-01   Storage      ENABLED  377487360 -       ACTIVE   -       -
sd NEWDISK-01   Storage-01   ENABLED  377487360 0       -        -       -
pl Storage-02   Storage      ENABLED  377487360 -       ACTIVE   -       -
sd OLDDISK-01   Storage-02   ENABLED  377487360 0       -        -       -
```

To confirm completion of the mirror task

- Use the `vxtask` command.

Removing old storage

After the mirroring process completes, you can remove the old storage.

To remove the old storage

1. Break the mirror.
2. Check the viability of the volume. Services do not need to be stopped during this phase.

3 Clean up the mirror from the old disk (*OLDDISK*).

```
root@ny-puredisk:vxassist -g PDDG remove mirror Storage alloc=\!OLDDISK  ←(note)
root@ny-puredisk:vxprint
Disk group: PDDG

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PDDG         PDDG         -        -        -        -        -       -

dm NEWDISK      sdc          -        417301712 -       -        -       -
dm OLDDISK      sdb          -        733894672 -       -        -       -

v  Storage      fsgen        ENABLED  377487360 -       ACTIVE   -       -
pl Storage-01   Storage      ENABLED  377487360 -       ACTIVE   -       -
sd NEWDISK-01   Storage-01   ENABLED  377487360 0       -        -       -
```

4 Remove the old storage from the diskgroup.

```
root@ny-puredisk:vxdg -g PDDG rmdisk OLDDISK
root@ny-puredisk:vxdisk list
DEVICE       TYPE            DISK         GROUP        STATUS
sda          auto:none       -            -            online invalid
sdb          auto:sliced                  -            online
sdc          auto:sliced     NEWDISK      PDDG         online
```

The use of the backslash is necessary to override the significance of "!" to the bash Shell which is the default shell for root user. Without the "\", the bash (or C Shell) command line interpreter would look for some history of command event.

Post-mirroring steps

The last step is to check on application services by running whatever utilities you have to ensure the application is up. At some point, a reboot should be done at this point to ensure that the system properly starts and can access the disks during a reboot. No additional modifications need to be made to the file system mount table (etc/fstab, for example) since all storage, disk group, and volume object names remain unchanged.

Chapter 20

Migrating data between platforms

This chapter includes the following topics:

- Overview of the Cross-Platform Data Sharing (CDS) feature
- CDS disk format and disk groups
- Setting up your system to use Cross-platform Data Sharing (CDS)
- Maintaining your system
- File system considerations
- Alignment value and block size
- Migrating a snapshot volume

Overview of the Cross-Platform Data Sharing (CDS) feature

This section presents an overview of the Cross-Platform Data Sharing (CDS) feature of Veritas InfoScale Storage Foundation. 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 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.

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 format and disk groups

This section provides additional information about CDS disk format and CDS disk groups.

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
- 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 232.

CDS disk types

The CDS disk format, `cdsdisk`, is recognized by all VxVM platforms. 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 235.

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

Private and public regions

A VxVM 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) 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.

About Cross-platform Data Sharing (CDS) disk groups

A Cross-platform Data Sharing (CDS) disk group allows cross-platform data sharing of Veritas Volume Manager (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 (VxVM disks with the disk format `cdsdisk`), and is only available for disk group version 110 and greater.

Starting with disk group version 160, CDS disk groups can support disks of greater than 1 TB.

Note: The CDS conversion utility, `vxcdsconvert`, is provided to convert non-CDS VxVM 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 232.

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 226.

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 are 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 243.

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 247.

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 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 240.

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 240.

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, an instant snap DCO volume has the following region requirements:

- Minimum region size of 16K
- Incremental region size of 8K

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 to use Cross-platform Data Sharing (CDS)

In order to migrate data between platforms using Cross-platform Data Sharing (CDS), set up your system to use CDS disks and CDS disk groups. The CDS license must be enabled. You can use the default files to configure the appropriate settings for CDS disks and disk groups.

Table 20-1 describes the tasks for setting up your system to use CDS.

Table 20-1 Setting up CDS disks and CDS disk groups

Task	Procedures
Create the CDS disks.	You can create a CDS disk in one of the following ways: ■ Creating CDS disks from uninitialized disks See "Creating CDS disks from uninitialized disks" on page 229. ■ Creating CDS disks from initialized VxVM disks See "Creating CDS disks from initialized VxVM disks" on page 230. ■ Converting non-CDS disks to CDS disks See "Converting non-CDS disks to CDS disks" on page 232.

Table 20-1 Setting up CDS disks and CDS disk groups *(continued)*

Task	Procedures
Create the CDS disk groups.	You can create a CDS disk group in one of the following ways: ■ Creating CDS disk groups See "Creating CDS disk groups " on page 231. ■ Converting a non-CDS disk group to a CDS disk group See "Converting a non-CDS disk group to a CDS disk group" on page 233.
Verify the CDS license.	Verifying licensingSee "Verifying licensing" on page 235.
Verify the system defaults related to CDS.	Defaults files See "Defaults files" on page 235.

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 235.
  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 232.

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 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 be performed on-line (that is, while access to the disk group continues), stop any applications that are accessing the disks.
2. Make sure that the disks have free space of at least 256 sectors before doing the conversion.
3. Add a disk to the disk group for use by the conversion process. The conversion process evacuates objects from the disks, reinitializes the disks, and relocates objects back to the disks.

   Note: If the disk does not have sufficient free space, the conversion process will not be able to relocate objects back to the disk. In this case, you may need to add additional disks to the disk group.

4. 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.

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 online (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 relaying 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 online (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 offline.

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 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 attribute specifies a list of disks (`disk11` through `disk14`) to which subdisks can be evacuated to disks by setting the `evac_subdisks_ok` option to yes.

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

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

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 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 228.

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 Veritas InfoScale Operations Manager 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 Veritas InfoScale Operations Manager.

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

You may need to perform maintenance tasks on the CDS disks and CDS disk groups. Refer to the respective section for each type of task.

- Disk tasks
  See "Disk tasks" on page 238.
- Disk group tasks
  See "Disk group tasks" on page 240.
- Displaying information
  See "Displaying information" on page 246.
- Default activation mode of shared disk groups
  See "Default activation mode of shared disk groups" on page 249.
- Additional considerations when importing CDS disk groups
  See "Defaults files" on page 235.

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 235.

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.

Note: For disks larger than 1 TB, `cdsdisks` use the EFI layout. The procedure to restore disk labels does not apply to `cdsdisks` with EFI layout.

Note: The platform block is no longer written in the backup label. `vxdisk flush` cannot be used to restore the CDS disk label from 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

Note: Root Disk Encapsulation (RDE) is not supported on Linux from 7.3.1 onwards.

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 247.

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 233.

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 attribute 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 *Storage Foundation Administrator'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 *Storage Foundation 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 *Storage Foundation Administrator'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 *Storage Foundation Administrator'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 other than 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

To set up or migrate volumes with VxFS file systems with CDS, you must consider the file system requirements. This section describes these requirements. It also describes additional tasks required for migrating or setting up in CDS.

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 252.

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 252.

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 233.

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

   See the *Veritas InfoScale 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 258.

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 238.

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 258.

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 238.

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.

Note: If you replicate data from a little-endian to a big-endian system (or vice versa), you must convert the application after the replication completes.

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 9 ctime Thu Jun 1 16:16:53 2006
   ```

   Only file systems with disk layout Version 7 or later can be converted. If the file system has an earlier disk layout version, convert the file system to disk layout Version 7 or later 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_OS -e special_device
```

target_OS specifies the operating system to which you are migrating the file system.

See "Specifying the migration target" on page 252.

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.

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 -e -f recoveryfile -t target_specifiers special
```

For example:

```
# fscdsconv -e -f /tmp/fs_recov/recov.file -t Linux \
/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 21

Migrating from Oracle ASM to Veritas File System

This chapter includes the following topics:

- About the migration
- Pre-requisites for migration
- Preparing to migrate
- Migrating Oracle databases from Oracle ASM to VxFS

About the migration

Veritas InfoScale supports real-time migration of standalone and Oracle RAC databases hosted on Oracle ASM disks to VxFS file systems mounted on VxVM disks.

The migration requires a source system where the database is hosted on Oracle ASM disks and a target that serves as a standby during the migration. That target contains VxVM disks on which the Veritas File System is mounted. The target disks can be on the same host as the source database or on a different host.

The migration is performed by the script `asm2vxfs.pl`. The script creates the target database on the designated VxFS mount point and automates most of the necessary configuration tasks, such as preparing the source and target databases for migration, configuring the listener on the target and other configuration changes. You can migrate multiple instances of database at a time in a RAC environment.

Applications can continue to access the database while the migration is in progress. Once the source and target databases are synchronized, another script `switchover.pl` switches the role of the source database to standby and that of the target database to primary. All applications connected to the source database must be manually stopped before the transition begins. After the roles of the source and target databases are switched, the applications must be started manually. This is the only downtime incurred during the migration process.

The total migration time depends on the following factors:

- The amount of redo information (load) being generated on the source system
- The amount of system resources available for the new target database
- The size of the source database

You can run the migration script on the command line using a configuration file.

Figure 21-1 illustrates the migration process with the target storage on the same host as the source.

Figure 21-1 Migration with the target storage on the same host as the source

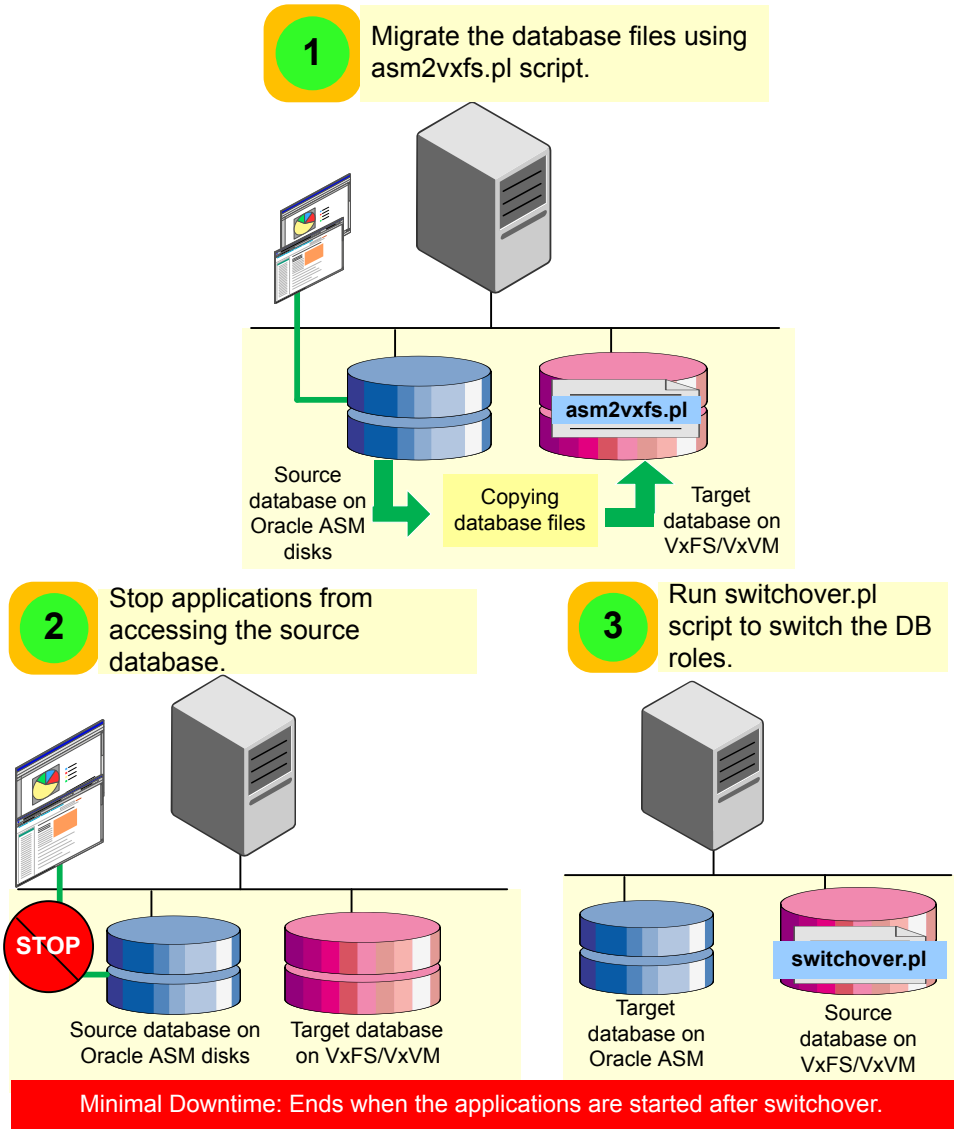
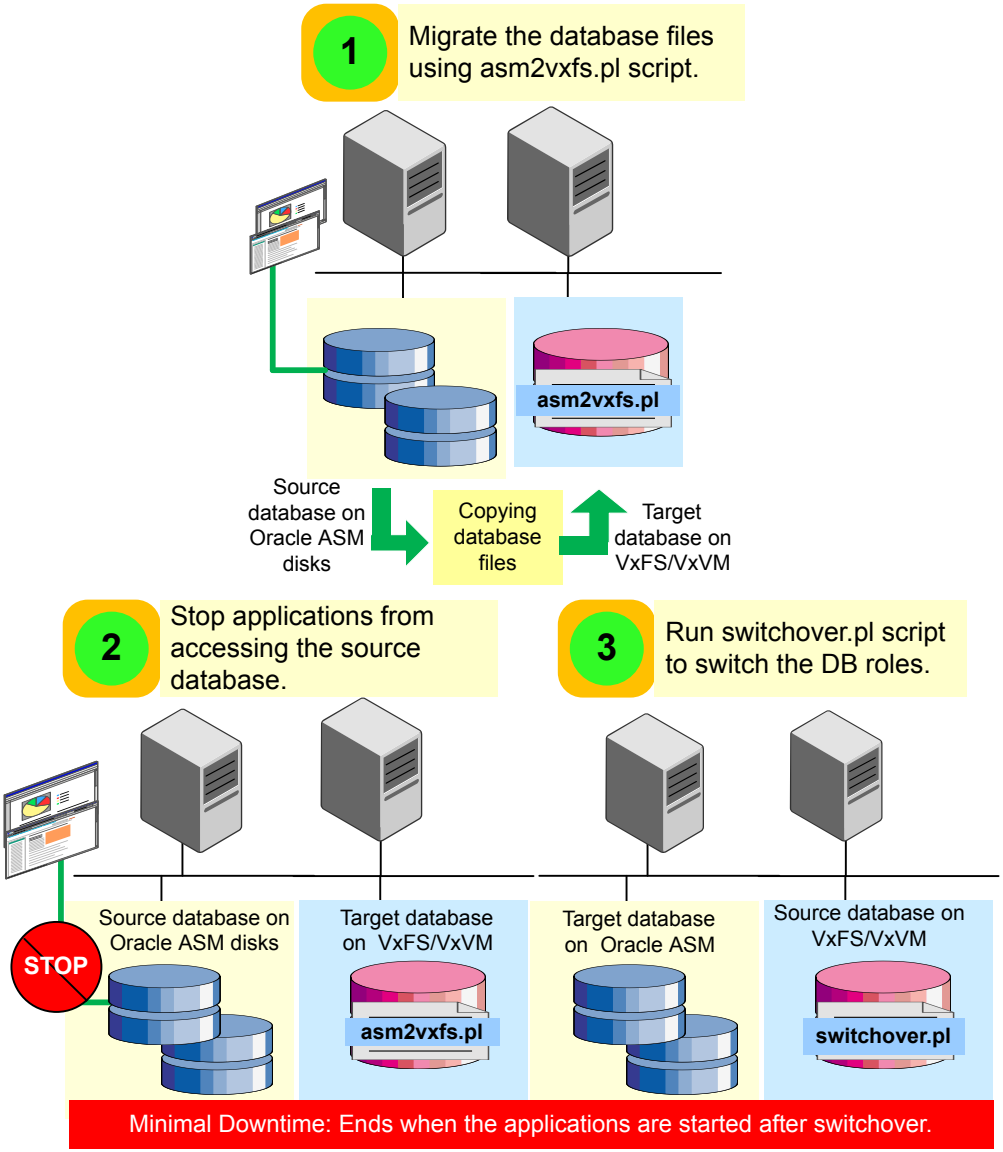


Figure 21-2 illustrates the migration process with the target storage on a different host.

Figure 21-2 Migration with the target storage on a host different from the source



Script and configuration file options

Table 21-1 lists the configuration file options that are used with the migration script.

Table 21-1 Configuration file options

Configuration file parameters (if you are using the configuration file)	Description
ORACLE_BASE	Path of the Oracle base directory on the target system.
PRIMARY	Name of the source database.
PRIMARY_INSTANCE	Any instance name of the source database.
STANDBY	Name of the target database.
STANDBY_INSTANCES	Instances of the target database. **Note:** Multiple database instances are specified as a comma-separated list.
PRIMARYHOST	Host name of the instance, specified in the parameter of PRIMARY_INSTANCE.
STANDBY_HOST	Host name of all the target instances. **Note:** Multiple host names are specified as a comma-separated list.
DATA_MNT	VxFS mount point.
RECOVERY_MNT	Destination path of the recovery files (on top of VxFS).
SYSTEM_PASSWORD	Password of the system user of the source database.
SYS_PASSWORD	Password of the SYS user of the source database.

Sample configuration file

Set the parameter values in the following format:

```
PARAMETER=value
```

A sample configuration file is as follows:

```
ORACLE_BASE=/oracle_base
PRIMARY=sourcedb
PRIMARY_INSTANCE=sourcedb1
STANDBY=std
STANDBY_INSTANCES=std,std2
```

```
PRIMARYHOST=example.com
STANDBY_HOSTS=example2.com
DATA_MNT=/data_mntpt
RECOVERY_MNT=/data_mntpt/recover_dest
SYSTEM_PASSWORD=system123
SYS_PASSWORD=sys123
```

Pre-requisites for migration

Ensure that you meet the following requirements before you migrate:

- Veritas InfoScale 7.4.1 is installed.
- The Enterprise edition of any supported Oracle version is installed.
- The `VRTSfssdk` RPM is manually installed.
- Run the `asm2vxfs.pl` script, which is located in the `/opt/VRTSfssdk/<Veritas InfoScale Version>/src/utils/asm2vxfs/` directory.
- The free space of the VxFS file system must be as large as the size of the source database.
- If the target is different from the source, the hardware architecture and the database version on the target must be the same as the source. The directory structure of ORACLE_HOME on source and target system must be same and if GRID is present, then the similar directory structure is applicable for GRID_HOME. The directory structure has same permission, ownership and the group membership.
- The size of the source database must not change until the migration is complete.
- The following configuration files `tnsnames.ora` and `listener.ora` must not be modified during the migration process.
  If the source and target systems are different, then the `listener.ora` file of the source database can be modified.
- The following parameters must not be modified during the migration:

  ```
  LOG_ARCHIVE_CONFIG, FAL_SERVER, FAL_CLIENT, LOG_ARCHIVE_DEST_N,
  STANDBY_FILE_MANAGEMENT
  ```

Preparing to migrate

Perform the following steps before you migrate the database.

To prepare the systems for migration

1. Enable `archivelog` mode on the source database.
   - Shut down the database instance if it is open.
   - Restart the instance with the mount option.

     ```
     SQL> startup mount
     ORACLE instance started.
     ```

   - Enable the `archivelog` mode.

     ```
     SQL> alter database archivelog;
     Database altered.
     ```

   - Open the database.

     ```
     SQL> alter database open;
     Database altered.
     ```

2. Create a VxFS file system on the volume and mount it.
   - For Mounting a VxFS file system. See*Storage Foundation Cluster File System High Availability Administrator's Guide*
   - Create a cluster-wide mount point for a RAC environment for administering CFS. See*Storage Foundation Cluster File System High Availability Administrator's Guide*

     Note: Ensure that the size of the file system is as large as the source database.

3. Set up passwordless SSH between the root users of the source and target systems.
4. Install the DBD::ORACLE perl module on the target systems, using the PERL binary which is present in the Oracle Home directory.

   For instructions, see the Perl documentation.

5 Set the following environment variables—ORACLE_HOME and LD_LIBRARY_PATH:

```
# export ORACLE_HOME=path of oracle_home
# export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME
```

6 Keep the following information handy:

- Oracle base directory path of the target system
- Name of the source database
- Any instance name of source database
- Name of the target database
- Name of all target database instances
- Name of the source database host
- Host name of all target instances
- VxFS mount point
- Destination path of the recovery files
- Password of system user and sys user for the source databases
- Absolute path of the configuration file.

Migrating Oracle databases from Oracle ASM to VxFS

You can perform the migration in the following way:

- Run the script using the configuration file

Perform the following steps to migrate Oracle database instances from Oracle ASM disks to VxFS.

To migrate Oracle databases from Oracle ASM to VxFS

1. Run the migration script `asm2vxfs.pl` on the target system using the configuration file:

   ```
   # asm2vxfs.pl -f config_file_name
   ```

   For example:

   ```
   # asm2vxfs.pl -f /root/config_file
   ```

   See the section called “Script and configuration file options” on page 268.

   See the section called “Sample configuration file” on page 269.

2. Stop all the applications connected to the source database.

3. Run the `switchover.pl` script on the target system using the configuration file. The script switches the role of the source database to standby and that of the target database to primary.

   Note: Downtime begins when the switch over operation starts.

   ```
   # switchover.pl -f config_file_name
   ```

   For example:

   ```
   # switchover.pl -f /root/config_file
   ```

4. After the `switchover.pl` script completes, you will be prompted to unlink the old primary database.

   ```
   Do you want to unlink the original primary? <y/n>
   ```

   Note: If you enter **y**, all connections to the old primary database are unlinked and reconfigured to use the new primary database.

5. Connect the applications with the new primary database on VxFS.

   Note: Downtime ends here.

Section 8

Just in time availability solution for vSphere

- Chapter 22. Just in time availability solution for vSphere

Chapter 22

Just in time availability solution for vSphere

This chapter includes the following topics:

- About Just In Time Availability
- Prerequisites
- Supported operating systems and configurations
- Setting up a plan
- Managing a plan
- Deleting a plan
- Viewing the properties
- Viewing the history tab
- Limitations of Just In Time Availability

About Just In Time Availability

The Just In Time Availability solution provides increased availability to the applications on a single node InfoScale Availability cluster in VMware virtual environments.

Using the Just In Time Availability solution, you can create plans for:

1. Planned Maintenance
2. Unplanned Recovery

Planned Maintenance

In the event of planned maintenance, the Just In Time Availability solution enables you to clone a virtual machine, bring it online, and failover the applications running on that virtual machine to the clone on the same ESX host. After the maintenance procedure is complete, you can failback the applications to original virtual machine. Besides failover and failback operations, you can delete a virtual machine clone, view the properties of the virtual machine and its clone, and so on.

Unplanned Recovery

When an application encounters an unexpected or unplanned failure on the original virtual machine on primary ESX, the Just In Time Availability solution enables you to recover the application and bring it online using the unplanned recovery feature.

With **Unplanned Recovery Policies**, the Just In Time Availability solution enables you to set up recovery policies as per your requirement to mitigate the unplanned failure that is encountered by an application. Just In Time Availability solution provides the following recovery policies for your selection. You may select one or all the recovery policies as per your need.

Unplanned Recovery Policies	Description
Restart Application	Just In Time Availability (JIT) solution attempts to restart the service group (SG), and bring the application online on the original virtual machine on primary ESX. Maximum three retry attempts are permitted under this policy. **Note:** If all the three attempts fail, application continues to remain in faulted state or continues with the next policy as selected while creating a plan.

Unplanned Recovery Policies	Description
Restart virtual machine (VM)	Just In Time Availability (JIT) solution performs the subsequent tasks such as bring the service group offline and shuts down the virtual machine; powers on the virtual machine; bring the service group online on the original virtual machine on primary ESX. You are provided with **Last attempt will be VM reset** option to reset the virtual machine. By default, this checkbox is selected and the default retry attempt value is one. If you retain the default settings, then VM reset operation is performed on the virtual machine at the first attempt itself. Maximum three retry attempts are permitted for this operation. If you deselect the checkbox, then the virtual machine reset (VM Reset) operation is not performed.
Restart VM on target ESX	Using this policy, you can recover the faulted application on the virtual machine. In this policy, the original virtual machine is unregistered from the primary ESX; registered on the target ESX; and the faulted application is brought online on the virtual machine. **Note:** While configuring **Restart VM on target ESX**policy, ensure that the ESX version of both the source and target is compatible with each other. The virtual machines on target ESX are registered with the same vmx file as on the source ESX.

Unplanned Recovery Policies	Description
Restore VM on target ESX	Using this policy, you can recover the faulted application on the virtual machine using a boot disk backup copy of the original virtual machine. In this policy, the original virtual machine is unregistered from the ESX and the boot disk backup copy of the original virtual machine is registered on target ESX. The faulted application is then brought online on the virtual machine. **Note:** While configuring **Restore VM on target ESX** policy, ensure that the ESX version of both the source and target is compatible with each other. The virtual machines on target ESX are registered with the same vmx file as on the source ESX.
Unplanned Failback	The **Unplanned Failback** operation lets you failback the application from the boot disk back up copy of virtual machine on the target ESX to the original virtual machine on primary ESX. If you have selected either **Restart VM on target ESX** or **Restore VM on target ESX** or both the recovery policies, you can perform the **Unplanned Failback** operation. On the **Plans** tab, in the plans table list, right-click the virtual machine and click **Unplanned Failback**. **Note: Unplanned Failback operation** operation is disabled and not available for the plans and the virtual machines which have **Restart Application** and **Restart VM** policies as the only selected options.

Based on the selected recovery policy for a plan, Just In Time Availability (JIT) solution performs the necessary operations in the sequential order.

For example, if you have selected **Restart Application** and **Restart VM** as the recovery policy, then in the event of unplanned application failure, first it performs tasks for **Restart Application** policy and if that fails, it moves to the next policy.

You may select one or all the recovery policies based on your requirement.

Table 22-1 lists the sequence of tasks that are performed for each Unplanned Recovery policy.

Table 22-1 Tasks performed for each Unplanned Recovery policy

Unplanned Recovery Policy	Tasks Performed
Restart Application	◆ Makes an attempt to restart the application.
Restart virtual machine (VM)	**1** Brings the service group(s) offline **2** Shuts down the virtual machine **3** Power on the virtual machine **4** Brings the service group(s) online
Restart VM on target ESX	**1** Brings the service group(s) offline **2** Shuts down the original virtual machine **3** Detaches the data disks from the original virtual machine **4** Unregisters the virtual machine from the primary ESX **5** Registers the original virtual machine on target ESX **6** Attaches the data disks back to the virtual machine **7** Power on the virtual machine **8** Brings the service group(s) online

Table 22-1 Tasks performed for each Unplanned Recovery policy *(continued)*

Unplanned Recovery Policy	Tasks Performed
Restore VM on target ESX	**1** Brings the service group(s) offline **2** Shuts down the virtual machine **3** Detaches the data disks from the virtual machine **4** Unregisters the original virtual machine from the ESX **5** Registers the boot disk backup copy of the original virtual machine to the target ESX **6** Attaches the data disks back to the virtual machine **7** Power on the virtual machine **8** Brings the service group(s) online
Unplanned Failback	**1** Brings the service group(s) offline **2** Shuts down the virtual machine **3** Detaches the data disks from the virtual machine **4** Unregisters the virtual machine from the target ESX **5** Registers the virtual machine using the original boot disk to the primary ESX **6** Attaches the data disks to the virtual machine **7** Power on the virtual machine on primary ESX **8** Brings the service group(s) online on the virtual machine

Scheduler Settings

While creating a plan for unplanned recovery, with **Scheduler Settings**, you can set up a schedule for taking a back up of boot disk of all the virtual machines that are a part of the plan.

To use the Just In Time Availability solution, go to **vSphere Web Client > Home view > Veritas AppProtect**.

See “Setting up a plan” on page 286.

Getting started with Just In Time Availability

You can access the Just In Time Availability solution from the **vSphere Web Client > Veritas AppProtect** interface.

The **Veritas AppProtect** is registered with Veritas InfoScale Operations Manager (VIOM), and is accessed from the **vSphere Web Client >Home** view.

Table 22-2 describes the Veritas AppProtect interface in detail.

Figure 22-1 Elements of the Veritas AppProtect interface

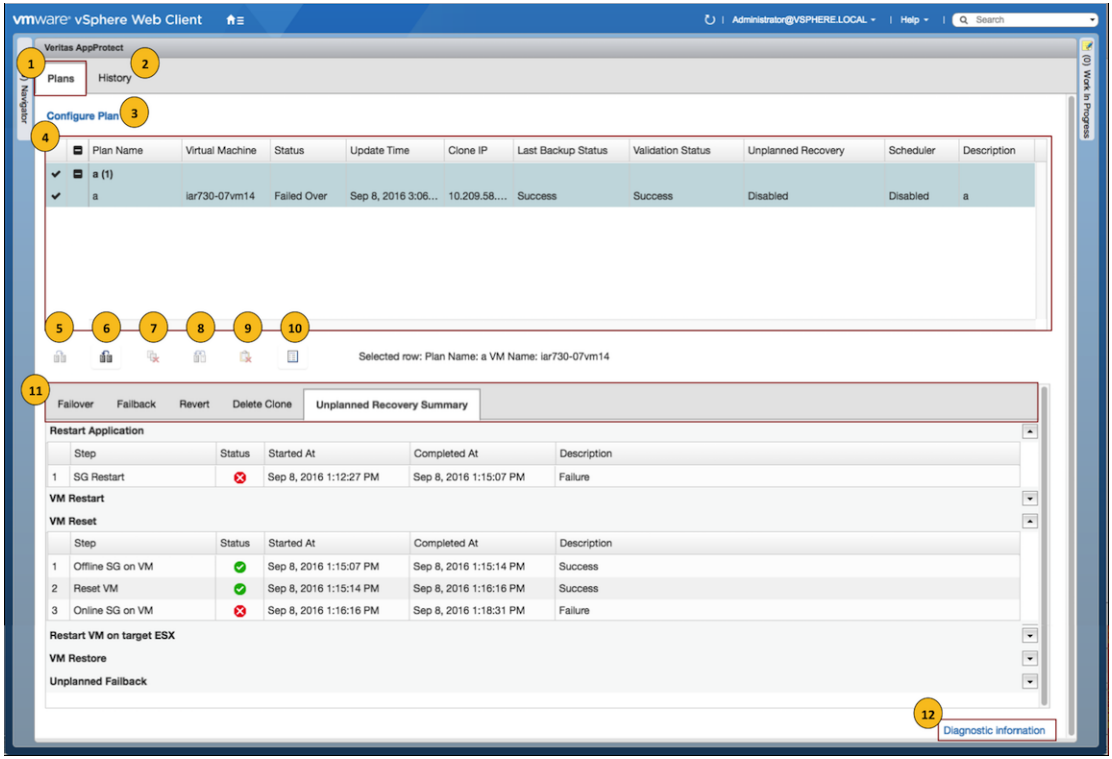


Table 22-2 Elements of the Veritas AppProtect interface and the description

Label	Element	Description
1	**Plans** tab	Enables setting up a plan for a planned failover and unplanned recovery. Displays the plan attributes, and the virtual machines that are added to the plan. Displays the status of virtual machines for unplanned recovery and schedule for virtual machine back up operation based on the criteria set while configuring or editing the plan. Shows the enabled or disabled failover, failback, delete clone, revert, delete plan, and properties operations icons based on the state of the selected plan for planned failover.
2	**History** tab	Displays the status and the start and the end time of the specific operation performed on the created plans.
3	**Configure Plan** link	Opens the **Plan Configuration** wizard.
4	**Plans** table	Displays the attributes of the plan.
5	**Failover** icon	Fails over the applications from the original virtual machine to the clone.
6	**Failback** icon	Fails back the applications from the clone to the original virtual machine.
7	**Delete Clone** icon	Deletes the cloned virtual machine.

Table 22-2 Elements of the Veritas AppProtect interface and the description *(continued)*

Label	Element	Description
8	**Revert State** icon	Reverts the failed operation, restores the applications to the original virtual machines, and delete the clone virtual machines.
9	**Delete Plan** icon	Deletes the plan.
10	**Properties** icon	Displays the attributes of each virtual machine and the clone.
11	Operation-specific tabs	Displays the sequence of the tasks that are performed for the selected operation. Based on the operation that is executed, the associate tab opens. **For Planned Maintenance** **1** Failover **2** Failback **3** Revert **4** Delete Clone **For Unplanned Recovery** ◆ Unplanned Recovery Summary
12	**Diagnostic information**	Displays the logs that are reported for the Veritas AppProtect interface.

See "Plan states" on page 299.

Prerequisites

Before getting started with Just In Time Availability, ensure that the following prerequisites are met.

- The Just In Time (JIT) solution feature cannot co-exist with VMware HA, VMware FT, and VMware DRS. This pre-requisite is applicable for **Unplanned Recovery** only.
- VIOM version 7.2 or later must be installed, and configured using fully qualified domain name (FQDN) or IP.
- Make sure that you have the admin privileges for vCenter.
- VMware Tools must be installed and running on the guest virtual machine.
- VIOM Control Host add-on must be installed on VIOM server or machine.
- The virtual machines must be added in VIOM. The virtual machines, vSphere ESX servers, and VIOM must have the same Network Time Protocol (NTP) server configured.
- Make sure to specify VIOM Central Server FQDN or IP in the SNMP Settings of the vCenter Server.
- vCenter Server and VIOM must be configured using the same FQDN or IP address. Make sure that if FQDN is used to configure vCenter in VIOM Server that is used during the configuration.
- If raw disk mapping (RDM) disks are added to the virtual machine, then make sure that the virtual machine is in the physical compatibility mode. Veritas AppProtect does not support the virtual compatibility mode for RDM disks.
- For Microsoft Windows operating system, make sure that you have the Microsoft Windows product license key. The key is required to run the Sysprep utility, which enables customization of the Windows operating system for a clone operation.
- For SUSE12 , RHEL7 and supported RHEL-compatible distributions, install the deployPkg plug-in file on the virtual machine.
  For more information on installing the plug-in, see
  https://kb.vmware.com/kb/2075048
- Make sure that the InfoScale Availability service group is configured with one of the Storage Agents such as Mount, DiskGroup, LVMVolumeGroup, VMNSDg (for windows), DiskRes agent (for windows) for the data disks. This configuration enables Veritas AppProtect to discover data disks for the applications. Also, ensure that the service group is online to determine data disk mapping.
- Virtual Machines which have snapshots associated with them are not supported.
- Virtual Machines with SCSI Bus Sharing are not supported.
- Make sure that the SNMP Traps are configured for the following from vCenter server to VIOM:

  - Registered virtual machine
  - Reconfigured virtual machine
  - Virtual machine which is getting cloned
- Make sure that the boot disk of VM's(vmdk) does not have spaces
- For HA console add on upgrade from VIOM 7.1 to later versions of VIOM, see the *Veritas InfoScale Operations Manager 7.4.1 Add-ons User's Guide* for more details.
- Make sure to set the vSphere DRS Automation Level to manual, if you want to configure **Restart VM on target ESX** or **Restore VM on target ESX** policies for your plan.
- Ensure to update or edit the plan, when a virtual machine is migrated or if there are any modifications made to the settings of the virtual machines which are configured for that plan.
- Ensure to increase the tolerance limit of disk agent resource to two, if you want to create a plan for unplanned recovery with **Restore VM on target ESX** as the unplanned recovery policy.

Note: This prerequisite is applicable for windows operating system.

Supported operating systems and configurations

Just In Time Availability supports the following operating systems:

- On Windows: Windows 2012, and Windows 2012 R2.
- On Linux: RHEL5.5, RHEL6, RHEL7 and supported RHEL-compatible distributions, SUSE11, SUSE12.

Just In Time Availability supports the following configurations:

- Veritas Cluster Server (VCS) 6.0 or later, or InfoScale Availability 7.1 and later.
- Veritas InfoScale Operations Manager managed host (VRTSsfmh) 7.1, 7.2 or later versions on the virtual machines.
  For more information about VRTSsfmh, see the *Veritas InfoScale Operations Manager 7.2 User Guide*.
- Veritas InfoScale Operations Manager (VIOM) 7.2 or later as a central or managed server.
- VMware vSphere 5.5 Update 2, Update 3, or 6.0 and 6.0 Update 1 version.

Setting up a plan

Plan is a template which involves a logical grouping of virtual machines so as to increase the availability of the application in the event of a planned failover and recovery of the application in the event of an unexpected application failure.

To set up a plan

1. Launch Veritas AppProtect from the **VMware vSphere Web Client > Home view > Veritas AppProtect** icon.
2. Click **Configure Plan**.

   The **Plan Configuration** wizard appears.
3. Specify a unique **Plan Name** and **Description**, and then click **Next**.

   The wizard validates the system details to ensure that all prerequisite requirements are met.
4. Select the virtual machines that you want to include in the plan, review the host and operating system details, and then click **Next**.

   The **Unplanned Recovery Settings** page appears.
5. On the **Unplanned Recovery Settings** page, you can configure the selected virtual machines for **Unplanned Recovery** as well.

   Deselect the **Configure selected VMs for Unplanned Recovery as well** check box, if you do not want to include the selected virtual machines for unplanned recovery.

   If you have selected the virtual machines for unplanned recovery, then set up the unplanned recovery policies as appropriate from the available options. You can set up policies to restart applications, restart virtual machines, restart virtual machine on target ESX, and restore a virtual machine on target ESX.

   If you have selected **Restore VM on target ESX** as the unplanned recovery policy, then you can set up a schedule to create a boot disk back up copy of the virtual machine within the configured plan. You can set the frequency as daily, weekly, monthly, or manual as per your requirement.

   After you have finished making necessary settings for Unplanned Recovery, Click **Next**.
6. The wizard validates the prerequisite attributes of the virtual machine and the ESX host, and adds the qualified virtual machines to the plan.

   Click **Next** after the validation process completes.

7 In the **Disks** tab, you can view the selected application data disks. Just In Time Availability solution uses the selected data disks to perform detach-attach operation during a planned failover and unplanned recovery.

Note: If the disks are not auto-marked as selected to perform detach-attach operation, then first refresh the VIOM server and then the VCentre server in VIOM and then create a plan.

---

8 In the **Network Configuration** tab, specify the network interface configuration details for the cloned virtual machine. Make sure to specify at least one public interface and valid IP details.

9 In the **Unplanned Recovery Target** tab, specify the target ESX server to restore the virtual machine, and the target ESX port details.

Note: The **Unplanned Recovery Target** tab is visible only when **Restart VM on target ESX** or **Restore VM on target ESX** is selected.

---

10 In the **Windows Settings** tab, specify the domain name, Microsoft Windows product license key, domain user name, domain password, admin password, and time zone index.

Note: The **Windows Settings** tab is visible only when a Windows virtual machine is selected in the plan.

---

11 Click **Next**. The **Summary** wizard appears.

12 In the **Summary** wizard, review the plan details such as the plan name, unplanned recovery policies, schedule, and so on.

Deselect the **Start backup process on finish** checkbox, if you do not want to initiate a backup process when the plan creation procedure is finished. By default, this checkbox is selected.

Click **Create**. The plan is created and saved.

13 Click **Finish** to return to the plans tab and view the created plans.

See “Managing a plan” on page 288.

See “Deleting a plan” on page 290.

Managing a plan

Planned Maintenance

After the maintenance plan is created, you can failover the applications to the clone virtual machine and failback the applications from the clone to the virtual machine. When the scheduled maintenance is complete, you can delete the cloned virtual machine or retain it for future use.

To perform failover, failback, revert, or delete clone operations, go to **Plans**, and select a plan. Based on the enabled operation, perform the following tasks:

To failover the applications to the cloned virtual machine

- Click the **Failover** icon.

  Just In Time Availability (JIT) performs the sequence of failover tasks, which includes taking the application offline, detaching the disks, cloning the virtual machine, attaching the disks, and so on.

To failback the applications from the clone to the primary virtual machine

- Click the **Failback** icon.

  Just In Time Availability (JIT) performs the sequence of failback tasks, which includes taking the application offline, detaching the disks, attaching the disks, and so on.

To revert a failover or a failback operation

- Click the **Revert** icon.

  If the failover or a failback operation fails, the revert operation restores the applications on the virtual machine, and deletes the clone if created.

To delete a clone

- Click the **Delete Clone** icon.

  After the failback operation is complete, you can delete the clone. By default, the revert operation deletes the clone.

Note: Alternatively, right-click **Plan** in the **Plans** table on the **Plans** wizard to perform failover, failback, revert, delete plan, and delete clone operations.

Unplanned Recovery

Once you have set up a plan for unplanned recovery during **Configure Plan** operation, based on the recovery policies selected for the plan, the application is recovered accordingly.

You can manage unplanned recovery policies settings by performing the following operations on the plan and its associated virtual machines.

Managing unplanned recovery settings

On the **Plans** tab, in the plans table which lists all the existing plans, navigate to the required plan and use the right-click option on the selected plan.

- **Edit**: Use this option to modify the configured plans settings such as adding or removing a virtual machine from the plan, and so on.
  The same **Configuration Plan** wizard using which you had set up or configured a plan is displayed with pre-populated details.
  See "Setting up a plan" on page 286.
- **Disable Unplanned Recovery**: Use this option to disable the Unplanned Recovery settings.
- **Enable Unplanned Recovery**: Use this option to enable the Unplanned Recovery settings.
- **Disable Scheduler**: Use this option to disable the scheduler settings.
- **Enable Scheduler**: Use this option to enable the scheduler settings.
- **Delete Plan**: Use this option to delete the created plan.
- **Properties**: Use this option to view the properties for unplanned recovery. It displays details such as the selected unplanned recovery policies and the associated operations for the selected policies. It also provides information about the selected scheduler mode for performing boot disk back up operation for the selected virtual machines.

Managing virtual machines settings

On the **Plans** tab, in the plans table which lists all the existing plans and its associated virtual machines, navigate to the required virtual machine. Select the required virtual machine and use the right-click option on the selected virtual machine.

- **Remove VM From Plan**: Use this option to delete the virtual machine from the selected plan.
- **Create Clone Backup**: Use this option to create a boot disk back up copy of the virtual machine.
- **Unplanned Failback**: Use this option to failback the application from the boot disk back up copy of the virtual machine on target ESX to the original virtual machine on primary ESX.

Note: This option is available only if you have set unplanned recovery policies as **Restart VM on target ESX** or **Restore VM on target ESX**.

- **Properties**: Use this option to view properties such as the last run time for backup operation, last successful backup attempt time and the target ESX details.

See "Plan states" on page 299.

Deleting a plan

After you have finished performing failback operations from the clone to the primary virtual machine in case of planned maintenance and recovery operations in case of unplanned recovery, you may want to delete the plan.

To delete a plan

1. Launch **Veritas AppProtect** from the VMware vSphere Web Client Home view.
2. In the **Plans** tab, select the plan that you want to delete.
3. Click **Delete Plan**.

Note: The **Delete plan** icon is enabled only when the selected plan is in **Ready For Failover**, **Failed to Revert**, or **Unplanned Failed to Failback** state.

Viewing the properties

Virtual Machine Properties

The **Virtual Machine Properties** window displays information about the virtual machine and its clone such as name, operating system, cluster name, service groups, DNS server, domain, IP addresses, and data disks.

To view the properties

1. On the **Plans** tab, select the virtual machine.
2. Click the **Properties** icon or right-click the virtual machine.

   The **Virtual Machine Properties** window opens and displays the attributes of the virtual machine and its clone.

Plan Properties

The **Plan Properties** window displays information about the unplanned recovery policies selected; scheduler mode set; and the time when the last backup operation was run and was successful for a virtual machine.

To view properties for the plan

1. In the Plan Name table, select the plan.
2. Right-click the selected plan. A window with a list of options is displayed.
3. Click **Properties**

   The **Plan Properties** window opens and displays the unplanned recovery policies selected and the schedule mode for virtual machine backup operation.

Viewing the history tab

On the **History** tab, you can view the detailed summary of the operations that are performed on the virtual machine. The details include the plan name, virtual machine name, operation, the status of the operation, the start and the end time of the operation, and the description of the operation status.

To view the summary

1. Launch **Veritas AppProtect** from the VMware vSphere Web Client Home view.
2. Click the **History** tab.

Limitations of Just In Time Availability

The following limitations are applicable to Just In Time Availability.

- On a single ESX host only ten concurrent failover operations are supported. Across ESX hosts, twenty concurrent failover operations are supported.
- Linked mode vCenter is not supported.
- Only three backup operations per data store are active , the rest will be queued. Only five backup operations per ESX host are active, the rest will be queued.

See “Supported operating systems and configurations” on page 285.

Section 9

Veritas InfoScale 4K sector device support solution

- Chapter 23. Veritas InfoScale 4k sector device support solution

Chapter 23

Veritas InfoScale 4k sector device support solution

This chapter includes the following topics:

- About 4K sector size technology
- Veritas InfoScale unsupported configurations
- Migrating VxFS file system from 512-bytes sector size devices to 4K sector size devices

About 4K sector size technology

Over the years, the data that is stored on the storage devices such as the hard disk drives (HDD) and Solid State Devices (SSD) has been formatted into a small logical block which is referred to as **Sector**. Despite of increase in storage densities over a period of time, the storage device sector size has remained consistent - 512 bytes. But, this device sector size proves to be inefficient for Solid State Devices (SSD).

Benefits of transition from 512 bytes to 4096 bytes or 4K sector

The 4K sector disks are the first advanced generation format devices. They help with the optimum use of the storage surface area by reducing the amount of space that is allocated for headers and error correction code for sectors. They are considered to be more efficient for larger files as compared to smaller files.

The advanced format devices with 4K sector size are considered to be beneficial over 512-bytes sector size for following reasons:

1. Improves the format efficiency
2. Provides a more robust error correction

Considering the benefits, many storage device manufacturers such as Hitachi, NEC, Fujitsu have started shipping 4K sector devices.

However, many aspects of modern computing still assume that the sectors are always 512-bytes. The alternative is to implement 4K sector transition that is combined with the 512-bytes sector emulation method. The disadvantage of 512-bytes sector emulation method is that it reduces the efficiency of the device.

Veritas InfoScale uses the Veritas Volume Manager and Veritas File System storage components to provide a solution that supports 4K sector devices (formatted with 4KB) in storage environment. Earlier, you were required to format 4K devices with 512-bytes. You can now directly use the 4K sector devices with Veritas InfoScale without any additional formatting.

Supported operating systems

You can use 4k sector devices with Veritas InfoScale 7.2 or later versions only on Linux (RHEL, SLES, and supported RHEL compatible distributions) and Solaris 11 operating systems.

See "Veritas InfoScale unsupported configurations" on page 294.

See "Migrating VxFS file system from 512-bytes sector size devices to 4K sector size devices" on page 295.

Veritas InfoScale unsupported configurations

This section lists the various Veritas InfoScale features that are not supported with 4K sector devices.

- **Volume Layout**: RAID-5 is not supported. All other volume layouts are supported
- **VxVM Disk Group support**: Only cross Platform Data Sharing (CDS) disk group format is supported. A disk group with a combination or a mix of 512-byte sector disks and 4K sector disks is not supported. Two different disk groups, one with 4K disks and other with 512-byte disks can co-exist
- **VxVM SmartIO configuration support**: If the sector size of the disk which hosts the application volume and the disk which hosts the cache differ, then caching is not enabled on that application volume.
- Storage area network (SAN) boot
- Root disk encapsulation
- Snapshot across disk groups with different sector size disks
- Volume level replication such as Veritas Volume Replicator(VVR)

- **VxFS File System support**: The file system **block size** and **logiosize** less than 4 KB is not supported on a 4K sector device

Migrating VxFS file system from 512-bytes sector size devices to 4K sector size devices

This section describes the procedure to migrate VxFS file system from 512 bytes to 4K sector size devices.

VxFS file systems on the existing 512-bytes sector devices might have been created with a file system block size of 1 KB or 2 KB, which is not supported on a 4K sector device. Hence, the traditional storage migration solutions, such as array level or volume level migration or replication may not work properly.

Starting With Veritas InfoScale 7.2 and later versions, you can migrate VxFS file system from 512-bytes sector size devices to 4K sector size devices using the standard file copy mechanism.

Note: The standard file copy mechanism may not preserve certain file level attributes and allocation geometry.

Note: Migration of VxFS file system from 512-bytes sector size to 4K sector size is supported only on Linux (RHEL, SLES, and supported RHEL compatible distributions) and Solaris 11 operating systems.

To migrate VxFS file system from 512-bytes sector size devices to 4K sector size devices:

1. Mount 512 bytes and 4K VxFS file system

```
# mount -t vxfs /dev/vx/dsk/diskgroup/volume_512B /mnt1

# mount -t vxfs /dev/vx/dsk/diskgroup/volume_4K /mnt2
```

2. Copy all the files from /mnt1 to /mnt2 manually

```
# cp -r /mnt1 /mnt2
```

3. Unmount both the VxFS file system - 512 bytes and 4K

```
# umount  /mnt1

# umount  /mnt2
```

See “About 4K sector size technology” on page 293.

See “Veritas InfoScale unsupported configurations” on page 294.

Section 10

Reference

- Appendix A. Veritas AppProtect logs and operation states
- Appendix B. Troubleshooting Veritas AppProtect

Appendix A

Veritas AppProtect logs and operation states

This appendix includes the following topics:

- Log files
- Plan states

Log files

The following log files are helpful for resolving the issues that you may encounter while using Veritas AppProtect:

- Console related logs:

  `/var/opt/VRTSsfmcs/logs/*`

  These log files show console messages and are useful for debugging console issues.

- Operations logs:

  `/var/opt/VRTSsfmh/logs/vm_operations.log`

  This log file shows the messages pertinent to the Veritas AppProtect interface.

- VMware vSphere 6.0 logs:

  `C:\ProgramData\VMware\vCenterServer\logs\vsphere-client\logs\*`

  These log files show the messages that are reported for the VMware vSphere Web Client version 6.0.

- VMware vSphere 5.5 U2 and U3 logs:

```
C:\ProgramData\VMware\vSphere Web Client\serviceability\logs\*
```

These log files show the messages that are reported for the VMware vSphere Web Client version 5.5 U2 and U3.

- Veritas AppProtect interface logs:
  The log file shows the logs that are reported for the Veritas AppProtect interface. To view the log files, on the **Planned Maintenance** tab or the **History** tab **> Diagnostic Information**.

Plan states

Based on the state of the plan, the operation icons are enabled and disabled on the **Plans** tab.

Table A-1 List of plan and operation states

Plan state	Failover	Failback	Revert	Delete clone	Delete Plan	Unplanned Failback	Create Clone backup	Properties
Ready For Failover	✓	–	–	✓ **Note:** Enabled when the selected maintenance plan has an associate clone.	✓ **Note:** Enabled when the selected maintenance plan does not have an associate clone.	–	✓	✓
Failed Over	–	✓	–	–	–	–	–	✓
Failed To Failover	–	–	✓	–	–	–	–	✓
Failed To Failback	–	–	✓	–	–	–	–	✓
Failed To Revert	–	–	✓	–	✓	–	–	✓
Unknown	–	–	✓	–	–	✓	–	✓
Failed To Delete Clone	–	–	–	✓	–	–	–	✓

Table A-1 List of plan and operation states *(continued)*

Plan state	Failover	Failback	Revert	Delete clone	Delete Plan	Unplanned Failback	Create Clone backup	Properties
Failover In Progress	–	–	–	–	–	–	–	✓
Failback In Progress	–	–	–	–	–	–	–	✓
Revert In Progress	–	–	–	–	–	–	–	✓
Delete Clone In Progress	–	–	–	–	–	–	–	✓
Application Faulted	–	–	–	–	–	–	–	✓
Failed To Restart VM	–	–	–	–	–	–	–	✓
Failed To Move VM	–	–	–	–	–	✓	–	✓
Failed To Restore VM	–	–	–	–	–	✓	–	✓
Unplanned	–	–	–	–	–	✓	✓	–
Unplanned Restored VM	–	–	–	–	–	✓	–	✓
Unplanned Failed to Failback	–	–	–	–	✓	–	–	–

Appendix B

Troubleshooting Veritas AppProtect

This appendix includes the following topics:

- Troubleshooting Just In Time Availability

Troubleshooting Just In Time Availability

Table B-1 lists the issues and the recommended solutions.

Table B-1 Issues and the corresponding resolutions

Issue	Recommended Solution
When setting up a maintenance plan, the registered virtual machine is not listed on the wizard.	To troubleshoot the issue, make sure the following: ■ ESX host on which the virtual machine resides, is connected to the vCenter. ■ The virtual machine is added as a managed host to Management Server. ■ On the virtual machine, at least one application is configured for monitoring, along with VCS. ■ The virtual machine is registered in VIOM. ■ VCS is configured on the virtual machine. ■ The virtual machine does not contain RHEL7 and SUSE 12, which are not supported. **Note:** Windows 2012R2 and 2008R2 are supported. ■ VCS is configured with the service groups.
When setting up a maintenance plan, the listed virtual machine is not available for selection.	To troubleshoot the issue, make sure the following: ■ The virtual machine is not configured for Global Cluster option (GCO). ■ Agents that support SAN are configured.
When Veritas AppProtect executes an operation, the timeout message is reported.	To troubleshoot the issue, perform the following: ■ If the failover or the failback operation fails, then click **Planned Maintenance > Revert** icon. Retry the operation. ■ If the delete plan or the delete clone operation fails, then retry the operation.
The revert operation failed.	Manually revert the virtual machine to its original state.