SpinFS® 2.5
Administration Guide

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

Part number 215-00837-B0
November 2004

Copyright and trademark information

Copyright information

Copyright © 1994–2004 Network Appliance, Inc. All rights reserved. Printed in the U.S.A.

Portions © 1998–2001 The OpenSSL Project. All rights reserved.

No part of this book covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Portions of this product are derived from the Berkeley Net2 release and the 4.4-Lite-2 release, which are copyrighted and publicly distributed by The Regents of the University of California.

Copyright © 1980–1995 The Regents of the University of California. All rights reserved.

Portions of this product are derived from NetBSD, which is copyrighted by Carnegie Mellon University.

Copyright © 1994, 1995 Carnegie Mellon University. All rights reserved. Author Chris G. Demetriou.

Permission to use, copy, modify, and distribute this software and its documentation is hereby granted, provided that both the copyright notice and its permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof, and that both notices appear in supporting documentation.

CARNEGIE MELLON ALLOWS FREE USE OF THIS SOFTWARE IN ITS “AS IS” CONDITION. CARNEGIE MELLON DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.

Software derived from copyrighted material of The Regents of the University of California and Carnegie Mellon University is subject to the following license and disclaimer:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notices, this list of conditions, and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment:

   This product includes software developed by the University of California, Berkeley and its contributors.
4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions of the software were created by Netscape Communications Corp.

The contents of those portions are subject to the Netscape Public License Version 1.0 (the “License”); you may not use those portions except in compliance with the License. You may obtain a copy of the License at http://www.mozilla.org/NPL/.

Software distributed under the License is distributed on an “AS IS” basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is Mozilla Communicator client code, released March 31, 1998.

The Initial Developer of the Original Code is Netscape Communications Corp. Portions created by Netscape are Copyright © 1998 Netscape Communications Corp. All rights reserved.

This software contains materials from third parties licensed to Network Appliance Inc. which is sublicensed, and not sold, and title to such material is not passed to the end user. All rights reserved by the licensors. You shall not sublicense or permit timesharing, rental, facility management or service bureau usage of the Software.

Portions developed by the Apache Software Foundation (http://www.apache.org/). Copyright © 1999 The Apache Software Foundation.

Portions Copyright © 1995–1998, Jean-loup Gailly and Mark Adler

Portions Copyright © 2001, Sitraka Inc.

Portions Copyright © 2001, iAnywhere Solutions

Portions Copyright © 2001, i-net software GmbH

Portions of this product are derived from version 2.4.11 of the libxml2 library, which is copyrighted by the World Wide Web Consortium.

Network Appliance modified the libxml2 software on December 6, 2001, to enable it to compile cleanly on Windows, Solaris, and Linux. The changes have been sent to the maintainers of libxml2. The unmodified libxml2 software can be downloaded from http://www.xmlsoft.org/.

Copyright © 1994–2002 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. http://www.w3.org/Consortium/Legal/

Software derived from copyrighted material of the World Wide Web Consortium is subject to the following license and disclaimer:

Permission to use, copy, modify, and distribute this software and its documentation, with or without modification, for any purpose and without fee or royalty is hereby granted, provided that you include the following on ALL copies of the software and documentation or portions thereof, including modifications, that you make:

1. The full text of this NOTICE in a location viewable to users of the redistributed or derivative work.
2. Any pre-existing intellectual property disclaimers, notices, or terms and conditions. If none exist, a short notice of the following form (hypertext is preferred, text is permitted) should be used within the body of any redistributed or derivative code: "Copyright © [$date-of-software] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. http://www.w3.org/Consortium/Legal/.
3. Notice of any changes or modifications to the W3C files, including the date changes were made.

THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR DOCUMENTATION.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to the software without specific, written prior permission. Title to copyright in this software and any associated documentation will at all times remain with copyright holders.

This product includes Linux Kernel software that was developed and modified under the terms of the GNU General Public License, version 2. For a complete machine-readable copy of the Linux software and Linux Kernel software distributed with this product, please send an email request to: support@netapp.com

This product includes the Java FTP Client Library that was developed under the terms of the GNU Lesser General Public License, version 2.1.

This product includes CIMOM software that was developed and modified under the terms of the Storage Networking Industry Association Public License, version 1.1. These software modifications are available via anonymous FTP from the Network Appliance web site.

Project Athena, Athena, Athena MUSE, Discuss, Hesiod, Kerberos, Moira, and Zephyr are trademarks of the Massachusetts Institute of Technology (MIT).

This product includes Kerberos software that was developed by M.I.T. and that was modified under the terms of the M.I.T. license.

This product includes Net-SNMP software that is covered by the following copyrights:

Copyright © 1989, 1991, 1992 Carnegie Mellon University. All rights reserved.

Derivative work—Copyright © 1996, 1998-2000 The Regents of the University of California. All rights reserved.

Copyright © 2001-2003, Network Associates Technology, Inc. All rights reserved.

Copyright © 2001-2003, Cambridge Broadband Ltd. All rights reserved.

Copyright © 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Software derived from copyrighted material of Network Appliance, Inc. is subject to the following license and disclaimer:

Network Appliance reserves the right to change any products described herein at any time, and without notice. Network Appliance assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by Network Appliance. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of Network Appliance.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark information

NetApp, the Network Appliance design, the bolt design, NetApp–the Network Appliance Company, FAServer, NearStore, NetCache, and WAFL are registered trademarks of Network Appliance, Inc. in the United States and other countries. Network Appliance is a registered trademark of Network Appliance, Inc. in Monaco and a trademark of Network Appliance, Inc. in the United States and Canada. SnapCopy is a registered trademark of Network Appliance, Inc. in the European Union and a trademark of Network Appliance, Inc. in the United States. DataFabric, FilerView, SecureShare, SnapManager, SnapMirror, SnapMover, SnapRestore, and SyncMirror are registered trademarks of Network Appliance, Inc. in the United States. Data ONTAP and Snapshot are trademarks of Network Appliance, Inc. in the United States and other countries. ApplianceWatch, BareMetal, Center-to-Edge, ContentDirector, FlexClone, FlexVol, FPolicy, gFiler, MultiStore, RAID-DP, SecureAdmin, Serving Data by Design, Smart SAN, SnapCache, SnapDrive, SnapLock, SnapVault, vFiler, and Web Filer are trademarks of Network Appliance, Inc. in the United States. Spinnaker Networks, the Spinnaker Networks logo, SpinAccess, SpinCluster, SpinFS, SpinHA, SpinMove, SpinServer, and SpinStor are registered trademarks of Spinnaker Networks, LLC in the United States, other countries, or both. SpinAV, SpinManager, SpinMirror, and SpinShot are trademarks of Spinnaker Networks, LLC in the United States, other countries, or both.

Apple is a registered trademark and QuickTime is a trademark of Apple Computer, Inc. in the United States and/or other countries.

Microsoft is a registered trademark and Windows Media is a trademark of Microsoft Corporation in the United States and/or other countries.

RealAudio, RealNetworks, RealPlayer, RealSystem, RealText, and RealVideo are registered trademarks and RealMedia, RealProxy, and SureStream are trademarks of RealNetworks, Inc. in the United States and/or other countries.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

Network Appliance is a licensee of the CompactFlash and CF Logo trademarks.

Network Appliance NetCache is certified RealSystem compatible.

SpinFS 2.5 Administration Guide

List of Figures ix
List of Tables xi

1.0 About This Guide 1
1.1 Typographical Conventions 1
1.2 Organization of the Guide 2
1.3 SpinServer Documentation 2

Part I: Prerequisites for SpinServer Administration 5

2.0 Before You Begin 7
2.1 IP Addresses 7
2.2 Naming Conventions 7

3.0 Navigating the Management Interface 9
3.1 Command Line Interface 9
Command Node Navigation 10
Command Line Editing 10
3.2 Graphical User Interface 12
Command Node Navigation 13
GUI Documentation Conventions 15

4.0 Connecting to the SpinServer 17
4.1 Secure Shell 17
UNIX Systems 17
Windows Systems 17
4.2 HTTPS 18
4.3 Telnet 18

Part II: Software Configuration 19

5.0 Initializing the SpinServer 21
5.1 Ethernet Ports 21
5.2 Installation and Configuration Tasks for the SpinServer 22
5.3 Checking System Component Connectivity 23
5.4 Installing SpinServer Software 23
Changing the Login Password 31
Changing the Root Password 31
Installation of Software Upgrades 32

6.0 Configuring the SpinServer System 33
6.1 Management Software 33
6.2 Logging In 33

6.3 Configuring HTTPS Access 34
Using HTTPS with Internet Explorer 34
Using HTTPS with Netscape or Mozilla 34
6.4 Configuring the SpinServer 35
Displaying SpinServer Information 35
Initializing a SpinServer Cluster 36
6.5 Setting Up Virtual Interfaces 38
6.6 Configuring RAID Components 39
RAID Levels 40
Reviewing the Disk Configuration 40
RAID Configuration 41
Creating RAID Sets 41
Initializing a RAID Set 43
Configuring a Hot Spare 43
Verifying Disk Setup 44
6.7 Creating a Storage Pool 44
6.8 Setting Up Domains 46
Specifying NIS Domains 46
Defining a CIFS Domain 47
6.9 Creating Virtual Servers 48
Creating a Virtual Server 49
6.10 Binding Network Virtual Interfaces 49
6.11 Setting Up Kerberos Authentication 50
Setting Up Kerberos Authentication for NFS 50
Setting Up Kerberos Authentication for CIFS 55
6.12 Setting Up Export Rule Sets 59
Creating an Export Rule Set 59
Adding a Rule to a Rule Set 60
6.13 Setting Up a Virtual File System 61
Creating a Virtual File System and Setting Its Quota 61
Mounting a Virtual File System 62
6.14 Completing CIFS Tasks 63
Creating and Configuring a CIFS Share 64
6.15 Verifying Network Connections 65

7.0 Integrating with HP OpenView 67
7.1 Integrating SpinManager with HP OpenView 67
Integrating SpinManager with HP OpenView on Windows 67
Integrating SpinManager with HP OpenView on Solaris and HP-UX 72
7.2 Creating SpinServer Objects in Network Node Manager 73

Part III: Data-Management Tasks 79

8.0 Managing Clusters 81
8.1 Cluster Management Concepts 81
Clustering Concepts 82
Quorum in Clusters 83
8.2 Cluster Management Tasks 83
Creating a Cluster 84
Joining a Cluster 84
Managing Clusters 85
Enabling Cluster Nodes 85

Designating a SpinServer as Epsilon 86
Deleting a Node from a Cluster 87
Displaying Information about Clusters 87
Setting a Route for a Cluster 88
Enabling High-Availability Mode in a Two-Node Cluster 89
Modifying the Cluster Configuration 90

9.0 Managing Networks 91
9.1 Network Management Concepts 91
9.2 Network Management Tasks 91
Creating and Configuring a Virtual Interface 92
Displaying Virtual Interface Status 92
Migrating a Virtual Interface 93
Displaying Virtual Interface Configurations 94
Deleting a Virtual Interface 95
Taking a Virtual Interface Offline 95
Modifying Virtual Interfaces 96
Reverting Virtual Interfaces 96
Adding a Failover Port 96
9.3 Port Management Tasks 97
Configuring the Management Port 98
Configuring the Server Ports 98
Configuring the Second Cluster Port 98
Adding a Static Route to a Virtual Interface 99
Displaying Network Ports 100
Taking a Network Port Down 101
Modifying a Network Port 102

10.0 Managing RAID Components 103
10.1 RAID Concepts 103
10.2 RAID Management Tasks 104
Managing RAID Sets 105
Managing RAID Subsystems 114
Managing Disks in a RAID Subsystem 118
Managing SpinStor Disk Enclosures 123
Managing RAID Controllers 125
Managing the Battery Backup Unit 127
Scheduling RAID Operations 129

11.0 Managing Storage 133
11.1 Storage Concepts 133
High Availability 134
11.2 Storage Tasks 135
Displaying Storage Units 135
Creating a Storage Pool 138
Displaying Information about Storage Pools 138
Modifying a Storage Pool 140
Taking a Storage Pool Offline 140
Bringing a Storage Pool Online 141
Disconnecting a Storage Pool 141
Deleting a Storage Pool 141
Synchronizing I/O Activities 142

Expanding the Size of a Storage Pool 142
Forcing the Discovery of a Storage Pool 142
Displaying the High-Availability Status of a Storage Pool 143
Adding a Passive Secondary to a Storage Pool 143
Setting the Priority for a Passive Secondary 144
Promoting a Passive Secondary to Active Secondary 144
Demoting an Active Secondary to Passive Secondary 145
Removing a Secondary 145
Failing Over a Storage Pool 145
Reverting Storage Pools 146

12.0 Managing Virtual Servers 147
12.1 Virtual Server Concepts 147
12.2 Virtual Server Tasks 147
Creating a Virtual Server 147
Displaying Information about Virtual Servers 148
Deleting a Virtual Server 149
Binding and Unbinding Virtual Interfaces 149
Binding and Unbinding All Virtual Interfaces 150
Modifying a Virtual Server 151
Mirroring the Root of a Virtual Server 151
12.3 Virtual-Server Route-Management Tasks 152
Creating and Configuring a Virtual Server Route 152
Displaying Virtual Server Routes 153
Deleting Virtual Server Routes 153

13.0 Managing Virtual File Systems 155
13.1 VFS Concepts 155
13.2 VFS Tasks 157
Defining Virtual File Systems 157
Creating Virtual File Systems 158
Mounting a Virtual File System 160
Displaying Information about Virtual File Systems 161
Deleting a Virtual File System 165
Creating a SpinShot of a Virtual File System 165
Copying a Virtual File System 165
Moving a Virtual File System 166
Mirroring a Virtual File System 166
Resetting the Export Rule Set for a VFS 169
Scheduling VFS Operations 169

14.0 Managing SpinAccess CIFS 177
14.1 CIFS Management Concepts 177
14.2 CIFS Management Tasks 179
Defining a CIFS Domain 179
Managing CIFS Servers 182
Managing CIFS Shares 186
Managing CIFS Superusers 190

15.0 Managing CIFS-NFS Translations 193
15.1 CIFS-NFS Translation Concepts 193
SpinAccess Conversion Rules 194

CIFS-to-NFS Translation Guidelines 195
Rules for CIFS-to-NFS Mappings 196
15.2 CIFS-NFS Translation Tasks 196
Managing NIS Domains 197
Managing NFS Servers 198
Managing CIFS-to-NFS Mappings 200

16.0 Managing Security 207
16.1 Security Concepts 207
Kerberos 207
Export Rule Sets 208
SpinAuth 209
16.2 Security Tasks 209
Configuring an NFS-Kerberos Environment 209
Configuring a CIFS-Kerberos Environment 210
Creating a Kerberos Realm Configuration 210
Creating and Managing Kerberos-NIS Mappings 213
Enabling Kerberos on an Interface 215
Configuring Export Rule Sets 217
Configuring SpinAuth 222

17.0 Protecting Against Viruses 225
17.1 Virus Scanning Concepts 225
17.2 Virus Scanning Tasks 225
Setting up Virus Scanning on Virtual Servers and VFSs 225
Configuring SpinAV for a Cluster 226
Configuring SpinAV on SpinServers 227

18.0 Managing Data Backup and Restoration 231
18.1 Backup and Restoration Concepts 231
18.2 Backup and Restoration Tasks 232
Configuring NDMP 232
Displaying Tape Devices 233
Configuring the SpinServer with Backup and Restoration Products 234

19.0 Managing Disaster Recovery 235
19.1 Disaster Recovery Concepts 235
Disaster Recovery Mechanisms 235
Preparing for Disaster Recovery 236
19.2 Disaster Recovery Tasks 237
Promoting a Read-Only VFS to Read-Write Parent 237
Removing a Mirror from a Group 238
Deleting a VFS's Mount Point 238
Forcibly Deleting a VFS 238
Forcing the Unmounting of a VFS 239
Making a VFS Read-Only 239
Making a VFS Read-Write 240
Making a VFS the Root of the Virtual Server 240
Renaming a VFS 240
Deleting a Specific File from a VFS 241
Deleting a VFS Family 241
Taking a VFS Offline 241

Forcibly Deleting a Storage Pool 242
Forcing a SpinServer to Take Over as Primary 242
Forcing a Storage Pool Offline 242
Deleting All Data from a Storage Unit 243
Recovering a Mirror Root VFS 243
Recovering a Nonmirror Root VFS 244
Moving a Storage Pool to a Different SpinServer 244

Part IV: System-Management Tasks..... 245

20.0 Managing SpinServers 247
20.1 System Management Concepts 247
20.2 System Management Tasks 248
Displaying Information about SpinServers 248
Modifying a SpinServer 249
Restarting a SpinServer 249
Shutting Down a SpinServer 250
Displaying Information about a SpinServer's Internal RAID Sets 250
Displaying System Software Information 252
Upgrading Software 252
Managing Startup Images 253
Managing the Root File-System Partitions 254
Managing Fibre Channel Interfaces 255
Managing Long-Running Jobs 258
Managing Health Monitoring 259
Setting and Maintaining the Time 260
Setting the Firewall Configuration on Ports 262

21.0 Managing System Events 265
21.1 Event Management Concepts 265
Monitoring Services 265
Event Information 265
Event Filters 266
Levels of Severity 266
Audit Log 266
21.2 Event Management Tasks 266
Displaying Event Alarms 267
Managing Spin Events 267
Viewing Log Records 269
Managing Event Filters 270
Managing Event Notifications 272
Viewing Event-Notification History 274
Managing Event-Notification Subscriptions 275
Managing Syslog Events 276
Viewing the Audit Log 278

22.0 Using Statistics 281
22.1 Statistics Concepts 281
22.2 Statistics Tasks 281
Displaying Client Statistics 281
Displaying Disk-Drive Statistics 282
Displaying File-System Statistics 283

Displaying Network-Access Statistics 290

23.0 Using Third-Party Management Utilities 301
23.1 Third-Party Management Concepts 301
23.2 Third-Party Management Tasks 301
Setting Up SNMP Trap Destinations 301

Part V: Troubleshooting 305

24.0 Advanced Administrative Tools 307
24.1 The spinsh Command 307
24.2 The buildexports Command 308
24.3 Pluggable Authentication Modules 309
24.4 The vfsdump and vfsrestore Commands 311
The vfsdump Command 312
The vfsrestore Command 312
Dump Files 313

25.0 Troubleshooting the SpinServer System 315
25.1 Displaying Environmental Information 315
Displaying Temperature Information 315
Displaying Fan Information 316
Displaying Power Information 316
Displaying UPS Information 317
25.2 Obtaining Crash Information 317
Deleting a Crash File 318
25.3 Sending Information to Technical Support 318
25.4 Parsing System Messages 319
25.5 Miscellaneous Tips 319
Resolving Problems With Joining a Cluster 319
Updating SpinServer Identification Information 319

Appendixes 321

26.0 Glossary 323

List of Figures

Figure 3-1. Initial view of the SpinManager GUI after login 12
Figure 3-2. The **Virtual Servers → Servers** command node selected in the GUI 13
Figure 3-3. The results of the **Show** command for a virtual server 14
Figure 5-3. The Welcome screen for SpinFS software installation 24
Figure 5-4. The installation prompt 25
Figure 5-5. An example creation message 26
Figure 5-6. An example package-installation message 26
Figure 5-7. The SpinServer automatically restarts after the software installation. 27
Figure 5-8. The hardware clock setting. 28
Figure 5-9. Enter the SpinServer ID in the hostname field. 29
Figure 5-10. Select DHCP or enter your IP addresses manually. 30
Figure 5-11. The login prompt 30
Figure 5-12. The SpinServer prompt after login 31
Figure 7-1. The Welcome screen 68
Figure 7-2. The License Agreement 68
Figure 7-3. The Destination Folder screen 69
Figure 7-4. The Change Current Destination Folder screen 70
Figure 7-5. The Ready to Install the Program screen 71
Figure 7-6. The InstallShield Wizard Completed screen 72
Figure 7-7. Launch OpenView. 73
Figure 7-8. Add Object Palette 74
Figure 7-9. The Add Object dialog box 75
Figure 7-10. The SpinServer server object 76
Figure 7-11. Launch SpinManager 77
Figure 7-12. SpinManager login screen 78

List of Tables

Table 5-1. SpinServer Ethernet Connections 21
Table 5-2. SpinServer Configuration Information 22
Table 6-1. SpinServer Configuration Example 35
Table 13-1. Differences among VFS Features 156
Table 13-2. VFS Scheduler Format Examples 172
Table 19-1. Comparison of Backup Types 236

1.0 About This Guide

The *SpinFS® 2.5 Administration Guide* is written for system administrators. It includes information that is needed to complete system setup and to perform system-maintenance procedures.

This document is intended to provide the information that a system administrator needs to configure and maintain a SpinServer® system. The system administrator needs to be familiar with UNIX commands.

Note: The command line interface (CLI) is used in the command examples in this guide.

1.1 Typographical Conventions

This document uses the following typographical conventions:

- **`Fixed-width bold text`** indicates commands and command options in syntax definitions, examples, and running text.
- *Italicized text* indicates user-supplied values. For example, **`-deviceid`** *`device_ID`* indicates that you supply a device ID for *`device_ID`* after the **`-deviceid`** parameter. Italics also indicate emphasis and the titles of books.
- **Bold text** indicates GUI elements (including menus, menu items, labels, buttons, icons, and folders), keys on the keyboard (for example, **Enter** and **ESC**), and literal values.
- `Fixed-width text` indicates screen output.
- The backslash character (`\`) is used to indicate a line break in a command. When issuing a command, you do not need to enter this symbol.
- Lists from which you must choose an item in syntax descriptions are enclosed in braces (`{ }`). For example, `{`**`true | false`**`}` indicates that you must choose between **`true`** and **`false`**. Do not enter the braces in the command.
- The vertical column (`|`) separates items in a list of choices enclosed in braces in syntax descriptions, as shown in the example above.
- Ellipses (**`...`**) in syntax descriptions indicate that you can repeat the preceding item one or more times. Ellipses in examples indicate that the information was omitted for the sake of brevity.
- An item in syntax descriptions enclosed in square brackets (`[ ]`) is optional. For example in the following command, the `[ ]` characters indicate that the parameter **`-nodetype`** is optional:
  `spinserver::>` **`spinaccess cifs servers new -vsname VS1 \`**

`-netbios NB1 -domain ENG` [**`-nodetype H`**]
Leaving the optional parameters out of the command line has no consequences for command execution. The brackets are not to be entered as part of the command.

- In command examples, the CLI prompt is represented as follows:
  `spinserver::>`
  On a SpinServer system, the word `spinserver` in the prompt is replaced with the name of the SpinServer itself. For instance, on a SpinServer named Server1, the CLI prompt is the following:
  `Server1::>`

1.2 Organization of the Guide

This guide provides you with information and examples for the specific tasks that you must perform to configure and maintain the SpinServer system. The document is divided into the following parts:

- “Part I: Prerequisites for SpinServer Administration” on page 5 lists the prerequisites needed for SpinServer administration and introduces the administrative interfaces.
- “Part II: Software Configuration” on page 19 provides instructions for installing the SpinFS software on a SpinServer and creating a basic configuration.
- “Part III: Data-Management Tasks” on page 79 discusses concepts and tasks for managing data on a SpinServer system.
- “Part IV: System-Management Tasks” on page 245 discusses concepts and tasks for managing a SpinServer system.
- “Part V: Troubleshooting” on page 305 provides information on working with system messages.

1.3 SpinServer Documentation

The following documentation is provided for the SpinServer:

- *SpinFS Architecture Basic Concepts Guide*
- *SpinFS 2.5 Administration Guide*
- *SpinFS 2.5 Administration Reference*
- *SpinFS 2.5 Messages and Troubleshooting Guide*
- *SpinFS Release Notes*
- *SpinServer 4100 Installation Guide*
- *SpinServer 4100 Quick Start Guide*
- *SpinServer Supported Storage Release Notes*
- *SpinServer Site Preparation Guide*

Documentation is available in PDF format and can be accessed from the SpinServer browser management interface or from the CD. To access the documentation from the CD, open the *root_directory*`/SpinDocs/index.html` file in a web browser, where *root_directory* is the root directory of the CD.

Part I: Prerequisites for SpinServer Administration

This part discusses prerequisites for administering the SpinServer, including naming conventions, the management interface, and supported methods of connecting to the SpinServer. It is organized into the following chapters:

Chapter 2. Before You Begin

"Before You Begin" on page 7 discusses tasks you must complete before administering a SpinServer system.

Chapter 3. Navigating the Management Interface

"Navigating the Management Interface" on page 9 provides information on using the SpinServer's command line interface and graphical user interface.

Chapter 4. Connecting to the SpinServer

"Connecting to the SpinServer" on page 17 provides information on connecting to the SpinServer system to perform administrative tasks.

2.0 Before You Begin

This chapter discusses the tasks you must complete before you install and configure a SpinServer. These tasks include obtaining necessary IP addresses and designing a naming convention for system entities.

2.1 IP Addresses

You need the following IP addresses before you begin the configuration of the SpinServer system components. Each set of ports must be on a separate IP subnet. Obtain these IP addresses from your network administrator.

- An IP address is needed for the management software port (mgmt0) for each SpinServer. For remote access of the management interface, the port can be configured on a separate management LAN.
- An IP address is needed for both SpinServer network ports (serv0 and serv1) for each SpinServer.
- For clustering, an IP address is needed for both cluster ports (clus0 and clus1). For security reasons, cluster ports must be attached to a secure network. The cluster port must be configured even if only a single SpinServer is being installed to assure that all system components work together properly.

2.2 Naming Conventions

For ease of management of the SpinServer system and the data stored on it, you need to consider the names you choose for SpinServer system entities, including the SpinServers, virtual network interfaces, storage pools, virtual servers, and virtual file systems.

To prevent confusion and potential loss or misplacement of data, the names you choose must be orderly and easily recognizable by all system administrators who maintain the SpinServer system. Any naming system can be chosen for the system, but the following rules must be observed:

- You enter the SpinServer ID into the system at the time you initially install the software. The name of the SpinServer is propagated from that initial installation into the management framework. The IDs for the SpinServers must be unique within a cluster.
- After the ID for a SpinServer has been set, you cannot modify the server name or host

name. To do so invalidates the failover port designation for the SpinServer and can result in the loss of data.

- Storage pools names can be up to 32 characters long.
- Virtual server names and VFS names can be 1 to 32 characters long, consisting of letters, numbers, period, and underscore. They must start with a letter or number. The names are case sensitive; for example, **`vs0`**, **`vS0`**, and **`Vs0`** are all acceptable, different names for a virtual server.
- Virtual file system names must be unique within a cluster. You cannot use the same virtual file system name on multiple virtual servers.

3.0 Navigating the Management Interface

This chapter discusses the SpinManager software, which provides a management interface to the SpinServer for system administration tasks. SpinManager is built by using the Common Information Model (CIM), an object-oriented model that provides a conceptual framework for describing management data.

SpinManager provides two ways of interacting with SpinServers. The SpinManager Command Line Interface (CLI) provides a command-based mechanism for interacting with the SpinServer that is similar to the tcsh shell in UNIX. The SpinManager Graphical User Interface (GUI) enables you to interact with SpinServer objects by using a web browser.

Regardless of the interface used (CLI or GUI), a set of similar objects is displayed as rows in a table. Each object/row is identified or located by a set of keys, which occupy the first *n* fields. The rest of the rows are occupied by parameters.

SpinManager commands are organized hierarchically into command sets, or *nodes*. Command nodes can contain other command nodes, which are referred to as *subnodes*. Subnodes, in turn, can contain *leaf nodes*. The commands contained within the command nodes enable you to manage the components of a SpinServer system.

This chapter includes the following sections:

- "Command Line Interface" on page 9 describes the command line interface (CLI).
- "Graphical User Interface" on page 12 describes the graphical user interface (GUI).

3.1 Command Line Interface

The command line interface provides a command-based view of the management interface. SpinManager commands are entered at the prompt, and command results are displayed in text.

The command line interface can be accessed from a terminal that is connected directly to a SpinServer on the COM1 serial port, or you can use secure shell (SSH) to access a SpinServer and use the command line interface to interact with the management interface remotely.

The CLI's command prompt is represented as `spinserver::>` in the documentation. On a running SpinServer, this prompt reflects the machine name. For instance, on a SpinServer named DataServer, the CLI prompt is `DataServer::>`.

Command Node Navigation

When using the command line interface, you can access a command node by typing the command node name at the prompt and then pressing the **Enter** key. The command node name is then included in the prompt text to indicate that you are interacting with this command node. To move deeper into the command node hierarchy, type the name of a command subnode followed by the **Enter** key; the subnode name is now included in the prompt text and the context shifts to that subnode.

Within a given command node, you can enter only commands that are available in that command node context, which includes all the subnodes beneath that command node. To see the list of commands that are available, enter **?** (question mark). To view the parameters for a given command, enter the command name and press **Tab**.

You can navigate several levels of commands if the entire command string is entered. For example, you can display a list of disk drives with information about disk ID, capacity, status, the RAID subsystem using it, channel, and target by entering **`raid diskdrives show`** at the prompt. The same information is obtained by navigating through one command node at a time as shown in the following example:

```
spinserver::> raid

spinserver:: raid> diskdrives

spinserver:: raid diskdrives> show
```

Also, you need to enter only the minimum number of letters in a command that makes the command unique to that level. For the previous example, enter **`r d sh`** to display disk drive information. For automatic completion of the command, press **Tab**.

Use the following actions to navigate from one command node to another:

- Type **`.`** and press **Enter** to display the command node at the current level.
- Type **`..`** and press **Enter** to go up one level in the command node hierarchy.
- Type **`top`** and press **Enter** to go to the top level of the command node hierarchy.

Command Line Editing

The command at the current shell prompt is the current active command. You can edit any part of the command. In the table below, ^ indicates that you must press the **Ctrl** key.

Edit Command	Action
^b	Move the cursor back one character.
^f	Move the cursor forward one character.
^a	Move the cursor to the beginning of the line.
^e	Move the cursor to the end of the line.

^k	Remove the contents of the edit buffer, from the cursor to the end of the line, and save it in the cut buffer.
^y	Yank the contents of the cut buffer, pushing it into the edit buffer at the cursor.
ESC b	Move the cursor back one word.
ESC f	Move the cursor forward one word.
ESC d	Cut the contents of the edit buffer, beginning at the cursor and continuing to the end of the following word.
ESC Backspace	Delete the word before the cursor.
^h	Delete the character before the cursor.
Backspace	Delete the character before the cursor.
^d	Delete the character after the cursor.
^p	Replace the current contents of the edit buffer with the previous entry on the history list. For each successive ^p action, the history cursor moves to the previous entry.
^n	Replace the current contents of the edit buffer with the next entry on the history buffer.
↓	Down history.
↑	Up history.
←	Go backward one character.
→	Go forward one character.
^q	TTY start output.
^s	TTY stop output.
^u	Clear the current edit buffer.
^v	Escapes a special mapping for the following character. For instance, to enter a question mark into a command's arguments, press **^v**, then press **?**.
?	Display context-sensitive help.

3.2 Graphical User Interface

The GUI provides the end user with an interactive management interface. SpinManager commands are selected from a menu of options, and command results are displayed in tables.

The GUI can be accessed by pointing a supported web browser to a SpinServer (for example, http://*server*, where *server* is the IP address or host name of the SpinServer). You must enter a valid administrative user name and password when connecting to a SpinServer.

Note: If a SpinManager GUI session remains idle for 60 minutes, the session times out. If you attempt to use a timed-out session, the GUI prompts you again for the administrative user name and password.

When you have provided a valid user name and password, the browser view resembles the following figure.

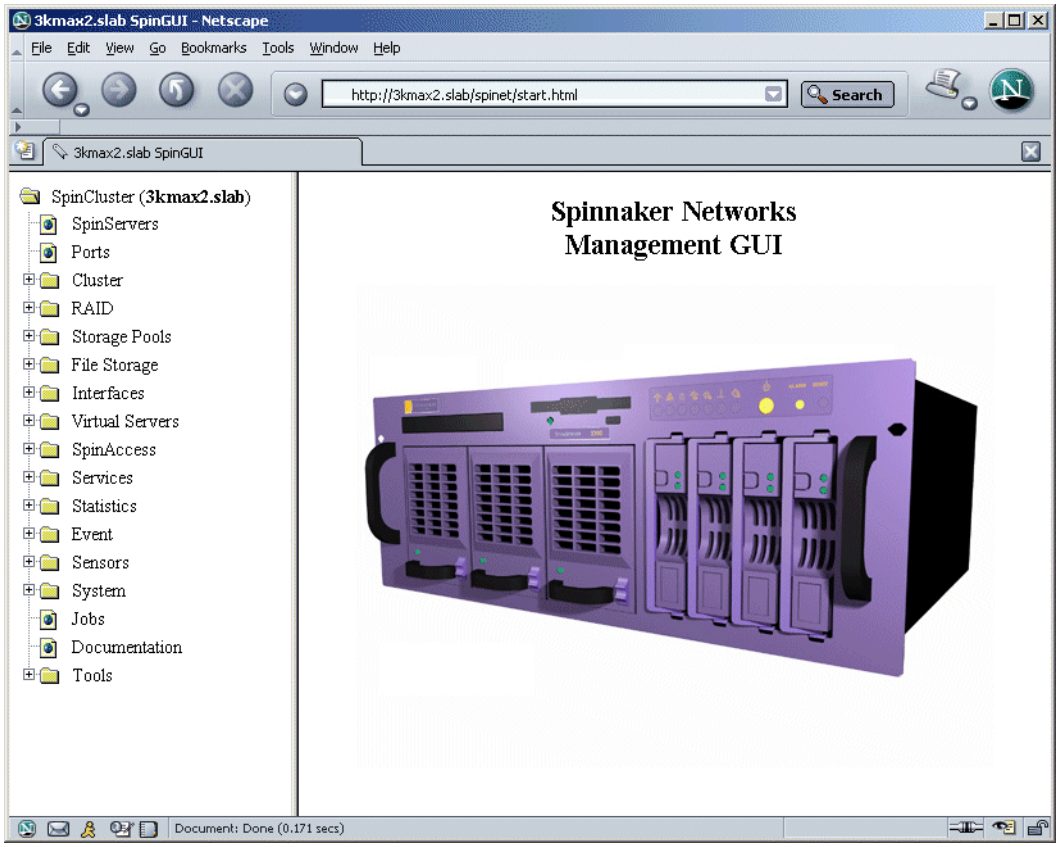


Figure 3-1. Initial view of the SpinManager GUI after login

Command Node Navigation

The left-hand pane, or *navigation pane*, of the browser lists the available command nodes. Some command nodes expand into subnodes; these nodes are indicated by a folder icon and can be expanded by clicking the icon. Similarly, some subnodes expand into leaf nodes. If a high-level command node is expanded, a minus sign (-) is displayed in the box to the left of the folder icon; if a high-level command node is not expanded, a plus sign (+) is displayed in the box. Leaf nodes are indicated by a page icon. Clicking a leaf node displays a list, in table format, of that node's objects in the right-hand pane, or *action pane*, of the browser.

The following figure shows the GUI with the high-level **Virtual Servers** command node expanded and the **Servers** command node selected in the navigation pane. A table that lists the SpinServer's virtual servers is displayed in the action pane.

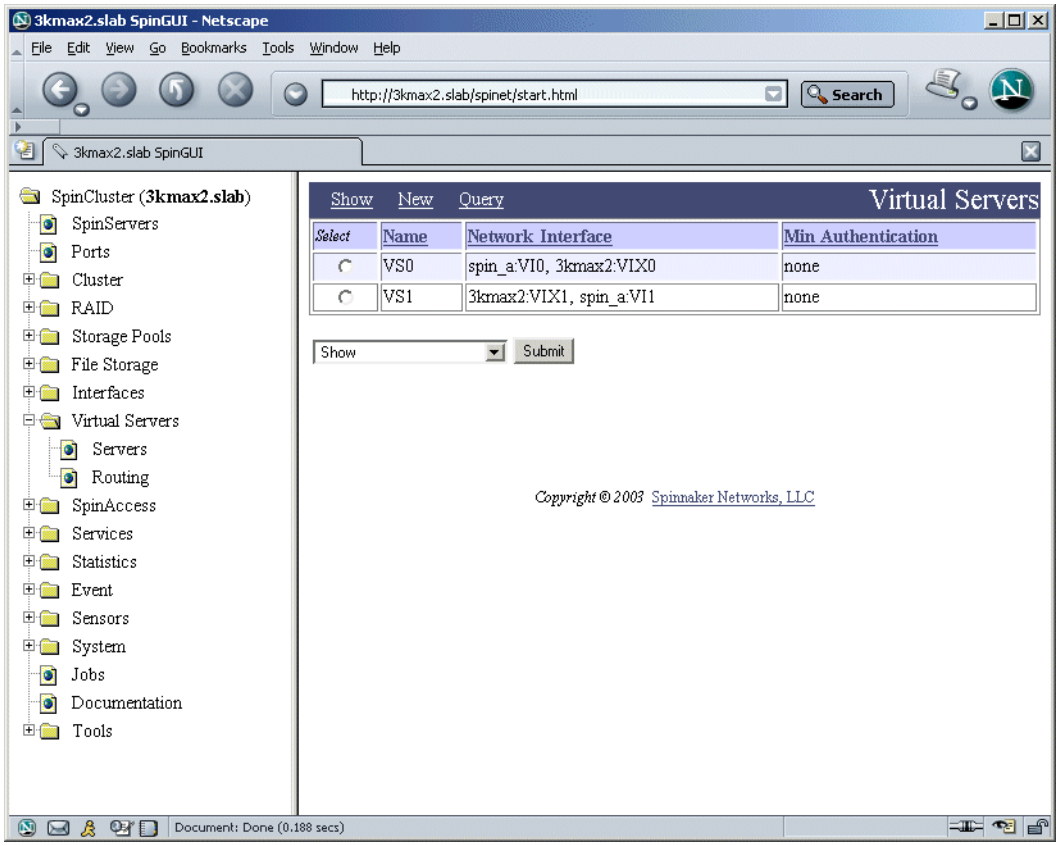


Figure 3-2. The **Virtual Servers ➔ Servers** command node selected in the GUI

When the command node is displayed in the action pane, commands that affect the entire node (that is, all objects that are displayed in the table) are displayed in the bar that runs across the top of the pane, and commands that affect specific objects within the node (that is, specific rows of the table) are available from the drop-down menu. You can perform the following actions:

- To reorder the table according to specific attributes, click the headings at the top of each column. For instance, if a table has columns labeled **Name** and **Network Interface**, clicking **Name** once orders objects by name, and clicking it again reorders them

in the opposite order (ascending or descending). Similarly, clicking **Network Interface** orders objects by their network interfaces, and clicking it again reorders them in the opposite order.

- To redisplay the default table at any time, click **Show** in the top bar.
- To create a new object (an operation that does not involve any objects currently listed in the table), click **New** in the top bar. The GUI then displays a dialog box into which you can enter the attributes of the new object. When you have entered the new object's attributes, click **Create** to create the object.
- To run a command on an object, select the object by clicking its corresponding button in the **Select** column, select the appropriate command from the drop-down menu, and click **Submit**. If additional information is needed or if confirmation is required before the command is run, the GUI prompts you for the required information.
- To show the attributes of an object that is listed in the table, select the object by clicking its corresponding button in the **Select** column, select **Show** from the drop-down menu, and click **Submit**. A sample result that shows the attributes of a virtual server is shown in the following figure.

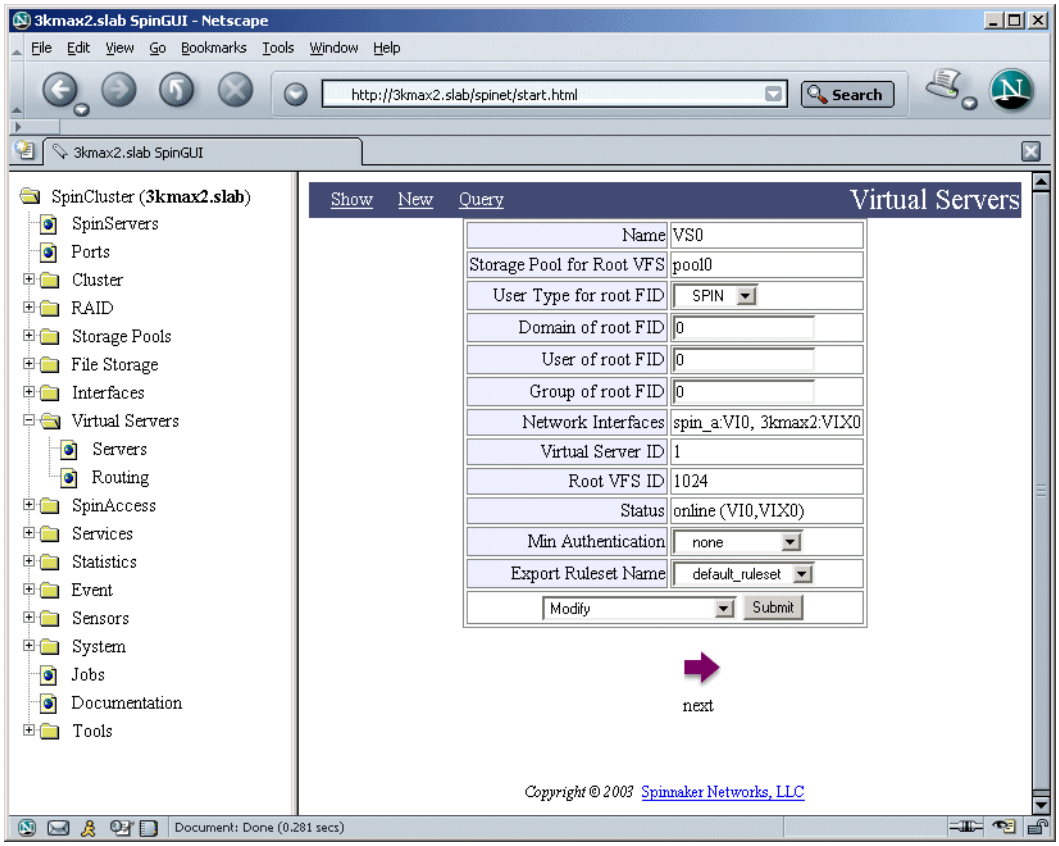


Figure 3-3. The results of the **Show** command for a virtual server

From this view, you can modify the attributes of the selected object by changing the values in editable attribute fields, selecting **Modify** from the drop-down menu, and clicking **Submit**. You can also run other commands on the object by selecting them from the drop-down menu and clicking **Submit**. With the exception of **Modify**, which replaces **Show**, the commands available from the drop-down menu in this view are the same that are available from the drop-down menu in the previous view.

GUI Documentation Conventions

The following conventions are used in the documentation to describe the use of the GUI:

- A right-pointing arrow (➔) indicates selections that must be made in the navigation pane. For instance, "Navigate to the **Storage Pools ➔ Storage Pools** node" indicates that you click the top-level **Storage Pools** node in the navigation pane, then click the **Storage Pools** subnode. Similarly, "Navigate to the **SpinAccess ➔ CIFS ➔ Shares** node" indicates that you click the **SpinAccess** node in the navigation pane, click the **CIFS** subnode, and then click the **Shares** leaf node.
- In the context of the GUI, the word *run* indicates that you need to select a command from the drop-down menu in the action pane, then click the **Submit** button to run the command. For example, "Select **vs1** from the table and run **Delete**" indicates that in the action pane you select the **vs1** entry from the table, select the **Delete** command from the drop-down menu below the table, and then click **Submit**.
- The GUI features two variants of the **Show** command. The **Show** link in the top bar of the action pane refreshes the table listing of all objects in the command node. The **Show** command that is available in the drop-down menu displays information for a selected item from the table; select the command from the drop-down menu and then click **Submit**. These two variants are distinguished in the documentation by phrasing such as "Click **Show** in the top bar of the action pane" and "Select an entry from the table and run **Show**," respectively.

Note: The **Query** command that is available from the top bar of the action pane for some command nodes is not currently supported. Do not use this command unless you are specifically instructed to do so by a support representative.

4.0 Connecting to the SpinServer

This chapter describes supported methods for connecting to a SpinServer.

4.1 Secure Shell

Secure shell (SSH) is a program used to log in to another machine by using an encrypted connection that provides secure communications over insecure channels. It is recommended for secure connections to a SpinServer system.

Note: Connect to the SpinServer by using the IP address or Domain Name System (DNS) name of the management port, not a data or cluster port.

You can obtain secure shell from a number of sources, including the following:

- http://openssh.org
- http://ssh.com
- OpenSSH is included in the Red Hat 7.2 Linux distribution.

UNIX Systems

To use SSH from a UNIX system, perform the following steps:

1. Enter **`ssh admin@spinserver`**. At the prompt, enter the root password to run a remote shell.

2. When prompted for the admin password, enter the administrator password. This is initially set to admin.

Windows Systems

To use SSH from a Windows system, perform the following steps:

1. Start an SSH client.

2. Connect to the SpinServer by using the IP address or DNS name of the management port.

3. Log in as **admin**.

4. When prompted for the admin password, enter the administrator password. This is initially set to admin.

4.2 HTTPS

To connect to a SpinServer by HTTPS, in the address field of a browser, type **https://***server*, where *server* is the ID of the SpinServer you want to manage (e.g., **fileserver1.engineeringlab**, **Server1**, or **10.2.2.237**).

See "Configuring HTTPS Access" on page 34 for information on setting up secure HTTPS on a SpinServer.

4.3 Telnet

Telnet access can be enabled by running the **chkconfig telnet on** command after logging in as root to a SpinServer.

Note: Connecting to the SpinServer via Telnet is not a secure connection.

Part II: Software Configuration

This part includes the following chapters:

Chapter 5: Initializing the SpinServer

"Initializing the SpinServer" on page 21 provides information about installing the SpinFS software on a SpinServer.

Chapter 6: Configuring the SpinServer System

"Configuring the SpinServer System" on page 33 provides the information for completing a basic configuration of the SpinServer system.

Chapter 7: Integrating with HP OpenView

"Integrating with HP OpenView" on page 67 provides information for running SpinManager on HP OpenView Network Node Manager.

5.0 Initializing the SpinServer

This chapter describes the initialization process for the SpinServer. Before initializing the SpinServer, ensure that all hardware components are installed as documented in the *Quick Start Guide* and *Installation Guide*.

The CLI SpinManager is used in the command examples in this guide. The GUI SpinManager has the same commands in the same command hierarchy.

Important: It is not recommended that the SpinServer Reset button be used to restart the SpinServer at any time because this shutdown method does not provide the system with the time necessary to properly save all data.

5.1 Ethernet Ports

A SpinServer has six Ethernet connections, which are identified in the following table.

Table 5-1. SpinServer Ethernet Connections

Ethernet Port	Name
mgmt0	Management port
mgmt1	Management port (not used)
serv0	Server port one
serv1	Server port two (optional)
clus0	Cluster port one
clus1	Cluster port two (optional)

A network virtual interface is associated with a network port and is assigned an IP address. A network port can be configured with one or more network virtual interfaces with IP addresses and additional failover interfaces.

5.2 Installation and Configuration Tasks for the SpinServer

Before installing SpinServer software, you must obtain configuration information from your network administrator. Use the following table to enter the information that is needed during the installation process.

Table 5-2. SpinServer Configuration Information

SpinServer ID	
Management IP address (if not using Dynamic Host Configuration Protocol (DHCP))	
Management netmask (if not using DHCP)	
Management default gateway IP (if not using DHCP)	
Management primary name server (if not using DHCP)	
Server IP address 1	
Server netmask 1	
Cluster IP address 1	
Cluster IP address 2 (optional)	
Cluster netmask	
Cluster broadcast	

Important: You enter the SpinServer ID at the time you initially install the software. The SpinServer ID can be an IP address (for example, **`10.1.1.14`**) or a name (for example, **`Server1`**). The name or ID of the SpinServer is propagated from that initial installation of the software into the SpinManager management framework. To prevent confusion and potential loss or misplacement of data, the SpinServer IDs you choose must be orderly and easily recognizable by all system administrators who maintain your SpinServer system. Any naming system can be chosen for your system, *but the IDs for the SpinServers must be unique within a cluster*. After the ID for a SpinServer has been set, you cannot modify the server name or ID or host name. To do so invalidates the failover port designation for the SpinServer and can result in the loss of data.

Note: To assure that all system components work together properly, one cluster port must be configured even if only a single SpinServer is being installed. Even if you intend to use only one of the cluster ports, it is *highly recommended* that you configure both cluster ports at initial installation of the software.

The serial number of a SpinServer is stored internally. During the installation of the SpinServer software, the SpinServer automatically restarts. At that time, the serial number is populated into the management framework.

You can use the **`spinservers modify`** command to further identify the SpinServer by entering this additional information: owner, contact information, asset tag, and location of the SpinServer. For additional information about this command, see "Modifying a SpinServer" on page 249.

5.3 Checking System Component Connectivity

Before installing the SpinServer software, ensure that all hardware has been correctly set up and cabled, as described in the *Installation Guide*. Check all cables and cords for the proper physical connections. The software installation and configuration procedures require the following connectivity and configuration.

- The SpinServer management port (mgmt0) is connected to the management network.
- The SpinServer server ports (serv0 and serv1) are connected to the server network.
- The Spin Server cluster port (clus0 and clus1) are connected to the cluster network.
- The SpinServer Fibre Channel port or ports are connected to the host port or ports of the RAID enclosure.
- The SpinServer UPS serial port (COM2) is connected to the fully charged UPS.
- The SpinServer console serial port (COM1) is connected to a console, and the console is configured as follows:
  - Bits per second: 9600
  - Data bits: 8
  - Parity: None
  - Stop bits: 1
  - Flow control: None

5.4 Installing SpinServer Software

For the initial installation of the SpinFS software, follow the procedures listed below.

Important: When you enter the SpinServer ID at the time you initially install the software, the SpinServer ID can be an IP address (for example, 10.1.1.14) or a name (for the examples used in this document, Server1).

Note: If you are configuring the SpinServer statically, you must enter the IP addresses manually and you need to have the following information available before you begin the software installation process. Obtain this information from your network administrator.

- Management port IP address
- Netmask
- Default gateway IP address
- Primary name server IP address

Note: All software installation output is directed to the serial console. No software installation output is visible on a VGA console.

1. If the SpinServer is on, power it down by pressing and holding the power button until the LEDs on the front panel power supply units turn red. (This takes about five seconds.)

2. When the SpinServer is off, press and release the power button one time to power up the SpinServer. During this process, place the software CD into the CD drive. From the serial console, you can monitor the initialization processes.

Note: Ensure that the CD is in place before the SpinServer searches for the drives from which to boot.

3. The **Welcome** screen displays the End User License Agreement.

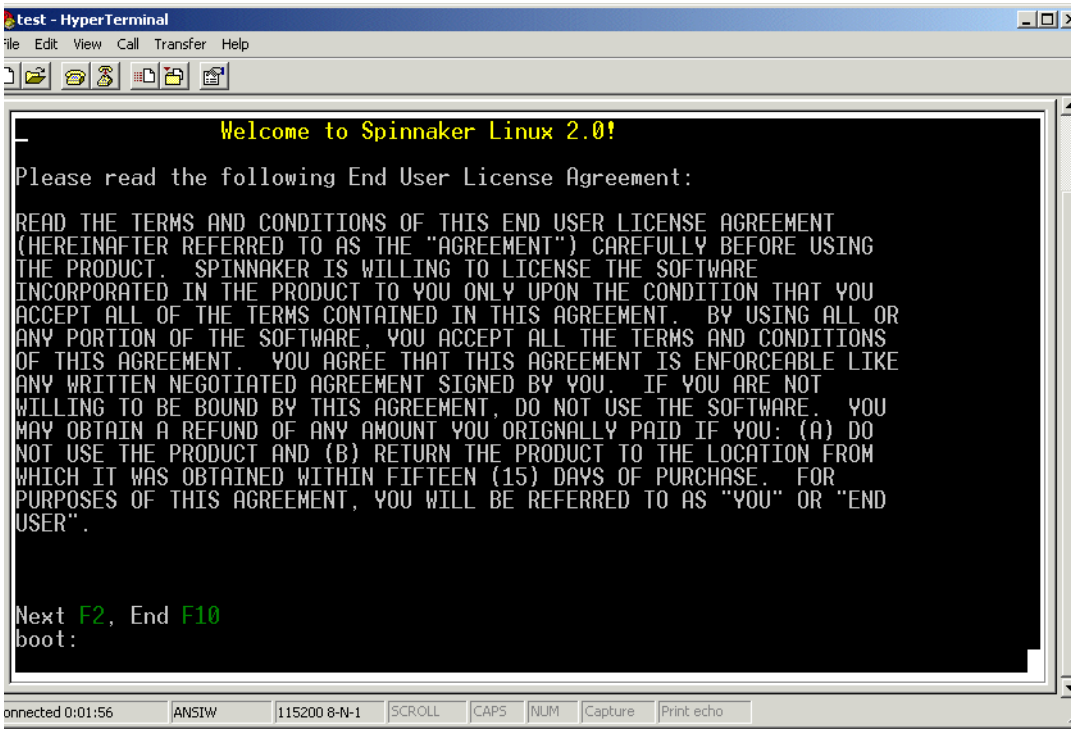


Figure 5-3. The Welcome screen for SpinFS software installation

4. To continue reading the End User License Agreement, page through the the screens by following the instructions at the bottom of each screen (for example, to read the second End User License Agreement screen, press **Ctrl-F 2**).

5. After the final End User License Agreement screen, the following screen is displayed. Type **`install`** and press **Enter** to continue the installation process.

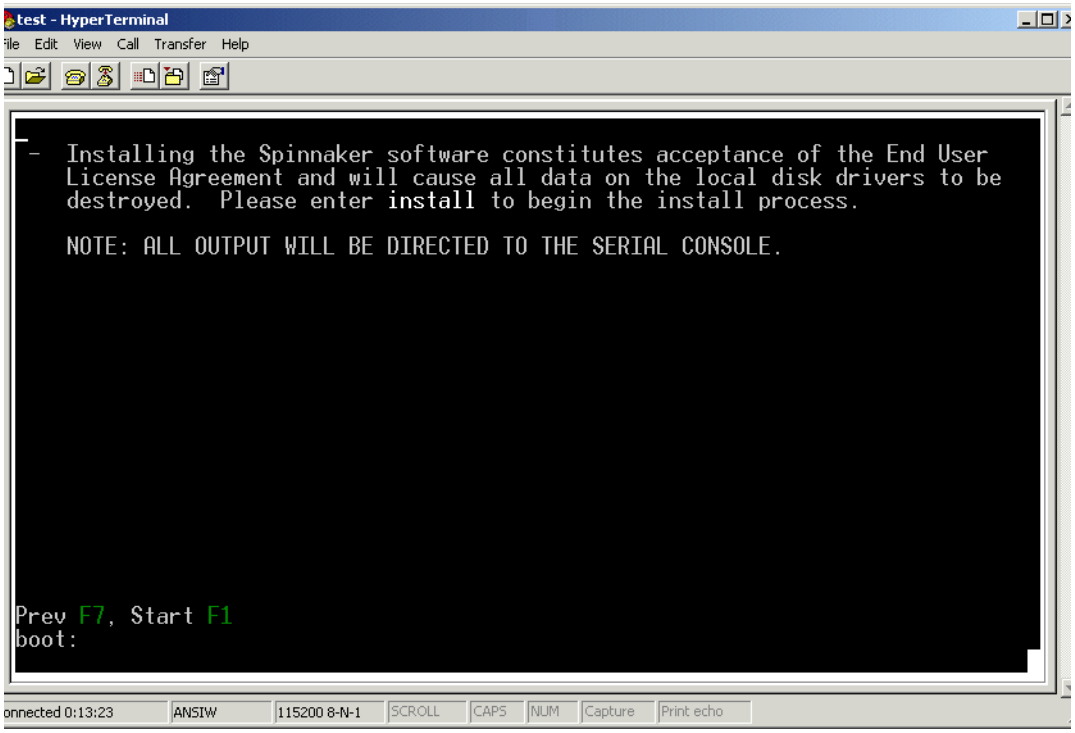


Figure 5-4. The installation prompt

6. As the initialization continues, on-screen messages indicate that the system is creating RAID sets, formatting file systems, installing software packages, and completing other procedures for configuring the system. The following figures show two example messages.

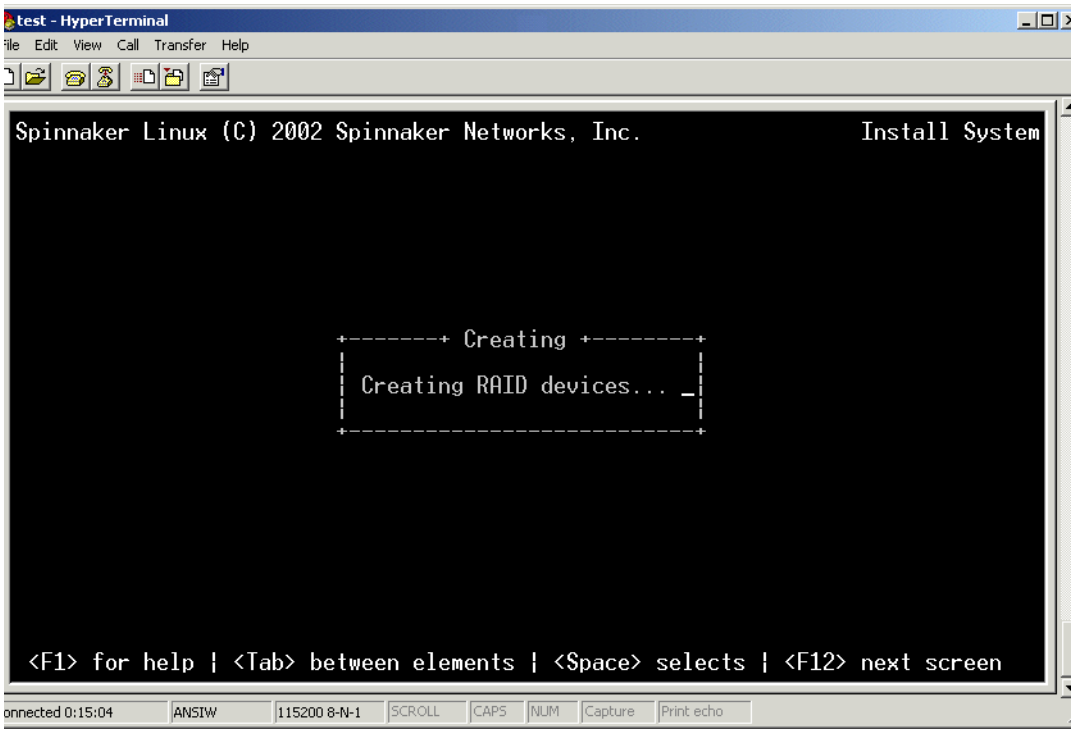


Figure 5-5. An example creation message

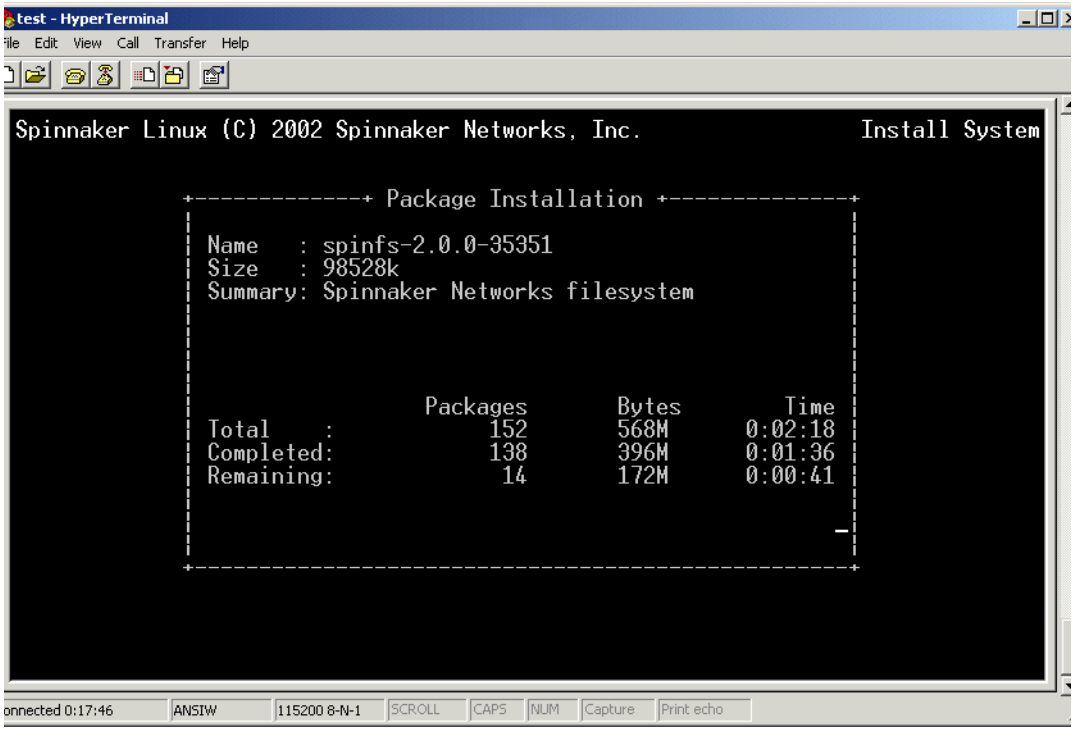


Figure 5-6. An example package-installation message

7. The SpinServer automatically restarts, as shown in the following figure, and ejects the CD. Remove the CD from the drive.

Caution: The software CD must be removed from the CD drive. If the SpinServer loses power and the CD is in the drive, the SpinServer installation process can restart when power to the SpinServer is restored. If installation of the software is restarted and completed, all data that is stored on the SpinServer is lost.

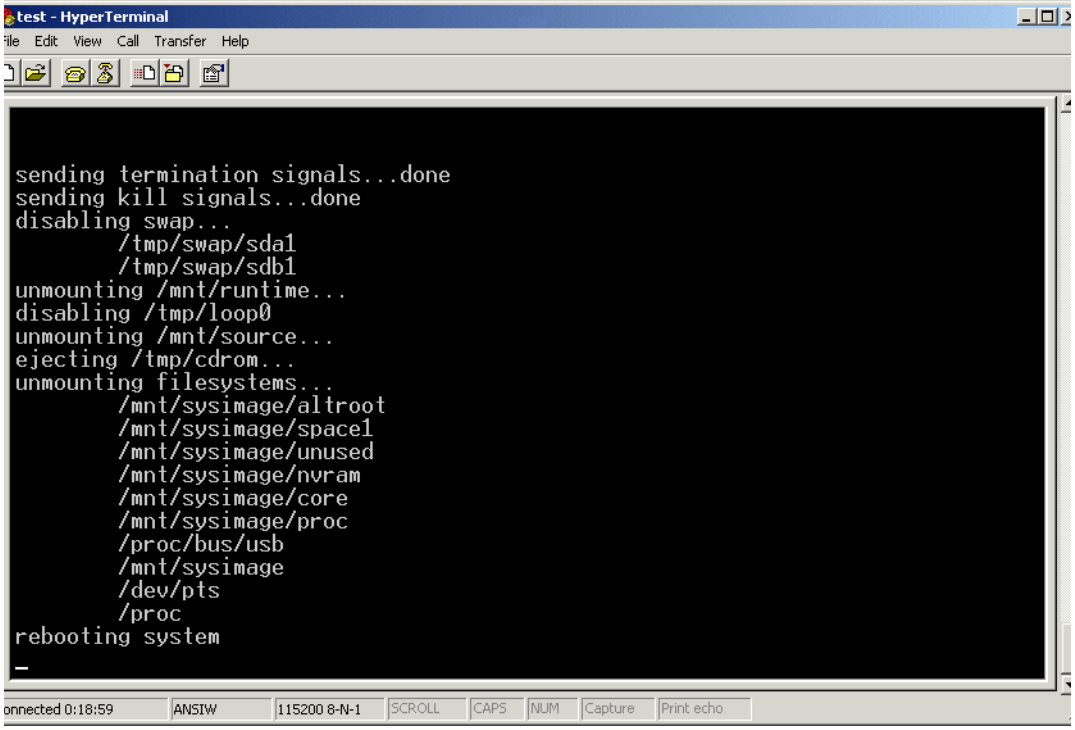


Figure 5-7. The SpinServer automatically restarts after the software installation.

8. Enter the clock settings into the system as shown in the following screen. Select the settings for your region and tab to **OK**, then press the space bar.

Note: The default setting for the hardware clock is GMT. Do not change this setting.

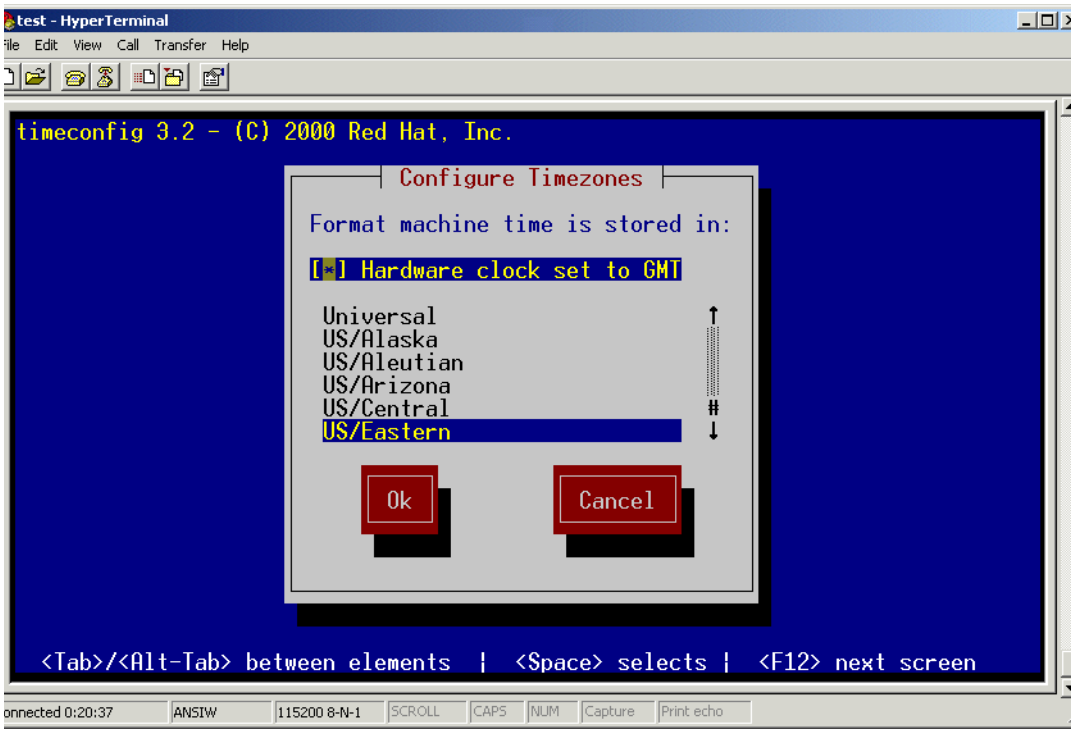


Figure 5-8. The hardware clock setting.

9. The initialization process prompts you for the name or ID of the SpinServer, as shown in the following screen. After you enter the SpinServer ID into the **Hostname** field, tab to the **OK** box and press the space bar.

Note: For the examples given in this chapter, the SpinServer ID is **`Server1`**.

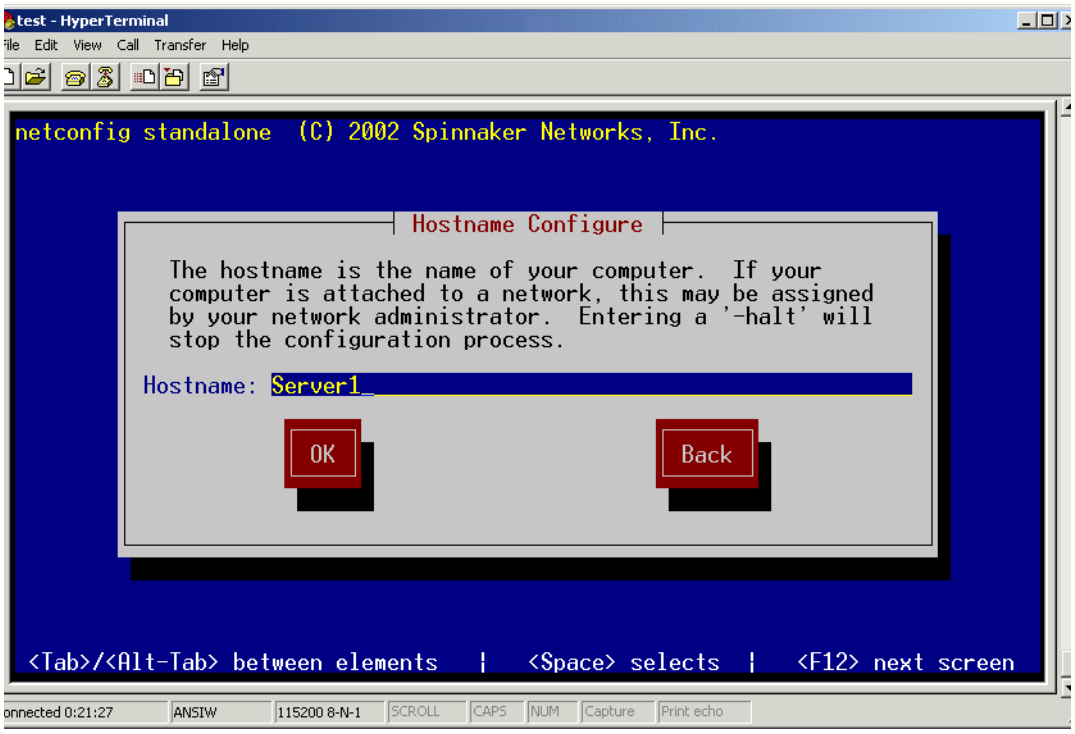


Figure 5-9. Enter the SpinServer ID in the hostname field.

10. After entering the SpinServer ID, configure the management port for the SpinServer. If the management port is to be configured by using DHCP, tab to **OK** and press the space bar. If you are manually entering the network configuration information, complete the fields with the configuration information provided by your network administrator. When finished, tab to **OK** and press the space bar.

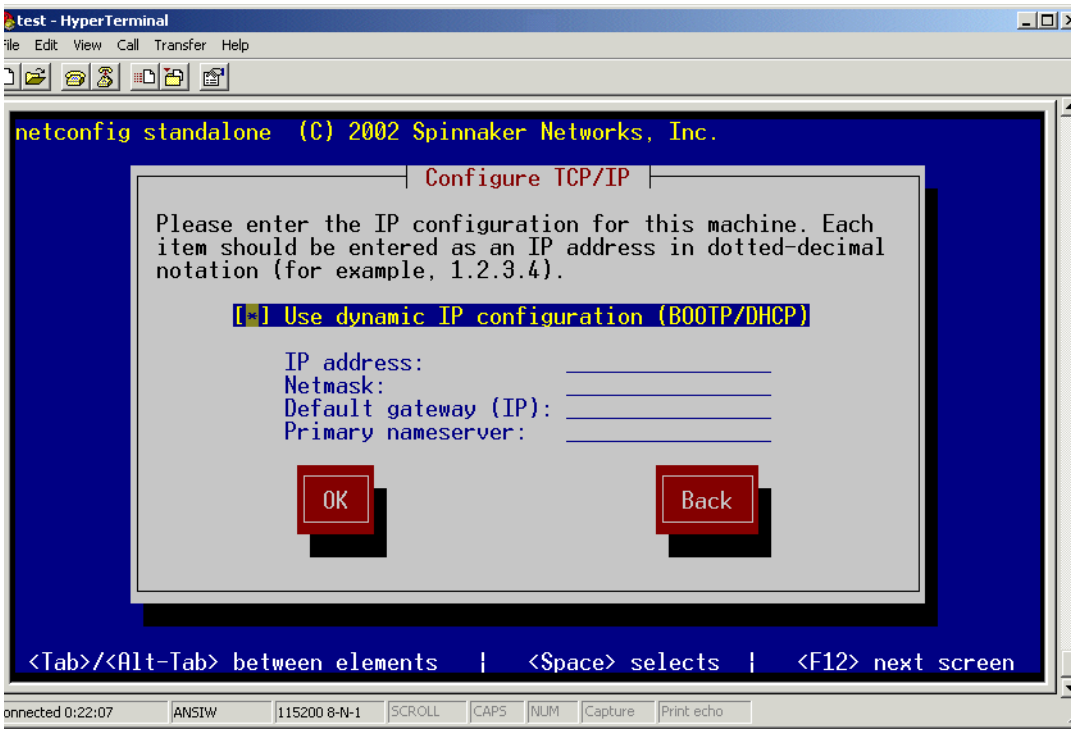


Figure 5-10. Select DHCP or enter your IP addresses manually.

11. When the installation of the SpinFS software is complete, the SpinServer presents the login prompt. At the login prompt, type **admin**. At the Password prompt, type **admin**.

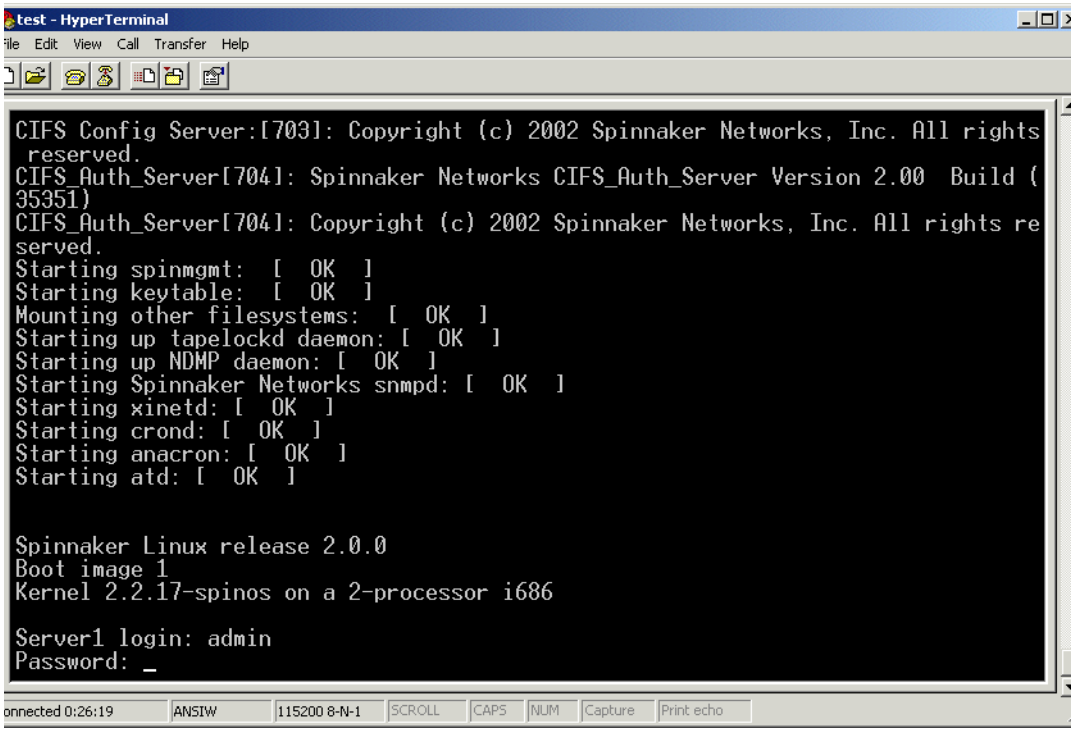


Figure 5-11. The login prompt

12. Your system is now ready for configuration, as shown in the following figure.

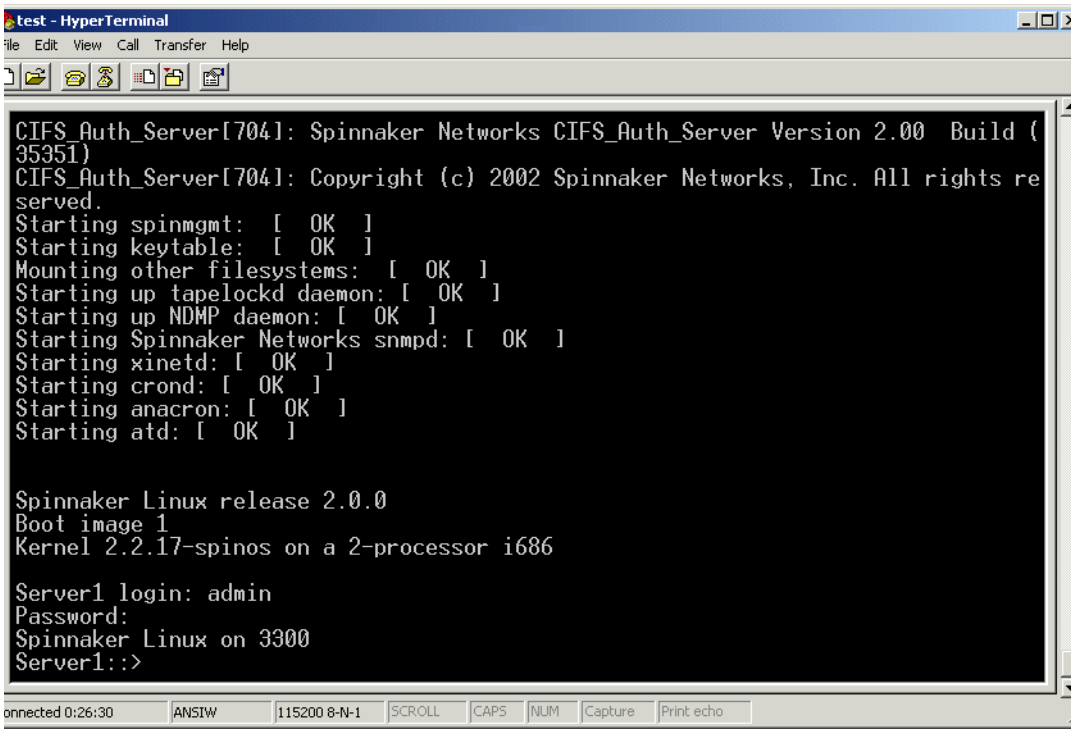


Figure 5-12. The SpinServer prompt after login

Note: The SpinServer shows local disk activity for several minutes as it mirrors the SCSI drives. This is normal behavior and does not interfere with the configuration or operation of the SpinServer system components.

Changing the Login Password

After initial installation of the SpinFS software on your SpinServer, the user password can be changed by following these steps:

1. Using secure shell (SSH), log in as root.
2. Using the **`passwd`** command (for example, **`passwd admin`**), enter your new password.
3. Enter the password a second time to confirm the password.

Changing the Root Password

After initial installation of the SpinFS software on your SpinServer, the root password can be changed by following these steps:

1. Using secure shell (SSH), log in as root.

2. Using the **`passwd`** command (for example, **`passwd root`**), enter your new password.

3. Enter the password a second time to confirm the password.

Installation of Software Upgrades

To install SpinFS software upgrades, see the *Release Notes* for installation instructions.

6.0 Configuring the SpinServer System

This chapter provides the steps needed to create a basic configuration of a SpinServer system, including the configuration of the SpinServer and the SpinStor 200. The steps provided in this chapter are examples; for more information on any of the steps, see the full description of each command in the following chapters.

The SpinManager CLI is used in the command examples. The SpinManager GUI has the same commands in the same hierarchy.

6.1 Management Software

To complete the administration tasks discussed in this chapter, use one of the options described in “Connecting to the SpinServer” on page 17.

6.2 Logging In

A console connected to the serial port (COM1) is needed to access the SpinManager CLI.

Note: Only the CLI is available to access the SpinManager when using a direct connection through the serial port.

With the console connected, press **Enter** twice. At the Login prompt, type **`admin`**. At the Password prompt, type **`admin`**.

After logging in, the prompt (*`SpinServerID`*`::>`) is displayed. You are ready to continue the configuration tasks for the SpinServer. For the examples in this guide, **`Server1`** is the SpinServer ID and is used in the SpinServer prompt in the command examples.

Note: To change the login password, see the chapter “Initializing the SpinServer” on page 21.

6.3 Configuring HTTPS Access

You can set up secure access to the SpinManager GUI by configuring HTTPS on the SpinServer system. By default, SpinFS Version 2.1 uses HTTPS for GUI connections.

By default, the SpinServer uses a self-signed Secure Sockets Layer (SSL) certificate. To use another certificate (for instance, a certificate from VeriSign), log onto the SpinServer as root and copy the certificate to `/var/spinmgmt/keys/spinmgmt.pem`. Contact Network Appliance Technical Support for assistance. The self-signed SSL certificate is valid for 365 days from installation.

Using HTTPS with Internet Explorer

The first time you use Internet Explorer to access the SpinManager GUI, the browser can generate the following warning: `Information you exchange with this site cannot be viewed or changed by others. However, there is a problem with the site's security certificate.` To install the certificate into Internet Explorer and avoid this message on subsequent sessions, perform the following steps:

1. Click **View Certificate** on the Security Alert dialog box. A new dialog box opens.
2. Click **Install Certificate**.
3. Complete the certificate installation with the defaults provided by the dialog box.
4. Click **OK** on the Security Alert dialog box.

Using HTTPS with Netscape or Mozilla

The first time you use Netscape or Mozilla to access the SpinManager GUI, the browser can generate the following warning: `Unable to verify the identity of` *`SpinServer`* `as a trusted site.` To install the certificate into the browser and avoid this message on subsequent sessions, perform the following steps:

1. On the dialog box, select **Accept this certificate permanently** and click **OK**.
2. The browser can generate the following message: `You have attempted to establish a connection with` *`SpinServer`*. This message is displayed only if the SpinServer's name is different from that contained in the certificate; for example, if the SpinServer's name is System1.lab but the name in the certificate is listed as System1. If this message appears, click **OK** on the dialog box.
3. The browser can generate the following warning: `You have requested an encrypted page that contains some unencrypted information.` To disable this warning, click **Edit → Preferences**, then open **Privacy and Security → SSL**. Under **SSL Warnings**, clear the **Viewing a page with an encrypted/unencrypted** mix check box and click **OK**.

6.4 Configuring the SpinServer

Command examples are given to illustrate the configuration process. For the command examples, the information obtained from a network administrator is given below.

Table 6-1. SpinServer Configuration Example

SpinServer ID	Server1
Management IP address (if not using DHCP)	
Management netmask (if not using DHCP)	
Management default gateway IP (if not using DHCP)	
Management primary name server (if not using DHCP)	
Server IP address 1	10.1.1.140
Server netmask 1	255.255.255.0
Server IP address 2 (optional)	
Server netmask 2 (optional)	
Cluster IP address 1	192.168.66.1
Cluster IP address 2 (optional)	192.168.66.2
Cluster netmask	255.255.0.0
Cluster broadcast	192.168.255.255

Displaying SpinServer Information

Before configuring the SpinServer, you can check the SpinServer identification information that was entered at the time the software was installed. To do this, use the **`spinservers show`** command.

To Display System SpinServers

To display information about the SpinServers in the system, use the following command.

```
Server1::> spinservers show
```

Initializing a SpinServer Cluster

Initialization of a SpinServer involves creating a new cluster or joining an existing cluster. Every SpinServer is part of a cluster, and although the cluster can contain only a single SpinServer, every SpinServer must be entered into a cluster before any other system management task is undertaken.

The **spinservers createcluster** and **spinservers joincluster** commands cause other system operations to take place. Both the **spinservers createcluster** and **spinservers joincluster** commands take the IP configuration for the cluster ports. This information is then stored in the interface configuration page after the node has completed the create or join cluster operation.

The **spinservers joincluster** command uses the cluster IP address of a SpinServer already in the cluster that you want another SpinServer to join. This SpinServer must be healthy and in active communication with the cluster. It can be any healthy host within the cluster.

Note: The **spinservers createcluster** and **spinservers joincluster** commands can take up to 30 seconds to complete.

To Initialize a SpinServer by Creating a Cluster

To initialize SpinServer Server1, use the **spinservers createcluster** command, as shown in this example.

```
Server1::> spinservers createcluster –ssid Server1 \
-clustername Cluster1 -ipaddress1 192.168.66.1 \
-ipaddress2 192.168.66.2 -netmask1 255.255.0.0 \
-broadcast1 192.168.255.255
```

where

Server1 is the ID given to the SpinServer.

Cluster1 is the name of the new cluster.

192.168.66.1 is the IP address of the first cluster interface.

192.168.66.2 is the IP address of the second cluster interface. It is optional.

255.255.0.0 is the netmask of the cluster network.

192.168.255.255 is the broadcast IP address. The broadcast IP address is needed only if the default is known to be insufficient.

For more information about this command, see “Creating a Cluster” on page 84.

To Initialize a SpinServer by Joining a Cluster

The **spinservers joincluster** command can be used to initialize additional SpinServers that you want to add to an existing cluster. The **spinservers joincluster** command is different from the **spinservers createcluster** command in that it uses the cluster IP address of a SpinServer already in the cluster.

Important: This command must not be used for any other purpose. If you mistakenly run the **spinservers createcluster** command instead of the **spinservers joincluster** command, you must reinstall the SpinServer (or completely delete all configuration data) to undo the operation. Before you reinstall the SpinServer or delete any configuration information, contact Network Appliance Technical Support.

```
Server1::> spinservers joincluster –ssid Server2 \
-clusternodeipaddress 192.168.66.1 -ipaddress1 \
192.168.66.3 -ipaddress2 192.168.66.4 -netmask1 \
255.255.0.0 -broadcast1 192.168.255.255
```

where

Server2 is the ID given to the SpinServer that is added to the cluster.

192.168.66.1 is the cluster IP address of a SpinServer already in the cluster that you want Server2 to join. This SpinServer must be healthy and in active communication with the cluster. It can be any healthy host within the cluster.

192.168.66.3 is the IP address of the first cluster interface.

192.168.66.4 is the IP address of the second cluster interface. It is optional.

255.255.0.0 is the netmask.

192.168.255.255 is the broadcast IP address. The broadcast IP address is changed only if it is known to be insufficient.

For more information about this command, see “Joining a Cluster” on page 84.

Note: After running the **spinservers joincluster** or **spinservers createcluster** command, you can use the **spinservers modify** command to further identify the SpinServer by entering this additional information: owner, contact information, asset tag, and location of the SpinServer. For additional information about this command, see “Modifying a SpinServer” on page 249.

6.5 Setting Up Virtual Interfaces

A virtual interface is named by the user and associated with a network port. You need to give some thought to the names you choose so that when other network virtual interfaces are added to your system, the names are orderly and recognizable. See your network administrator if you have questions about the naming scheme to be used in the SpinServer system.

The **`if configuration new`** command creates an interface and identifies it by its unique SpinServer ID and interface name.

To Create and Configure Virtual Interfaces

Use the **`if configuration new`** command to create a virtual interface, as shown in the following example.

```
Server1::> if configuration new -ssid Server1 -ifname \
if0 -netportname serv0 -ipaddress 10.1.1.140 -netmask \
255.255.255.0
```

where

`Server1` is the ID given to the SpinServer.

`if0` is the name of the new interface.

`serv0` is the name of the physical port to which the new virtual interface is bound.

`10.1.1.140` is the IP address you assigned to the new interface.

`255.255.255.0` is the netmask you assigned to the new interface.

For more information about this command, see “Creating and Configuring a Virtual Interface” on page 92.

To Configure a Failover Port for High Availability

If you have purchased the optional SpinHA software, perform the following steps to configure the SpinServer system for high availability (HA):

Use the **`if configuration addfailoverport`** command to set Server1’s virtual interface to fail over to Server2.

Failover ports are critical components of HA configurations. Use the **`if configuration addfailoverport`** command to set Server1’s virtual interface to fail over to Server2.

```
Server1::> if configuration addfailoverport -ssid Server1 \
-ifname if0 -ssidto Server2 -netportnameto serv0
```

where

`Server1` is the ID given to the SpinServer.

`if0` is the name of the virtual interface on Server1.

`Server2` is the name of the SpinServer with the failover port.

`serv0` is the network port that is designated as a failover port on Server2.

For more information about this command, see "Adding a Failover Port" on page 96. For more information about HA configurations, see *SpinFS Architecture Basic Concepts Guide*.

6.6 Configuring RAID Components

All of the disks in a single RAID enclosure, along with the disks in any attached expansion enclosure, are controlled by the pair of RAID controllers in the RAID enclosure. This pair of RAID controllers and their attached disks form a single *RAID subsystem*. The SpinServer has as many RAID subsystems as it has SpinStor 200 RAID enclosures.

Each RAID subsystem is configured individually. By using the management software, you can identify which disks belong to which RAID subsystem, and the disks' states. Normally, individual disks can be part of a RAID set, be used as a hot spare, or be uncommitted. A disk cannot belong to more than one RAID set.

RAID subsystems must be configured with individual disks of the same speed, but can use individual disks of different capacities. However, if you mix disks of different capacities in a RAID set, the SpinServer uses all disks drives in the RAID set as having the same capacity as the smallest disk drive in the RAID set. The remaining disk drive capacity is not used.

Important: This section describes the configuration of RAID components when SpinStor 200 RAID enclosures are used. SpinStor 200 enclosures are used only with legacy SpinServer systems. For information on configuring the RAID components of a new SpinServer system that uses FAS270c RAID enclosures, see the Data ONTAP documentation, including the *Block Access Management Guide for FCP* and *Storage Management Guide*.

Note: When using disks of different sizes in the same RAID set, any disk designated as a hot spare must be of the larger disk size.

Note: When replacing a disk drive with a disk drive of a larger capacity, the **`raid diskdrive show`** command displays the capacity of the original disk drive instead of the new larger disk drive size until the system is rebooted.

RAID Levels

The SpinServer supports the following RAID levels:

- RAID 0—This level supports a striped disk array, but without fault tolerance. A RAID 0 RAID set must include at least two disks.
- RAID 1—This level supports mirroring. A RAID 1 RAID set must include only two disks.
- RAID 3—This level supports parallel data transfer with parity. A RAID 3 RAID set must include at least three disks.
- RAID 5—This level supports independent data disks with distributed parity blocks. A RAID 5 RAID set must include at least three disks.
- RAID 0+1—This level supports high performance of data transfer. A RAID 0+1 RAID set must include at least four disks.

All RAID levels combine a set of disks to form a RAID set. A RAID set is a group of disks that work together to form a *logical disk* that is larger, faster, and more reliable than the individual disks. After a RAID set has been created, it can be configured as one large, fast disk for the purpose of setting up file systems.

With a hot spare disk available, the RAID controller can reconstruct the data that is stored on a failed disk by placing the data on the hot spare. After this rebuilding process is complete, the RAID set regains its performance and reliability.

Note: If a hot spare is not available, the RAID set cannot be rebuilt until the failed disk is replaced with a new disk.

There can be more than one hot spare disk in a RAID subsystem. In this case, when a hot spare is needed to rebuild a RAID set, the choice is arbitrary as to which hot spare disk is used. The hot spare does not need to be physically located near the rest of the RAID set.

An uncommitted disk is available to be combined with other uncommitted disks to form a RAID set or to be designated as a hot spare.

Reviewing the Disk Configuration

If your SpinServer system includes RAID sets that were preconfigured at the factory, they are ready for you to use to create storage pools and virtual file systems. If the RAID sets were not preconfigured, you must configure and initialize the RAID sets.

Before proceeding with the configuration and initialization of the RAID sets, the current disk configuration must be reviewed. To review the disk configuration, use the **`raid diskdrives show`** command. Each of the recognized disks is identified in the Device field by its RAID subsystem (**`r`***n*, where **`r`** stands for RAID subsystem) and its disk ID (**`d`***n*, where **`d`** stands for disk). See “Displaying System Disk Information” on page 119 for more information about this command.

If the RAID sets and disks are already configured as needed for your system, then no more RAID configuration is necessary. However, if your system requires additional RAID sets, they must be configured and initialized as described in the following section.

RAID Configuration

Before using a RAID enclosure, confirm that it is in the expected configuration by checking the following:

- Determine that the target numbers of all of the disks match the target numbers set on the disk enclosure. Use the following formula to determine the target numbers:
  Disk ID = 2 x target number + default channel number
  To review the target numbers of the disk drives, use the **`raid diskdrives show`** command. For additional information about setting the target numbers, see the *Installation Guide*.
- Determine that several disks in each enclosure can be correctly located by using the **`raid diskdrives locate`** command. See “Locating a Disk Drive” on page 119 for information.
- Determine that RAID sets can be correctly located as they are created by using the **`raid sets locate`** command. See “Locating a RAID Set” on page 108 for information.
- Determine that the enclosure’s temperature, power supply, and blower monitoring functions are working by observing the Fault LEDs on the enclosure and by using the **`raid enclosures show`** command. See “Locating an Enclosure” on page 124 for information.
- Determine that half of the disks in each enclosure are using Channel 0 and the rest are using Channel 1 by using the **`raid diskdrives show`** command. See “Displaying System Disk Information” on page 119 for information.
- Determine that all of the RAID controllers’ status is `Active` by using the **`raid controllers show`** command. `Critical` status is not acceptable during initial configuration. See “Displaying RAID Controller Information” on page 125 for more information.

Creating RAID Sets

You must create RAID sets before you create storage pools.

The device identification for a RAID set is in the format **r***subsystem_number***s***raidset_number*, where *subsystem_number* indicates the RAID subsystem and *raidset_number* indicates the specific RAID set.

If the SpinServer system does not include RAID sets that were preconfigured at the factory or if you are adding a new RAID enclosure to the system, your first step is to create a RAID set. The RAID set will be online after you create it because you designate its status as online.

Note: When the first RAID set is created on a RAID subsystem, the creation process can take as long as two minutes. The process takes this long to create the first RAID set on a RAID subsystem because a reset command is issued to the RAID controllers after the creation process is completed. The reset is necessary because the strip size and cache line size are updated from the previous settings. This type of change requires the RAID controllers to be reset. If the RAID set on a RAID subsystem is not the first one created, the creation process is completed in a shorter time period.

To Create and Configure a RAID Set

To create a RAID set, use the following command with a list of the disks that you want to be included in the RAID set.

```
Server1::> raid sets new -raidlevel 5 -stripsize 16 \
-writecacheenabled false -status Online -disks [r0d1, \
r0d2, r0d5, r0d6, r0d9, r0d10, r0d13, r0d14, r0d17, \
r0d18, r0d21 r0d22, r0d25, r0d26, r0d29]
```

where

5 is the RAID level.

16 is the RAID strip size in KB. The default setting is **8** KB. Other setting options are **16** KB, **32** KB, or **64** KB. A RAID strip is a piece of a single disk that is used for data storage. A RAID stripe is a set of the corresponding strips from all of the disks in a RAID set. The RAID strip size must match the RAID strip size that is currently set on the RAID controller.

false disables the write cache. This is the default setting for the write cache. Enabling the write cache can result in higher system performance but can cause the loss of data if an error occurs.

Online is the status of the RAID set.

[r0d1, r0d2, r0d5, r0d6, r0d9, r0d10, r0d13, r0d14, r0d17, r0d18, r0d21 r0d22, r0d25, r0d26, r0d29] are the disk IDs of the disks to be included in the RAID set.

Note: If you make mistakes in the configuration of a RAID set, you must delete the RAID set. RAID sets cannot be deleted randomly. The RAID set created most recently must be deleted first.

See “Creating and Configuring a RAID Set” on page 105 for more information about this command.

Initializing a RAID Set

After RAID sets are created and configured, initialization of the RAID set begins as a background process upon first use of the RAID set. For example, if you create storage pools and assign storage units (RAID sets) to the storage pools, background initialization automatically starts.

As background initialization proceeds, you can use one of the storage pools to create a virtual server. After the virtual server is created, you create a virtual interface, specifying an appropriate IP address and netmask. After creating and mounting a virtual file system, you can write data to the RAID set as background initialization continues.

Note: Background initialization can be a lengthy process. It can take about 1 hour for every 70 GB of disk capacity. For example, initializing a 240-GB RAID set can take about three and one-half hours. However, as mentioned above, you can use this RAID set while the initialization is in progress.

See “Starting Background Initialization” on page 109 for more information about background initialization. Alternatively, for information about foreground initialization, see “Initializing a RAID Set” on page 111.

Configuring a Hot Spare

The **`raid diskdrives makehotspare`** command identifies a given disk by its disk ID and designates its status as a hot spare in your RAID subsystem. A hot spare is a replacement disk that can be used to rebuild all user data from a failed disk in the RAID subsystem. Use the command to create as many hot spares as the RAID subsystem requires.

Note: Disks configured as hot spares can protect RAID sets only on the same RAID subsystem.

To Create a Hot Spare

To create a hot spare for a RAID subsystem, use the following command:

```
Server1::> raid diskdrives makehotspare -deviceid r0d26
```

where

`r0d26` is the disk ID for the disk designated as a hot spare in the RAID subsystem **`r0`**.

See “Creating a Hot Spare Disk” on page 120 for more information about this command.

Verifying Disk Setup

After setting up your disk enclosure, use the **`raid diskdrives show`** and **`raid sets show`** commands to check that all of the disks and RAID sets are correctly configured and initialized. If the configuration is incorrect, you can use **`raid`** commands to modify the configuration. See “RAID Management Tasks” on page 104 for more information.

6.7 Creating a Storage Pool

A *storage pool* is a logical grouping of physical storage. Storage pools are comprised of one or more storage units. A storage pool can be comprised of 1 to 128 storage units. A single storage unit represents a single external RAID set or a single JBOD disk, providing a much larger block address space than is possible with only one RAID set or one JBOD disk. The functional purpose of a storage pool is to serve as a logical container for virtual file systems (VFSs).

If you have purchased the optional SpinHA software and have configured your system for high availability, each storage pool is served by one primary SpinServer and one or more secondary servers so that the SpinServer system has no single point of failure. The primary SpinServer for a storage pool controls the read and write operations for the storage pool. If the primary SpinServer fails, one of the secondary SpinServers assumes control of the storage pool.

The SpinServer on which the storage pool is created becomes the primary SpinServer for that storage pool.

Before setting up a storage pool, use the **`storage units show`** command to obtain the storage unit names assigned by the system. This command provides a view of the external RAID sets or JBOD disks as seen from the Fibre Channel host bus adapters (HBAs).

To Display Storage Units

The following command example displays configuration information about all storage units. After identification of the storage units has been made, the following information is displayed: device ID, World Wide name, logical unit number (LUN), file system device ID, channel status, primary loop ID, secondary host channel status, secondary host channel, secondary loop ID, RAID set ID, type, vendor, product, revision number, status (online, offline, or unused), damaged status (true or false), error status (true or false), last error code, error description, and storage space size (in GB) on the storage unit.

The following command example displays information about the storage units attached to all SpinServers in a cluster.

```
Server1::> storage units show
```

See “Displaying Storage Units” on page 135 for more information.

To Create a Storage Pool

A storage pool is named by the user. Any assigned name can be chosen as long as it is unique within a cluster. To create a storage pool, use the **storage pools new** command, as in the following example. Depending on the number of physical disks that are included in the RAID sets that are being used, this command can take several minutes to complete.

```
Server1::> storage pools new -deviceid sp0 -layout \
stripe -freespacealarm 10 -sunits [su0,su1,su2]
```

where

sp0 is the name or ID you assigned to the storage pool you created.

stripe is the configuration of the storage pool.

10 specifies the percentage of free space remaining on the storage pool before the free-space alarm is triggered.

[su0,su1,su2] is the list of storage units (RAID sets) assigned to the storage pool you created.

See “Creating a Storage Pool” on page 138 for more information.

To Configure a Storage Pool for High Availability

After creating a new storage pool, for HA, you must designate a passive secondary SpinServer for the storage pool and then promote that passive secondary to an active secondary. See “High Availability” on page 134 for general information about high availability.

To Designate a Passive Secondary SpinServer for HA

Add Server2 as a passive secondary by using the **storage pools addpassivesecondary** command, as in the following example

```
Server1::> storage pools addpassivesecondary -deviceid \
sp0 -passivesec Server2 -priority 1
```

where

sp0 is the name or ID you assigned to the storage pool you created.

Server2 is the SpinServer to be added as a passive secondary server for the storage pool.

1 is the takeover priority of the passive secondary server.

See “Adding a Passive Secondary to a Storage Pool” on page 143 for more information.

To Promote a Passive Secondary SpinServer to an Active Secondary for HA

Promote **Server2** as an active secondary by using the **storage pools promotepassivesecondary** command.

```
Server1::> storage pools promotepassivesecondary \
-deviceid sp0 -passivesec Server2
```

where

sp0 is the name or ID you assigned to the storage pool you created.

Server2 is the SpinServer to be promoted as an active secondary server for the storage pool.

Note: If the number of active secondary servers for the storage pool is set to greater than zero, the first passive secondary server that was created is automatically promoted to active secondary.

See "Promoting a Passive Secondary to Active Secondary" on page 144 for more information.

6.8 Setting Up Domains

Before creating a virtual server or VFS, you can define Network Information Service (NIS) or CIFS domains. Defining NIS or CIFS domains at this point in the configuration process provides you with the opportunity to specify the ownership of the new VFS by name rather than by number. Also, CIFS domains are required for the CIFS protocol, and NIS domains are required for performing mapping of CIFS users to NFS users, and for NFS access via Kerberos.

Note: Domain configuration can be omitted if your site does not use NIS or CIFS, or if you would prefer to do the domain configuration later.

Specifying NIS Domains

NIS is widely used to provide simple name network lookup service and user authentication service. Its purpose is to provide information that needs to be available to the entire network, such as login names, passwords, home directories, and group information.

The SpinServer system's virtual servers can use NIS to translate NFS user IDs to names and vice versa. This process enables you to specify the owner of a VFS by name (rather than ID), and can be used to support mapping of CIFS users to NFS users. Your first task is to specify the NIS domains to be used.

Use the **spinaccess nfs domains new** command to specify the NIS domains to be used by identifying their domain names and NIS servers to be used for the domain. The NIS server option for this command specifies a comma-separated list of NIS servers that handle NIS requests for the specified domain. If the option is omitted, a broadcast search is used to look for a NIS server for the domain.

Note: To configure a NIS domain at this point, it is assumed that the NIS server's IP address is on the same subnet as the management port (mgmt0). If the NIS server's subnet is the same as the subnet connected to the network ports (serv0 and serv1), then a virtual server and a bound virtual interface must first be configured for the subnet. See "Creating a Virtual Server" on page 49 and "Binding Network Virtual Interfaces" on page 49 for details.

To Specify NIS Domains

Use the following command to specify a NIS domain.

```
spinserver::> spinaccess nfs domains new -domainname \
acmebricks.com -nisservers [10.2.2.1, 10.3.3.1]
```

where

acmebricks.com is the NIS domain name.

[10.2.2.1, 10.3.3.1] is the square-bracketed, comma-separated list of NIS servers.

See "Specifying NIS Domains" on page 197 for more information.

Defining a CIFS Domain

To define the CIFS domain, use the **spinaccess cifs domain new** command. CIFS virtual servers, which are created later in this configuration process, must exist within a CIFS domain as defined here. By configuring the CIFS domains, you inform the SpinServer of the Windows NT and Windows 2000 domains that exist on the network. You must enter every CIFS domain on the network that users use to access the SpinServer.

Note: Domain names, domain controller names, and NetBIOS names can be up to 15 characters long. The SpinServer does not accept the fully qualified DNS name style that Windows 2000 supports. CIFS domain names are converted to all upper case when they are created.

Note: To successfully register a new domain with the SpinServer, the SpinServer must be able to contact the primary domain controller (PDC) that is specified by the **-pdcname** and **-pdcip** parameters in the command. The virtual interface that is bound to the virtual server that you plan to add to the new domain must already be running and have access to the PDC.

Note: To configure a CIFS domain at this point, it is assumed that the PDC's IP address is on the same subnet as the management port (mgmt0). If the PDC's subnet is the same as the subnet connected to the network ports (serv0 and serv1), then a virtual server and a bound virtual interface must first be configured for the subnet. See "Creating a Virtual Server" on page 49 and "Binding Network Virtual Interfaces" on page 49 for details.

To Define a CIFS Domain

To define a CIFS domain, use the **`spinaccess cifs domains new`** command.

```
Server1::> spinaccess cifs domains new -domainname ENG \
-winsip 10.1.1.116 -pdcname STORM -pdcip 10.1.1.116 \
-bdcname1 RAIN -bdcip1 10.1.1.118
```

where

`ENG` is the name of the CIFS domain.

`10.1.1.116` is the IP address of the domain's WINS server.

`STORM` is the name of the domain's primary domain controller (PDC).

`10.1.1.116` is the IP address identifying the PDC.

`RAIN` is the name of the first backup domain controller (BDC).

`10.1.1.118` is the IP address of the first BDC.

Note: If specifying BDCs, the fields with the lowest values must be filled in first. For example, if you have one BDC, fill in **`-bdcname1`** and **`-bdcip1`**. If you have two, fill in **`-bdcname1`**, **`-bdcip1`**, **`-bdcname2`**, and **`-bdcip2`**. All must be filled in before specifying a third backup domain controller.

See "Creating a CIFS Domain" on page 179 for more information.

6.9 Creating Virtual Servers

A virtual server represents a separate file system namespace with separate network access. When a virtual server is created, its root VFS is automatically created. To create a virtual server, use the **`vservers servers new`** command.

For more information about creating virtual servers, see "Creating a Virtual Server" on page 147.

Creating a Virtual Server

The following command example creates a virtual server.

To Create a Virtual Server

```
Server1::> vservers servers new -vsname vserver1 \
-spdevid sp1 -spinauth false -floatingsuperuser false \
-minauth none -rulesetname default_ruleset -showspinshot \
true -restrictedacl false
```

where

vserver1 is the name or ID of the virtual server to be created.

sp1 is the name or ID you assigned to the storage pool in which the new virtual server's root VFS is to be created.

false specifies that the SpinAuth authentication mechanism is not to be used on this virtual server.

false specifies that a floating superuser is not permitted on this virtual server.

none is the minimum authentication for the virtual server. Possible values are **none**, **cifs-ntlm**, **cifs-kerberos**, and **nfs-kerberos** (**kerberos** is accepted as a synonym for **nfs-kerberos**).

default_ruleset specifies the name of the export rule set that is to be applied to this virtual server.

true specifies that the `.spinshot` directory is visible to clients on all VFSs created on this virtual server.

false specifies that CIFS access control lists (ACLs) are used on all VFSs created on this virtual server.

6.10 Binding Network Virtual Interfaces

After setting up a virtual server, you must bind one or more network virtual interfaces to the virtual server. To bind a network virtual interface to the virtual server, use the **vservers servers bindif** command. The **vservers servers bindif** command identifies a given interface by its SpinServer and interface name and binds the network virtual interface to your virtual server.

Note: As detailed earlier, the network virtual interfaces specified here must have been previously created.

To Bind a Network Virtual Interface

The following command example identifies the interface by its SpinServer and the interface name you assigned to it, and binds the network virtual interface to the designated virtual server.

```
Server1::> vservers servers bindif -vsname vserver0 \
-ssid Server1 -ifname if0
```

where

vserver0 is the name or ID of the virtual server to which the network virtual interface named **if0** is bound.

Server1 is the ID given to the SpinServer.

if0 is the name or ID of the network virtual interface.

See “Binding and Unbinding Virtual Interfaces” on page 149 for more information.

6.11 Setting Up Kerberos Authentication

This section provides instructions for setting up Kerberos for NFS or CIFS.

Setting Up Kerberos Authentication for NFS

To enable the SpinServer to use Kerberos authentication for the NFS protocol, complete the following tasks:

1. Define the NIS domain in which Kerberos users are defined. This task is described in “Specifying NIS Domains” on page 46.

2. Define the Kerberos realm or realms that are to be used to control access to the SpinServer.

3. Create the virtual server. This task is described in “Creating a Virtual Server” on page 49.

4. Define a set of regular expressions that are to be used to map Kerberos client principal names to NIS user names.

5. Define the Service Principal Names (SPNs) to be used by each virtual interface that is to be enabled for Kerberos authentication.

To complete these tasks, you must obtain configuration information from your network and Kerberos administrator or administrators. Use the following table to enter the information that is needed for this process.

NIS domain name	

For each Kerberos realm, enter the following information.

Realm name	
KDC vendor	
KDC IP address	
KDC port	
Admin IP address	
Admin port	
kpasswd IP address	
kpasswd port	
Active Directory server IP address	
Administrator’s user name	
Administrator’s password	

For each virtual interface, enter the following information.

Service Principal Name	

Setting Up Kerberos

To use Kerberos on a SpinServer, you must configure a Kerberos realm on the SpinServer by using the **spinaccess kerberos realm new** command.

To define a Kerberos realm, complete the following command.

```
Server1::> spinaccess kerberos realm new -configname eng \
-realm ENG.EXAMPLE.COM -kdc_vendor Other -kdc 10.1.4.55 \
-kdcport 88 -adminaddr 10.1.4.55 -adminport 749 \
-passwdaddr 10.1.4.55 -passwdport 464 -clockskew 300
```

where

eng is the configuration name that SpinManager uses to refer to this realm.

ENG.EXAMPLE.COM is the Kerberos realm name. Kerberos realm names are case sensitive.

Other is the Kerberos KDC vendor. Possible values for this option are **Microsoft** and **Other**. Use **Microsoft** only for Kerberos realms served by Microsoft Windows 2000-compatible Active Directory-based realms. Note that CIFS can be used only with **Microsoft** realms, while NFS can be used with either type.

10.1.4.55 is the address of the key distribution center (KDC), which, in this example, is also the admin server and kpasswd server.

88 is the KDC port number. (This is the default value.)

749 is the admin port number. (This is the default value.)

464 is the kpasswd port number. (This is the default value.)

300 is the clock skew, in seconds, permitted between clients and the server. (This is the default value.)

For more information, see "Creating a Kerberos Realm Configuration" on page 210.

Setting Up Kerberos-NIS User Mappings

Kerberos clients accessing files on a SpinServer via NFS must have their identities mapped to numeric IDs. This mapping is done by first applying a set of regular expressions to the clients' Kerberos principal names, and then by looking up the results of those regular expressions in NIS.

To configure this mapping, you must first associate the NIS domain and Kerberos realms with the virtual server, and then define the set of regular expressions to be used in the mapping.

Note: For further information about regular expressions, see *Mastering Regular Expressions* published by O'Reilly and Associates, Inc. at http://www.oreilly.com/catalog/regex. Additional information can be obtained from a UNIX prompt by running **info regex**. For further information and examples, run the command **man 7 regex**.

To Associate a NIS Domain with a Virtual Server

To associate a NIS domain with a virtual server, use the **spinaccess nfs servers new** command, as in the following example.

```
Server1::> spinaccess nfs servers new -vsname fin_svr \
-domainname example.com -kerbrealms [eng, finance]
```

where

fin_svr is the name or ID of the virtual server to be associated with the NIS domain and Kerberos realms.

example.com is the name of the NIS domain to be used for NIS lookups for this virtual server. This must be a domain that has been previously configured by using the **spinaccess nfs domains** commands, as described in "Specifying NIS Domains" on page 46.

[eng, finance] is the list of Kerberos realms that is visible to this virtual server. Multiple values must be separated by commas and enclosed in square brackets, as shown here. Kerberos realms in this list must have been previously configured by using the **spinaccess kerberos realm** commands, as described previously.

For more information, see "Configuring an NFS Server's NIS Domains" on page 198.

To Set Up Kerberos-NIS User Mappings

To set up Kerberos-NIS user mappings, use the **spinaccess kerb-nis new** command, as in the following examples.

```
Server1::> spinaccess kerb-nis new -vsname fin_svr \
-clientpattern "[^@]+@ENG.EXAMPLE.COM" \
-clientpatternreplace eng_guest

Server1::> spinaccess kerb-nis new -vsname fin_svr \
-clientpattern "([^@/]+)(/[^@]+)?@FINANCE.EXAMPLE.COM" \
-clientpatternreplace \1
```

where

fin_svr is the name or ID of the virtual server for which these mappings are to be defined.

[^@]+@ENG.EXAMPLE.COM is the regular expression to be matched in the first mapping. (This regular expression matches any principal in the Kerberos realm **ENG.EXAMPLE.COM**.)

eng_guest is the replacement to be used when the pattern is matched. This first mapping, therefore, maps any principal in the Kerberos realm **ENG.EXAMPLE.COM** to the NIS user **eng_guest**.

([^@/]+)(/[^@]+)?@FINANCE.EXAMPLE.COM is the regular expression to be matched in the second mapping. (This regular expression matches any principal in the Kerberos realm **FINANCE.EXAMPLE.COM**, and groups any set of characters up to the first / character.)

\1 is the replacement to be used when the pattern is matched. The notation **\1** means to replace the pattern with the first grouped set of matched characters; in this case any characters from the original client principal name before the first / character. This second mapping, therefore, maps any principal in the Kerberos realm **FINANCE.EXAMPLE.COM** to the NIS user corresponding to the principal's base name with any instance names removed. (For example, **anne/admin@FINANCE.EXAMPLE.COM** becomes **anne**.)

For more information, see "Creating a Kerberos-NIS Mapping" on page 213.

Defining NFS Service Principal Names

A virtual interface that is to use Kerberos authentication for NFS must have a service principal name (SPN). NFS SPNs are required to be of the form **nfs/***hostname*@*REALM*, where *hostname* is the DNS name that clients use to mount file systems on this virtual interface (that is, the DNS name associated with this virtual interface's IP address), and *REALM* is the Kerberos realm name of the virtual interface.

To assign an SPN to a virtual interface for use with NFS, use the **spinaccess kerberos ifconfig enablekerberos** command, as in the following example.

```
Server1::> spinaccess kerberos ifconfig enablekerberos \
-ssid Server1 -ifname vif2 \
-spn nfs/spin1.eng.example.com@ENG.EXAMPLE.COM \
-uname tom/admin -passwd s3cre!
```

where

Server1 is the name or ID of the SpinServer on which the virtual interface is defined.

vif2 is the name or ID of the virtual interface whose SPN is to be assigned. This virtual interface must be one that has already been bound to a virtual server.

nfs/spin1.eng.example.com@ENG.EXAMPLE.COM is the Service Principal Name (SPN) for this virtual interface. Service Principal Names are case sensitive. This name must include a realm part, and it must be in a realm that is visible to the virtual server to which this virtual interface has been bound. If the principal does not already exist, it will be created; if it does already exist, its service key will be randomized.

tom/admin is the Kerberos user name of a principal with privileges to create a new principal or randomize its key in the SPN's realm. You can give a name that includes a realm part here, but if you do, it must be the same realm as the SPN's realm. If you omit the realm part (as shown here), the SPN's realm will be used.

s3cre! is the password for the **tom/admin@EXAMPLE.COM** account. Note that, if you are using the CLI, the password is echoed here in plain text; to mask this password as it is entered, perform this step from the GUI.

The administrative user name and password entered in this step are not stored; they are used to create the principal named by the SPN (or to randomize its key) and are then discarded.

If you use a single DNS name to refer to multiple IP addresses, you can use the same SPN for multiple virtual interfaces. If you do, the key is not rerandomized for each virtual interface, and the administrator's user name and password can be omitted when setting SPNs for the second and subsequent virtual interfaces.

For more information, see "Enabling Kerberos on an Interface" on page 215.

Setting Up Kerberos Authentication for CIFS

To enable the SpinServer to use Kerberos authentication for the CIFS protocol, complete the following tasks:

1. Define CIFS domain settings for the target domain. This task is described in “Defining a CIFS Domain” on page 47.

2. Define the Kerberos realm or realms that are to be used to control access to the SpinServer.

3. Set up a CIFS domain for Kerberos authentication and account management.

4. Create a CIFS virtual server in the target domain or realm.

To complete these tasks, you must obtain configuration information from your network and Kerberos administrator or administrators. Use the following table to enter the information that is needed for this process.

CIFS domain name	

For each Kerberos realm, enter the following information.

Realm name*	
KDC vendor	
KDC IP address	
KDC port	
Admin IP address	
Admin port	
kpasswd IP address	
kpasswd port	
Active Directory server IP address	
Administrator’s user name	
Administrator’s password*	

For each CIFS virtual server, enter the following information:

NetBIOS host name	

*This parameter is case sensitive and must be entered accordingly. The CIFS realm name must always be typed in full uppercase, as it is in the previous examples.

Defining a Kerberos Realm

To use Kerberos on a SpinServer, you must inform the SpinServer about the configuration details of the Kerberos realm. To do this, use the **spinaccess kerberos realm new** command.

To define a Kerberos realm, complete the following command.

```
Server1::> spinaccess kerberos realm new -configname \
mykerbrealm -realm MYDOMAIN.COM -kdc_vendor Microsoft \
-kdc 10.10.10.0 -kdcport 88 -adminaddr 10.10.10.0 \
-adminport 749 -passwdaddr 10.10.10.0 -passwdport 464 \
-clockskew 300
```

where

mykerbrealm is the configuration name that SpinManager uses to refer to this realm.

MYDOMAIN.COM is the Kerberos realm name. Kerberos realm names are case sensitive.

Microsoft is the Kerberos KDC vendor. Possible values for this option are **Microsoft** and **Other**. Use **Microsoft** only for Kerberos realms served by Microsoft Windows 2000 compatible Active Directory-based realms. Note that CIFS can be used only with **Microsoft** realms, while NFS can be used with either type.

10.10.10.0 is the address of the KDC which, in this example, is also the admin server and kpasswd server.

88 is the KDC port number. (This is the default value.)

749 is the admin port number. (This is the default value.)

464 is the kpasswd port number. (This is the default value.)

-clockskew 300 is an optional parameter that the defines the amount of time, in seconds, that the clocks on Kerberos clients can differ from the SpinServer. If the clocks differ by more than this amount, clients fail Kerberos preauthentication. (This is the default value.)

For more information, see “Creating a Kerberos Realm Configuration” on page 210.

Setting Up a New CIFS Domain for Kerberos Authentication

Windows 2000 clients that participate in the Windows Domain Security Model can take advantage of Kerberos-style authentication. A SpinServer can authenticate clients that attempt to use its resources by using the Kerberos tickets that are presented by the client when attempting to access resources that are shared by the SpinServer. To accomplish this, the SpinServer must know the Kerberos realm and the Windows domain to which it belongs. These values are often the same.

To create a Windows domain that uses Kerberos authentication, complete the following command:

```
Server1::> spinaccess cifs domains new -domainname \
MYDOMAIN -winsip 10.10.10.0 -pdcname mypdc -pdcip \
10.10.10.0 -kerbconf mykerbrealm
```

where

MYDOMAIN is the CIFS domain to be modified with Kerberos configuration information.

10.10.10.0 is the (optional) address for a WINS server in the target domain.

mypdc is the name of the Primary Domain Controller in the target domain.

10.10.10.0 is the IP address of the Primary Domain Controller in the target domain.

mykerbrealm specifies the Kerberos configuration that is to be associated with the CIFS domain. You must specify a Kerberos configuration whose KDC vendor type is Microsoft.

For more information, see “Creating a CIFS Domain” on page 179.

Specifying an Existing CIFS Domain for Kerberos Authentication

To specify that an existing Windows domain use Kerberos-style authentication by binding a Kerberos realm to the CIFS domain, complete this command:

```
Server1::> spinaccess cifs domains modify –domainname \
MYDOMAIN –kerbconf mykerbrealm
```

where

MYDOMAIN is the CIFS domain to be modified with Kerberos configuration information

mykerbrealm specifies the Kerberos configuration that is to be associated with the CIFS domain. You must specify a Kerberos configuration whose KDC vendor type is **Microsoft**.

For more information, see “Modifying a CIFS Domain” on page 180.

Creating a Kerberos-Enabled CIFS Virtual Server

A CIFS virtual server must belong to a domain. If the domain to which it belongs has a Kerberos realm bound to it (see previous step), the virtual server can optionally participate and use Kerberos to authenticate clients. In addition, down-level clients (clients that are not capable of Kerberos authentication) can still authenticate themselves, as long as the Windows Domain Security Policy for the CIFS domain has been set appropriately and does not restrict authentication.

To create a CIFS virtual server that uses Kerberos to authenticate clients, complete the following command:

```
Server1::> spinaccess cifs servers new -vsname myvs0 \
-nbname MYCIFSSRV0 -domainname MYDOMAIN –visibledomains \
[MYDOMAIN, MYOTHERDOMAIN] -nodetype H -usekerberos Yes \
-kerbprivuser Administrator -kerbprivpass secret
```

where

myvs0 is the name of an existing virtual server that is to be enabled for CIFS access

MYCIFSSRV0 is the NetBIOS name for the virtual server that is advertised on the local network. NetBIOS names must be unique on their subnet. Another host on any subnet in which the virtual server has a virtual interface must not use this name. The name is advertised in all subnets. A name conflict in any of them is difficult to discover.

MYDOMAIN is the domain to which the virtual server belongs. The domain that is specified with this command must already have been created, and must be enabled for Kerberos access. See “Defining a CIFS Domain” on page 47.

[MYDOMAIN, MYOTHERDOMAIN] is a square-bracketed, comma-separated list of domains that are considered trusted domains that enable users from those domains to be authenticated and granted access to the SpinServer. Users from domains not listed in the visible domains are not granted access to the SpinServer, even though those domains have a Windows trust relationship with one another.

H is the NetBIOS node type, which is a way of specifying how the CIFS virtual server announces itself to hosts on the subnet. For more information see RFCs 1000 and 1001. Other possible values for the NetBIOS node type are **B**, **P**, and **M**.

Yes specifies that Kerberos is used to authenticate users. See the previous section for details on how to create a Kerberos realm.

Administrator is the name of an account that has sufficient privileges to create Service Principal Accounts in the Active Directory for the domain. This account need not be the administrator for the domain, but often this account is the only account that has been granted the required privilege.

secret is the password for the account specified by the value of the **-kerbprivuser** parameter. The password is echoed in clear text to the CLI console; it is therefore recommended that you perform the entire CIFS virtual server creation step from the GUI. The GUI masks the password as it is entered.

For more information, see "Creating and Configuring a CIFS Server" on page 182.

6.12 Setting Up Export Rule Sets

This section provides instructions on setting up export rules and rule sets. Export rules and rule sets enable you to require different levels of authentication based on client IP addresses.

The following steps are involved in setting up export rule sets:

1. Create an export rule set, as described in "Creating an Export Rule Set" on page 59.
2. Add rules to the rule set, as described in "Adding a Rule to a Rule Set" on page 60.
3. Configure one or more VFSs to use the rule set, as described in "Setting Up a Virtual File System" on page 61.

Creating an Export Rule Set

An export rule set holds export rules; each rule set can hold approximately 8,186 rules. Each VFS is associated with exactly one rule set. Each rule set is identified by a unique name and a unique numeric ID. A SpinCluster can contain up to 1,024 rule sets. Each cluster has at least one rule set with ID 0, which contains no rules; every VFS in the cluster is, by default, associated with this rule set. This rule set cannot be deleted, although it can be renamed or modified. A newly installed or upgraded cluster contains only the default rule set.

Use the **spinaccess exportrule new** command to create a new export rule set. The following example creates an export rule set named **rs0**:

```
Server1::> spinaccess exportrule new -rulesetname rs0
```

For more information, see "Creating a New Rule Set" on page 217.

Adding a Rule to a Rule Set

Use the **spinaccess exportrule addrule** command to add a rule to an export rule set that was created by using the **spinaccess exportrule new** command. To add a rule, you need to specify the name of the rule set, the client IP address or netgroup name to which the rule applies, the network mask that the rule must match (if a client IP address is specified), the strength of the read-only rule, and the strength of the write rule. The strengths of the read-only and write rules, listed in order of increasing strength, are as follows:

- **any**—A matching client can access the VFS regardless of authentication strength.
- **cifs-ntlm**—A matching client can access the VFS if it is authenticated by CIFS NTLM or a stronger mechanism.
- **cifs-kerberos**—A matching client can access the VFS if it is authenticated by CIFS Kerberos or a stronger mechanism.
- **nfs-kerberos**—A matching client can access the VFS if it is authenticated by NFS Kerberos or a stronger mechanism.
- **never**—Matching clients cannot access the VFS.

Note: Write access is contingent upon read-only access, so specifying a value for the **-rwrule** parameter that is not as strong as the value of the **-rorule** parameter has no effect. For example, if you give the settings **-rorule nfs-kerberos -rwrule any**, CIFS clients cannot access the VFS in any way, even though the value of the **-rwrule** parameter is **any**.

To specify a client IP address, enter the IP address as the value of the **-clientmatch** parameter and specify a netmask by using the **-netmask** parameter. To specify a netgroup, specify the netgroup as the value of the **-clientmatch** parameter in the form **@***netgroup_name*, and do not specify the **-netmask** parameter.

The following example adds an export rule to the rule set named **rs0**. The rule applies to the client IP address **192.55.0.0**, and the network mask **255.255.128.0** is used to match the rule. Rules that are completely redundant are rejected by the SpinServer. The rule permits read-only access to **any** matching clients, but requires that matching clients have **nfs-kerberos** authentication to write files.

```
Server1::> spinaccess exportrule addrule -rulesetname \
rs0 -clientmatch 192.55.0.0 -netmask 255.255.128.0 \
-rorule any -rwrule nfs-kerberos
```

For more information, see “Adding a Rule to a Rule Set” on page 219.

6.13 Setting Up a Virtual File System

The root VFS for a virtual server is automatically created when a virtual server is created. The file system for a particular virtual server can be expanded beyond this single VFS by creating additional VFSs and linking them together with the **`filestorage vfs mount`** command.

Note: By default, the root VFS has a maximum quota of 10 MB. You can expand its quota by using the **`filestorage vfs modify`** command.

Creating a Virtual File System and Setting Its Quota

To create a VFS and set quotas for it, use the **`filestorage vfs new`** command.

Minimum and maximum quotas can be specified when creating a VFS. A quota refers to the amount of data storage on a VFS and is specified in units of 1-MB blocks.

The minimum quota guarantees that the specified amount of storage is available. This is done by reserving storage from the underlying storage pool. The maximum quota limits the amount of data that can be written to a VFS, but no reservation is made. Oversubscription is possible. The quotas can be modified at any time.

The following example creates of a VFS named **`vfs0`** for the virtual server **`vs0`** on storage pool **`sp0`** with a **`5000`**-MB minimum quota and **`10000`**-MB maximum quota.

Note: You cannot use the same VFS name or ID on multiple, different virtual servers. VFS names or IDs must be unique to a cluster.

```
Server1::> filestorage vfs new -vfsname vfs0 –vsname vs0 \
-mounttype context -mountpath /vfs1 -qmin 5000 –qmax \
10000 -freespacealarm 10 -spdevid sp0 -usertype SPIN \
-domain 0 -user 0 -group 0 -rulesetname rs0 \
-showspinshot true -restrictedacl false
```

where

`vfs0` is the name or ID of the virtual file system.

`vs0` is the name or ID of the virtual server.

`context` specifies that this VFS inherits the context (read-write or read-only) from its parent VFS.

`/vfs1` is the mount path for the VFS

`5000` is the minimum quota, specified in MB.

`10000` is the maximum quota, specified in MB.

10 is the percentage of free space remaining on the VFS when the free-space alarm is triggered.

sp0 is the name or ID of the storage pool.

SPIN is the type of the user. This is a generic user type that is used for initial setup procedures. Use the **SPIN** user type if there is not an NIS or CIFS domain configured or available.

0 is the name or ID of the domain.

0 is the name or ID of the user.

0 is the name or ID of the user's group.

rs0 is the name of the export rule set that is applied to this VFS.

true specifies that this VFS's `.spinshot` directory is visible to clients.

false specifies that CIFS ACLs are used on files on this VFS.

For more information, see "Creating Virtual File Systems" on page 158.

Mounting a Virtual File System

Any number of VFSs can be created on a given storage pool. Each VFS can be mounted in another VFS. If you do not specify the mount path by using the **-mountpath** parameter of the **filestorage vfs new** command when the VFS is created, use the **filestorage vfs mount** command to create a mount path for the virtual file system, as in the following example. The VFS is not accessible until it is mounted.

With the **filestorage vfs mount** command, you associate a file system path name with a VFS.

```
Server1::> filestorage vfs mount -vfsname vfs0 \
-mounttype explicit -mountpath /vfs0 -expoverride false
```

where

vfs0 is the name or ID of the virtual file system.

explicit is the type of VFS that is to be mounted. Possible values are **context** (for a VFS that inherits the type of the VFS that it is mounted on), **explicit** (for a read-write VFS), **mirror** (for a read-only mirror), and **clone** (for a read-only clone).

/vfs0 is the mount path of the virtual file system. You access the VFS by this path from root (**/**).

false specifies that the VFS's associated export rule set is not overridden.

For more information, see “Mounting a Virtual File System” on page 160.

6.14 Completing CIFS Tasks

To enable the SpinServer for the CIFS protocol, complete the following tasks:

1. Define the CIFS domain in which the virtual server resides. This task is described in “Defining a CIFS Domain” on page 47.

2. Create a CIFS server. This task is described in “Creating a Kerberos-Enabled CIFS Virtual Server” on page 58.

3. Bind virtual interfaces to the CIFS server.

4. Create CIFS shares.

To complete the CIFS tasks, you must obtain configuration information from your network administrator. Use the following table to enter the information that is needed for this process.

CIFS domain name or ID	
CIFS domain’s WINS server IP address	
CIFS domain’s primary domain controller name	
CIFS domain’s primary domain controller IP address	
CIFS domain’s backup domain controller name	
CIFS domain’s backup domain controller IP address	
Virtual server name or ID	
Visible domain names or IDs	
NetBIOS name	
Node type	
Share name or ID	
Share’s directory path	

Note: For detailed information on CIFS and CIFS-to-NFS mappings, see "Managing SpinAccess CIFS" on page 177 and "Managing CIFS-NFS Translations" on page 193.

For the CIFS command examples that follow, the information obtained from a network administrator is given below.

CIFS domain name or ID	**ENG**
CIFS domain's WINS server IP address	**10.1.1.116**
CIFS domain's primary domain controller name or ID	**STORM**
CIFS domain's primary domain controller IP address	**10.1.1.116**
CIFS domain's backup domain controller name or ID	**RAIN**
CIFS domain's backup domain controller IP address	**10.1.1.118**
Virtual server name or ID	**VS0**
Visible domain names or IDs	**MARKETING, ENG**
NetBIOS name	**NB0**
Node type	**M**
Share name or ID	**ENG_BILL**
Share's directory path	**/development/engineering**

Creating and Configuring a CIFS Share

To share data, a CIFS share must be created. The **spinaccess cifs shares new** command node enables you to define a share, which essentially defines the directory within the file system that is accessible to Windows clients. Multiple shares can be created on a single virtual server.

To Create and Configure a CIFS Share

To create a new CIFS share, complete the following command.

```
Server1::> spinaccess cifs shares new -vsname VS0 \
-sharename ENG_BILL -path /development/engineering \
-remark "This is a new share for VS0."
```

where

VS0 is the name or ID of the virtual server to which the share is added.

ENG_BILL is the name of the share to be added.

/development/engineering is the directory path that is to be shared. This path must exist in the virtual file system.

"**This is a new share for VS0.**" is the description for the share that is to be made available to Windows clients as a comment (optional).

For more information, see “Creating and Configuring a CIFS Share” on page 186.

6.15 Verifying Network Connections

To verify network connections, use the Linux **ping** and **traceroute** commands.

7.0 Integrating with HP OpenView

SpinFS includes integration tools that enable you to manage your SpinServer system by using SpinManager in HP OpenView. After integration, HP OpenView Network Node Manager (NNM) discovers SpinServers in your network, receives event notifications, and displays event messages.

Using the SpinManager integration with HP OpenView NNM is supported on the following platforms:

- Windows NT
- Windows 2000
- Solaris
- HP-UX

Note: The integration with SpinManager does not work if HP OpenView NNM was not properly installed on your machine before the integration process began.

7.1 Integrating SpinManager with HP OpenView

Integrating SpinManager with HP OpenView on Windows

To install HP OpenView support for SpinManager on a Windows machine, perform the following steps:

1. Insert the SpinFS software CD into the CD drive.

2. From the software CD, run the **\Spinnaker\windows\HP\SpinHPOVSetup.exe** program. The installation wizard provides you with the following steps to complete the integration of SpinManager with HP OpenView.

3. After running **SpinHPOVSetup**, the **Welcome** screen is displayed. Click **Next**.



Figure 7-1. The **Welcome** screen

4. Read the license agreement and select **I accept the terms in the license agreement**. Click **Next**.

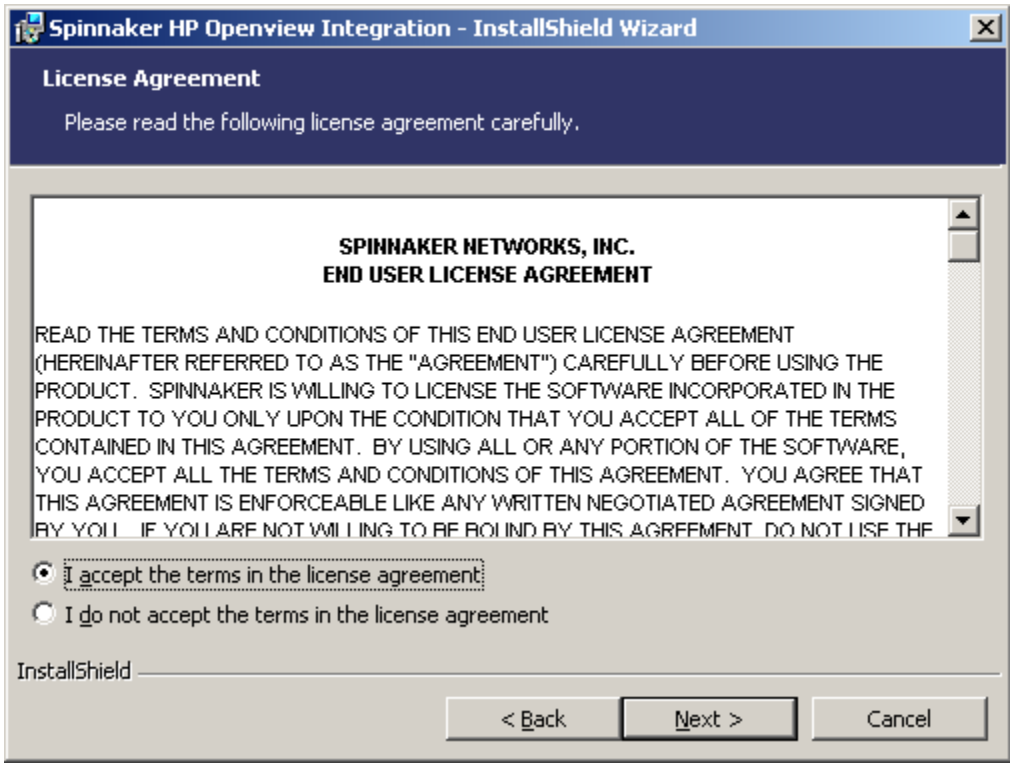


Figure 7-2. The License Agreement

5. In the **Destination Folder** screen, accept the installation folder indicated and click **Next** or click **Change** to select a different installation folder.

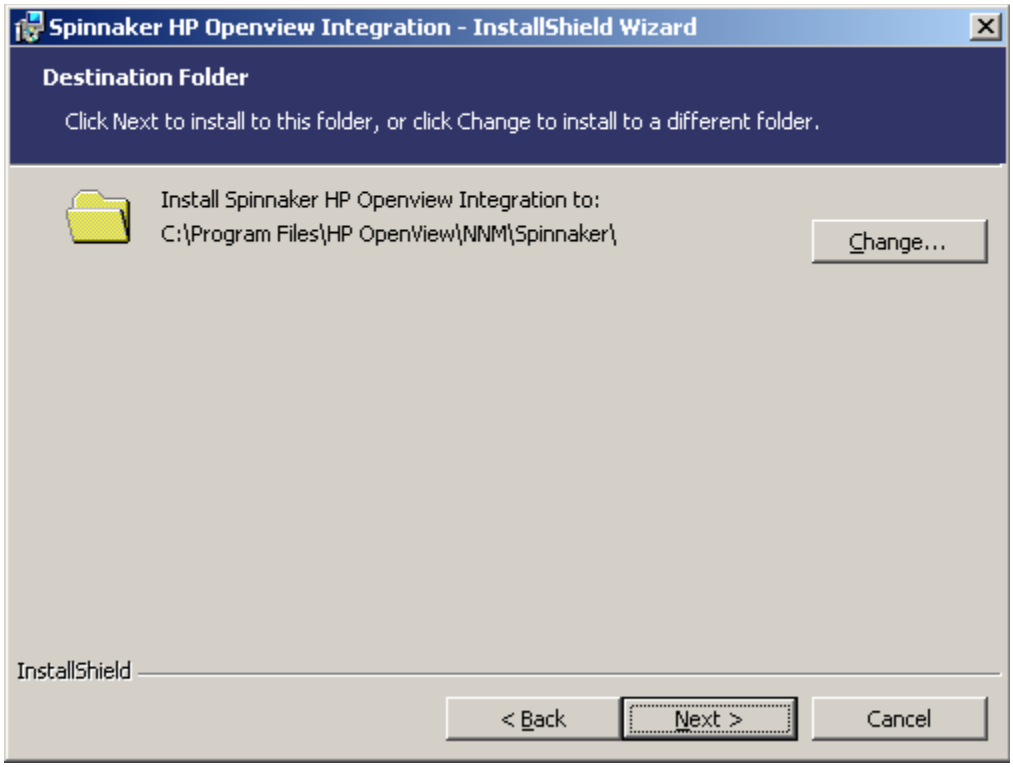


Figure 7-3. The **Destination Folder** screen

6. If you need to install to a different folder, type the path for that folder in the **Folder name** field and click **OK**.

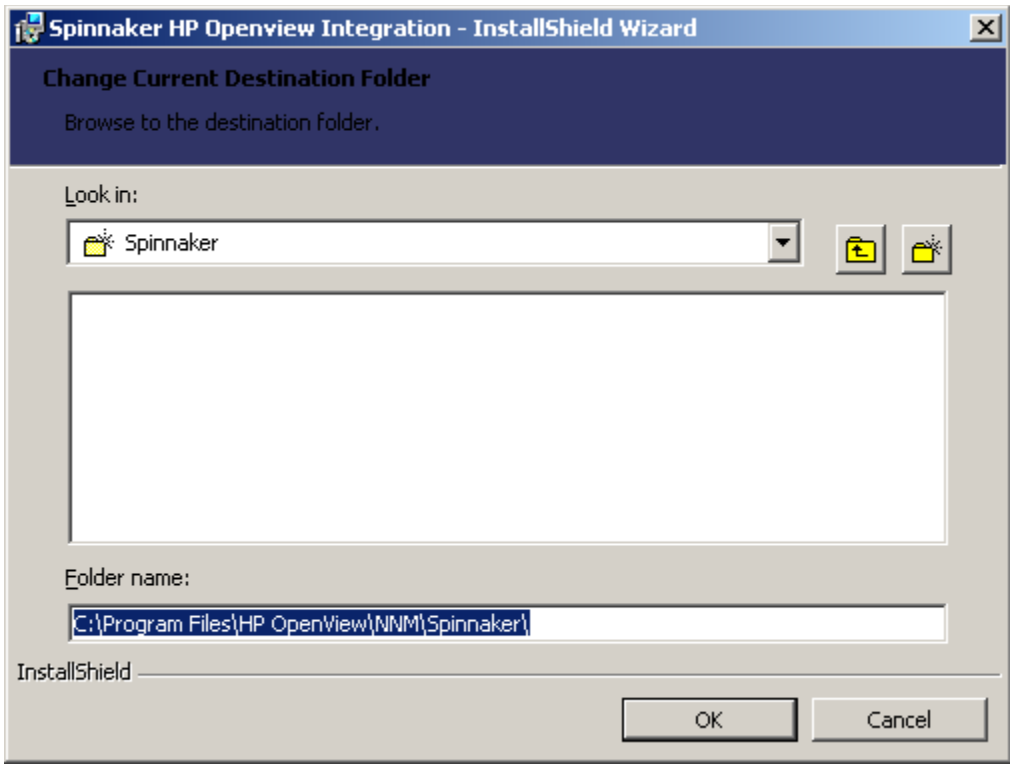


Figure 7-4. The **Change Current Destination Folder** screen

7. The **Ready to Install the Program** screen is displayed. Click **Install** to begin the installation process.

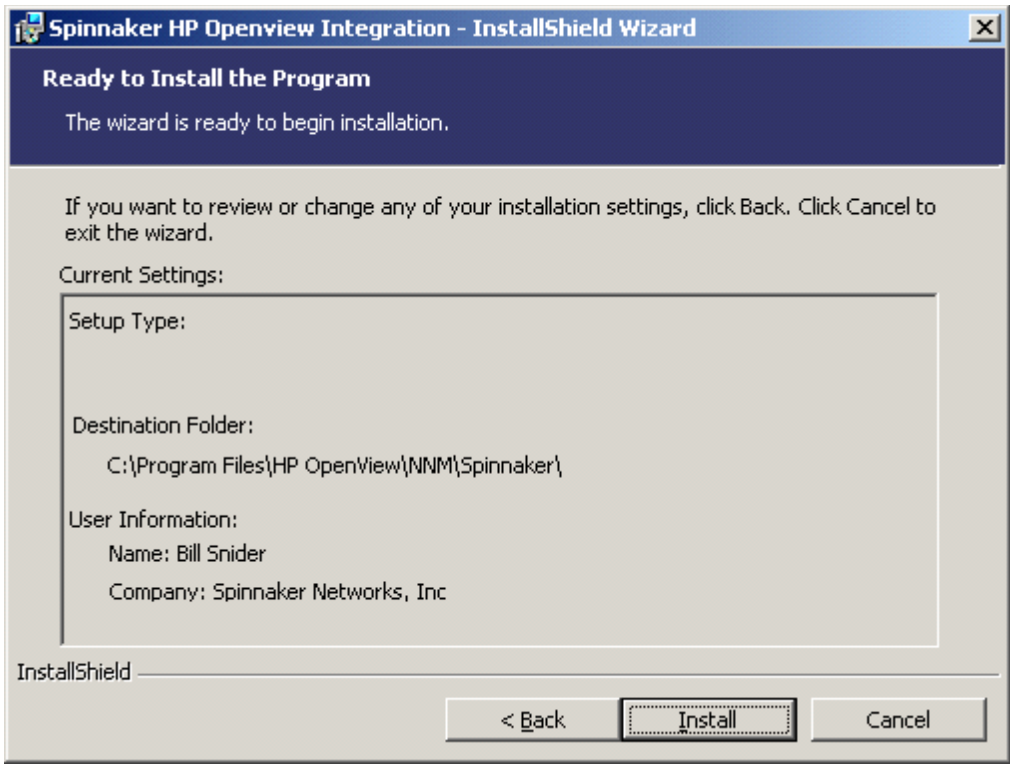


Figure 7-5. The **Ready to Install the Program** screen

8. After the installation process is completed, the **InstallShield Wizard Completed** screen is displayed. Click **Finish**.



Figure 7-6. The **InstallShield Wizard Completed** screen

Integrating SpinManager with HP OpenView on Solaris and HP-UX

To install SpinServer support for the HP OpenView Network Node Manager (NNM) onto a Solaris or HP-UX system, perform the following steps:

1. Close all NNM sessions on the machine on which HP OpenView is installed. You do not need to stop the NNM server processes.

2. Insert the software CD-ROM into the CD-ROM drive of the machine on which HP OpenView is installed. If necessary, mount the CD-ROM by using standard operating-system commands.

3. Source the **`ov.envvars.sh`** script from the HP OpenView installation's **`bin`** directory. This script sets environment variables that are required for the installation of SpinServer support.
   ```
   $ $OV_BIN/ov.envvars.sh
   ```

4. Verify that the HP OpenView installation's **`bin`** directory is in your path.

5. Change to the **`/cdrom/Spinnaker/unix/HP/`** directory:
   ```
   $ cd /cdrom/Spinnaker/unix/HP
   ```

6. Run the **`install.sh`** script:
   ```
   $ ./install.sh
   ```

7. When the installation completes, the following message is displayed on the console:
   `Spinnaker Server successfully installed in HP OpenView Network Node Manager.`

7.2 Creating SpinServer Objects in Network Node Manager

To create SpinServer objects in Network Node Manager (NNM), perform the following steps.

1. After you have completed the installation of SpinManager in HP OpenView, launch OpenView.

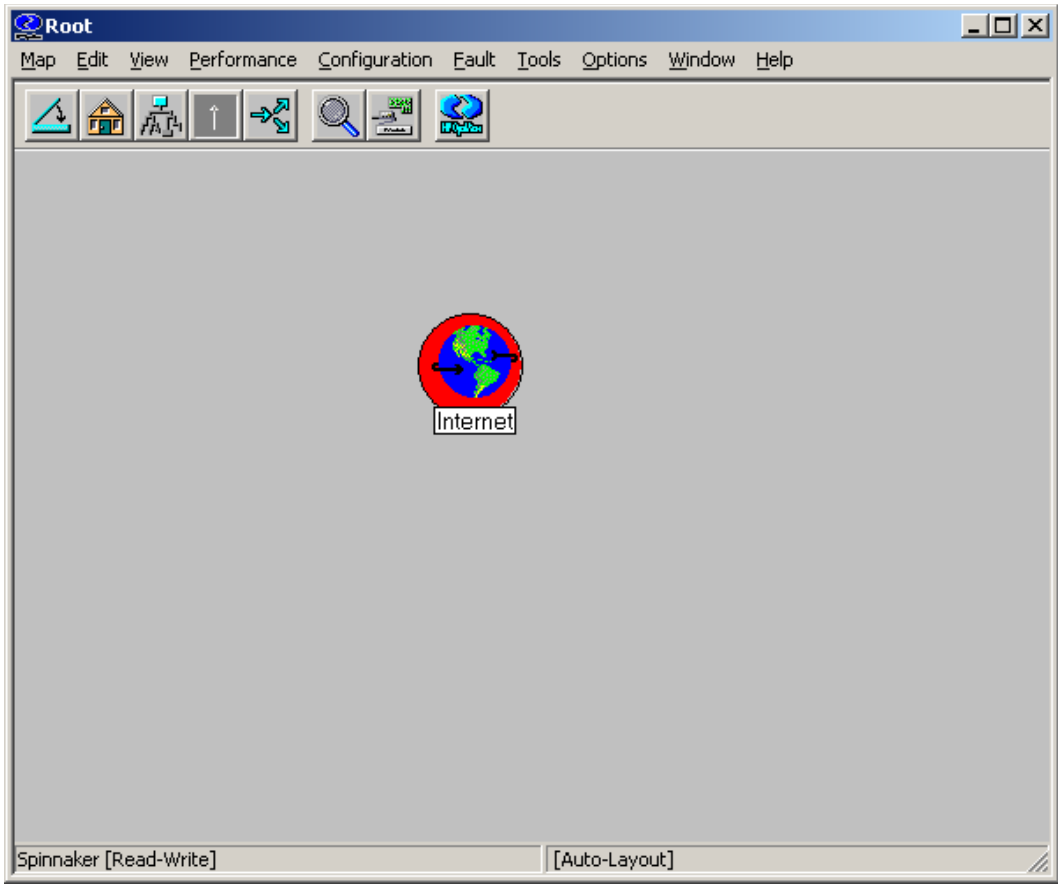


Figure 7-7. Launch OpenView.

2. From the **Edit** menu, select **Add Object**. The **Add Object Palette** opens.

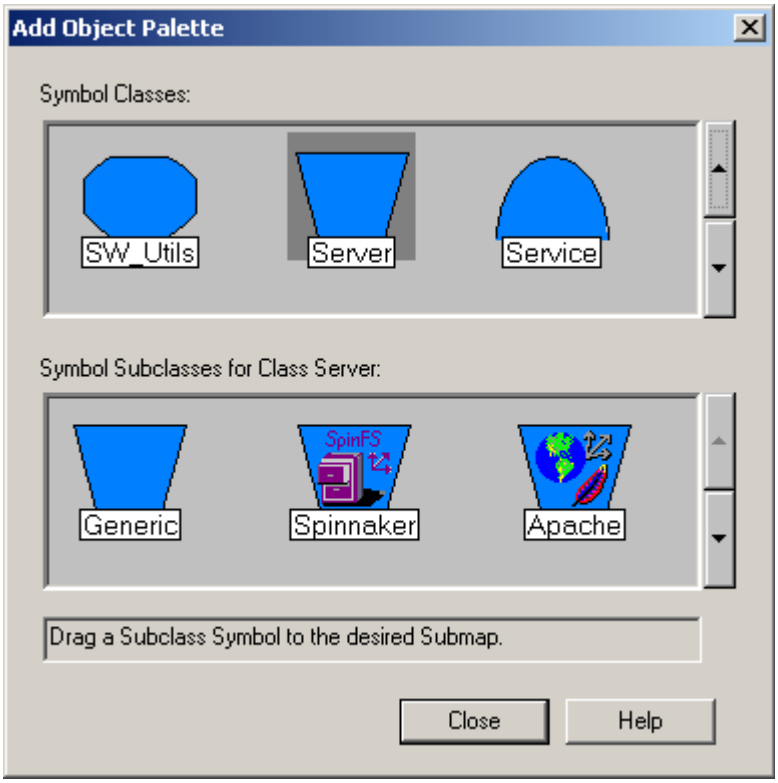


Figure 7-8. Add Object Palette

3. Scroll through the **Symbol Classes** choices and select the **Server** object.

4. Scroll through the **Symbol Subclasses for Class Server** choices and drag the **Spinnaker** object to the network map. The **Add Object** dialog box opens.

5. In the **Label** field, enter the name of the SpinServer, as shown in the following example. The SpinServer name must be the DNS name of the SpinServer management port.

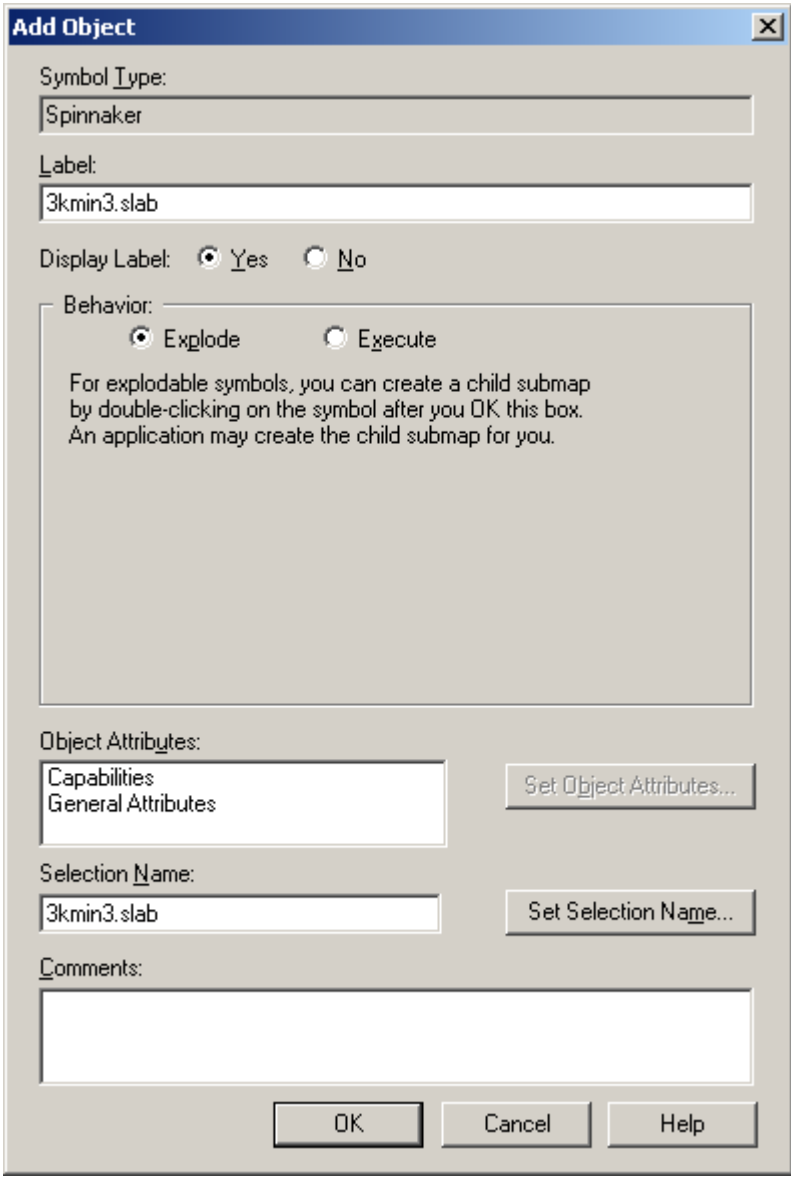


Figure 7-9. The **Add Object** dialog box

6. Click **OK**. The SpinServer object is created and is displayed on the network map.

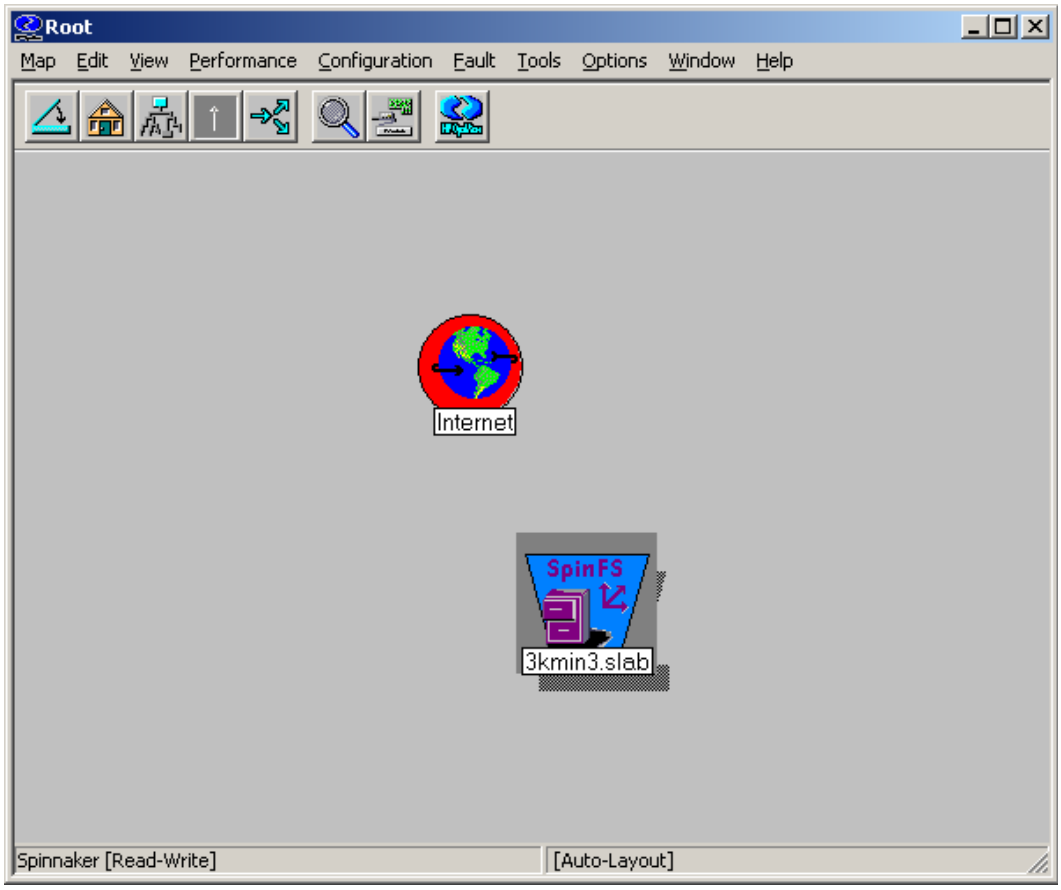


Figure 7-10. The SpinServer server object

7. To use SpinManager to manage the SpinServer, right-click the SpinServer object and select **Launch SpinManager**.

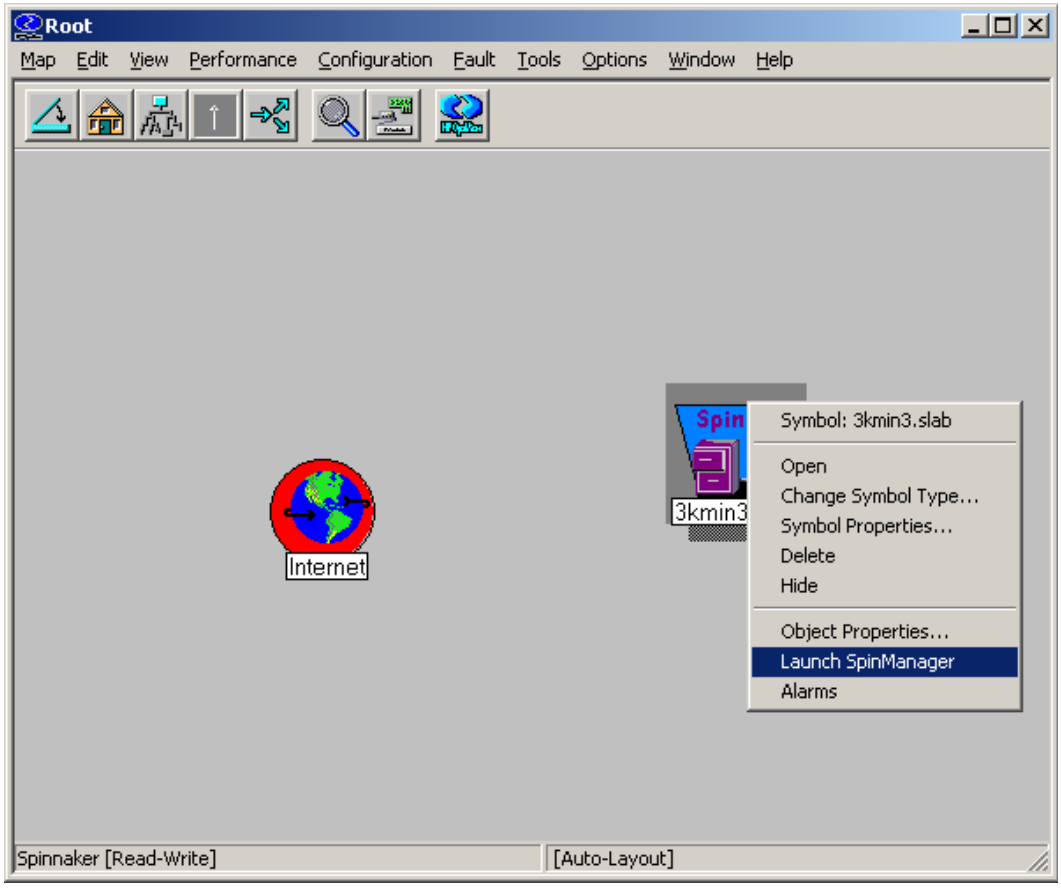


Figure 7-11. Launch SpinManager.

8. SpinManager is opened in your browser and the **SpinServer Login** screen is displayed.

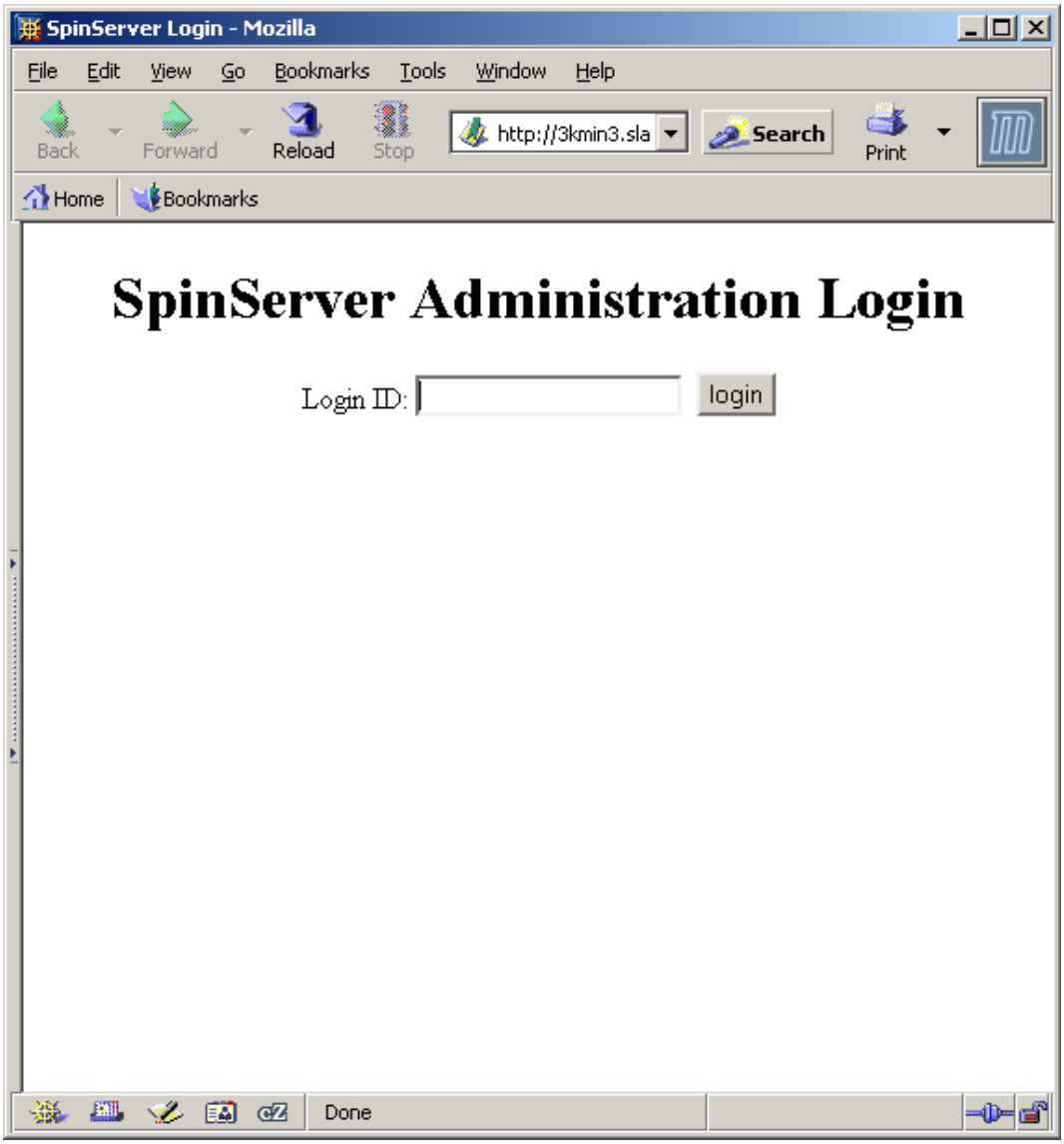


Figure 7-12. SpinManager login screen

9. Complete the integration between HP OpenView and the SpinServer by performing the following steps:

   a. Set up an SNMP trap destination on the SpinServer for HP OpenView. See “Creating an SNMP Trap Destination” on page 301 for details.

   b. Create an event filter on the SpinServer for HP OpenView. See “Creating a Spin Events Filter” on page 268 for details.

   c. Create an event-notification handler on the SpinServer for HP OpenView. See “Creating Event-Notification Handlers” on page 273 for details.

   d. Create an event-notification subscription on the SpinServer for HP OpenView. See “Creating Event-Notification Subscriptions” on page 275 for details.

Part III: Data-Management Tasks

This part discusses tasks and techniques you can use to manage data on a SpinServer system. It is organized into the following chapters:

Chapter 8. Managing Clusters

"Managing Clusters" on page 81 provides information on setting up and managing clusters.

Chapter 9. Managing Networks

"Managing Networks" on page 91 provides information on setting up and managing ports and network interfaces.

Chapter 10. Managing RAID Components

"Managing RAID Components" on page 103 provides information on managing RAID components that are used to store data on the SpinServer system.

Chapter 11. Managing Storage

"Managing Storage" on page 133 provides information on setting up and managing storage units and storage pools. It also provides information on setting up and managing a high-availability (HA) system.

Chapter 12. Managing Virtual Servers

"Managing Virtual Servers" on page 147 provides information on creating and managing virtual servers.

Chapter 13. Managing Virtual File Systems

"Managing Virtual File Systems" on page 155 provides information on setting up and managing virtual file systems (VFSs), which are the basic units of storage in the SpinServer model.

Chapter 14. Managing SpinAccess CIFS

"Managing SpinAccess CIFS" on page 177 provides information on setting up a SpinServer system to use the Common Internet File System (CIFS).

Chapter 15. Managing CIFS-NFS Translations

"Managing CIFS-NFS Translations" on page 193 provides information on mapping CIFS users and Network File Systems (NFS) users so that users of the two different protocols can share data.

Chapter 16. Managing Security

"Managing Security" on page 207 provides information on setting up and managing Kerberos authentication on a SpinServer system.

Chapter 17. Protecting Against Viruses

XREF discussing using the SpinAV component to protect data on your SpinServer system from viruses.

Chapter 18. Managing Data Backups and Restorations

"Managing Data Backup and Restoration" on page 231 provides information on configuring a SpinServer system to use a third-party backup and restore mechanism.

Chapter 19. Managing Disaster Recovery

"Managing Disaster Recovery" on page 235 provides information on recovering data after a catastrophic event such as a storage pool failure.

8.0 Managing Clusters

This chapter discusses the management of clusters. Clustering forms the basis of the SpinServer system's scaling and high-availability capacities. Every SpinServer in a system must be part of a cluster. A cluster can contain one or more SpinServers; however, a SpinServer can belong to only one cluster at a time.

8.1 Cluster Management Concepts

This section discusses concepts and terms of cluster technology.

For ease of management of a SpinServer system and the data stored on it, you need to give some thought to the names you choose for the SpinServers in a cluster. To prevent confusion and potential loss or misplacement of data, the names you choose must be orderly and easily recognizable by all system administrators who maintain the SpinServer system.

Any naming system can be chosen for the system, *but the IDs for the SpinServers must be unique within a cluster*. After the ID for a SpinServer has been set, you cannot modify the server name or host name. To do so invalidates the failover port designation for the SpinServer and can result in the loss of data.

You enter the SpinServer ID into the system when you initially install the SpinFS software on the SpinServer. The name of the SpinServer is propagated from the initial installation into the management framework.

Initialization of a SpinServer involves joining an existing cluster or creating a new cluster. Every SpinServer is part of a cluster, and although the cluster can contain only a single SpinServer, each SpinServer must be entered into a cluster before any other system management task is undertaken.

Use the **`spinservers createcluster`** command to enter the first SpinServer into a cluster, and use the **`spinservers joincluster`** command to add a new SpinServer into an already existing cluster. The **`spinservers joincluster`** command provides all active SpinServers in the cluster with all of the communication details for contacting the newly configured SpinServer. The new SpinServer joins the cluster and quorum is achieved. (See "Clustering Concepts" on page 82 and "Quorum in Clusters" on page 83 for information about quorum.) At that point, the new SpinServer is updated with the clusterwide view.

The **spinserver createcluster** and **spinserver joincluster** commands cause other system operations to take place. Both the **spinserver createcluster** and **spinserver joincluster** commands take the IP configuration for the cluster ports. This information is then stored in the interface configuration page after the node has completed the create or join cluster operation.

Clustering Concepts

SpinServer clustering is based on the following concepts:

A *quorum* is formed when a majority of the known SpinServers, or *nodes*, in a cluster are healthy and in contact with one another. In this healthy state, they run a voting protocol that elects one node as the *master*; the remaining nodes are known as *secondary* nodes. (Note that the master/secondary designation in a cluster is unrelated to the concept of primary, active secondary, and passive secondary servers in a high-availability configuration.) When quorum is formed, it is maintained by continual voting of the nodes. Because of the majority requirement, there can be only one quorum at a time. Other nodes can join an existing quorum (for example, on restart). In a cluster's steady, healthy state, it is normal for a quorum to exist.

When a new quorum forms, a distributed recovery algorithm synchronizes all of the nodes. When this process successfully completes, all of the member nodes go online. When a new node joins an existing quorum, it individually proceeds through the recovery process and goes online.

In a steady state in a well-connected network, healthy nodes are online and each node is capable of running all defined SpinServer system operations. Changes made to one node are simultaneously propagated to all online nodes. When a node is offline (for instance, if there is no quorum, or if the node is unable to contact the quorum), it is not capable of running all defined SpinServer system operations.

In a cluster with an even number of nodes, *epsilon* enables a quorum to form by using slightly weaker requirements. Epsilon is an extra fractional voting weight held by, at most, one node. For example, two out of four eligible nodes are sufficient if one of those two nodes holds epsilon. Epsilon is a configuration setting for a node in a cluster.

An *eligible* node is one that has been designated by the administrator as an active participant in the cluster. Its presence impacts on the quorum membership requirements, and it is online (if at all possible). An administrator can choose to make the node ineligible for many reasons, such as recent software upgrades, network reconfiguration, and coping with failures. Use the **cluster status enablenode** and **cluster status disablenode** commands to configure node eligibility. If a node needs to be removed completely from the cluster, it must first be marked ineligible, and then removed from the cluster by using the **cluster status disablenode** command.

Quorum in Clusters

Managing clusters requires a working familiarity with the concepts that provide the framework for cluster technology in the SpinServer system. The primary concept that needs to be understood is the maintenance of quorum. Quorum is a precondition for a fully functioning cluster. When quorum is lost, the cluster loses the ability to accomplish normal cluster operations.

For single-node clusters, a majority is achieved when the single node is configured with epsilon. Achieving majority defines whether or not quorum is maintained; without epsilon, the majority is lost for the single-node cluster.

For this reason, a two-node cluster presents some unique challenges in maintaining quorum. In a two-node cluster, epsilon is not held by either node; instead, both nodes are continuously polled to ensure that if one node fails, the other has full read-write access to data as well as access to virtual interfaces and management functions.

In larger clusters, the effect of epsilon is most often felt when the cluster can be partitioned. In general, assuming reliable connectivity among the various nodes of the cluster, a larger cluster is more stable than a smaller cluster. The quorum requirement of a simple majority of half the nodes plus epsilon is easier to maintain in a cluster of 50 nodes than in a cluster of 2 nodes. However, when the connectivity between two equal portions of a large cluster fails, the group of SpinServers containing epsilon maintain quorum, assuming that all the SpinServers are healthy. For example, if a single link is established between 25 SpinServers in one room and 25 SpinServers in another room to comprise a 50-node cluster and the link fails, then the group of SpinServers that holds epsilon maintains quorum.

Only one collection of SpinServers can have quorum at any one time because all of the SpinServers collectively share a single view of the data. Therefore, if two noncommunicating SpinServers are permitted to modify the data in divergent ways, it is no longer possible to reconcile the data into a singular data view.

8.2 Cluster Management Tasks

Managing clusters includes creating clusters, adding and removing SpinServers, and querying their status. Creating a cluster and adding SpinServers to it uses the **`spinservers createcluster`** and **`spinservers joincluster`** commands; the remaining tasks in the following list use the **`cluster`** command node.

- Creating a cluster.
- Adding additional SpinServers to the cluster.
- Enabling or disabling a cluster node.
- Adding or removing epsilon.
- Enabling or disabling high-availability (HA) mode in a two-node cluster.
- Removing a node from a cluster.

- Displaying a cluster's configuration information.
- Setting a route for a cluster.
- Setting the audit-log size for a cluster.
- Setting the cache size for a cluster.

Creating a Cluster

Use the **spinservers createcluster** command to create a cluster from one SpinServer. The syntax of the command follows.

```
spinservers createcluster -ssid spinserver -clustername
cluster_name -ipaddress1 IP_address [-ipaddress2
IP_address] -netmask1 netmask [-netmask2 netmask]
[-broadcast1 IP_address] [-broadcast2 IP_address] [-mtu1
maximum_transmission_unit] [-mtu2
maximum_transmission_unit]
```

Important: This command must not be used for any purpose other than to create a new cluster. If you mistakenly run the **spinservers createcluster** command in attempt to add a SpinServer to an existing cluster, you must reinstall the SpinServer (or completely delete all configuration data) to undo the operation. Before you reinstall the SpinServer or delete any configuration information, contact Network Appliance Technical Support.

The following example initializes SpinServer **Server1** into a cluster named **Cluster1** with a cluster-network netmask of **255.255.0.0**, an IP address of **192.168.66.1** for the first cluster interface, an IP address of **192.168.66.2** for the second cluster interface, a netmask of **255.255.0.0**, a broadcast IP address of **192.168.255.255**, and a maximum transmission unit (MTU) setting of **9000** bytes.

```
spinserver::> spinservers createcluster –ssid Server1 \
-clustername Cluster1 -ipaddress1 192.168.66.1 \
-ipaddress2 192.168.66.2 -netmask1 255.255.0.0 \
-broadcast1 192.168.255.255 -mtu1 9000
```

Note: The **spinservers createcluster** command can take up to 30 seconds to complete.

Joining a Cluster

Use the the **spinservers joincluster** command to join a SpinServer to a cluster. The syntax of the command follows.

```
spinservers joincluster -ssid spinserver
-clusternodeipaddress IP_address -ipaddress1 IP_address
[-ipaddress2 IP_address] -netmask1 netmask [-netmask2
netmask] [-broadcast1 IP_address] [-broadcast2 IP_address]
[-mtu1 maximum_transmission_unit] [-mtu2
maximum_transmission_unit]
```

The following example adds a SpinServer named **Server2** to a cluster that is hosted by a SpinServer with the IP address **192.168.66.10**. The first cluster interface has the IP address **192.168.66.1**, the second cluster interface has the IP address **192.168.66.2**, the broadcast IP address is **192.168.255.255**, the network mask is **255.255.0.0**, and the MTU setting is **9000** bytes.

```
spinserver::> spinservers joincluster -ssid Server2 \
-clusternodeipaddress 192.168.66.10 -ipaddress1 \
192.168.66.1 -ipaddress2 192.168.66.2 -netmask1 \
255.255.0.0 -broadcast1 192.168.255.255 -mtu1 9000
```

Note: The **spinservers joincluster** command can take up to 30 seconds to complete.

Managing Clusters

After creating a cluster by using the **spinservers createcluster** command and adding other SpinServers to the cluster by using the **spinservers joincluster** command, you can perform the following cluster-management tasks:

- Enabling and disabling quorum voting
- Displaying information about cluster nodes
- Adding epsilon to and removing epsilon from cluster nodes
- Enabling and disabling two-node high-availability (HA) mode
- Setting cluster routes
- Setting the audit-log size for the cluster

Enabling Cluster Nodes

To add or remove a SpinServer for quorum voting in a cluster, use the **cluster status enablenode** and **cluster status disablenode** commands. Enabling a SpinServer to participate in the quorum voting marks the node as eligible and changes the conditions under which quorum can be maintained. Quorum is maintained when half or more of the SpinServers that are eligible to vote are responding in a positive, healthy manner.

Note: The **cluster status enablenode** and **cluster status disablenode** commands cannot be run on the node that you are enabling or disabling. Run these commands on a node that is already a member of the cluster, and specify the node that is to be enabled or disabled by using the **-ssid** parameter.

To Enable a Spinserver for Quorum Voting

Use the **cluster status enablenode** command to add a SpinServer to the quorum voting in a cluster. The command has the following syntax:

cluster status enablenode -ssid *spinserver*

The following example enables quorum voting for a SpinServer named **Server1**:

```
spinserver::> cluster status enablenode -ssid Server1
```

To Disable a Spinserver from Quorum Voting

To remove a SpinServer from the quorum voting in a cluster and mark it as ineligible as an active participant in the cluster, use the **cluster status disablenode** command, as in the following example, which disables quorum voting for a SpinServer named **Server1**.

```
spinserver::> cluster status disablenode -ssid Server1
```

From the browser interface, the following confirmation message appears:

```
Disabling a node from the cluster means the node can no
longer participate in cluster updates or voting. Are you sure
you want to disable this node?
```

Click **OK** to confirm the command.

Designating a SpinServer as Epsilon

To add or remove the epsilon designation for a SpinServer in a cluster, use the **cluster status addepsilon** and **cluster status removeepsilon** commands. Epsilon enables a quorum to form by using slightly weaker requirements; for example, two out of four eligible nodes are sufficient if one of those two nodes holds epsilon. Quorum is maintained when half or more of the SpinServers that are eligible to vote are responding in a positive, healthy manner. A SpinServer can be designated as epsilon to add weight to its voting in a cluster with an even number of nodes. Only one SpinServer can be designated as epsilon at any given time in a cluster.

To Designate a SpinServer as Epsilon

The **cluster status addepsilon** command is used to designate a SpinServer in a cluster as epsilon. The following example designates a SpinServer named **Server1** as epsilon.

```
spinserver::> cluster status addepsilon -ssid Server1
```

To Remove a Designation of a SpinServer as Epsilon

The **cluster status removeepsilon** command is used to remove the designation of a SpinServer in a cluster as epsilon. The following example removes the designation of epsilon from a SpinServer named **Server1**

```
spinserver::> cluster status removeepsilon -ssid Server1
```

Deleting a Node from a Cluster

To remove a node from a cluster, use the **cluster status deletenode** command. This command deletes all configuration data from the SpinServer. The command's **-force** parameter specifies whether the deletion operation is forced; the default is **false**. To return a deleted node to its original cluster or to add it to a different cluster, use the **spinservers joincluster** command.

The command has the following syntax:

```
cluster status deletenode -ssid spinserver -force {true |
false}
```

The following example deletes a SpinServer named **Server1** from its cluster. The deletion operation is not forced.

```
spinserver::> cluster status deletenode -ssid Server1 \
-force false
```

Displaying Information about Clusters

The **cluster status show** command displays the following information about nodes in a cluster:

- Host name
- Primary cluster IP address
- Secondary cluster IP address
- Epsilon designation
- Node state
- Health of node

The following example displays configuration information about the SpinServers in a cluster.

```
spinserver::> cluster status show
Hostname   Cluster IP 1     Cluster IP 2   Epsilon Node State  Health

Server1   10.3.128.20      10.3.128.28      true    Eligible   Online
Server2   10.3.128.21      10.3.128.27      false   Eligible   Online
```

Setting a Route for a Cluster

You can set a specific network route for the cluster to control the flow of traffic across a complex network. Use the **cluster routes** commands to work with cluster routes.

Creating a Route for a Cluster

Use the **cluster routes new** command to create a new cluster route. The command has the following syntax:

```
cluster routes new -portname port_name -nexthop
next_hop_IP_address -netnumber network_IP_address
-netmask netmask -metric metric
```

The following example creates a cluster route on port **clus0** with a next-hop address of **127.127.0.0**, a network IP address of **127.64.7.1**, and a netmask of **255.255.0.0**.

```
spinserver::> cluster routes new -portname clus0 \
-nexthop 127.127.0.0 -netnumber 127.64.7.1 -netmask \
255.255.0.0 -metric 1
```

Displaying a Cluster Route

The **cluster routes show** command displays information about one or more cluster routes. The command has the following syntax:

```
cluster routes show [-portname port_name -nexthop
next_hop_IP_address -netnumber network_IP_address
-netmask netmask]
```

The following example displays information about all cluster routes:

```
spinserver::> cluster routes show
Port Name         Gateway Address  Network Address    Network Mask   Metric
clus0             127.127.0.0      127.64.7.1         255.255.0.0    1
clus1             127.127.0.0.     127.64.8.1         255.255.0.0    1
```

Deleting a Cluster Route

Use the **cluster routes delete** command to delete a cluster route. The command has the following syntax:

```
cluster routes delete -portname port_name -nexthop
next_hop_IP_address -netnumber network_IP_address
-netmask netmask
```

The following example deletes the route on the port **clus0**, which has a next-hop address of **127.127.0.0**, a network IP address of **127.64.7.1**, and a network mask of **255.255.0.0**:

```
spinserver::> cluster routes delete -portname clus0 \
-nexthop 127.127.0.0 -netnumber 127.64.7.1 -netmask \
255.255.0.0
```

Enabling High-Availability Mode in a Two-Node Cluster

High availability (HA) in a two-node cluster is a special case and must be specifically enabled by using the **cluster configuration enablehamode** command. If the cluster contains three or more nodes, you do not specifically need to enable HA mode in the cluster. See “High Availability” on page 21 and “Managing Storage” on page 133 for more information on HA configurations.

Note: Before enabling HA mode, you must have a storage pool configured with an active secondary. See “Adding a Passive Secondary to a Storage Pool” on page 143 and “Promoting a Passive Secondary to Active Secondary” on page 144 for details.

To enable HA mode in a two-node cluster, use the **cluster configuration enablehamode** command. The command has the following syntax:

cluster configuration enablehamode -ssid *spinserver*

The following example enables HA mode in a two-node cluster that contains a SpinServer named **Server1**:

```
spinserver::> cluster configuration enablehamode -ssid \
Server1
```

Disabling High-Availability Mode in a Two-Node Cluster

Use the **cluster configuration disablehamode** command to disable HA mode in a two-node cluster. The command has the following syntax:

cluster configuration disablehamode -ssid *spinserver*

The following example disables HA mode in a two-node cluster that contains a SpinServer named **Server1**:

```
spinserver::> cluster configuration disablehamode -ssid \
Server1
```

Modifying the Cluster Configuration

The audit log stores a record of commands that are issued on a SpinServer cluster. By default, the number of commands recorded by the log is 10; commands are recorded across the entire cluster, not on individual SpinServers. The SpinServer's health-monitoring facility can be used to send periodic reports on the SpinServer's condition to an email address; this requires a mail-from address and outbound mail server (see "Managing Health Monitoring" on page 259 for details). If the cluster is predominantly serving either very large files or very small files, you can specify the use of a unified cache scheme to improve efficiency. You can specify whether or not a message is issued when a SpinServer in the cluster is restarted. Because restarts are categorized as critical, they often trigger pager alerts. You can set the number of commands that are contained in the audit log, the email properties, the cache scheme, and the sending of restart messages for a SpinServer cluster by using the **`cluster configuration modify`** command. The command has the following syntax:

```
cluster configuration modify -clustername cluster_name
-auditlogsize audit_log_size -mailfrom mail_from_address
-mailserver outbound_mail_server -unifiedcache {true |
false} -sendmessage {true | false}
```

The following example changes the number of commands recorded in the audit log to **`2000`** in a cluster named **`cluster1`**. The mail-from address is set to **`postmaster@spincluster.example.com`** and the outbound mail server is specified as **`smtp1.example.com`**. The unified cache scheme is enabled, and restart messages are disabled.

```
spinserver::> cluster configuration modify -clustername \
cluster1 -auditlogsize 2000 -mailfrom \
postmaster@spincluster.example.com -mailserver \
smtp1.example.com -unifiedcache true -sendmessage false
```

Displaying Information about Cluster Configuration

Use the **`cluster configuration show`** command to display configuration information for a SpinServer cluster. The command has the following syntax:

```
cluster configuration show [-clustername cluster_name]
```

The following example displays configuration information for a cluster named **`cluster1`**:

```
spinserver::> cluster configuration show -clustername \
cluster1
Cluster Name                 cluster1
AuditLog Size                2000
Mail From                    postmaster@10.3.129.40
Outging SMTP Server          foo
Unified Cache Scheme Enabled  true
```

9.0 Managing Networks

This chapter discusses the tasks associated with managing networks. Managing networks includes the creation, configuration, and management of virtual interfaces and network ports.

9.1 Network Management Concepts

A virtual interface is associated with a network port and is assigned an IP address. A SpinServer port can be configured with one or more virtual interfaces with IP addresses and additional failover interfaces. Assigning failover interfaces is critical for creating a high-availability (HA) configuration.

In this chapter, you create a virtual interface and use a storage pool to create a virtual server. For information on creating storage pools, see “Managing Storage” on page 133. After the virtual server is created, you bind the virtual interface to it.

9.2 Network Management Tasks

Virtual interfaces must be created before virtual servers are created. After creating the virtual servers, you bind the virtual interfaces you have created to the virtual servers.

To complete the network-management tasks, perform the following steps:

1. Create and configure a virtual interface.

2. Create a virtual server, which creates the root virtual file system.

3. Bind the virtual interface to the virtual server

4. Associate a virtual interface with a network port.

The first step in managing virtual interfaces is to create and configure as many interfaces as the system needs. You name the virtual interfaces and associate them with an Ethernet port. When the interfaces are available to the system, there are many management tasks that are done occasionally, such as modifying the configuration of interfaces or deleting them.

Creating and Configuring a Virtual Interface

You configure the interface as you create it. To configure a virtual interface, you need to obtain an IP address and netmask from your network administrator.

In the table below, enter the configuration information.:

IP address	
Netmask	

The **`if configuration new`** command creates an interface and identifies it by its host SpinServer, interface name, network port name, IP address, and netmask. The syntax of the command is as follows:

```
if configuration new -ssid spinserver -ifname
interface_name -netportname network_port_name -ipaddress
IP_address -netmask netmask
```

The command example creates a new interface named **`if0`** on the SpinServer named **`Server1`**. The interface has a network port name of **`serv0`**, an IP address of **`10.1.1.140`**, and a netmask of **`255.255.255.0`**.

```
spinserver::> if configuration new -ssid Server1 \
-ifname if0 -netportname serv0 -ipaddress 10.1.1.140 \
-netmask 255.255.255.0
```

Displaying Virtual Interface Status

After creating virtual interfaces for the network, use the **`if status show`** command to display and review information about the virtual interfaces. The following information about the interfaces is displayed by the system:

- SpinServer ID
- Name
- Network port
- IP address
- Netmask
- Failover port
- Cluster or server role in the network

The syntax of the command is as follows:

```
if status show [-ssid spinserver -ifname interface_name]
```

The following example displays information about all interfaces.

```
spinserver::> if status show
SpinServer    Name         Address      Stat   Active   Active Virtual
   ID                                          Server    Port

spsrv10    clus_if1    10.3.128.61      up   spsrv10    clus1  false
spsrv10    clus_if0    10.3.128.60      up   spsrv10    clus0  false
spsrv10    mgmt_if0    10.3.2.115       up   spsrv10    mgmt0  false
```

The following example displays information about the interface named **rData0** on the SpinServer named **spsrv10**:

```
spinserver::> if status show -ssid spsrv10 -ifname rData0
SpinServer ID   spsrv10
Name            rData0
Port            serv1
Address         10.5.15.58
Status          up
Role            Server
Active Server   spsrv10
Active Port     serv1
Virtual         true
```

Migrating a Virtual Interface

To migrate a virtual interface from one port to another, you can use the **if status migratevif** command. This command is useful when you are working with high-availability (HA) configurations and you need to move a virtual interface from a failed primary server to a secondary server.

The syntax of the command follows.

```
if status migratevif -ssid spinserver -ifname
virtual_interface_name -destssid destination_spinserver
-destport destination_port
```

The following example moves a virtual interface named **vif0** on SpinServer **Server1** to port **serv0** on SpinServer **Server2**.

```
spinserver::> if status migratevif -ssid Server1 -ifname \
vif0 -destssid Server2 -destport serv0
```

Displaying Virtual Interface Configurations

After creating virtual interfaces for the network, use the **if configuration show** command to display and review the configuration information for the virtual interfaces. The following information about the interfaces is displayed by the system:

- SpinServer ID
- Name
- Network port
- IP address
- Netmask
- Failover port

The syntax of the command is as follows:

if configuration show -ssid *spinserver* **-ifname** *interface_name*

The following example displays information about all the configuration of all interfaces.

```
spinserver::> if configuration show
SpinServer ID    Name     Port      IP Address        Netmask       Failover
                                                                      Port

spsrv10        rData0    serv1   10.5.15.58       255.255.0.0
spsrv10        Data0     serv0   10.5.10.58       255.255.0.0
spsrv10        rData1    serv0   10.5.15.162      255.255.0.0
spsrv10        rData2    serv1   10.5.15.161      255.255.0.0
spsrv10        rData3    serv0   10.5.15.160      255.255.0.0
spsrv10        rData4    serv1   10.5.15.157      255.255.0.0
spsrv10        clus_if0  clus0   10.3.128.20      255.255.128.0
spsrv10        rData5    serv0   10.5.15.158      255.255.0.0
spsrv10        clus_if1  clus1   10.3.128.28      255.255.128.0
```

The following example identifies an interface named **if0** on a SpinServer named **spsrv10** and displays its configuration information.

```
spinserver::> if configuration show -ssid spsrv10 \
-ifname if0
SpinServer ID   spsrv10
Name            rData0
Port            serv1
IP Address      10.5.15.58
Netmask         255.255.0.0
Failover Port
VIF ID          6
Role            Server
```

Deleting a Virtual Interface

If a virtual interface is not needed or becomes obsolete, you can remove it from the network by using the **if configuration delete** command. The command has the following syntax:

```
if configuration delete -ssid spinserver -ifname
interface_name
```

The following example deletes a virtual interface named **if0** on a SpinServer named **spsrv10**.

```
spinserver::> if configuration delete -ssid spsrv10 \
-ifname if0
```

Taking a Virtual Interface Offline

The configuration of a virtual interface can be modified when the virtual interface is online. The configuration change takes effect when the interface is restarted (that is, after making the change, you must take the interface offline, then bring it back online). Adding or removing a virtual interface route takes effect immediately.

To Take a Virtual Interface Offline

The **if configuration offline** command identifies an interface by its SpinServer ID and interface name and designates the interface's status as offline. The syntax of the command is as follows:

```
if configuration offline -ssid spinserver -ifname
interface_name
```

The following example takes an interface named **if0** on a SpinServer named **spsrv10** offline.

```
spinserver::> if configuration offline -ssid spsrv10 \
-ifname if0
```

To Bring an Interface Online

After making changes in the configuration of a virtual interface, use the **if configuration online** command to bring the interface back online. The syntax of the command is as follows:

```
if configuration online -ssid spinserver -ifname
interface_name
```

The following example brings an interface named **if0** online on a SpinServer named **spsrv10**.

```
spinserver::> if configuration online -ssid spsrv10 \
-ifname if0
```

Modifying Virtual Interfaces

After virtual interfaces are created, changes to the configuration of the interfaces are sometimes necessary. Use the **`if configuration modify`** command to modify configuration information for a virtual interface.

The **`if configuration modify`** command identifies a given interface by its SpinServer ID and interface name and enables you to modify the IP address and netmask of an interface. Changing the netmask changes the virtual interface's subnet assignment.

The syntax of the command is as follows:

```
if configuration modify -ssid spinserver -ifname
interface_name -ipaddress IP_address -netmask netmask
```

This following example changes the IP address to **`10.1.1.130`** of an interface named **`if0`** on a SpinServer named **`spsrv10`**. The interface has the netmask **`255.255.255.0`**.

```
spinserver::> if configuration modify -ssid spsrv10 \
-ifname if0 -ipaddress 10.1.1.130 -netmask 255.255.255.0
```

Reverting Virtual Interfaces

After a failover, you can revert virtual interfaces automatically to their home ports by using the **`if status revertvifs`** command. The command takes no parameters; issuing the command reverts all virtual interfaces to their home ports, as in the following example.

```
spinserver::> if status revertvifs
```

Adding a Failover Port

Failover is an important fault-tolerance function of systems that rely on constant accessibility, including high-availability (HA) configurations. Failover automatically and transparently redirects user requests from the failed or downed network to a backup network that takes over the operations of the primary network. Each virtual interface can have multiple failover ports. A virtual interface (IP address) migrates to the failover port in the event of a failure.

A failover port can be purely a standby port, or it can be a fully configured port with active virtual interfaces. Adding a failover port provides the system with a standby interface that automatically switches to a standby SpinServer or network if the primary network fails or is temporarily shut down for servicing. A failover port can be a port on either the local node or a remote node.

When a virtual interface is bound to a network port, that port is marked as the interface's primary port and serves as the default failover port in the unlikely event of multiple failures. If the virtual interface fails over and a failure occurs on the failover port, the interface can then be redirected to the original port.

Cluster link failover is enabled automatically, without explicit configuration. The cluster link fails over to the peer cluster link on the same node if the peer link is available.

To Add a Failover Port

To assign a failover port to the interface, use the **`if configuration addfailport`** command. The **`if configuration addfailport`** command identifies an interface by its SpinServer and interface name and assigns a failover port to the interface. The syntax of the command follows:

```
if configuration addfailport -ssid spinserver -ifname
interface_name -ssidto failover_spinserver -netportnameto
network_failover_port_name
```

The following example identifies an interface named **`if0`** on a SpinServer named **`spsrv10`** and adds a server port named **`serv2`** on a SpinServer named **`spsrv12`** as a failover port.

```
spinserver::> if configuration addfailport -ssid spsrv10 \
-ifname if0 -ssidto spsrv12 -netportnameto serv2
```

To Remove a Failover Port

To remove a failover port from the interface, use the **`if configuration removefailport`** command. The syntax of the command follows.

```
if configuration removefailport -ssid spinserver -ifname
interface_name -ssidfrom failover_spinserver
-netportnamefrom network_failover_port_name
```

The following example removes a failover port named **`if2`** on a SpinServer named **`spsrv12`** from an interface named **`if0`** on a SpinServer named **`spsrv10`**.

```
spinserver::> if configuration removefailport -ssid \
spsrv10 -ifname if0 -ssidfrom spsrv12 -netportnamefrom if2
```

9.3 Port Management Tasks

Ports represent physical Ethernet connections. The Ethernet ports are identified in Table 5-1 on page 21.

Configuring the Management Port

The management port on a SpinServer is the connection to the management interface. The management port is automatically configured when the SpinServer is initially started.

Configuring the Server Ports

A server port can be configured with one or more virtual interfaces with IP addresses. Virtual interfaces are named by the user and associated with an Ethernet port.

To Configure a Virtual Interface on the First Server Port

A virtual interface is assigned an IP address by using the **`if configuration new`** command. The syntax of the command follows.

```
if configuration new -ssid spinserver -ifname
interface_name -netportname network_port_name -ipaddress
IP_address -netmask netmask
```

The following example configures a virtual interface named **`if0`** on the server port named **`serv0`** (the first of two server ports) and the SpinServer named **`Server1`**. The port is given the IP address **`10.1.1.139`** and the netmask **`255.255.255.0`**.

```
spinserver::> if configuration new –ssid Server1 -ifname \
if0 -ipaddress 10.1.1.139 –netmask 255.255.255.0 \
-netportname serv0
```

To Configure a Virtual Interface on the Second Server Port

The following example configures a virtual interface named **`if0`** on the server port named **`serv1`** (the second of two server ports) and the SpinServer named **`Server1`**. The port is given the IP address **`10.1.1.140`** and the netmask **`255.255.255.0`**.

```
spinserver::> if configuration new –ssid Server1 -ifname \
if0 -ipaddress 10.1.1.140 –netmask 255.255.255.0 \
-netportname serv1
```

Configuring the Second Cluster Port

The first cluster port is configured when you use either the **`spinserver createcluster`** command or the **`spinserver joincluster`** command.

Note: To assure that all system components work together properly, the cluster port must be configured even if only a single SpinServer is being installed. Even if you intend to use only one of the cluster ports, it is *strongly recommended* that you configure both cluster ports at initial installation of the software.

If you did not configure the second cluster port at the time you initially installed the software, use the **if configuration new** command to configure the second cluster port.

After configuration is completed, the SpinServer can recognize disk drives and RAID sets. A cluster port can be configured with only one IP address. With the cluster port configured, accessing one SpinServer also provides you with access to files that reside on disks that are attached to another SpinServer. You can perform administrative tasks with RAID sets and virtual file systems on both SpinServers.

The following example configures a virtual interface named **if0** on the second cluster port (**clus1**) on a SpinServer named **Server1**. The port is given the IP address **10.1.1.145** and the netmask **255.255.255.0**.

```
spinserver::> if configuration new –ssid Server1 -ifname \
if0 -ipaddress 10.1.1.145 –netmask 255.255.255.0 \
-netportname clus1
```

Adding a Static Route to a Virtual Interface

You can add a static route to a virtual interface to make network traffic move more efficiently. Use the **if configuration addroute** command to add a route to a virtual interface. The command has the following syntax:

```
if configuration addroute -ssid spinserver -ifname
interface_name -destaddr destination_IP_address -destmask
destination_mask -nexthop next_hop_IP_address
```

The following example adds a route to a virtual interface named **vif0** on a SpinServer named **Server1**. The route has a destination IP address of **192.40.5.1**, a destination mask of **255.255.255.0**, and a next-hop IP address of **192.40.6.1**.

```
spinserver::> if configuration addroute -ssid Server 1 \
-ifname vif0 -destaddr 192.40.5.1 -destmask \
255.255.255.0 -nexthop 192.40.6.1
```

Deleting a Route from a Virtual Interface

Use the **if routes delete** command to delete a static route from a virtual interface. The command has the following syntax:

```
if routes delete -ssid spinserver -ifname interface_name
-destaddr destination_IP_address -destmask
destination_mask -nexthop next_hop_IP_address
```

The following example deletes a route from a virtual interface named **vif0** on a SpinServer named **Server1**. The route has a destination IP address of **192.40.5.1**, a destination mask of **255.255.255.0**, and a next-hop IP address of **192.40.6.1**.

```
spinserver::> if routes delete -ssid Server 1 -ifname \
vif0 -destaddr 192.40.5.1 -destmask 255.255.255.0 \
-nexthop 192.40.6.1
```

Displaying Information about Routes on a Virtual Interface

Use the **if routes show** command to display information about routes on one or more virtual interfaces. The command has the following syntax:

```
if routes show [-ssid spinserver -ifname interface_name]
```

The following example displays route information about a virtual interface named **vif0** on a SpinServer named **Server1**:

```
spinserver::> if routes show -ssid Server1 -ifname vif0
SpinServer    Name      Destination IP      Destination         Next Hop
    ID                      Address            Netmask

Server1       vif0      192.40.5.1          255.255.255.0       192.40.6.1
```

Displaying Network Ports

To display configuration information about all ports or information about one port, use the **ports show** command. To display information about one network port, use the command with the **-ssid** and **-netportname** parameters. To display configuration information about a list of ports on the network, use no parameters.

The command identifies the network port by its SpinServer ID and network port name. After identification is made, the following information about the network port or ports is displayed:

- SpinServer ID
- Name
- Maximum Transmission Unit (MTU)
- Media access control (MAC) address
- Online status (`up` or `down`)
- Link status
- Whether autonegotiation is enabled (`true` or `false`)
- Whether full duplexing is enabled (`true` or `false`)
- Port speed

The syntax of the command follows.

```
ports show [-ssid spinserver -netportname
network_port_name]
```

The following example displays configuration information about all network ports in the cluster.

```
spinserver::> ports show
SpinServer ID   Name    MTU   Status  Auto   Full    Speed
                                      Negoti Duplex

p416            clus0  1500    up     true   true   1000
p416            clus1  1500    up     true   true   1000
p416            mgmt0  1500    up
p416            serv1  1500    up     true   true   1000
p416            serv0  1500    up     true   true   1000
p48             mgmt0  1500    up
p48             clus0  1500    up     true   true   1000
p48             serv1  1500    up     true   true   1000
p48             clus1  1500    up     true   true   1000
p48             serv0  1500    up     true   true   1000
```

The following example displays configuration information about the network port named **serv1** on the SpinServer named **Server1**.

```
spinserver::> ports show -ssid Server1 -netportname serv1
SpinServer ID    Server1
Name             serv1
MTU              1500
MAC Address      00:04:76:F8:D2:FC
Status           up
Link Status      link
Auto Negotiate   true
Full Duplex      true
Speed            1000
```

Taking a Network Port Down

To make an IP address unavailable to clients while doing network maintenance tasks, take the port down before changes are made.

Note: Using this command takes down all virtual interfaces assigned to the port.

To Take a Network Port Down

To take down a port from the network, use the **ports down** command. The syntax of the command follows.

```
ports down -ssid spinserver -netportname network_port_name
```

The following example takes down the **serv1** port on the SpinServer named **Server1**.

```
spinserver::> ports down -ssid Server1 -netportname serv1
```

To Bring a Network Port Up

To bring up a port for the network, use the **ports up** command. The syntax of the command follows.

```
ports up -ssid spinserver -netportname network_port_name
```

The following example brings up the **serv1** port on the SpinServer named **Server1**.

```
spinserver::> ports up -ssid Server1 -netportname serv1
```

Modifying a Network Port

To modify the settings for a network port, use the **ports modify** command. An MTU is the largest physical packet size, measured in bytes, that a network can transmit. Any messages larger than the MTU are divided into smaller packets before being sent.

The **ports modify** command modifies the maximum transmission unit (MTU) size, autonegotiation setting, full-duplexing setting, and port speed of a network port. The changes take effect immediately, without the port needing to be restarted. By default, the server ports have an MTU value of **1500** bytes, an autonegotiation setting of **true**, a full-duplexing setting of **true**, and a speed of **1000**. The syntax of the command follows.

```
ports modify -ssid spinserver -netportname
network_port_name -mtu maximum_transmission_unit
-autonegotiate {true | false} -fullduplex {true | false}
-speed speed
```

The following example identifies a port named **serv1** on a server named **Server1** and modifies the maximum transmission unit size to **9000** bytes, the autonegotiation value to **true**, the full duplexing value to **false**, and the port speed to **1000**.

```
spinserver::> ports modify -ssid Server1 -netportname \
serv1 -mtu 9000 -autonegotiate true -fullduplex false \
-speed 1000
```

10.0 Managing RAID Components

This chapter discusses the creation and maintenance of RAID sets that comprise a specific list of physical disks. All of the disks in a single RAID enclosure, along with the disks in any attached expansion enclosure, are controlled by the pair of RAID controllers in the RAID enclosure. This pair of RAID controllers and their attached disks form a single RAID subsystem. The SpinServer has as many RAID subsystems as it has RAID enclosures.

Each RAID subsystem is configured individually. By using the management software, you can identify which disks belong to which RAID subsystem and the state of each disk. Normally, individual disks can be part of a RAID set, be used as a hot spare, or be labeled as uncommitted.

Note: The information in this chapter applies only to SpinStor storage. This information does not apply to Network Appliance FAS270c storage or to any third-party vendors' storage. For information on managing LUNs on Network Appliance storage, see the FAS270c and Data ONTAP documentation.

10.1 RAID Concepts

A RAID set is a group of disks that work together to form a logical disk that is larger, faster, and more reliable than the individual disks. After a RAID set has been created, it can be configured as one large, fast disk for the purpose of setting up file systems.

If a disk from a RAID set fails, a hot spare is used to replace it. Due to this redundancy, the RAID set can operate even after one of its disks has failed. If a hot spare disk is available, the RAID controller can reconstruct the data that is stored on the failed disk and place the data onto the hot spare. After this rebuilding process is complete, the RAID set regains its performance and reliability. There can be more than one hot spare disk in a RAID subsystem. In this case, when a hot spare is needed to rebuild a RAID set, any hot spare disk can be used. The hot spare does not need to be physically located near the rest of the RAID set.

If a hot spare is not available, the RAID set cannot be rebuilt until a hot spare is provided. You can do this by removing the disk that has failed and replacing it with a new disk.

An uncommitted disk can be used in different ways. It can be combined with other uncommitted disks to form a RAID set, or it can be designated as a hot spare.

Note: The **`raid`** commands report the sizes of disk drives in gigabytes; however, the commands interpret "gigabyte" as 2^{30} (1,073,741,824) bytes, not one billion bytes. Most disk-drive manufacturers interpret "gigabyte" as one billion bytes, so there can be a discrepancy between the output of the commands and the label on the disk. For instance, the commands can report a 36-GB drive as having a physical size of 34 GB.

10.2 RAID Management Tasks

If the SpinServer includes RAID sets that were preconfigured at the factory and background initialization is enabled, you are ready to create storage pools and virtual file systems. However, if the SpinServer does not include RAID sets that were preconfigured at the factory, creating a RAID set is your first RAID management task.

Furthermore, for every system, preconfigured or not, routine RAID management tasks must be done. These routine RAID management tasks are described in this chapter and include the following:

- Create RAID sets when a new SpinStor RAID enclosure is added to the system
- Modify configurations
- Locate RAID sets
- Replace faulty disks

Before using a RAID enclosure, you must confirm that it is in the expected configuration by checking the following:

- Determine that five or six disks in each enclosure can be correctly located by using the **`raid diskdrives locate`** command.
- Determine that RAID sets can be correctly located as they are created by using the **`raid sets locate`** command.
- The enclosure's temperature, power supply, and blower monitoring functions are working as demonstrated by the Fault LEDs on the enclosure and by using the **`raid enclosures show`** command.
- Half of the disks in each enclosure are using Channel 0 and the rest are using Channel 1, as shown by the **`raid diskdrives show`** command.
- The status of all RAID controllers is `Active`, as shown by the **`raid controllers show`** command. `Critical` status is not acceptable during initial configuration.

Managing RAID Sets

The device identification for a RAID set is in the format of **r***subsystem_number***s***raidset_number*, where *subsystem_number* indicates the RAID subsystem and *raidset_number* indicates the specific RAID set. For example, a device ID of **r0s1** indicates that the device is on RAID subsystem **0** and has a RAID set number of **1**.

Creating and Configuring a RAID Set

If the SpinServer does not include RAID sets that were preconfigured at the factory, or if you are adding a new SpinStor RAID enclosure to a system, your first step is to create a RAID set. If you set the new RAID set's status to `online`, the RAID set is available as soon as you create it. However, if you designate the status as `offline`, the RAID set is created but is not immediately available for storage-pool creation. Use the **raid sets makeonline** command to create a storage pool.

Note: When you create the first RAID set on a RAID subsystem, the creation process can take as long as two minutes. The process takes this long because a **Reset** command is issued to the RAID controllers after the creation process is completed. This is necessary because the strip size and cache line size are updated from the previous settings. This type of change requires the RAID controllers to be reset. If the RAID set on a RAID subsystem is not the first one created, the creation process is completed in a shorter time period.

RAID sets have five possible RAID levels: 0, 1, 3, 5, and 0+1. These levels have the following requirements:

- A RAID set of level 0 must have at least two (2) disks
- A RAID set of level 1 must have exactly two (2) disks
- A RAID set of level 3 must have at least three (3) disks
- A RAID set of level 5 must have at least three (3) disks
- A RAID set of level 0+1 must have at least four (4) disks

After the creation and configuration of a RAID set, initialization of the RAID set begins as a background process at the first use of the RAID set. For example, if you create storage pools and assign storage units (RAID sets or JBOD disks) to the storage pools, background initialization automatically starts. As background initialization proceeds on the RAID set, you can use the newly created storage pool attached to the RAID set to create a virtual server. After the virtual server is created, you can create a virtual interface, specifying an appropriate IP address and netmask. After creating and mounting a VFS, you can write data to the RAID set as background initialization continues.

Note: Background initialization can be a lengthy process. It can take approximately one hour for every 70 GB of disk capacity. For example, initializing a 240-GB RAID set can take about three and one-half hours.

To create a RAID set, use the **raid sets new** command. The syntax of the command follows.

```
raid sets new -raidlevel {0 | 1 | 3 | 5 | 0+1} -stripsize
{8 | 16 | 32 | 64} -writecacheenabled {true | false}
-status RAID_status -disks [list_of_disks]
```

The following example creates a new RAID set. Explanations of the values are listed after the example.

```
spinserver::> raid sets new -raidlevel 5 -stripsize 8 \
-writecacheenabled true -status Online -disks [r0d1, \
r0d2, r0d5, r0d6, r0d9, r0d10, r0d13, r0d14, r0d17, \
r0d18, r0d21, r0d22, r0d25, r0d26, r0d29, r0d30]
```

where

5 is the RAID level. Possible RAID levels are **0**, **1**, **3**, **5**, and **0+1**. The default is **5**.

8 is the RAID strip size in KB. The default setting is **32** KB. Other setting options are **16** KB, **32** KB, or **64** KB. A RAID *strip* is a piece of a single disk that is used for data storage. A RAID *stripe* is a set of the corresponding strips from all of the disks in a RAID set. The RAID strip size must match the RAID strip size that is currently set on the RAID controller. For best performance when using hardware-based RAID systems such as SpinStor disk arrays, NetApp recommends a total stripe size of 128 KB. To determine the proper strip size, divide 128 KB by the number of data disks, excluding parity disks, in the RAID set. For instance, if the RAID set has 16 data disks, use a strip size of 8 KB (that is, 128 KB/16 disks).

true enables the write cache. This is the default setting for the write cache. Enabling the write cache can result in higher system performance but can cause data loss if an error occurs.

Online is the status of the RAID set. This is the default value.

[r0d1, r0d2, r0d5, r0d6, r0d9, r0d10, r0d13, r0d14, r0d17, r0d18, r0d21, r0d22, r0d25, r0d26, r0d29] are the disk IDs of the disks to be included in the RAID set. The maximum number of disks in a single RAID set is 16. By default, all unused disk drives (up to 16) are included. If you are using the CLI and press the **Tab** key after the **-disks** parameter, all unused disk drives (up to 16) are automatically listed.

Displaying RAID Sets

After creating one or more RAID sets, you can display information about the RAID sets by using the **raid sets show** command. The command can be used without any parameters to display information about all RAID sets or with the **-deviceid** parameter to display information about a specific RAID set. The command displays the following information:

- RAID level
- Logical size (MB)
- Physical size (MB)
- Strip size (KB)
- Cache line size (KB)
- Initialization status
- Status of write cache (`enabled` or `disabled`)
- RAID set status (`online` or `offline`)
- Number of disks in the RAID set
- List of disks by ID number
- LongOp in progress status (LongOp refers to operations that take a long time to complete; for example, RAID set initialization, rebuilding, and consistency checking)

The syntax of the command follows.

```
raid sets show [-deviceid device_ID]
```

The following example displays information about a RAID set with the device ID **r0s0**.

```
spinserver::> raid sets show -deviceid r0s0
Device ID              r0s0
RAID Level             5
Logical Size (MB)      69424
Physical Size (MB)     104136
Strip Size (KB)        8
Cache Line Size (KB)   8
Initialized?           true
Write Cache Enabled?   false
Status                 Online
# Disks                3
Disks                  r0d2, r0d6, r0d10
LongOp In Progress     None
```

Modifying RAID Sets

To toggle the status of the write cache for a RAID set, use the **raid sets modify** command.

To enable the write cache, use **true** as the argument. To disable the write cache, use **false** as the argument. The default setting is **false**.

Enabling the write cache can result in higher system performance, but can cause data loss if an error occurs.

The syntax of the command follows.

```
raid sets modify -deviceid device_ID -writecacheenabled
{true | false}
```

The following example toggles the status of the write cache for a RAID set with an ID of **r0s0** to **true**.

```
spinserver::> raid sets modify -deviceid r0s0 \
-writecacheenabled true
```

Deleting a RAID Set

If data on a RAID set becomes obsolete and is moved to an archive, you can delete the RAID set and free the storage space for other projects by using the **raid sets delete** command. Deleting a RAID set removes all information stored on the physical disks in the RAID set. For this reason, if the data is to be maintained, it is important to transfer the data to another location before deleting the RAID set.

The syntax of the command follows.

```
raid sets delete -deviceid device_ID
```

The following example deletes a RAID set with the ID **r0s0**.

```
spinserver::> raid sets delete -deviceid r0s0
```

Note: RAID sets in a RAID subsystem must be deleted in the reverse order in which they were created. For example, if the RAID sets in a RAID subsystem are **r0s0, r0s1**, and **r0s2**, you must delete **r0s2** before deleting **r0s1**, and delete **r0s0** last.

Note: Long-running operations including background initialization, rebuilding, and consistency checking must be stopped or allowed to complete before you delete a RAID set. If the **raid sets delete** command is issued while a long-running operation is running, a warning message is displayed and the RAID set is not deleted.

Locating a RAID Set

To determine the physical location of the disks that are included in a specific RAID set, use the **raid sets locate** command. The **raid sets locate** command causes the front panel LEDs of the disks in the RAID set to blink red for the time specified in the command. The syntax of the command follows.

```
raid sets locate -deviceid device_ID -time minutes
```

The following example locates a RAID set with the ID **r0s0** and causes the front panel LEDs of the disks in the RAID set to blink red for **2** minutes.

```
spinserver::> raid sets locate -deviceid r0s0 -time 2
```

Displaying Disk IDs

To display the IDs of the disks that are included in a RAID set, use the **raid sets show-disks** command. The command can be used without any parameters to display information about the disks in all RAID sets in the system or with the **-deviceid** parameter to display information about a specific RAID set. The syntax of the command follows.

```
raid sets show-disks [-deviceid device_ID]
```

The following example displays information about the disks in a RAID set with the ID **r0s0**.

```
spinserver::> raid sets show-disks -deviceid r0s0
```

Starting Background Initialization

When background initialization is enabled, the initialization process runs in the background. Although the initialization process can be very lengthy when it is performed in the background, other system processes can continue. Initialization can take about one hour for every 70 GB of disk capacity. For example, initializing a 240-GB RAID set can take about three and one-half hours.

Background initialization is enabled by default. To manually start background initialization, use the **raid sets startbkginit** command. The command has the following syntax:

```
raid sets startbkginit -deviceid device_ID
```

The following example starts background initialization on a RAID set named **r0s0**:

```
spinserver::> raid sets startbkginit -deviceid r0s0
```

Checking the Status of Background Initialization

To check the status of background initialization, use the **raid sets bkginitstatus** command.

Note: The **raid sets bkginitstatus** command works only if background initialization is enabled. Background initialization is enabled by default.

The syntax of the command follows.

```
raid sets bkginitstatus -deviceid device_ID
```

The following example checks the status of the background initialization of a RAID set with the ID **r0s0**.

```
   spinserver::> raid sets bkginitstatus -deviceid r0s0
Action "Background Init Status" returned with the following status:
Not in progress
```

Troubleshooting Background Initialization

If a bad block is encountered on a disk during background initialization, the initialization process fails. To work around this problem, use one of the following approaches:

- Replace the bad disk and continue with background initialization, as follows:
  a. Identify the disk on which the bad block is located, and ensure that no problems exist on the other disks in the array.
  b. Pull the affected disk from the enclosure and wait until the status of the RAID set is reported as `Critical`.
  c. Reinsert the affected disk into the enclosure. The rebuilding process starts automatically.

  The advantage of this approach is that the RAID set remains online during the procedure. Disadvantages include the need to locate and manipulate the affected disk array, and the possibility of RAID-set corruption if there is a bad block on any other disk in the array.
- Use foreground initialization to initialize the RAID set, as follows:
  a. Ensure that background initialization is complete on all other RAID sets.
  b. Use the **filestorage vfs move** command to move any VFSs from the storage pool or pools that are located on the affected RAID set. For more information, see “Moving a Virtual File System” on page 166.
  c. Use the **storage pools delete** or **tools storage pools forcedelete** command to delete the affected storage pool or pools. For more information, see “Deleting a Storage Pool” on page 141 and “Forcibly Deleting a Storage Pool” on page 242, respectively.
  d. Use the **raid sets modify** command to disable background initialization on the affected RAID set or sets. For more information, see “Modifying RAID Sets” on page 107.
  e. Use the **raid sets startinit** command to initialize the affected RAID set or sets. For more information, see “Initializing a RAID Set”. Note that foreground initialization can be a lengthy process.
  f. When initialization has completed, recreate the storage pool or pools on the affected RAID set, then move the VFSs back to the storage pool or pools.

  The advantages of this approach are the ability to perform all tasks remotely through the management interface and the relative safety of the procedure (that is, because VFSs are moved off of the affected storage pool during the procedure). Disadvantages include the longer time requirement and the need for space on other storage pools to hold VFSs while the RAID set is being initialized.

After the RAID set is fully initialized, a bad block does not interfere with the correct operation of the RAID set.

Initializing a RAID Set

If background initialization is disabled, initialize a RAID set by using the **raid sets startinit** command. The syntax of the command follows.

```
raid sets startinit -deviceid device_ID
```

Note: Initializing a RAID set with the **raid sets startinit** command can be a lengthy process. Depending on the size and RAID level of the RAID set, it can take up to four hours. During this time, the RAID set is unusable for other processes. For these reasons, background initialization is recommended.

The following example initializes a RAID set with the device ID **r0s0**.

```
spinserver::> raid sets startinit -deviceid r0s0
```

To Stop Initialization of a RAID Set

To stop initialization of a RAID set when background initialization is disabled, use the **raid sets stopinit** command. The syntax of the command follows.

```
raid sets stopinit -deviceid device_ID
```

The following example stops the initialization of a RAID set with an ID of **r0s0**.

```
spinserver::> raid sets stopinit -deviceid r0s0
```

To Check the Status of Initialization

The degree of completion of the initialization of a RAID set can be determined by using the **raid sets initstatus** command. The syntax of the command follows.

```
raid sets initstatus -deviceid device_ID
```

The following example display the initialization status of a RAID set with an ID of **r0s0**.

```
spinserver::> raid sets initstatus -deviceid r0s0
```

Taking a RAID Set Offline

When maintenance on more than one disk in a RAID set is necessary, you must take the RAID set offline before you begin the disk-maintenance tasks. Such tasks include the physical replacement of one or more malfunctioning disks or changing the physical location of a disk; that is, moving a disk from one slot to another, thereby causing the disk ID to change within the disk enclosure.

To take a RAID set offline, use the **raid sets makeoffline** command. The syntax of the command follows.

```
raid sets makeoffline -deviceid device_ID
```

The following example takes a RAID set with an ID of **r0s0** offline.

```
spinserver::> raid sets makeoffline -deviceid r0s0
```

Bringing a RAID Set Online

After completing disk-maintenance tasks, bring a RAID set back online by using the **raid sets makeonline** command.

Note: The **raid sets makeonline** command is used to bring a RAID set back online after the **raid sets makeoffline** command has been used to take a RAID set offline. If the RAID set already is online, using the **raid sets makeonline** command has no effect.

The syntax of the command follows.

```
raid sets makeonline -deviceid device_ID
```

The following example brings a RAID set with an ID of **r0s0** online.

```
spinserver::> raid sets makeonline -deviceid r0s0
```

Rebuilding a RAID Set

When maintenance on more than one disk in a RAID set is necessary, you must rebuild the RAID set by using the **raid sets startrebuild** command after the maintenance has been completed. Such disk-maintenance tasks include the physical replacement of one or more malfunctioning disks or changing the physical location of a disk; that is, moving a disk from one slot to another, thereby causing the disk ID to change within the disk enclosure. The syntax of the command follows.

```
raid sets startrebuild -deviceid device_ID
```

The following example begins the process of rebuilding a RAID set with the ID **r0s0**.

```
spinserver::> raid sets startrebuild -deviceid r0s0
```

Stopping the Rebuilding of a RAID Set

Use the **raid sets stoprebuild** command to stop the rebuilding process of a specified RAID set. The syntax of the command follows.

```
raid sets stoprebuild -deviceid device_ID
```

The following example stops the rebuilding process of a RAID set with the ID **r0s0**.

```
spinserver::> raid sets stoprebuild -deviceid r0s0
```

Checking the Status of the Rebuilding of a RAID Set

Use the **raid sets rebuildstatus** command to check the status of the rebuilding process of a RAID set. The syntax of the command follows.

```
raid sets rebuildstatus -deviceid device_ID
```

The following example checks the status of the rebuilding process of a RAID set with the ID **r0s0**.

```
spinserver::> raid sets rebuildstatus -deviceid r0s0
```

Checking the Consistency of RAID Set Configuration

By using the **raid sets startconsistencycheck** command, you can check that the parity information stored in a RAID set is correctly established. The RAID set uses parity to ensure that the data across a stripe is valid and that if a disk fails, the lost data can be accurately reconstructed.

Starting a RAID Set Consistency Check

To start a RAID set consistency check, use the **raid sets startconsistencycheck** command. Use the **-restoreonerror** parameter to set whether or not consistency errors are corrected; the default is **false** (that is, consistency errors are not corrected). The syntax of the command follows.

```
raid sets startconsistencycheck -deviceid device_ID
-restoreonerror {true | false}
```

The following example starts a consistency check on the RAID set with ID **r0s0**, and does not correct errors in consistency (**false**).

```
spinserver::> raid sets startconsistencycheck -deviceid \
r0s0 -restoreonerror false
```

Stopping a RAID Set Consistency Check

Use the **raid sets stopconsistencycheck** command to halt a consistency check on a specified RAID set. The syntax of the command follows.

```
raid sets stopconsistencycheck -deviceid device_ID
```

The following example stops a consistency check on the RAID set with ID **r0s0**.

```
spinserver::> raid sets stopconsistencycheck -deviceid \
r0s0
```

Checking the Status of a RAID Set Consistency Check

To determine the degree of completion of the consistency check of a RAID set, use the **raid sets consistencycheckstatus** command. The syntax of the command follows.

```
raid sets consistencycheckstatus -deviceid device_ID
```

The following example checks the status of a consistency check being run on the RAID set with ID **r0s0**.

```
spinserver::> raid sets consistencycheckstatus -deviceid \
r0s0
Action "Consistency Check Status" returned with the following
status:
In progress 50%
```

Forcibly Deleting a RAID Set

In some circumstances it can be necessary to forcibly delete a RAID set that cannot be deleted by using the **raid sets delete** command. In this case, use the **tools raid sets forcedelete** command. The command has the following syntax:

```
tools raid sets forcedelete -deviceid device_ID
```

The following example forcibly deletes a RAID set with the ID **r0s0**:

```
tools raid sets forcedelete -deviceid r0s0
```

Managing RAID Subsystems

A RAID subsystem consists of all the RAID controllers, disk enclosures, disk drives, and RAID sets that are connected to a single SpinServer. A RAID set is a grouping of three or more disks to create a storage space that is larger in size and more reliable than a single disk drive.

The device identification for a RAID subsystem is in the format of **r***subsystem_number*, where *subsystem_number* indicates the specific RAID subsystem. An example of an ID for the RAID subsystem is **r0**.

Displaying RAID Subsystem Information

The **raid subsystems show** command identifies a given RAID subsystem or all RAID subsystems on the SpinServer system. After identification is made, the following information about the RAID subsystem or subsystems is displayed by the system:

- Subsystem ID
- Names of servers connected to the RAID subsystem
- Identification of the RAID controllers connected to the RAID subsystem
- Disk enclosures connected to the RAID subsystem
- Number of disk enclosures connected to the RAID subsystem
- Number of disks in the RAID subsystem
- Number of RAID sets included in the RAID subsystem
- Background initialization setting (`true` or `false`).
- Rebuild rate (percentage)

The syntax of the command follows.

raid subsystems show [**-id** *ID*]

The following example displays configuration information about a RAID subsystem with an ID of **r0**.

```
spinserver::> raid subsystems show -id r0
ID                      r0
Status                  Online
Host Connections        (p417 -- r0c0) , (p417 -- r0c1)
SpinServers Connected   p417
Controllers             r0c0, r0c1
RAID sets               r0s0, r0s1, r0s2, r0s3, r0s4, r0s5
Enclosures              r0e1, r0e2
Battery                 r0b1
# Enclosures           2
# Disks                32
# RAID sets            6
Background Init         true
Rebuild Rate (%)        0
```

Displaying RAID Subsystem Events

The **raid subsystems events** command displays events that have occurred on a particular RAID controller. This is useful for diagnosing problems with RAID arrays. The command has the following syntax:

raid subsystems events -id *ID* **-number** *number_of_events*

In this command, **-id** *ID* specifies the ID of the RAID controller whose events are to be displayed, and **-number** *number_of_events* specifies the number of events that are to be displayed. If this value is higher than the number of messages from the RAID controller, then all of the messages from the controller are displayed.

The following command displays the first **10** events for the RAID subsystem **r0**.

```
   spinserver::> raid subsystems events -id r0 -number 10
Action "Raid Events" returned with the following status:
0: System (sys: r0) started successfully.[12 Aug 09:43:19]
1: Automatic reboot count has changed on System (sys: r0).[12 Aug
09:43:19]
2: Dual controllers enabled on System (sys: r0).[12 Aug 09:43:19]
3: Updated partner's status on System (sys: r0).[12 Aug 09:43:19]
4: Dual controllers entered nexus on System (sys: r0).[12 Aug
09:43:19]
5: System (sys: r0) started successfully.[12 Aug 09:43:19]
6: Automatic reboot count has changed on System (sys: r0).[12 Aug
09:43:19]
7: Dual controllers enabled on System (sys: r0).[12 Aug 09:43:19]
8: Updated partner's status on System (sys: r0).[12 Aug 09:43:19]
9: Controller started completed  on System (sys: r0).[12 Aug
09:43:19]
```

Modifying RAID Subsystem Configuration

Use the **raid subsystems modify** command to modify the background initialization setting and the rebuild rate for the RAID subsystem, expressed as a percentage. The syntax of the command follows.

```
raid subsystems modify -id ID -bkginit {true | false}
-rebuildrate rebuild_rate
```

The following example enables background initialization (**true**) and sets the rebuild rate to **5** percent of the controller's CPU capacity for a RAID subsystem named **r0**.

```
spinserver::> raid subsystems modify -id r0 -bkginit \
true -rebuildrate 5
```

Disabling and Enabling SMART for a RAID Subsystem

The Self-Monitoring, Analysis, and Reporting Technology (SMART) is an open standard for developing disk drive and software systems that automatically monitors a disk's functional status and reports potential problems to the system. The reports from SMART enable you to proactively respond to disk failures. SMART is enabled by default.

Disabling SMART

To disable the SMART for all the disk drives of a RAID subsystem, use the **raid subsystems disablesmart** command. The syntax of the command follows.

raid subsystems disablesmart -id *ID*

The following example disables SMART for a RAID subsystem with the ID **r0**.

```
spinserver::> raid subsystems disablesmart -id r0
```

Enabling SMART

To enable SMART for all the disk drives of a RAID subsystem, use the **raid subsystems enablesmart** command. The syntax of the command follows.

raid subsystems enablesmart -id *ID*

The following example enables SMART for a RAID subsystem with the ID **r0**.

```
spinserver::> raid subsystems enablesmart -id r0
```

Discovering Disk Enclosures

The **raid subsystems discoverenclosures** command forces the SpinServer system to recognize a newly attached disk enclosure, either RAID or JBOD. The syntax of the command follows.

raid subsystems discoverenclosures -id *ID*

The following example forces the RAID subsystem named **r0** to recognize a newly attached disk enclosure.

```
spinserver::> raid subsystems discoverenclosures -id r0
```

Updating Disk Firmware

The **raid subsystems updatediskfirmware** command updates the firmware on all physical disks in a RAID subsystem from an absolute file path specified by the value of the **-imagefile** parameter. Note the following before upgrading disk firmware on a RAID subsystem:



Caution: Upgrading the disk firmware can possibly but rarely cause data loss. It is recommended that you back up all of the data on the RAID subsystem or copy it to another storage pool before upgrading the disk firmware

- If a RAID set is in a critical state, it is not fault tolerant, and its disks' firmware cannot be upgraded until the RAID set is rebuilt.
- If a failure occurs during the upgrade process, some of the disks can have the new firmware while other disks do not. In addition, one of the RAID sets can become critical because it has lost use of a disk. To restore normal operation, use a standard recovery technique such as rebuilding the RAID set by using a spare disk.
- To modify the firmware on a specific disk, use the **raid diskdrives update-**

firmware command.

The syntax of the command follows.

```
raid subsystems updatediskfirmware -id ID -imagefile
image_file_path
```

The following example updates the firmware on all physical disks in a RAID subsystem with the ID **r0** from a firmware image located at **/usr/ima/spin.ima**.

```
spinserver::> raid subsystems updatediskfirmware -id r0 \
-imagefile /usr/ima/spin.ima
```

Resetting the RAID Subsystem

In the event of an error, use the **raid subsystems reset** command to reset the RAID subsystem. The RAID subsystem needs to be reset to refresh its status. The syntax of the command follows.

```
raid subsystems reset -id ID
```

The following example resets a RAID subsystem with the ID **r0**.

```
spinserver::> raid subsystems reset -id r0
```

Dumping the RAID Subsystem Configuration Information

To provide support personnel with configuration information to assist you in debugging a SpinServer system's RAID controllers, use the **raid subsystems dumpconfig** command. The command writes configuration information to a file specified by the value of the **-file** parameter.

The syntax of the command follows.

```
raid subsystems dumpconfig -id ID -file file_path
```

The following example prints the configuration information for a RAID subsystem with the ID **r0** to a file named **/tmp/r0.dump**.

```
spinserver::> raid subsystems dumpconfig -id r0 -file \
/tmp/r0.dump
```

Managing Disks in a RAID Subsystem

Managing disk drives in a RAID subsystem includes the tasks of locating a particular disk or set of disks, displaying disk status information, and preparing a disk for removal from the enclosure.

The ID of the disks in a RAID subsystem is in the format **r***subsystem_number***d***disk_number*, where *subsystem_number* indicates the RAID subsystem and *disk_number* indicates the specific disk drive.

Displaying System Disk Information

Use the **raid diskdrives show** command to display information about all the disks in a RAID subsystem or information about one disk. The syntax of the command follows.

```
raid diskdrives show [-id ID]
```

The following example displays information about all disks in a RAID subsystem.

```
spinserver::> raid diskdrives show
Device ID  Capacity     Status      Used By   Channel Target

r0d1       33 GB    Online         r0s0       1       0
r0d10      33 GB    Online         r0s1       0       5
r0d13      33 GB    Online         r0s1       1       6
r0d14      33 GB    Online         r0s1       0       7
r0d17      33 GB    Online         r0s1       1       8
r0d18      33 GB    Online         r0s1       0       9
r0d2       33 GB    Online         r0s0       0       1
r0d21      33 GB    Online         r0s2       1       10
r0d22      33 GB    Online         r0s2       0       11
r0d25      33 GB    Online         r0s2       1       12
r0d26      33 GB    Online         r0s2       0       13
r0d29      33 GB    Online         r0s2       1       14
r0d5       33 GB    Online         r0s0       1       2
r0d6       33 GB    Online         r0s0       0       3
r0d9       33 GB    Online         r0s0       1       4
r1d33      33 GB    Online         r1s3       1       16
r1d34      33 GB    Online         r1s3       0       17
r1d37      33 GB    Online         r1s3       1       18
r1d38      33 GB    Online         r1s3       0       19
r1d41      33 GB    Online         r1s3       1       20
r1d42      33 GB    Online         r1s4       0       21
-- Type 'q' to quit and any key to continue --
```

Locating a Disk Drive

The **raid diskdrives locate** command locates a disk in a system. The front-panel LED of the disk blinks red for the number of minutes specified by the value of the **-time** parameter to indicate the location of the disk in the SpinStor. The default value of the **-time** parameter is **0**, indicating that the LED blinks for less than one minute. The syntax of the command follows.

```
raid diskdrives locate -deviceid device_ID -time minutes
```

The following example locates a disk named **r0d14** in a RAID subsystem and causes the front-panel LED of the disk to blink red for **2** minutes.

```
spinserver::> raid diskdrives locate -deviceid r0d14 \
-time 2
```

Checking Sense Data

The **raid diskdrives checksense** command returns sense data for a specified disk drive. Sense data is useful for diagnosing problems with the physical disk drive. The syntax of the command follows.

```
raid diskdrives checksense -deviceid device_ID
```

The following example returns sense data for a disk drive named **r0d1**.

```
spinserver::> raid diskdrives checksense -deviceid r0d1
Action "Check Sense" returned with the following status:
70 00 00 04 45 dc 6d 0a 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Creating a Hot Spare Disk

The **raid diskdrives makehotspare** command identifies a disk by its disk ID and designates its status as a hot spare in a RAID subsystem. A hot spare is a replacement disk that can be used to rebuild all user data from a failed disk in the RAID subsystem. Use this command to create as many hot spares as the RAID subsystem requires. The syntax of the command follows.

```
raid diskdrives makehotspare -deviceid device_ID
```

The following example creates a hot spare for the RAID subsystem on the disk with ID **r0d154**.

```
spinserver::> raid diskdrives makehotspare -deviceid \
r0d154
```

Replacing a Disk in an Enclosure

Removing and replacing a disk from a disk enclosure involves the following steps:

1. Take the disk offline. See “Taking a Disk Offline” on page 121 for more information.
2. Inform the system that the disk is going to be removed. See “Preparing a Disk for Removal” on page 121 for more information.
3. Remove the disk.
4. Insert the replacement disk.
5. Bring the replacement disk online. See “Bringing a Disk Online” on page 121 for more information.

Taking a Disk Offline

Before removing a disk from a RAID or attached expansion JBOD enclosure, you must take the disk offline.

Taking a disk offline is necessary when doing maintenance on the disk or replacing a faulty disk with a new one. The **raid diskdrives makeoffline** command identifies a disk by its disk ID and designates it for offline status. The syntax of the command follows.

```
raid diskdrives makeoffline -deviceid device_ID
```

The following example takes a disk with ID **r0d14** offline.

```
spinserver::> raid diskdrives makeoffline -deviceid r0d14
```

Preparing a Disk for Removal

Before physically removing a disk from a system, use the **raid diskdrives prepareforremoval** command to alert the RAID subsystem to the removal of the disk. The syntax of the command follows.

```
raid diskdrives prepareforremoval -deviceid device_ID
```

Important: After disengaging a drive from the enclosure, wait 30 seconds before pulling the carrier out of the enclosure. This enables the drive to properly spin down and shut down before you completely remove it from the enclosure.

The following example tells the system to prepare to have the disk with ID **r0d14** removed.

```
spinserver::> raid diskdrives prepareforremoval \
-deviceid r0d14
```

Bringing a Disk Online

After completing the required maintenance on the disk, or when replacing a disk with a new one, use the **raid diskdrives makeonline** command to bring a disk back online. The syntax of the command follows.

```
raid diskdrives makeonline -deviceid device_ID
```

The following example brings the disk with ID **r0d12** online.

```
spinserver::> raid diskdrives makeonline -deviceid r0d12
```

Performing a Disk Self-Test

Use the **raid diskdrives selftest** command to perform a check of the status of a physical disk in a RAID subsystem. The syntax of the command follows.

```
raid diskdrives selftest -deviceid device_ID -option
options
```

The following example checks the status of a physical disk with the ID **r0d12** by using default options.

```
spinserver::> raid diskdrives selftest -deviceid r0d12 \
-option default
```

Checking the Status of a Disk Self-Test

To determine the degree of completion of a disk self-test, use the **raid diskdrives selftestst atus** command. The syntax of the command follows.

```
raid diskdrives selfteststatus -deviceid device_ID
```

The following example checks the status of the self-test of a disk with the ID **r0d12**.

```
spinserver::> raid diskdrives selfteststatus -deviceid \
r0d12
```

Stopping a Disk Self-Test

To stop a disk self-test, use the **raid diskdrives stopselftest** command. The syntax of the command follows.

```
raid diskdrives stopselftest -deviceid device_ID
```

The following example stops the self-test of a disk with the ID **r0d12**.

```
spinserver::> raid diskdrives stopselftest -deviceid r0d12
```

Updating Disk Firmware

The **raid diskdrives updatefirmware** command updates the firmware on a specific disk drive in a RAID subsystem. The command has the following syntax.

```
raid diskdrives updatefirmware -deviceid device_ID
-imagefile file_path
```

Note: To modify the firmware on all physical disks in a RAID subsystem, use the **raid subsystems updatediskfirmware** command.

The following example updates the firmware of a disk with the ID **r0d4** from a firmware image located at **/usr/ima/spin.ima**.

```
spinserver::> raid diskdrives updatefirmware -deviceid \
r0d4 -imagefile /usr/ima/spin.ima
```

Checking the Status of a Disk Firmware Update

To determine the degree of completion of the update of the firmware on a specific disk in a RAID subsystem, use the **raid diskdrives firmwareupdatestatus** command. The command has the following syntax.

```
raid diskdrives firmwareupdatestatus -deviceid device_ID
```

The following example displays the status of the firmware update for a disk with ID **r0d4**.

```
spinserver::> raid diskdrives firmwareupdatestatus \
-deviceid r0d4
```

Managing SpinStor Disk Enclosures

Managing disk enclosures in a RAID subsystem includes the tasks of verifying the enclosure settings and silencing the alarms in the event of an error. The ID of a disk enclosure in a RAID subsystem is in the format **r***subsystem_number***e***enclosure_number*, where *subsystem_number* indicates the RAID subsystem and *enclosure_number* indicates the specific enclosure.

Displaying Disk Enclosure Settings

To view the settings of RAID or attached expansion JBOD enclosures, use the **raid enclosures show** command. Disk enclosures are attached by Fibre Channel to the SpinServer. The **raid enclosures show** command displays information about one disk enclosure or all disk enclosures. To display information about one disk enclosure, use the command with the **-deviceid** parameter. To display a list of all enclosures in the system, use no parameters.

After identification is made, the following information about the disk enclosure or enclosures is displayed by the system:

- Device name
- Status
- Manufacturer
- Model number
- RAID subsystem
- World Wide Name
- Enclosure service type
- Number of populated disk slots
- Controller status

- Fan status
- Power-supply status
- Temperature status
- Alarm status
- UPS status
- Daisy-chain status (`true` or `false`)
- System type (`RAID` or `JBOD`)
- Temperature warning settings
- Temperature failure settings
- Alarm-enablement status

The command has the following syntax.

raid enclosures show [**-deviceid** *device_ID*]

The following example displays information about all RAID enclosures.

```
   spinserver::> raid enclosures show
Device ID       Model         Status  Access  Type

r0e1      FR15-F22-2SM3.    OK       OK      SES
r1e1      FR15-F22-2Sxx.    OK       OK      SES
r1e2      FR15-F22-2SM3.    OK       OK      SES
```

Locating an Enclosure

To locate a RAID enclosure, use the **raid enclosures locate** command. This command causes all of the LEDs on the enclosure to flash. The syntax of the command follows.

raid enclosures locate -deviceid *device_ID*

The following example causes all of the LEDs on the enclosure named **r0e0** to flash.

```
spinserver::> raid enclosures locate -deviceid r0e0
```

Silencing the Alarm

In the event that there is a problem with a disk enclosure, a beeping sound emanates from the disk enclosure. To silence this alarm, use the **raid enclosures mute** command. The syntax of the command follows.

raid enclosures mute -deviceid *device_ID*

The following example silences the alarm on a disk enclosure with the ID **r0e1**.

```
spinserver::> raid enclosures mute -deviceid r0e1
```

Managing RAID Controllers

Managing the RAID controllers in a RAID subsystem includes the tasks of monitoring controller information and status, updating controller firmware, and taking the controller's partner offline for maintenance. The ID of the RAID controller is in the format **r***subsystem_number***c***controller_number*, where *subsystem_number* indicates the RAID subsystem and *controller_number* indicates the specific RAID controller.

Displaying RAID Controller Information

The **raid controllers show** command displays information about one RAID controller or all RAID controllers in the system. After identification is made, the following information about the RAID controller or controllers is displayed by the system:

- Device name
- Status
- Model number
- Firmware version number
- RAID subsystem
- Enclosure ID
- World Wide Name
- Cache size
- Memory size
- Battery backup unit
- Number of host ports
- Number of drive channels
- Bus type
- Bus width
- Bus speed
- Number of physical devices connected to the controller (including itself and its partner)
- Number of logical drives
- Partner status
- Fault tolerant status
- Port device name
- Number of ports

The command has the following syntax:

```
raid controllers show [-deviceid device_ID]
```

Use the following example to receive information about all RAID controllers in a system.

```
   spinserver::> raid controllers show
Device ID      Status         Model  Version  Partner

r0c0      Active            FFx-2   8.50     Active
r0c1      Active            FFx-2   8.50     Active
```

Taking a RAID Controller's Partner Offline

When maintenance is required on a RAID controller's partner, such as replacing a faulty RAID I/O module, you must take the RAID controller's partner controller offline. To take a RAID controller's partner controller offline, use the **raid controllers setpartneroffline** command. The command has the following syntax:

```
raid controllers setpartneroffline -deviceid device_ID
```

The following example sets the partner of a RAID controller with the ID **r0c1** offline:

```
spinserver::> raid controllers setpartneroffline \
-deviceid r0c1
```

Bringing a RAID Controller's Partner Online

When maintenance is complete on a RAID controller's partner, use the **raid controllers setpartneronline** command to bring the RAID controller's partner controller back online. The command has the following syntax:

```
raid controllers setpartneronline -deviceid device_ID
```

The following example sets the partner of a RAID controller with the ID **r0c1** online:

```
spinserver::> raid controllers setpartneronline \
-deviceid r0c1
```

Updating Firmware for RAID Controllers

Use the **raid controllers updatefirmware** command to update the firmware for a RAID controller in the system. The command has the following syntax:

```
raid controllers updatefirmware -deviceid device_ID
-imagefile file_path
```

The following example updates the firmware for a RAID controller with the ID **r0c10** with the image located at **/usr/ima/spin.ima**.

```
spinserver::> raid controllers updatefirmware -deviceid \
r0c10 -imagefile /usr/ima/spin.ima
```

Managing the Battery Backup Unit

The battery backup unit (BBU) maintains the RAID controller's data in random access memory (RAM) in the event that AC power is lost to the disk storage system. The BBU must be replaced when it can no longer take or hold a charge.

The device identification for the battery backup unit is in the format of **r***subsystem_number***b***battery_ID*, where *subsystem_number* is the ID of the RAID subsystem and **b** indicates the BBU.

Displaying BBU Information

The **raid bbu show** command displays information about the BBU. This command can be used without any parameters to display information about all battery backup units in the RAID subsystem or with the **-deviceid** parameter to display information about a battery backup unit in a specific RAID subsystem.

After identification is made, the following information about the BBU is displayed by the system:

- Device ID
- Type of battery (for example, NiCAD)
- Version number of the RAID controller
- Status
- Current power (minutes)
- Maximum power (minutes)
- Charge level (%)
- Low-power threshold (minutes)

The current power level of the battery is expressed in minutes of charge. The value changes as the battery is discharged and recharged.

The maximum power is the highest level of charge of the battery. It is expressed in minutes of charge. It changes as the battery ages and is reconditioned. The older the battery, the less charge it can hold.

The charge level is expressed in percentage of the battery's current charge level. The battery is charged to its maximum level when the charge level is 100%.

The syntax of the command follows.

```
raid bbu show [-deviceid device_ID]
```

The following example displays information for a BBU with the ID **r0b1.**

```
spinserver::> raid bbu show -deviceid r0b1
Device ID                 r0b1
Type                      NiCAD
Version                   1
```

```
Status                       OK
Current Power (Min)          1500
Max Power (Min)              1500
Charge Level (%)             100
Low Power Threshold (Min)    1200
```

Modifying the Low-Power Threshold Setting

The low-power threshold is expressed in minutes of charge that the BBU can hold. If the power level drops below the low-power threshold that you set, an alarm is triggered.

Use the **raid bbu modify** command to modify the low-power threshold of the BBU. The syntax of the command follows.

```
raid bbu modify -deviceid device_ID -threshold threshold
```

The following example sets the low-power threshold of the BBU with ID **r0b1** to a value of **1200**, so that an alarm is triggered when the BBU's reserve power drops to 1,200 minutes.

```
spinserver::> raid bbu modify -deviceid r0b1 -threshold \
1200
```

Reconditioning the BBU

A full battery-reconditioning cycle consists of discharging and recharging the battery. The reconditioning cycle must start with a fully charged battery. Normally, the battery is automatically recharged, but in a new system or in a system that has experienced a power failure, the battery must go through the reconditioning process. A fast recharge occurs when the RAID controller loses power and is powered on again.

The reconditioning cycle must be completed without an interruption of power. Also, do not interrupt the reconditioning cycle by using the **raid bbu stop** command to stop the reconditioning process. If the cycle is interrupted for any reason, the battery must be recharged and the reconditioning process restarted. The reconditioning process can take several hours.

Under normal usage, a BBU needs to be reconditioned if it has gone through 30 or more discharge/recharge cycles.

Initiating a Reconditioning of a BBU

Use the **raid bbu recondition** command to initiate the reconditioning of a BBU. The syntax of the command follows.

```
raid bbu recondition -deviceid device_ID
```

The following example initiates the reconditioning of a BBU with the ID **r0b1**.

```
spinserver::> raid bbu recondition -deviceid r0b1
```

Stopping the Reconditioning of a BBU

Use the **raid bbu stop** command to halt the reconditioning of a BBU. The command has the following syntax.

raid bbu stop -deviceid *device_ID*

The following example stops the reconditioning of a BBU with the ID **r0b1**.

```
spinserver::> raid bbu stop -deviceid r0b1
```

Initiating a Fast Charge

If a BBU is fully discharged, it must be recharged before use by the RAID controller.

To initiate a fast charge on the BBU, use the **raid bbu fastcharge** command; its syntax follows.

raid bbu fastcharge -deviceid *device_ID*

The following example initiates a fast charge on the BBU with ID **r0b1**.

```
spinserver::> raid bbu fastcharge -deviceid r0b1
```

If you are using the browser interface, the following confirmation message appears:

```
Are you sure you want to initiate a fast charge on the
battery?
```

Click **OK** to confirm.

Scheduling RAID Operations

You can schedule automatic consistency checks of RAID sets by using the **raid schedules** commands.

Creating a RAID Schedule

The **raid schedules new** command creates a new RAID schedule. A schedule consists of a schedule name, an action to be performed, the device or devices on which the action is performed, and the time interval at which the action is performed. The schedule can initially be enabled or disabled. Currently, the only supported value for the **-raidaction** parameter is **ConsistencyCheck** for a RAID consistency check. The command has the following syntax:

raid schedules new -name *schedule_name* **-raidaction** *action_name* **-devices** *device_name_or_pattern* **-schedule** {**#-Minutely** | **hourly** | **daily** | **weekly** | **monthly** | **yearly**} **-enabled** {**true** | **false**}

The following command example creates a RAID schedule named **raid.sched.1**. It runs a consistency check on all devices named **r1s*** on a **daily** basis. It is enabled.

```
spinserver::> raid schedules new -name raid.sched.1 \
-raidaction ConsistencyCheck -devices r1s* -schedule \
daily -enabled true
```

Displaying Information about RAID Schedules

Use the **raid schedules show** command to display information about a single RAID schedule (if the **-name** parameter is specified) or all RAID schedules. The command displays the following information:

- Schedule name
- Action
- Device or devices
- Schedule (time interval)
- Whether or not the schedule is enabled
- Time of next execution (GUI only)
- Time of last execution

The command has the following syntax:

```
raid schedules show [-name schedule_name]
```

The following command example displays information about all RAID schedules.

```
spinserver::> raid schedules show
Name         Action       Devices        Schedule      Enabled?    Last Execution

raid.sched.1 ConsistencyCheck r0s6       weekly        true
```

Enabling or Disabling a RAID Schedule

The **raid schedules modify** command enables or disables a RAID schedule. The command has the following syntax:

```
raid schedules modify -name schedule_name -enabled {true | false}
```

The following example disables a RAID schedule named **raid.sched.1**.

```
spinserver::> raid schedules modify -name raid.sched.1 \
-enabled false
```

Deleting a RAID Schedule

The raid schedules delete command deletes a RAID schedule. The command has the following syntax:

raid schedules delete -name *schedule_name*

The following example deletes a RAID schedule named **raid.sched.1**.

```
spinserver::> raid schedules delete -name raid.sched.1
```

11.0 Managing Storage

This chapter provides information about working with storage pools, including creating storage pools out of storage units, maintaining storage pools, and adding storage pools into a high-availability configuration.

The **`storage`** command node displays all the storage pools that are located on a SpinServer. Storage pools comprise one or more storage units. The primary purposes of storage pools are as follows:

- To provide a logical block address space that is larger than is possible with a single RAID set
- To provide a secure and robust log-recovery mechanism
- To serve as a container for virtual file systems

The ranges and limitations for storage pools are as follows:

- The minimum number of storage units per storage pool is 1. The maximum number of storage units per storage pool is 128.
- The minimum number of storage pools per SpinServer is 1. The maximum number of storage pools per SpinServer is 16.
- Storage pools can be expanded, but not reduced. To reduce the size of a storage pool, you must delete it and recreate it.
- Sizes are listed in 1-MB (1,048,576-byte) blocks.
- The device IDs for storage pools and storage units must be unique within a cluster.

With the optional SpinHA software, storage pools can also be set up in a high-availability (HA) configuration. An HA configuration requires at least two SpinServers in a cluster to provide service to storage pools.

11.1 Storage Concepts

In the SpinServer model, data is stored on logical devices called storage pools. Each storage pool is backed by one or more storage units. A storage unit is the abstraction of the physical storage (RAID sets and JBOD disks) that is attached to the SpinServer cluster.

High Availability

You can use two or more SpinServers and the optional SpinHA software to set up a high-availability (HA) system that has no single point of failure. HA systems are used to provide storage to applications that require the utmost in availability, such as high-volume databases. In an HA system, two or more SpinServers share responsibility for a particular storage pool. If one SpinServer fails, the other automatically takes control of its disks and continues without losing any committed data.

The HA configuration guards against a SpinServer failure but not against a catastrophic failure of multiple disks (a failure that is beyond the ability of the RAID subsystem to contain). Because RAID sets are highly reliable, the HA configuration is a preferred solution for many storage administrators.

In a high-availability system, one SpinServer acts as the *primary* and another SpinServer acts as the *active secondary*. If additional SpinServers are available in the same cluster, they can be designated as *passive secondaries*. The roles of these different machines are as follows:

- Primary—The primary machine receives client requests and propagates them to both the storage pool and, over a *failover stream*, to the active secondary.
- Active secondary—The active secondary holds the failover stream from the primary until the data it holds is written to disk. If the primary stops sending the failover stream, or if the storage pool indicates that it has stopped receiving data from the primary, the active secondary writes the uncommitted data from the failover stream to the storage pool and takes over as primary. It then looks for a passive secondary that it can promote to active secondary.
- Passive secondary—The passive secondary does not receive the failover stream from the primary. If either the primary or the active secondary fails, the passive secondary is promoted to active secondary. A passive secondary is promoted to active secondary on the basis of its *priority*. For instance, if there are three passive secondaries with priorities of 1, 2, and 3, the passive secondary with priority 1 is the first to be promoted to active secondary, followed by the passive secondaries with priorities 2 and 3.

To illustrate a simple HA configuration, assume that you have three SpinServers named A, B, and C. Server A is the primary, Server B is the active secondary, and Server C is the passive secondary. The following scenarios can occur:

- If Server A fails, Server B takes over as primary and Server C takes over as active secondary.
- If Server B fails, Server A continues as primary and Server C takes over as active secondary.
- If Server C fails, Server A and Server B continue in their respective roles as primary and active secondary, and both servers look for additional servers that are designated as passive secondaries in case A or B fails. The search for a passive secondary happens only when A or B fails.

High-Availability Design Considerations

When a server in an HA configuration fails, its load is redistributed to other servers in the cluster. When designing an HA configuration, you must consider all possible failure scenarios and ensure that in a worst-case scenario of multiple server failures, the load on a single SpinServer does not exceed maximum capacities. An individual SpinServer has the following maximum capacities:

- 22 TB of total storage capacity
- 16 storage pools
- Four (4) storage pools that the SpinServer serves as secondary (active secondary or passive secondary) in an HA configuration

For instance, you cannot have an HA configuration in which three 10-TB storage pools, each served by a primary SpinServer, are all configured to fail over to a fourth SpinServer. In the event that all three primary SpinServers failed in rapid succession, the 30 TB of total storage is beyond the capacity of the fourth SpinServer to absorb.

11.2 Storage Tasks

A single storage unit represents a single external RAID set that you have already created, or a single JBOD disk. One or more storage units can be used to create a single storage pool.

After creating storage pools, routine management tasks include the following:

- Displaying all storage pools on a server
- Deleting a storage pool
- Displaying the list of storage units that comprise a storage pool
- Designating a storage pool as high availability, and managing the HA attributes of the storage pool

Use the **`storage units`** commands to work with storage units. Use the **`storage pools`** commands to create storage pools and assign storage units to them.

Displaying Storage Units

The **`storage units show`** command provides a view of the external RAID sets or JBOD disk as seen from the Fibre Channel host bus adapters (HBAs). A single storage unit represents a single external RAID set that has already been created, or individual JBOD disks. Storage units are used in the creation of storage pools. A storage pool can be comprised of 1 to 128 storage units. The following ranges or limitations are defined:

- Up to 1024 storage units (RAID sets or JBOD disks) per SpinServer.
- Up to 127 storage units (RAID sets or JBOD disks) per Fibre Channel loop.
- Up to 32 logical unit numbers (LUNs) per RAID array.

Use the **storage units show** command to display information about a storage unit. With no parameter, the **storage units show** command displays information about all storage units attached to a SpinServer. To display information about a single storage unit, use the command with the **-deviceid** parameter. The command displays the following information:

- Device ID
- World wide name
- LUN
- Fibre Channel device (FCD) ID
- D-function (SpinFS) device ID
- Primary host channel status
- Primary host channel
- Primary loop ID
- Secondary host channel status
- Secondary host channel
- Secondary loop ID
- RAID set ID
- Type of disk chassis
- Vendor
- Product name
- Revision number
- Status (`online`, `offline`, or `unused`)
- Damaged status (`true` or `false`)
- Error (`true` or false)
- Last error code (`0` - `6`)
- Error description
- Storage space size (in GB) on the storage unit

The command has the following syntax:

storage units show [**-deviceid** *device_ID*]

The following example displays information about a storage unit with the ID **su0**.

```
spinserver::> storage units show -deviceid su0
Device ID                       su0
World Wide Name                 20000080e511632b
LUN                             0
FCD ID                          0
D-Function Device ID            7
Primary Host Channel Status     connected
Primary Host Channel            1
Primary Loop ID                 133
```

```
Secondary Host Channel Status   connected
Secondary Host Channel          0
Secondary Loop ID               132
RAID Set                        r1s0
Type                            RAID 5
Vendor                          SPINNET
Product                         1316-06v2
Revision                        5785
Status                          Online
Damaged?                        false
Error?                          false
Last Error Code                 0
Error Description               success
Size (GB)                       136
```

Setting the Primary Host Channel for a Storage Unit

To set the primary host channel for a storage unit, use the **`storage units modify`** command.

Possible values for the desired primary host channel include **`Auto`**, **`0`**, **`1`**, **`2`**, and **`3`**. If the value is specified as **`Auto`**, the SpinServer uses the primary channel that is selected by the Fibre Channel device; if the value is specified as **`0`**, **`1`**, **`2`**, or **`3`**, the SpinServer uses that value as the primary channel. The default value for a new storage unit is **`Auto`**.

The command has the following syntax:

```
storage units modify -deviceid device_ID
-primaryhostchannel primary_host_channel -desiredphc
{Auto | 0 | 1 | 2 | 3}
```

The following example sets the sets the storage unit named **`su0`** to use primary host channel **`1`** and sets the desired primary host channel to **`Auto`**:

```
spinserver::> storage units modify -deviceid su0 \
-primaryhostchannel 1 -desiredphc Auto
```

Clearing an Error Status from a Storage Unit

A storage unit can be marked with an error in the event of a power failure or other system problem. You can determine whether a storage unit is marked with an error by using the **`storage units show`** command; error status is listed in the `Error?` column of the command's output. If a storage unit is marked with an error status, use the **`storage units clearerror`** command to clear the error status. The command has the following syntax:

```
storage units clearerror -deviceid device_ID
```

The following example clears the error status from a storage unit named **`su0`**:

```
spinserver::> storage units clearerror -deviceid su0
```

Creating a Storage Pool

To create a storage pool, use the **storage pools new** command. Depending on the number of physical disks that are included in the RAID sets or JBOD disks that are being used, this command can take several minutes to complete. If the storage pool is to be part of an HA configuration, the SpinServer on which the command is issued becomes the primary SpinServer for the storage pool; the primary can be changed by using other **storage pools** commands. The command has the following syntax:

```
storage pools new -deviceid device_ID -layout
{concatenation | stripe} -freespacealarm
free_space_alarm_threshold -sunits
[list_of_storage_units]
```

If you do not provide a value for the **-deviceid** parameter, a device name of the form **pool***number* is automatically generated, where *number* is a sequential integer (for instance, **0**, **1**, **2**, and so on). This is to ensure that new storage pools are given unique device IDs across the cluster.

The **-sunits** parameter is required. If a value is not specified for it, all unused storage units are used by default. If you are using the CLI and press the **Tab** key after the **-sunits** parameter, all unused storage units are automatically listed.

The following example creates a storage pool named **sp1**. The storage pool uses the **stripe** layout, has a free-space alarm threshold of **10**%, and uses five storage units, numbered **su0** through **su4**.

```
spinserver::> storage pools new -deviceid sp1 -layout \
stripe -freespacealarm 10 -sunits [su0,su1,su2,su3,su4]
```

Displaying Information about Storage Pools

To display configuration information about all storage pools or information about one storage pool, use the **storage pools show** command. To display information about a specific storage pool, use the command with the **-deviceid** parameter; this parameter displays the status of the storage pool as it is seen by its primary server. To display configuration information about a list of all storage pools known to the SpinServer, use no parameter with the command. The command displays the following information:

- Device ID (storage pool name)
- UUID
- Status (`online`, `offline`, `busy`, `active`, or `passive`)
- Layout
- Group size (in MB blocks)
- Total storage pool size (in MB blocks)
- MB blocks used
- MB blocks available

- Percentage of MB blocks used
- Free-space alarm threshold (percentage)
- Number of VFSs on the storage pool or pools
- List of storage units that are known to the SpinServer
- List of storage units that are required for the storage pool
- Maximum number of active servers
- Home server name
- Primary server name
- List of active secondary servers
- List of passive secondary servers
- Pool version
- NVRAM version
- Server priorities for failover

The syntax of the command follows.

storage pools show [**-deviceid** *device_ID*]

The following example displays information about a storage pool named **sp0**.

```
spinserver::> storage pools show -deviceid sp0
 Device ID                    sp0
UUID                        66b1714033b80b00336b6d696e370000
 Status                        online
 Layout                        stripe
 Group Size (MB)                512
 Size (MB)                     138848
 Used                          20336
 Avail                         118512
 Use%                          14.6%
 Free Space Alarm Threshold (%)   1
 # VFS                         172
 Storage Units Present          su0
 Storage Units Required         su0
 Max # Actives                  1
 Home Server                   Server1
 Primary Server                 Server1
Active Servers                Server2
 Passive Servers
 Pool Version                   17321
 NVRAM Version                  17321
 Server Priorities                 Server2:1, Server1:64
```

To display the home SpinServer for one or more storage pools, use the **storage pools show-home** command. The command has the following syntax:

storage pools show-home [**-deviceid** *device_ID*]

The following example displays the home SpinServer of all storage pools.

```
   spinserver::> storage pools show-home
Device    Home Server     Primary Server
   ID

pool0    Server1             Server1
pool1    Server1             Server2
pool2    Server1             Server1
pool3    Server1             Server2
```

The home SpinServer for a storage pool is also listed in the output of the **storage pools show** and **tools storage pools show** commands, if the commands are issued with the **-deviceid** parameter to display information about a specified storage pool.

To display information about one or more storage pools as seen by a local SpinServer (that is, a SpinServer that is not necessarily the primary server for the storage pool or pools), use the **storage pools show-local** command. The command has the following syntax:

```
storage pools show-local [-deviceid device_ID]
```

The command displays the following information about the storage pool or pools:

- Device ID
- UUID
- Status (`online`, `offline`, `busy`, `active`, or `passive`)

Modifying a Storage Pool

The **storage pools modify** enables you to modify the free-space alarm threshold and the home SpinServer for a storage pool. You must run this command on the SpinServer that is the primary server for the storage pool. The command has the following syntax:

```
storage pools modify -deviceid device_ID [-freespacealarm
free_space_alarm_threshold] [-homeserver home_spinserver]
```

The following example modifies a storage pool named **sp1** such that the free-space alarm threshold is set to **5**% and the home server is set to a SpinServer named **Server0**:

```
spinserver::> storage pools modify -deviceid sp1 \
-freespacealarm 5 -homeserver Server0
```

Taking a Storage Pool Offline

The **storage pools setoffline** command takes a storage pool offline. Only online or damaged storage pools can be taken offline. The use of the **storage pools setoffline** command prohibits further access to the storage pool and takes all the pool's VFSs offline. The storage pool remains offline across restarts. The command has the following syntax:

```
storage pools setoffline -deviceid device_ID
```

Note: Only storage pools in the offline state can be deleted. Run the **storage pools setoffline** command before using the **storage pools disconnect** command or the **storage pools delete** command.

The following example takes a storage pool named **sp0** offline.

```
spinserver::> storage pools setoffline -deviceid sp0
```

Bringing a Storage Pool Online

The **storage pools setonline** command brings a storage pool back online. The argument of the **-newprimary** parameter specifies the SpinServer that the storage pool is to use as its primary server; by default, this is the storage pool's home server. The command has the following syntax:

```
storage pools setonline -deviceid device_ID -newprimary
primary_server
```

The following example brings online a storage pool named **sp0** on a primary SpinServer named **Server1**:

```
spinserver::> storage pools setonline -deviceid sp0 \
-newprimary Server1
```

Disconnecting a Storage Pool

Use the **storage pools disconnect** command to disconnect a storage pool from the SpinServer. Before using the **storage pools disconnect** command, run the **storage pools setoffline** command. The **storage pools disconnect** command disconnects a storage pool from the SpinServer. The SpinServer has no knowledge of the storage pool. If the SpinServer is restarted, the storage pool is again registered with the SpinServer. Do not use this command if you need to reuse a storage pool's storage units to create a new storage pool. For that purpose, use the **storage pools delete** command.

The syntax of the command follows.

```
storage pools disconnect -deviceid device_ID
```

The following example disconnects a storage pool named **sp0**.

```
spinserver::> storage pools disconnect -deviceid sp0
```

Deleting a Storage Pool

The **storage pools delete** command destroys the contents of the storage pool, freeing all the resources that the pool occupied. This is primarily useful for reusing the storage pool's storage units to create a new storage pool. To create a new storage pool after deleting the old one, use the **storage pools new** command.

Note: Only storage pools in the offline state can be deleted. Before deleting a storage pool, run the **storage pools setoffline** command.

The command has the following syntax:

```
storage pools delete -deviceid device_ID
```

The following example deletes a storage pool named **sp0**.

```
spinserver::> storage pools delete -deviceid sp0
```

Synchronizing I/O Activities

To prevent the loss of data when shutting down a system, you must synchronize from the cache to the disk all the input and output (I/O) activities that are pending on a single storage pool. To accomplish this synchronization, use the **storage pools sync** command. The command has the following syntax:

```
storage pools sync -ssid spinserver -deviceid device_ID
```

The following example synchronizes all I/O activities that are pending on a storage pool named **sp0**.

```
spinserver::> storage pools sync -deviceid sp0
```

Expanding the Size of a Storage Pool

To expand the size of a storage pool, you must first determine whether you have sufficient storage disks, either RAID sets or JBOD disks, that are unused. With storage units available, use the **storage pools grow** command to increase the storage capacity of a storage pool by adding additional storage units (RAID sets or JBOD disks) to the storage pool.

Note: You cannot decrease the size of a storage pool.

The command has the following syntax:

```
storage pools grow -ssid spinserver -deviceid device_ID
-sunits [list_of_storage_units]
```

The following example adds storage units **su0**, **su1**, and **su2** to a storage pool named **sp0**.

```
spinserver::> storage pools grow -deviceid sp0 -sunits \
[su0,su1,su2]
```

Forcing the Discovery of a Storage Pool

In the event of an error in the RAID subsystem, the storage pools need to be rediscovered to bring the pools back online. To force the discovery of the pools, use the **storage pools forcediscovery** command. After the command is run, the storage pools enter a busy state while a quick verification process is performed. After verification is complete, the pools are online and ready for normal VFS operations.

The command has the following syntax:

```
storage pools forcediscovery
```

The following example forces the discovery of all storage pools.

```
spinserver::> storage pools forcediscovery
```

Displaying the High-Availability Status of a Storage Pool

To determine the current high-availability status of a storage pool, and to list a storage pool's primary, active secondary, and passive secondary servers, use the **storage hastatus show** command. Specify the **-deviceid** parameter to display information about a specific storage pool, or issue the command without the parameter to display information about all storage pools.

The command has the following syntax:

```
storage hastatus show [-deviceid device_ID]
```

The following example displays HA status for all storage pools in a cluster.

```
spinserver::> storage hastatus show
Pool    Primary Server   Active Servers      Passive Servers

pool0    ServerA           ServerB             ServerC
pool1    ServerA           ServerC             ServerB
pool2    ServerC           ServerB             ServerA
```

Adding a Passive Secondary to a Storage Pool

Each secondary server for a storage pool must initially be added as a passive secondary. A passive-secondary server can then be promoted to active-secondary status. Use the **storage pools addpassivesecondary** command to add a passive secondary to a storage pool. Required information for the command includes the device ID of the storage pool, the name of the SpinServer, and the priority that the passive secondary is to hold. Possible values for the **-priority** parameter range from **1** (lowest priority) to **64** (highest priority). For instance, if an HA configuration includes a server named ServerA that has a priority of **2** and a server named ServerB that has a priority of **1**, ServerA is promoted as an active secondary before ServerB if the primary or active secondary fails.

Note: If the number of active secondary servers for the storage pool is set to greater than zero, the first passive secondary server that was created is automatically promoted to active secondary.

Note: An individual SpinServer can serve as secondary to no more than four storage pools, whether as active secondary or passive secondary.

The syntax of the command follows.

```
storage pools addpassivesecondary -deviceid device_ID
-passivesec spinserver -priority priority
```

The following example adds a SpinServer named **NewServer** as a passive secondary to a storage pool named **sp0**. The priority of the new server is set to **2**.

```
spinserver::> storage pools addpassivesecondary \
-deviceid sp0 -passivesec NewServer -priority 2
```

Setting the Priority for a Passive Secondary

You can set the priority for a passive secondary by using the **storage pools setpriority** command. The priority of a passive secondary determines the order in which it is promoted to an active secondary in the event that the primary fails. Priority values range from **1** to **64**, with **64** being the highest; that is, a passive secondary set to priority **64** is the first to be selected as an active secondary, followed by a passive secondary set to priority **63**, and so forth.

The syntax of the command follows.

```
storage pools setpriority -passivesec spinserver -priority
priority
```

The following example sets the priority to **3** for a SpinServer named **Server2.**

```
spinserver::> storage pools setpriority -passivesec \
Server2 -priority 3
```

Promoting a Passive Secondary to Active Secondary

You can manually promote a passive secondary to active-secondary status. This is useful to increase the number of active secondaries for a storage pool, or to provide an extra measure of high availability before you manually fail over the current active secondary.

Note: If the number of active secondary servers for the storage pool is set to greater than zero, the first passive secondary server that was created is automatically promoted to active secondary.

Use the **storage pools promotepassivesecondary** command to promote a passive secondary. The syntax of the command follows.

```
storage pools promotepassivesecondary -deviceid device_ID
-passivesec spinserver
```

The following example promotes a SpinServer named **Server2** from passive secondary to active secondary for a storage pool named **sp1**.

```
spinserver::> storage pools promotepassivesecondary \
-deviceid sp1 -passivesec Server2
```

Demoting an Active Secondary to Passive Secondary

To change the arrangement of secondaries in an HA configuration, you possibly need to demote an active secondary to a passive secondary. Use the **storage pools demoteactivesecondary** command to demote an active secondary.

The syntax of the command follows.

```
storage pools demoteactivesecondary -deviceid device_ID
-secondary spinserver
```

The following example demotes a SpinServer named **Server3** from active secondary to passive secondary for a storage pool named **sp1**.

```
spinserver::> storage pools demoteactivesecondary \
-deviceid sp1 -secondary Server3
```

Removing a Secondary

To remove a secondary server from an HA configuration (for example, before taking the secondary server down for maintenance) use the **storage pools removesecondary** command. The syntax of the command follows.

```
storage pools removesecondary -deviceid device_ID
-secondary spinserver
```

The following example removes a SpinServer named **Server2** as a secondary for a storage pool named **sp0**.

```
spinserver::> storage pools removesecondary -deviceid \
sp0 -secondary Server2
```

Failing Over a Storage Pool

To force a storage pool to fail over to an active secondary, use the **storage pools failover** command. The syntax of the command follows.

```
storage pools failover -deviceid device_ID -activesec
spinserver
```

The following example forces a storage pool named **sp0** to fail over to the active secondary named **Server2**.

```
spinserver::> storage pools failover -deviceid sp0 \
-activesec Server2
```

Reverting Storage Pools

In a high-availability (HA) configuration, storage pools can fail over from one SpinServer to another. You can return one or more storage pools directly to their home SpinServers.

By default, a storage pool's home SpinServer is the SpinServer on which the storage pool was created.

The **storage pools revertpools** command returns one or more storage pools to their home SpinServers. The argument of the **-deviceids** parameter specifies the square-bracketed, comma-separated list of pools that are to be reverted. If you are using the CLI, press **Tab** after typing the parameter to get a complete list of storage pools.

The syntax of the command follows.

```
storage pools revertpools -deviceids [list_of_pools]
```

The following example reverts storage pools **pool0** and **pool1** to their home SpinServers.

```
spinserver::> storage pools revertpools -deviceids \
[pool0,pool1]
```

12.0 Managing Virtual Servers

This chapter discusses creating and managing virtual servers. Virtual servers provide data access to both CIFS and NFS clients.

12.1 Virtual Server Concepts

In “Managing Networks” on page 91, you created virtual interfaces and configured the network ports. In “Managing Storage” on page 133, you created storage pools. Now, you are ready to create virtual servers. After creating the virtual servers, you bind the virtual interfaces to the virtual servers.

12.2 Virtual Server Tasks

Use the **`vservers`** command node to work with virtual servers.

Creating a Virtual Server

To create a virtual server, use the **`vservers servers new`** command. A virtual server name can be 1 to 32 characters long and can consist of letters, numerals, and the period and underscore characters. The name must start with a letter or numeral.

To Create a Virtual Server

The **`vservers servers new`** command creates a virtual server. The virtual server’s attributes include its virtual server name, the storage pool on which it is created, virus-scan defaults, and authentication settings. See “Managing Security” on page 207 for information about authentication.

The syntax of the command follows.

```
vservers servers new -vsname virtual_server_name -spdevid
storage_pool_ID -spinauth {true | false}
-floatingsuperuser {true | false} -minauth {none |
cifs-ntlm | cifs-kerberos | nfs-kerberos} -rulesetname
rule_set_name -virusscanrwvfs {true | false}
-virusscanrovfs {true | false} -showspinshot {true |
false} -restrictedacl {true | false}
```

The following example creates a virtual server named **vs0** that is attached to a storage pool named **sp0**. The virtual server does not use SpinAuth or permit a floating superuser. The virtual server uses **nfs-kerberos** as its minimum authentication level. A rule set named **default_rules** is applied to the virtual server. By default, virus scanning is enabled on any read-write VFSs and disabled on any read-only VFSs created on the virtual server. Any `.spinshot` directories on new VFSs created on the virtual server are visible to clients by default. CIFS ACLs are not disabled on new VFSs created on the virtual server.

```
spinserver::> vservers servers new -vsname vs0 \
-spdevid sp0 -spinauth false -floatingsuperuser false \
-minauth nfs-kerberos -rulesetname default_rules \
-virusscanrwvfs true -virusscanrovfs false -showspinshot \
true -restrictedacl false
```

Displaying Information about Virtual Servers

To display general configuration information about all virtual servers or specific information about one virtual server, use the **vservers servers show** command. To display information about one virtual server, use the command with the **-vsname** parameter. To display configuration information about a list of virtual servers in the system, do not use the parameter. The command has the following syntax.

```
vservers servers show [-vsname virtual_server_name]
```

The following example shows the configuration information for all virtual servers in a cluster.

```
spinserver::> vservers servers show
  Name             Network Interface        Min Authentication

  vs1             Server1:vif2, Server1:vif3     nfs-kerberos
  vs2             Server1:vif4                   none
```

The following example shows the configuration information for a virtual server named **vs0**.

```
spinserver::> vservers servers show -vsname vs0
Name                            vs0
Storage Pool for Root VFS       pool0
User Type for root FID          NFS
Domain of root FID              example.com
User of root FID                0
Group of root FID               0
Network Interfaces              Server12:Data, Server:Data
```

```
Virtual Server ID               15
Root VFS ID                     5369
Status                          online (Data,Data)
Enable SpinAuth             false
Enable Floating SuperUsers  false
Min Authentication          nfs-kerberos
Export Ruleset Name          default_ruleset
VirusScan for R-W VFS       true
VirusScan for R-O VFS        true
```

Deleting a Virtual Server

To remove a virtual server from the network, use the **vservers servers delete** command. The syntax of the command follows.

vservers servers delete -vsname *virtual_server_name*

Note: Deleting a virtual server deletes all virtual file systems (VFSs) associated with that virtual server.

Note: If a CIFS virtual server or NFS server interface that is configured to use Kerberos is deleted, or is changed to use a different Service Principal Name (SPN), its original service principal is not automatically deleted from or disabled in the Kerberos realm. If you no longer need the service principal, it is recommended that you manually delete or disable the principal. This requires the Kerberos realm administrator's user name and password.

To ensure the clean deletion of a virtual server, it is recommended that you perform the following steps:

1. Disable SpinShots and mirror replications for all virtual file systems. See "Modifying a VFS Scheduler Job" on page 174 for more information.

2. Delete all VFSs on the virtual server that is to be deleted. See "Deleting a Virtual File System" on page 165 for details.

3. Delete the virtual server.

4. Re-enable SpinShots and mirror replications for all virtual file systems. See "Modifying a VFS Scheduler Job" on page 174 for more information.

The following example deletes a virtual server named **vs0**.

```
spinserver::> vservers servers delete -vsname vs0
```

Binding and Unbinding Virtual Interfaces

This section discusses binding a virtual interface to a virtual server and unbinding a virtual interface from a virtual server by using the **vservers servers bindif** and **vservers servers unbindif** commands, respectively.

When a virtual interface is bound to a virtual server, the virtual server is ready to serve NFS requests. If the minimum authentication level for the virtual server is **nfs-kerberos**, NFS requests fail until Kerberos is configured; if the minimum authentication level is **none**, requests can be served immediately. To enable the virtual server to serve CIFS requests, see "Managing SpinAccess CIFS" on page 177.

To Bind a Virtual Interface

To bind a virtual interface to a virtual server, use the **vservers servers bindif** command. The syntax of the command follows.

```
vservers servers bindif -ssid spinserver -vsname
virtual_server_name -ifname interface_name
```

The following example identifies an interface named **if0** on a SpinServer named **Server1** and binds the virtual interface to a virtual server named **vs0**.

```
spinserver::> vservers servers bindif -ssid Server1 \
-ifname if0 -vsname vs0
```

To Unbind a Virtual Interface

To unbind a virtual interface from a virtual server, use the **vservers servers unbindif** command. The syntax of the command follows.

```
vservers servers unbindif -ssid spinserver -vsname
virtual_server_name -ifname interface_name
```

The following example identifies an interface named **if0** on a SpinServer named **Server1** and unbinds the virtual interface from a virtual server named **vs0**:

```
spinserver::> vservers servers unbindif -ssid Server1 \
-ifname if0 -vsname vs0
```

Binding and Unbinding All Virtual Interfaces

This section discusses binding multiple virtual interfaces to a virtual server and unbinding multiple virtual interfaces from a virtual server by using the **vservers servers online** and **vservers servers offline** commands, respectively.

To Unbind All Virtual Interfaces

To unbind all virtual interfaces bound to a specific virtual server, use the **vservers servers offline** command. The **vservers servers offline** command identifies a virtual server and takes down all virtual interfaces bound to it, thus making it inaccessible to NFS and CIFS clients. The syntax of the command follows.

```
vservers servers offline -vsname virtual_server_name
```

The following example unbinds all virtual interfaces from a virtual server named **vs0**.

```
spinserver::> vservers servers offline -vsname vs0
```

To Bind Multiple Virtual Interfaces

To bind multiple virtual interfaces to a specific virtual server, use the **vservers servers online** command. The **vservers servers online** command identifies a virtual server and binds multiple virtual interfaces to it. The syntax of the command follows.

```
vservers servers online -vsname virtual_server_name
```

The following example binds multiple virtual interfaces to a virtual server named **vs0**.

```
spinserver::> vservers servers online -vsname vs0
```

Modifying a Virtual Server

Use the **vservers servers modify** command to modify configuration information for a virtual server. See “Managing Security” on page 207 for information about authentication. The syntax of the command follows.

```
vservers servers modify -vsname virtual_server_name
-spinauth {true | false} -floatingsuperuser {true | false}
-minauth {none | cifs-ntlm | cifs-kerberos | nfs-kerberos}
-rulesetname rule_set_name -virusscanrwvfs {true | false}
-virusscanrovfs {true | false} -showspinshot {true |
false} -restrictedacl {true | false}
```

The following example modifies a virtual server named **vs0**.

```
spinserver::> vservers servers modify -vsname vs0 \
-spinauth false -floatingsuperuser true -minauth \
cifs-kerberos -rulesetname rs0 -virusscanrwvfs false \
-virusscanrovfs false -showspinshot false -restrictedacl \
true
```

Mirroring the Root of a Virtual Server

It is recommended that the root VFS of each virtual server be mirrored for redundancy and backup. You can mirror the root VFS of a virtual server even if you do not have the optional SpinMirror software package. To mirror the root of a virtual server, use the **vservers servers newrootmirror** command. The syntax of the command follows.

```
vservers servers newrootmirror -vsname virtual_server_name
-sinkvfs sink_VFS -sinksp sink_storage_pool
```

The following example mirrors a virtual server named **vs0** to a VFS named **vs0.mirror**. The mirror VFS is located on a storage pool named **sp1**.

```
spinserver::> vservers servers newrootmirror -vsname vs0 \
-sinkvfs vs0.mirror -sinksp sp1
```

For information about recovering the root VFS of a virtual server in the event of a catastrophic disk failure, see “Recovering a Mirror Root VFS” on page 243.

12.3 Virtual-Server Route-Management Tasks

Virtual server routes are static routes. When packets are sent and the destination IP address is not on the local subnet or network, a default IP address is used as the destination IP address. This default address is usually a router or firewall on the local network.The default IP address is used when no other IP address matches.

Creating and Configuring a Virtual Server Route

Before you add a static route to a virtual server, you must add a default gateway for the route to use.

Adding a Default Gateway to a Virtual Server

Use the **vservers servers adddefaultgateway** command to add a default gateway for a static route on a virtual server. The command has the following syntax:

```
vservers servers adddefaultgateway -vsname
virtual_server_name -gateway gateway_IP_address
```

The following command example adds a gateway IP address of **10.1.1.104** to a virtual server named **Server1vserver1**.

```
spinserver::> vservers servers adddefaultgateway -vsname \
Server1vserver1 -gateway 10.1.1.104
```

Adding a Route to a Virtual Server

Use the **vservers servers addstaticroute** command to create a route for a virtual server. The command has the following syntax:

```
vservers servers addstaticroute -vsname
virtual_server_name -gateway gateway_IP_address
-netaddress network_IP_address -netmask netmask -metric
metric
```

The following command example adds a new static route to a virtual server named **Server1vserver1**. The route is defined with a gateway IP address of **10.1.1.104**, a destination IP address of **10.1.1.14**, a netmask of **255.255.255.0**, and a metric of **1**.

```
spinserver::> vservers servers addstaticroute -vsname \
Server1vserver1 -gateway 10.1.1.104 -netaddress \
10.1.1.14 -netmask 255.255.255.0 -metric 1
```

Displaying Virtual Server Routes

Use the **vservers routes show** command to display route configuration information about all virtual servers or information about one virtual server.

To display route information about one route, use the command with the **-vsname**, **-nexthop**, **-netnumber**, and **-netmask** parameters. To display route configuration information about a list of all routes, do not use any parameters. The command displays the following information about the virtual server or servers:

- Virtual server name
- Default gateway address
- Network address
- Network mask
- Metric

The syntax of the command follows.

```
vservers routes show [-vsname virtual_server_name -nexthop
nexthop_IP_address -netnumber network_IP_address -netmask
netmask]
```

The following example identifies a virtual server named **vs0** that has a next-hop IP address of **10.1.1.104**, a network IP address of **10.2.2.0**, and a netmask of **255.255.255.0** and displays route configuration information for it.

```
spinserver::> vservers routes show -vsname vs0 -nexthop \
10.1.1.104 -netnumber 10.2.2.0 -netmask 255.255.255.0
```

Deleting Virtual Server Routes

The **vservers routes delete** command identifies a virtual server and removes its route. The syntax of the command follows.

```
vservers routes delete -vsname virtual_server_name
-nexthop nexthop_IP_address -netnumber network_IP_address
-netmask netmask
```

The following example identifies a virtual server named **vs0** that has a next-hop IP address of **10.1.1.104**, a network IP address of **10.2.2.0**, and a netmask of **255.255.255.0** and removes its route.

```
spinserver::> vservers routes delete -vsname vs0 \
-nexthop 10.1.1.104 -netnumber 10.2.2.0 -netmask \
255.255.255.0
```

13.0 Managing Virtual File Systems

This chapter discusses the creation and management of virtual file systems (VFSs). Before you create a VFS, you must first select a virtual server and a storage pool in which to locate the VFS.

Each VFS in a cluster must be labeled with a name that is used by management commands to refer to that particular VFS. Any assigned name can be chosen, provided that it is unique within a cluster. You need to give some thought to the names you choose so that when other VFSs are added to the cluster, the names are orderly and recognizable. For example, if you are deploying a software-development project on a SpinServer system, you might choose to create the following VFSs:

- A VFS named **`user.`***`user_name`* for each user
- A VFS named **`src.`***`component`* for source code for each component
- A VFS named **`build.`***`component`* for compiled images of each component

Uniform naming schemes make it easier to manage large numbers of VFSs.

The root virtual file system for a virtual server is automatically created and mounted when the virtual server is created. The virtual server is named **`root.`*`virtual_server`**, where *`virtual_server`* is the name or ID you have given the virtual server. It is mounted at / (root).

The file system for a particular virtual server is expanded by creating additional VFSs and linking them together by using the **`filestorage vfs mount`** command.

13.1 VFS Concepts

The basic unit of storage on a SpinServer is the VFS, which is a conceptual container for a set of logically related directories and files. Use the **`filestorage`** commands to create and manage VFSs.

You can perform the following operations on VFSs:

- SpinShot—The SpinShot is SpinFS's snapshot implementation. A SpinShot of a VFS is a new VFS with read-only access. It is located on the same storage pool as the original VFS and represents the contents of the original VFS at a particular point in time. An original VFS and a SpinShot of it share disk space for all blocks that have not been modified between the creation of the VFS and the time the SpinShot is made, thereby making SpinShots lightweight. Similarly, two SpinShots share disk space for those blocks that were not modified between the times that the two SpinShots were created.

A chain of SpinShots can be created to represent the state of a VFS at a number of points in time. SpinShots can be accessed online, enabling users to retrieve their own data from past copies, rather than asking a system administrator to restore data from tape. A SpinShot is automatically mounted. SpinShots provide the basis for other VFS management features, including copying, mirroring, and moving.

- Move—You can move a VFS from one storage pool to another. The original VFS is completely copied to a new storage pool with the same name, and then the original VFS is deleted. This is useful for data management and load balancing. For instance, if one storage pool holds 500 VFSs with similar sizes and another storage pool of the same size holds only 200 VFSs of the same approximate sizes, you can move VFSs from the first storage pool to the second to balance the load and improve performance.
- Copy—A VFS copy is a full copy of the original VFS with the same access (read-only or read-write) as the original VFS. It does not share blocks with the original VFS; thus, a copy of a 2-GB VFS uses 2 GB of disk space, unlike a SpinShot of a VFS, which might use only a few kilobytes of disk space because of shared blocks. After the copy is made, no operations made on the copy or on the original affect the other. For instance, if you write data to the original VFS, the data is not written to the copy. A VFS copy is not automatically mounted when it is created.
- Mirror—A VFS mirror is a read-only copy of a VFS on the same or a different storage pool. If it is on the same storage pool as the original, it shares blocks with the original; if it is on a different storage pool, it does not share blocks with the original. It can be synchronized with the original VFS whenever the administrator desires. After it is created, a VFS mirror must be replicated with data from the original VFS before it is ready for use. A mirror is automatically mounted only if the entire VFS tree is mirrored, and each VFS is mounted with the **`context`** mount type.

Table 13-1 summarizes the differences among SpinShots, copies, mirrors, and moved VFSs.

Table 13-1. Differences among VFS Features

Feature	Access type (read-write or read-only)	Auto-mounted?	Full copy or shared blocks?	Location
SpinShot	Read-only	Yes	Shared blocks	Same storage pool and SpinServer
Copy	Same as the original VFS (read-write or read-only)	No	Full copy	Same or different storage pool; same or different SpinServer
Mirror	Read-only	Yes (upon replication)	Full copy	Different storage pool; same or different SpinServer
Move	Same as the original VFS (read-write or read-only)	Yes	Full move, then deletion of original	Different storage pool; same or different SpinServer

All of these VFS operations must occur within the context of the same virtual server. For instance, you cannot move a VFS from a virtual server named vs0 to a virtual server named vs1. However, a virtual server can span multiple SpinServers and multiple storage pools.

You can schedule regular occurrences of SpinShots and mirrors by using the VFS Scheduler. See “Scheduling VFS Operations” on page 169 for information on using the VFS Scheduler.

VFS operations can take a long time to complete, depending on the size of the VFS and the load on the cluster. For instance, moving a 1-TB VFS might take several hours to complete. See “Managing Long-Running Jobs” on page 258 for information on managing long-running jobs.

13.2 VFS Tasks

To create a VFS, select a virtual server and a storage pool, set quotas, and mount the VFS. Additional management tasks include the following:

- Unmounting a VFS
- Removing a VFS from the system by deleting it
- Modifying the VFS’s quotas
- Taking a one-time SpinShot of a VFS
- Displaying configuration information for VFSs, their mount paths, and virtual servers
- Copying a VFS
- Mirroring a VFS
- Moving a VFS
- Resetting the export rule set for a VFS
- Scheduling regular tasks (such as creating SpinShots and replicating mirrors) for a VFS

To manage virtual file systems, you must first define the purposes that the VFSs are to serve so that you can set the quotas for the amount of data that is stored in the VFS. After creating VFSs and setting the quotas, you mount the VFSs.

Defining Virtual File Systems

The purposes of the VFSs are defined by the needs of the organization. If the data contained in a VFS is crucial to the organization and not reproducible, it must be backed up on a clearly defined schedule so that it is easily recoverable in the event of a disaster. On the other hand, if the data in the VFS is a working file and easily reproduced, it does not need to be backed up. The size of the VFS is determined by the user’s need for storage and working space.

After you define and create the VFSs, provide users with the definitions of their purpose and use so that the storage space is used appropriately and efficiently.

For example, a user can be assigned the following VFSs:

- Home directory
- Build directory
- Source directory

The Home directory is typically small in size, about 10 MB, and contains configuration and initialization files. The Home VFS is scheduled for backup. The Build VFS is a working space is often large in size (10 GB) and used for binaries and compiled object files. It can easily be reproduced, so it is not scheduled for backup. The Source VFS usually is small in size, about 100 MB, and contains daily program files, such as documents or source code in progress. The Source VFS is scheduled for backup.

Creating Virtual File Systems

To create a VFS, use the **`filestorage vfs new`** command.

Minimum and maximum quotas can be specified when creating a VFS. A *quota* refers to an amount of data storage and is specified in units of 1-MB blocks.

The minimum quota guarantees that the specified amount of storage is available. This is done by reserving storage from the underlying storage pool. The maximum quota limits the amount of data that can be written to a VFS, but no reservation is made. Oversubscription is possible, and the quotas can be modified at any time.

When creating a VFS, you can specify the export rule set name that applies to it. Export rule sets enable you to restrict access to the VFS based on clients' IP addresses and authentication levels; see "Managing Security" on page 207 for more information.

The number of VFSs that can be created within a given storage pool is limited by the underlying storage pool space.

A VFS name can be 1 to 32 characters long and can consist of letters, numerals, and the period and underscore characters. The name must start with a letter or numeral. You cannot use the same VFS name on multiple, different virtual servers. VFS names must be unique to a cluster.

The syntax of the **`filestorage vfs new`** command follows.

```
filestorage vfs new -vfsname VFS_name -vsname
virtual_server_name -spdevid storage_pool_name -usertype
user_type -domain domain_name -user user_name -group
group_name -qmin quota_minimum -qmax quota_maximum
-mounttype {context | explicit | mirror | spinshot}
-mountpath mount_point -rulesetname rule_set_name
-virusscanrw {true | false} -virusscanro {true | false}
-showspinshot {true | false} -restrictedacl {true | false}
```

The following notes apply to the command:

- The **-mounttype** parameter specifies the type of VFS that is to be created. Possible values include **context** for a VFS that assumes the context of its parent mount point (read-write or read-only), **explicit** for a VFS that is read-write regardless of its context (for instance, a read-write VFS mounted below a VFS mirror), **mirror** for a read-only mirror, and **spinshot** for a read-only SpinShot. The default is **context**.
- The **-usertype** field has four options: **SPIN**, **NFS**, **CIFS**, and **NFS-Kerberos**. Use the **SPIN** value if an NIS or CIFS domain is not configured or available. If you create an NFS or CIFS virtual server or VFS, a NIS or a CIFS domain must already be configured on the SpinServer. The **CIFS** user type is used only for CIFS or mixed-security (NFS and CIFS) environments. If you specify the **NFS-Kerberos** user type, you must specify a Kerberos realm name as the value of the **-domain** parameter and a Kerberos principal name as the value of the **-user** parameter. If the **NFS-Kerberos** user type is specified, the value of the **-group** parameter is disregarded and can be omitted from the command.

  If the **NFS-Kerberos** user type is specified, the ownership of the VFS is assigned to the user that is mapped to the Kerberos principal name specified by the value of the **-user** parameter. The virtual server must have a NIS domain or other mapping mechanism (for instance, a pluggable authentication module) associated with it, and a Kerberos-NIS mapping that matches the Kerberos principal name must exist.
- To have the VFS inherit the export rule set of the VFS on which the new VFS is mounted, specify a value of **inherit** for the **-rulesetname** parameter. This is a reserved value that applies only to the **filestorage vfs new** command, and only when the new VFS is mounted at the same time it is created.
- The **-showspinshot** parameter specifies whether the VFS's `.spinshot` directory is visible to clients. The default is **true**.
- The **-restrictedacl** parameter specifies whether CIFS access control lists (ACLs) are used on files on the VFS. The default is **false** (that is, ACLs are used on the VFS). If ACLs are not used on the VFS, then the only permissions you can set on files are UNIX mode bits, and attempting to modify ACLs from a CIFS client results only in changes to the mode bits.

The following example creates a VFS named **vfs1** for the virtual server **vs1** on storage pool **sp0** with a **5000**-MB minimum quota and **10000**-MB maximum quota. The user type is **SPIN**, the user name is **0**, the domain name is **0**, and the group is **0**. The mount type is set to **context**, and the mount path is set to **/vfs1**. The export rule set is inherited from the VFS on which the new VFS is mounted (root). Virus scanning is enabled for read-write and read-only VFSs (for example, mirrors and SpinShots of this VFS). The VFS's `.spinshot` directory is not visible to clients. CIFS ACLs are not used on files on the VFS.

```
spinserver::> filestorage vfs new -vsname vs1 -spdevid \
sp0 -vfsname vfs1 -usertype SPIN -domain 0 -user 0 \
-group 0 -qmin 5000 -qmax 10000 -mounttype context \
-mountpath /root -rulesetname inherit -virusscanrw true \
-virusscanro true -showspinshot false -restrictedacl true
```

Modifying a VFS

The settings for a virtual file system can be modified at any time. Use the **filestorage vfs modify** command to modify the minimum and maximum quotas for a virtual file system that were specified when the VFS was created or that are in current use, and to modify the export rule set that is applied to the VFS (see "Managing Security" on page 207 for information about export rule sets), and to change the virus-scanning settings for a VFS. The command has the following syntax.

```
filestorage vfs modify -vfsname VFS_name [-usertype
user_type] [-domain domain_name] [-user user_name] [-group
group_name] [-qmin quota_minimum] [-qmax quota_maximum]
[-rulesetname rule_set_name] [-virusscanrw {true | false}]
[-virusscanro {true | false}] [-showspinshot {true |
false}] [-restrictedacl {true | false}]
```

The following example sets a new maximum quota of **20000** MB for the VFS named **vfs1**.

```
spinserver::> filestorage vfs modify -vfsname vfs1 -qmax \
20000
```

Mounting a Virtual File System

If you do not specify a mount path when you create a VFS, use the **filestorage vfs mount** command to mount the VFS. A VFS can be mounted in another VFS. The command has the following syntax:

```
filestorage vfs mount -vfsname VFS_name -mountpath
mount_path -mounttype {context | explicit| mirror |
spinshot} -expoverride {true | false}
```

The **-mounttype** parameter specifies the type of VFS that is to be mounted. Possible values include **context** for a VFS that assumes the context of its parent mount point (read-write or read-only), **explicit** for a VFS that is read-write regardless of its context (for instance, a read-write VFS mounted below a mirror VFS), **mirror** for a read-only mirror, and **spinshot** for a read-only SpinShot. The default is **context**.

The **-expoverride** parameter specifies whether the VFS's associated export rule set is overridden. (See "Managing Security" on page 207 for information about export rule sets.) The default is **false**. The following rules apply to export rules when a VFS is mounted:

- If the associated export rule set is **none**, the mounted VFS's associated rule set is changed to be the same as the VFS it is mounted on, unless the parent VFS's associated export rule set is also **none**.
- If the mounted VFS has an associated export rule set, the rule set is not changed unless the value of the **-expoverride** parameter is set to **true**. If this value is set as **true**, the mounted VFS has its rule set changed to match the parent VFS's associated rule set, unless the parent VFS's associated export rule set is **none**.
- Rule-set changes are not recursive. To change rule sets recursively down a chain of

VFSs, use the **filestorage vfs resetexportruleset** command, as described in "Resetting the Export Rule Set for a VFS" on page 169.

The following example sets a mount path of **/Server1/engineering** for the virtual file system named **vfs1**. The VFS is mounted in the same context (read-write or read-only) as the parent VFS on which it is mounted. The associated export rule set is not overridden.

```
spinserver::> filestorage vfs mount -vfsname vfs1 \
-mountpath /Server1/engineering -mounttype context \
-expoverride false
```

To Unmount a Virtual File System

To point a virtual file system to a different directory path, you must unmount it from the current path. Use the **filestorage vfs unmount** command to identify a virtual file system by its virtual file system name and unmount it. The command has the following syntax:

```
filestorage vfs unmount -vfsname VFS_name
```

The following command unmounts the virtual file system named **vfs1**.

```
spinserver::> filestorage vfs unmount -vfsname vfs1
```

To Display the Mount Path for a Virtual File System

To display the mount path for a virtual file system, use the **filestorage vfs show-mount** command. This command can be used without the **-vfsname** parameter to display all VFS mount paths or with the **-vfsname** parameter to display the mount path for a specific VFS. The syntax of the command follows.

```
filestorage vfs show-mount [-vfsname VFS_name]
```

The following example displays the mount path for the virtual file system named **vfs1**.

```
spinserver::> filestorage vfs show-mount -vfsname vfs1
```

Displaying Information about Virtual File Systems

Use the **filestorage vfs show** command to display and review the configuration information for virtual file systems. This command can be used without the **-vfsname** parameter to display information about all read-write VFSs in the system or with the parameter to display information about a specific VFS (read-write or read-only). The syntax of the command follows.

```
filestorage vfs show [-vfsname VFS_name]
```

The command displays the following information about the VFS or VFSs. Some fields are displayed only when you issue the command for a specific VFS.

- VFS name
- VFS family name
- Number of VFSs in the family
- Virtual server name
- Storage pool name
- VFS ID (specific VFS only)
- Type (`read-write` or `read-only`) (specific VFS only)
- Creation time (specific VFS only)
- Virtual server from which the VFS was mounted
- Mount type (`context`, `explicit`, `mirror`, or `spinshot`; specific VFS only)
- Mount path
- Minimum quota (in MB)
- Maximum quota (in MB)
- Free-space alarm threshold (percentage)
- Amount of data storage in use (in MB)
- Storage pool name
- Amount of data storage that is available on the storage pool (in MB)
- User type (specific VFS only)
- Domain (specific VFS only)
- User (specific VFS only)
- Group (specific VFS only)
- Whether virus scanning is enabled for read-write VFSs
- Whether virus scanning is enabled for read-only VFSs
- Whether the `.spinshot` directory is visible to clients
- Status of the VFS (`online` or `offline`)
- Last status error, if any

The following example shows information about a VFS named **`user.1`**.

```
 spinserver::> filestorage vfs show -vfsname user.1
 VFS                          user.1
Family Name                   user.1
Family Size                    1
Virtual Server                 mike0
VFS ID                        1026
Type                         read-write
Create Time                  11/10/2003 4:26 PM
Mounted From                  root.mike0
Mount Type                    context
```

```
Mount Path                     /user.1
Minimum Quota (MB)              0
Maximum Quota (MB)              1000
Free Space Alarm Threshold (%)   0
Used (MB)                      1
Storage Pool                   pool1
 SP Avail (MB)                 32255
User Type of root FID           SPIN
Domain of root FID              0
User of root FID                0
Group of root FID               0
Export Ruleset Name            default_ruleset
 VirusScan for R-W              false
VirusScan for R-O              false
 Show Spinshot Directory        false
 Status                        online
 Status Error
```

The **filestorage vfs show-access**, **filestorage vfs show-mount**, **filestorage vfs show-vs** commands can be used to display subsets of VFS information pertaining to export rule sets, mount paths, and virtual servers, respectively. These commands display information about read-write VFSs only.

Displaying Information about All Virtual File Systems

The **filestorage vfs show** command, when issued without the **-vfsname** parameter, displays only read-write VFSs. To display all VFSs, both read-only and read-write, use the **filestorage vfs show-all** command. The command has the following syntax:

filestorage vfs show-all

The following example displays information about both read-only and read-write VFSs:

```
spinserver::> filestorage vfs show-all
VFS                 Fam MaxQ (MB)  Used (MB)   Storage   SP Avail (MB) Status
                    Siz                          Pool

root.vs0            1   10         1          pool0      47806         online
root.vs1            1   10         1          pool1      63883         online
vs0pool0_11         1   10         1          pool0      47806         online
vs0pool0_12         1   10         1          pool0      47806         online
vs0pool0_13         1   10         1          pool0      47806         online
vs0pool0_14         1   10         1          pool0      47806         online
vs0pool0_15         1   10         1          pool0      47806         online
vs0pool0_16         1   10         1          pool0      47806         online
vs0pool0_17         1   10         1          pool0      47806         online
vs0pool0_18         1   10         1          pool0      47806         online
vs0pool0_19         1   10         1          pool0      47806         online
vs0pool0_20         1   10         1          pool0      47806         online
vs0pool0_21         1   10         1          pool0      47806         online
vs0pool0_22         1   10         1          pool0      47806         online
vs0pool0_23         1   10         1          pool0      47806         online
```

Displaying Information about Read-Only Virtual File Systems

The **filestorage vfs show** command, when issued without the **-vfsname** parameter, displays only read-write VFSs. To display read-only VFSs (for instance, SpinShots and mirrors) that are associated with a read-write VFS, use the **filestorage vfs show-family** command. The command has the following syntax:

filestorage vfs show-family -vfsname *VFS_name*

The following example displays information about the read-only VFSs associated with a read-write VFS named **vfs1**.

```
spinserver::> filestorage vfs show-family -vfsname vfs1
VFS        Type        Mount Path         Create Time            Storage      Status
                                                                   Pool

san.67     spinshot    .spinshot/hourly   09/22/2004 8:58 AM      pool0       online
san.77     spinshot    .spinshot/hourly   09/22/2004 6:23 AM      pool0       online
san.16     spinshot    .spinshot/weekly   09/12/2004 12:44 AM     pool0       online
san.m1     mirror      /san               09/22/2004 9:30 AM      pool1       online
san        read-write  /san               07/06/2004 1:32 PM      pool0       online
san.23     spinshot    .spinshot/hourly   09/22/2004 11:28 AM     pool0       online
san.19     spinshot    .spinshot/daily    09/22/2004 12:38 AM     pool0       online
san.m3     mirror      /san               09/22/2004 9:30 AM      pool19      online
san.m0     mirror      /san               09/22/2004 9:30 AM      pool0       online
san.m5     mirror      /san               09/22/2004 9:30 AM      pool15      online
san.m4     mirror      /san               09/22/2004 9:30 AM      pool14      online
san.36     spinshot    .spinshot/daily    09/15/2004 12:37 AM     pool0       online
san.m2     mirror      /san               09/22/2004 9:31 AM      pool99      online
```

To display detailed information about a specific VFS, whether read-write or read-only, issue the **filestorage vfs show** command with the **-vfsname** parameter.

Displaying Backup Information about Virtual File Systems

The **filestorage backup show** command displays the following information about backup operations for one or more VFSs:

- VFS name
- VFS ID
- Storage pool ID
- Time of last full backup
- Time of last backup (levels 0 through 9)
- Version number of backups 0 through 9
- Creation time of backups 0 through 9
- Error status, if any

The command has the following syntax:

filestorage backup show [**-vfsname** *VFS_name*]

Deleting a Virtual File System

To remove a virtual file system from a system, use the **filestorage vfs delete** command. The syntax of the command follows.

```
filestorage vfs delete -vfsname VFS_name
```

The following example deletes the virtual file system named **vfs1**.

```
spinserver::> filestorage vfs delete -vfsname vfs1
```

Creating a SpinShot of a Virtual File System

After identifying an existing virtual file system by its virtual file system name, you can create a one-time SpinShot of that virtual file system by using the **filestorage vfs spinshot** command. The SpinShot is named by the SpinServer. The SpinServer gives the SpinShot the original VFS name and appends a new VFS ID to the name. This combination of original VFS name and the new VFS ID becomes the SpinShot's name. The SpinShot is mounted at a location specified by the value of the **-mountpointname** parameter; the SpinShot is then mounted under the source SpinShot at *vfs*/.spinshot/*spinshot_name*. For instance, if you create a SpinShot of a VFS named **foo** and specify a mount point of **bar**, the SpinShot is mounted at **foo/.spinshot/bar**. The syntax of the command follows.

```
filestorage vfs spinshot -src source_VFS_name
-mountpointname mount_point_name
```

The following example creates a one-time SpinShot of the virtual file system **vfs1** and gives it the mount point **myTuesData**, so that the SpinShot is mounted at **vfs1/.spinshot/myTuesData**.

```
spinserver::> filestorage vfs spinshot -src vfs1 \
-mountpointname myTuesData
```

Copying a Virtual File System

A VFS copy is a complete version of the original VFS with the same access (read-write or read-only) as the original. It can be on the same server or a different server, and in the same storage pool or on a different storage pool. It contains a complete copy of all data from the original VFS; it does not share blocks with the original VFS. For example, a copy of a 2-GB VFS uses 2 GB of disk space, whereas a SpinShot of the same VFS might use only a few kilobytes of disk space, depending on the extent of changes that have been made to the original since it was created. After you create a copy, mount it by using the **filestorage vfs mount** command.

Use the **filestorage vfs copy** command to copy a VFS. The syntax of the command follows.

```
filestorage vfs copy -vfsname VFS_name -copyvfsname
name_of_VFS_copy -copyspname storage_pool_name
```

The following example makes a copy of a VFS named **dev.testdata**. The copy is named **dev.copydata** and is located on a storage pool named **sp0**.

```
spinserver::> filestorage vfs copy -vfsname dev.testdata \
-copyvfsname dev.copydata -copyspname sp0
```

Moving a Virtual File System

You can move a VFS from one storage pool to another. This is useful for load balancing; for instance, if one storage pool has only 10 GB of free space and another storage pool of equal capacity has 200 GB of free space, you can move several VFSs from the first storage pool to the second pool so that the storage requirements are more equally balanced.

If a read-write VFS that has associated SpinShots or mirrors is moved, the SpinShots and mirrors remain in their original locations. It is recommended that you move or delete any SpinShots or mirrors that are associated with the parent read-write VFS to avoid potential confusion and to ensure efficient use of storage.

Use the **filestorage vfs move** command to move a VFS. The syntax of the command follows.

```
filestorage vfs move -vfsname VFS_name -spname
storage_pool_name
```

The following example moves a VFS named **user.fred** to a storage pool named **sp3**.

```
spinserver::> filestorage vfs move -vfsname user.fred \
-spname sp3
```

Mirroring a Virtual File System

A mirror of a VFS is a read-only VFS that contains a full copy of the data from the original (that is, it does not share blocks with the original). Data from the original must be replicated to the mirror; this can be performed manually or scheduled by using the VFS Scheduler. Mirrors can be used for backup and for disaster recovery.

Use the **filestorage mirror** commands to work with VFS mirrors.

Note: The **filestorage mirror** commands require the SpinMirror software package, which is licensed separately from the base SpinFS software package.

Creating a VFS Mirror

Use the **filestorage mirror new** command to create a VFS mirror. The syntax of the command follows.

```
filestorage mirror new -sourcevfsname source_VFS_name
-sinkvfsname sink_VFS_name -sinkspname storage_pool_name
```

The following example creates a mirror of a VFS named **user.john**. The mirror is named **mirror.john** and is located on a storage pool named **pool1**.

```
spinserver::> filestorage mirror new -sourcevfsname \
user.john -sinkvfsname mirror.john -sinkspname pool1
```

Replicating a VFS Mirror

After a VFS mirror is created, it needs to be replicated with data from the original VFS. Additional replications need to occur periodically to keep the mirror current. When a mirror is replicated, only the data that has changed is pushed to the mirror. Use the **filestorage mirror replicate** command to replicate the data in a VFS to a mirror VFS and to specify the number of mirror replicas that are kept. After the specified number is reached, the oldest replica is deleted as each new replica is created. The syntax of the command follows.

```
filestorage mirror replicate -sourcevfsname
source_VFS_name -sinkvfsname sink_VFS_name -maxspinshots
number_of_replicas
```

The following example replicates the data on a VFS named **user.john** to a mirror named **mirror.john**. The command keeps **3** replicas.

```
spinserver::> filestorage mirror replicate \
-sourcevfsname user.john -sinkvfsname mirror.john \
-maxspinshots 3
```

To avoid having to use this command to replicate VFSs to their mirrors manually, use the VFS Scheduler to schedule regular replications of VFSs. See “Scheduling VFS Operations” on page 169 for more information.

Displaying Information about VFS Mirrors

Use the **filestorage mirror show** command to display information about all VFS mirrors or about a specific VFS mirror. The syntax of the command follows.

```
filestorage mirror show [-sourcevfsname source_VFS_name
-sinkvfsname sink_VFS_name]
```

The following example displays mirror information about a VFS named **user.john** and its mirror, which is named **mirror.john**.

```
spinserver::> filestorage mirror show -sourcevfsname \
vfs1 -sinkvfsname m.vfs1
Source VFS  Sink VFS  Source SP   Sink SP    Sink Vers       Status

vfs1        m.vfs1    pool0       pool1      1               online
```

You can display information about the mirrors associated with a particular source VFS by using the **filestorage mirror show-group** command. Issue the command with the name of the source VFS and one of its mirrors; the command then displays information about the source VFS and all of its mirrors. Use this command if you need to locate all mirrors associated with a particular VFS. The command has the following syntax:

```
filestorage mirror show-group -sourcevfsname
source_VFS_name -sinkvfsname sink_VFS_name
```

The following example displays information about the mirrors of a source VFS named **san**, with one mirror named **san.m1**:

```
spinserver::> filestorage mirror show-group \
-sourcevfsname san -sinkvfsname san.m1
Source VFS        Sink VFS        Source SP Sink SP   Sink Ver Status

san               san.m1          pool0     pool1     2550018  online
san               san.m3          pool0     pool19    2550018  online
san               san.m0          pool0     pool0     2550018  online
san               san.m5          pool0     pool15    2550018  online
san               san.m4          pool0     pool14    2550018  online
san               san.m2          pool0     pool99    2550018  online
```

Deleting a VFS Mirror

To delete VFS mirrors, use the **filestorage mirror delete** command. This command deletes the specified VFS mirror of a source VFS. When this command is run, the original VFS remains intact, but the specified mirror is deleted.



Caution: If you have a VFS that is mirrored to two storage pools and you want to delete one of the mirrors, ensure that all mirrors are replicated immediately before deleting any mirrors, or otherwise ensure that all mirrors are synchronized with the source VFS before deleting any mirrors. This prevents clients from receiving potentially out-of-date date from an obsolete mirror.

The syntax of the command follows.

```
filestorage mirror delete -sourcevfsname source_VFS_name
-sinkvfsname sink_VFS_name
```

The following example deletes a VFS mirror named **mirror.john**, which is a mirror of a VFS named **user.john**.

```
spinserver::> filestorage mirror delete -sourcevfsname \
user.john -sinkvfsname mirror.john
```

Resetting the Export Rule Set for a VFS

Export rule sets enable you to restrict access to a VFS based on clients' IP addresses and levels of authentication. See "Managing Security" on page 207 for information on setting up export rule sets. To reset the export rule set on a single VFS, it is most efficient to use the **filestorage vfs modify** command with the **-rulesetname** parameter. To reset the export rule set for one or more VFSs, use the **filestorage vfs resetexportruleset** command. The command can propagate changes to export rule sets recursively down a chain of VFSs. The export rule set is changed on the VFS specified by the value of the **-vfsname** parameter; it can also be changed on all VFSs mounted below that VFS.

The **-rulesetname** and **-expoverride** parameters specify how to handle export rule sets on subordinate VFSs that are not using the export rule set named **none**. Subordinate VFSs are VFSs that are mounted below the VFS specified by the value of the **-vfsname** parameter. Each subordinate VFS is evaluated and changed as follows:

- If the rule set specified by the value of the **-rulesetname** parameter is **none**, then the rule set on each subordinate VFS is set to **none**, thereby clearing the rule sets on the target VFS and all subordinate VFSs.
- If the rule set of a subordinate VFS is currently set to **none**, then the rule set of the subordinate VFS is changed to the rule set specified by the value of the **-rulesetname** parameter.
- If the rule set of a subordinate VFS is currently set to a value other than **none**, then the ruleset of the subordinate VFS is changed only if the value of the **-expoverride** parameter is **true**. (The default is **false**.)

The command has the following syntax:

```
filestorage vfs resetexportruleset -vfsname VFS_name
-rulesetname rule_set_name -expoverride {true | false}
```

The following example resets the rule set for a virtual file system named **vfs1** to a rule set named **newRS**. The new rule set is applied to all VFSs mounted below **vfs1** regardless of their current rule sets.

```
spinserver::> filestorage vfs resetexportruleset \
-vfsname vfs1 -rulesetname newRS -expoverride true
```

Scheduling VFS Operations

The VFS Scheduler enables you to schedule VFS operations including SpinShots and mirrors. In VFS Scheduler terminology, a *job* is a *task* (such as replicating a mirror or creating a SpinShot) that runs on a specific *schedule*. Use the **filestorage scheduler** commands to create and modify VFS Scheduler tasks, schedules, and jobs.

Creating a VFS Scheduler Task

Use the **filestorage scheduler tasks new** command to create a new task. A task consists of a task name, the type of task (**mirror** or **spinshot**), the name of the VFS on which the task is to occur, and the number of task iterations that you want to be retained (that is, the number of mirror replicas or the number of SpinShots). The syntax of the command follows.

```
filestorage scheduler tasks new -name task_name -type
VFS_type -vfs VFS_name -limit limitation
```

You can specify a pattern as the value of *VFS_name*. For instance, to specify all VFSs that begin with the characters `dept.`, use a value of **dept.***. You can specify VFSs that are not to be included by using the **NOT{***expression***}** syntax. For instance, to specify all VFSs named **dept.*** except for the VFS named **dept.eng**, use the following syntax:

```
dept.*, NOT{dept.eng}
```

You can combine multiple expressions. In the following example, VFSs named **user.*** and **dept.*** are included, but VFSs named **user.z*** and **dept.acctg** are excluded.

```
user*., dept.*, NOT{user.z*}, NOT{dept.acctg}
```

The following example creates a new **mirror** task called **mirrorUserVFSs** that mirrors all VFSs that match the pattern **user***. A limit of **6** replicas is set.

```
spinserver::> filestorage scheduler tasks new -name \
mirrorUserVFSs -type mirror -vfs user* -limit 6
```

Displaying Information about a VFS Scheduler Task

Use the **filestorage scheduler tasks show** command to display information about one or more tasks. The syntax of the command follows.

```
filestorage scheduler tasks show [-name task_name]
```

The following example displays information about all tasks known to the VFS Scheduler.

```
spinserver::> filestorage scheduler tasks show
Name            Type       VFS            Limitation

mirrorAdmin     mirror     Admin          1
ssAdmin         spinshot   Admin          31
mirrorUserVFSs  mirror     user*          6
```

Modifying a VFS Scheduler Task

Use the **filestorage scheduler tasks modify** command to modify a task. The syntax of the command follows.

```
filestorage scheduler tasks modify -name task_name -type
VFS_type -vfs VFS_name -limit limitation
```

The following example modifies a task called **mirrorUserVFSs** so that it mirrors all VFSs that match the pattern **usr.***. A limit of **3** replicas is set.

```
spinserver::> filestorage scheduler tasks modify -name \
mirrorUserVFSs -type mirror -vfs usr.* -limit 3
```

Deleting a VFS Scheduler Task

Use the **filestorage scheduler tasks delete** command to delete a task. The syntax of the command follows.

```
filestorage scheduler tasks delete -name task_name
```

The following example deletes a task called **mirrorUserVFSs**.

```
spinserver::> filestorage scheduler tasks delete -name \
mirrorUserVFSs
```

Deleting All SpinShots or Mirror Replicas Created by a VFS Scheduler Task

The **filestorage scheduler tasks cleanup** command deletes all SpinShots or mirror replicas created by a VFS Scheduler task. If the task is a SpinShot task, the command deletes all SpinShots that were created by the task. If the task is a mirror task, the command deletes all replicas that were created by the task. The command has the following syntax:

```
filestorage scheduler tasks cleanup -name task_name
```

The following example cleans up all mirror replicas associated with a mirror VFS Scheduler task named **mirrorUserVFSs**:

```
spinserver::> filestorage scheduler tasks cleanup \
-name mirrorUserVFSs
```

Creating a VFS Scheduler Schedule

After you have created tasks, you need to create schedules on which the tasks run. Use the **filestorage scheduler schedules new** command to create a schedule. The command requires the name of the new schedule and the time at which the schedule is to run. The syntax of the command follows.

```
filestorage scheduler schedules new -name schedule_name
-minutes minutes -hours hours -months months -dom
days_of_month -dow days_of_week
```

Each schedule is assigned one of the following time types: `minutely`, `hourly`, `daily`, `weekly`, `monthly`, or `yearly`. The type is automatically determined, and is displayed in the output of the **`filestorage scheduler schedules show`** command.

Each time parameter can be specified as a range by using the following expressions:

- As a single value. For instance, specify a value of **`-dow Tuesday`** for a schedule that is to run every Tuesday.
- As a comma-separated list of values. For instance, specify a value of **`-hours 5,9,15`** for a schedule that is to run at 5:00 a.m., 9:00 a.m., and 3:00 p.m.
- As a slash-delimited interval. For instance, specify a value of **`-minutes /15`** for a schedule that is to run every 15 minutes.
- As a dash-delimited range. For instance, specify a value of **`-dom 20-31`** for a schedule that is to run every day from the 20th of the month through the 31st of the month.
- As an asterisk (*) to specify all possible values. For instance, specify a value of **`-months *`** for a schedule that is to run in all months.

Note: These expressions can be used only in the CLI. To specify a time range or interval in the GUI, select the appropriate values from the drop-down boxes.

Table 13-2. VFS Scheduler Format Examples

Time specifier	Single value	Example of comma-separated list	Example of slash-delimited interval	Example of dash-delimited range
Minutes	**0** through **59**	**0, 30, 45**	**/20**	**0-10**
Hours	**0** through **23**	**0, 6, 18**	**/2**	**0-6**
Months	**January** through **December**	**January, May, September**	**/3**	**June-August**
Days of the month	**1** through **31**	**1, 10, 25**	**/5**	**5-10**
Days of the week	**Sunday** through **Saturday**	**Monday, Wednesday, Friday**	**/3**	**Monday-Friday**

The following example creates a new schedule called **`noonOnWeekdays`** that runs every weekday (Monday through Friday) at 12:00 noon.

```
spinserver::> filestorage scheduler schedules new -name \
noonOnWeekdays -minutes 0 -hours 12 -months * -dom * \
-dow Monday-Friday
```

Note: Scheduled SpinShots are named and mounted according to their scheduling interval; for instance, hourly SpinShots are mounted under the `./.spinshot/hourly.`*n* directory, where *n* is an integer that increases as the number of SpinShots grows. The most recent hourly SpinShot is located in the `hourly.0` directory, the next most recent in the `hourly.1` directory, and so on. SpinShots must be scheduled according to a scheme that matches the naming convention. For instance, you cannot schedule hourly SpinShots of the same VFS to occur on both the hour and the half-hour, as this breaks the naming convention. Because a task and a schedule are separate until they are bound together into a job, the SpinManager software does not enforce this limitation when tasks and schedules are set up.

This limitation does not apply to scheduled mirrors.

Note: For any given VFS or set of VFSs, you cannot have two VFS Scheduler SpinShot jobs of the same time type (that is, minutes, hours, month, days of month, or days of week) whose tasks overlap each other. For instance, you cannot schedule two SpinShot jobs that each run at 15 minutes past the hour on all VFSs named `usr.*`.

Displaying Information about a VFS Scheduler Schedule

Use the **`filestorage scheduler schedules show`** command to display information about one or more schedules. The command has the following syntax:

`filestorage scheduler schedules show` [**`-name`** *`schedule_name`*]

The following example displays information about all schedules:

```
spinserver::> filestorage scheduler schedules show
 Name         Minutes   Hours    Months  DaysOfMont DaysOfWeek


daily              0       0       *       *          *
hourly             35      *       *       *          *
monthly            0       0       *       1          *
weekly             0       0       *       *          Sunday
yearly             0       0       January 1          *
noonOnWeekdays     0       12      *       *          Monday-Friday
```

Modifying a VFS Scheduler Schedule

Use the **`filestorage scheduler schedules modify`** command to modify a schedule. The command has the following syntax:

`filestorage scheduler schedules modify -name` *`schedule_name`* [**`-minutes`** *`minutes`*] [**`-hours`** *`hours`*] [**`-months`** *`months`*] [**`-dom`** *`days_of_month`*] [**`-dow`** *`days_of_week`*]

The following example modifies a schedule named **`15thDay`** so that it runs on the fifteenth day of each month at 5:00 p.m.:

```
spinserver::> filestorage scheduler schedules modify \
-name 15thDay -minutes 0 -hours 17 -dom 15
```

Deleting a VFS Scheduler Schedule

Use the **filestorage scheduler schedules delete** command to delete a schedule. The command has the following syntax:

```
filestorage scheduler schedules delete -name schedule_name
```

The following example deletes a schedule named **every5days**:

```
spinserver::> filestorage scheduler schedules delete \
-name every5days
```

Creating a VFS Scheduler Job

To create a job, you bind a task to a schedule. The task then runs on the schedule. To create a job, use the **filestorage scheduler jobs new** command. The command has the following syntax:

```
filestorage scheduler jobs new -task task_name -schedule
schedule_name -autorun {true | false}
```

The following example creates and enables a new job by binding the task named **mirrorUsers** to the schedule named **atNoon**:

```
spinserver::> filestorage scheduler jobs new -task \
mirrorUsers -schedule atNoon -autorun true
```

Modifying a VFS Scheduler Job

To modify a job, use the **filestorage scheduler jobs modify** command. The command has the following syntax:

```
filestorage scheduler jobs modify -task task_name
-schedule schedule_name -autorun {true | false}
```

The following example stops the automatic running (that is, disables) the job that consists of the task named **mirrorUsers** bound to the schedule named **atNoon**:

```
spinserver::> filestorage scheduler jobs modify -task \
mirrorUsers -schedule atNoon -autorun false
```

Displaying Information about a VFS Scheduler Job

Use the **filestorage scheduler jobs show** command to display the following information about one or more jobs:

- Task name
- Schedule name
- Whether the job automatically runs
- Time of next execution (specific job only)
- Time of last execution
- Host of last execution (specific job only)
- Status of last execution

The command has the following syntax:

```
filestorage scheduler jobs show [-task task_name -schedule
schedule_name]
```

To display information about all jobs, issue the command without the **-task** and **-schedule** parameters. To display information about a specific job, specify the parameters.

The following example displays information about all jobs:

```
spinserver::> filestorage scheduler jobs show

  Task          Schedule      Autorun?    Last Execution            Status
all_1           weekly          true       7/6/03 12:00 AM   Completed Successfully
all_2           daily           true       7/11/03 12:00 AM  Completed Successfully
all_3           hourly          true       7/11/03 3:00 PM   Completed Successfully
```

Testing a VFS Scheduler Job

Use the **filestorage scheduler jobs test** command to run a test of a VFS Scheduler job immediately, without having to wait for the next scheduled time interval.

The command has the following syntax:

```
filestorage scheduler jobs test -task task_name -schedule
schedule_name
```

The following example runs a test of a task named **mirrorUsers** bound to the schedule named **atNoon**:

```
spinserver::> filestorage scheduler jobs test -task \
mirrorUsers -schedule atNoon
```

Cleaning Up a VFS Scheduler Job

Use the **filestorage scheduler jobs cleanup** command to remove obsolete SpinShots and mirror replicas associated with a specified VFS Scheduler job. The command has the following syntax:

```
filestorage scheduler jobs cleanup -task task_name
-schedule schedule_name
```

The following example cleans up all SpinShots or mirror replicas associated with a job that includes a task named **spinshotUsers** and a schedule named **halfHour**:

```
spinserver::> filestorage scheduler jobs cleanup -task \
spinshotUsers -schedule halfHour
```

Deleting a VFS Scheduler Job

Use the **filestorage scheduler jobs delete** command to delete a job. The command has the following syntax:

```
filestorage scheduler jobs delete -task task_name
-schedule schedule_name
```

The following example deletes a job that includes a task named **mirrorUsers** and a schedule named **atNoon**:

```
spinserver::> filestorage scheduler jobs delete -task \
mirrorUsers -schedule atNoon
```

14.0 Managing SpinAccess CIFS

When you have created a virtual server and it is available to the network, you can mount the file system by using the Network File System (NFS) for UNIX clients.

The process described above enables the system for NFS clients, but the virtual server is not yet available to Windows clients, and the file system cannot yet be accessed by using Windows file sharing. With the Common Internet File System (CIFS), users with different platforms and computers can share files. Your next step is to make the virtual server available to Windows clients and enable file sharing. Virtual servers that serve CIFS clients must be members of a CIFS domain.

14.1 CIFS Management Concepts

Windows network clients that need to use file server resources must be able to locate the file server on the network. CIFS does this with NetBIOS names. A NetBIOS name is a case-insensitive, 16-character name that uniquely identifies a host on a subnetwork. You must provide a unique NetBIOS name for each of the CIFS virtual servers you define. This name is used when CIFS clients connect to the file server.

The Windows Internet Name Service (WINS) was created as an extension to the original NetBIOS naming specification. WINS performs the same basic function as a DNS server; that is, it accepts name registrations from hosts that must register or publish their network services. It resolves names into IP addresses for hosts that need to access those resources. WINS servers also communicate with each other and exchange host information so that hosts in one subnet can be known in another subnet. WINS does, however, complicate the name registration process for hosts.

The NetBIOS protocol partitions groups of servers into multiple subnets, thus requiring the client to know the IP address of the host if it exists on another subnet. The WINS server provides ways to look up server IP addresses for servers that are on a separate subnet. The NetBIOS node types determine the process by which the original NetBIOS protocol and WINS are used to locate a server.

Four different mechanisms are currently in place to register NetBIOS host names on a network. These are described as follows:

- The **B** or Broadcast NetBIOS Node Type is a purely broadcast mechanism, where the host announces its name to the subnet.
- The **P** or Peer NetBIOS Node Type is a WINS mechanism, where the host is configured with the name of a WINS server and informs the WINS server of its intent to use a

NetBIOS host name.

- The **H** or Hybrid NetBIOS Node Type is a mixed-mode announcement mechanism, where the host first contacts the WINS server and negotiates the use of its NetBIOS name and then broadcasts its name to the local subnet.
- The **M** or Microsoft NetBIOS Node Type is another mixed-mode announcement mechanism, where the host first broadcasts its name to the local subnet and then contacts a WINS server and negotiates for use of its NetBIOS host name.

The default NetBIOS Node Type is **M**.

Note: A WINS Server IP address is required if any virtual servers are using the **M**, **H**, or **P** NetBIOS node types. Without a WINS Server IP address, virtual servers can have only a **B** NetBIOS node type.

See the Windows documentation for further information about WINS and NetBIOS.

Each CIFS virtual server must participate in the Windows NT Domain Security model and have a corresponding account on the Windows Primary Domain Controller for the domain it participates in. Also, to prevent users of one CIFS virtual server in one domain from seeing another CIFS virtual server in another domain, NetBIOS name announcements must be limited to the subnet in which the virtual server is located.To accomplish this, NetBIOS name announcements are limited by the subnet mask that is specified in the management interface when a virtual interface is created.

To define a CIFS domain, you need to have the following information available.

Domain Name	
Primary Domain Controller (PDC) Name	
Primary Domain Controller IP address	
WINS Server IP address (Optional, but recommended)	
Backup Domain Controller (BDC) Name 1 (Optional)	
Backup Domain Controller IP address 1 (Optional)	
Backup Domain Controller Name 2 (Optional)	
Backup Domain Controller IP address 2 (Optional)	
Backup Domain Controller Name 3 (Optional)	
Backup Domain Controller IP address 3 (Optional)	
Backup Domain Controller Name 4 (Optional)	
Backup Domain Controller IP address 4 (Optional)	

14.2 CIFS Management Tasks

To enable Windows file sharing, you must complete the following tasks:

1. Define the Windows domain in which the virtual server resides.
2. Define domains that are visible to the virtual server.
3. Enable the virtual server for CIFS.
4. Create CIFS shares.

Defining a CIFS Domain

Use the **spinaccess cifs domains new** command to create and configure a CIFS domain.

A CIFS server must exist within a configured domain. By configuring the CIFS domains, you inform the SpinServer of the Windows NT or Windows 2000 domains that exist on the network and that have access to the SpinServer.

Note: Regardless of the case used to enter the new domain name, CIFS domain names are converted to all uppercase when they are created.

Creating a CIFS Domain

Use the **spinaccess cifs domains new** command to create a new CIFS domain. The command has the following syntax:

```
spinaccess cifs domains new -domainname CIFS_domain_name
[-winsip WINS_IP_address] -pdcname
primary_domain_controller_name -pdcip
primary_domain_controller_IP_address [-bdcname1
backup_domain_controller_name_1 -bdcip1
backup_domain_controller_IP_address_1] [-kerbconf
Kerberos_configuration_name]
```

The following example creates a new CIFS domain named **ENG**. The domain's WINS server has an IP address of **10.1.1.116**. The primary domain controller is named **STORM** and has an IP address of **10.1.1.116**. The backup domain controller is named **RAIN** and has an IP address of **10.1.1.118**.

```
spinserver::> spinaccess cifs domains new -domainname \
ENG -winsip 10.1.1.116 -pdcname STORM -pdcip 10.1.1.116 \
-bdcname1 RAIN -bdcip1 10.1.1.118
```

Note: If you specify backup domain controllers, the BDC fields with the lowest values must be entered first. For example, if you have one backup domain controller, enter **-bdcname1** and **-bdcip1**. If you have two, enter **-bdcname1**, **-bdcip1**, **-bdcname2**, and **-bdcip2**. If you have three, enter **-bdcname1**, **-bdcip1**, **-bdcname2**, **-bdcip2**, **-bdcname3**, and **-bdcip3**. All must be entered before a fourth backup domain controller is specified.

Displaying Information about CIFS Domains

To review the list of CIFS domains you have created and configured, use the **spinaccess cifs domains show** command.

To display general configuration information about all CIFS domains, do not use the parameter. To display information about one specific CIFS domain, use the command with the parameter.

The following information about the CIFS domains is displayed by the system when you do not specify a particular domain: Domain name, PDC name, and Kerberos configuration name. When you specify a domain, the following additional information is displayed: WINS, PDC name and IP address, Kerberos configuration name, and BDC names and IP addresses.

The **spinaccess cifs domains show** command has the following syntax:

spinaccess cifs domains show [**-domainname** *domain_name*]

The following example displays information about a CIFS domain named **ENG**.

```
spinserver::> spinaccess cifs domains show -domainname ENG
Domain                        ENG
Windows SID                   S-1-5-21-393965755-660102513-868425949
WINS                          10.1.1.41
PDC Name                      XAVIER
PDC IP                        10.1.1.41
Kerberos Configuration Name   example.com
BDC 1 Name                    MAGNETO
BDC 1 IP                      10.1.1.42
BDC 2 Name
BDC 2 IP
BDC 3 Name
BDC 3 IP
BDC 4 Name
BDC 4 IP
```

Modifying a CIFS Domain

Use the **spinaccess cifs domains modify** command to modify a CIFS domain. The command has the following syntax:

```
spinaccess cifs domains modify -domainname
CIFS_domain_name [-winsip WINS_IP_address]
-pdcname primary_domain_controller_name
-pdcip primary_domain_controller_IP_address
[-bdcname1 backup_domain_controller_name]
[-bdcip1 backup_domain_controller_IP_address] [-kerbconf
Kerberos_configuration_name]
```

Note: The **-winsip** parameter is optional. When a virtual server is added to the domain, it must be of type **B** if the domain does not have a WINS IP address. If you create a CIFS domain with a WINS IP address and add a virtual server with a node type that is not **B** (that is, **M**, **P**, or **H**), the domain must have a WINS IP address if modified.

Note: The SpinServer supports only pre-Windows 2000 domain names.

The following example modifies a CIFS domain named **ENG**. The domain's WINS server has an IP address of **10.1.1.11**. The primary domain controller is named **STORM** and has an IP address of **10.1.1.11**. The backup domain controller is named **RAIN** and has an IP address of **10.1.1.117**.

```
spinserver::> spinaccess cifs domains modify -domainname \
ENG -winsip 10.1.1.11 -pdcname STORM -pdcip 10.1.1.11 \
-bdcname1 RAIN -bdcip1 10.1.1.117
```

Removing a Backup Domain Controller from a Domain

If you need to remove one or more backup domain controllers (BDCs) from a CIFS domain, use the **spinaccess cifs domains removebdc** command. The command has the following syntax:

```
spinaccess cifs domains removebdc -domainname domain_name
-bdcs list_of_BDCs
```

The following example removes a BDC named **FOG** from a CIFS domain named **ENG**:

```
spinserver::> spinaccess cifs domains removebdc \
-domainname ENG -bdcs FOG
```

Deleting a CIFS Domain

To remove a CIFS domain from a system, use the **spinaccess cifs domains delete** command. A CIFS domain cannot be deleted if there are any CIFS servers within the CIFS domain (that is, CIFS virtual servers that were created by specifying the CIFS domain) or if a CIFS server lists the domain as visible.

Note: A cluster is limited to 4,096 domains. However, deleting a domain does not free up that space in the maximum domain list. For security reasons, all domain information ever entered into the cluster is kept indefinitely.

The **spinaccess cifs domains delete** command has the following syntax:

spinaccess cifs domains delete -domainname *domain_name*

The following example deletes a CIFS domain named **ENG**.

```
spinserver::> spinaccess cifs domains delete -domainname \
ENG
```

Renaming a CIFS Domain

In some circumstances it can be necessary to rename a CIFS domain without changing the domain's SID. Use the **tools cifs domains rename** command to rename a CIFS domain.

Caution: Changing the name of a CIFS domain improperly can result in the CIFS domain becoming inaccessible. Use extreme care when running this command.

The command has the following syntax:

tools cifs domains rename -domainname *domain_name* **-newname** *new_domain_name*

The following example changes the name of a CIFS domain from **WATERMELON** to **CANTALOUPE**.

```
spinserver::> tools cifs domains rename -domainname \
WATERMELON -newname CANTALOUPE
```

Managing CIFS Servers

After defining and creating a CIFS domain, you must enable a virtual server for CIFS.

Creating and Configuring a CIFS Server

A CIFS server consists of a virtual server, one or more virtual interfaces, and information about the server's domain. When creating a CIFS server, you must specify the NetBIOS node type, which determines how the server registers itself on the Windows Network.

If a domain is not defined to be *visible* to a virtual server, the CIFS virtual server ignores the domain, even if that domain has a Windows NT trust relationship with the CIFS virtual server's primary domain. Even if the Access Control List (ACL) on a file contains an entry from another domain, CIFS does not allow the domain to be considered during the access check because domains that are not defined as visible to the virtual server are masked.

The same NetBIOS name can be used on different CIFS domains.

The maximum number of CIFS servers per cluster is eight; the maximum number of CIFS shares per CIFS server is 128.

Use the **spinaccess cifs servers** command to enable a server for CIFS by configuring it and defining its domain visibility list. The command has the following syntax:

```
spinaccess cifs servers new -vsname virtual_server_name
-nbname NetBIOS_name -visibledomains domain_list
-domainname domain_name -nodetype { H | M | P |B }
-usekerberos {Yes | No} [-kerbprivuser Kerberos_user_name
-kerbprivpass Kerberos_password]
```

The following example enables a CIFS server:

```
spinserver::> spinaccess cifs servers new -vsname VS1 \
-nbname NB1 -visibledomains [MARKETING, EXE] -domainname \
ENG -nodetype M -usekerberos No
```

where

VS1 is the name of the virtual server. This name must already exist as a virtual server that was created by using the **vservers servers new** command.

NB1 is the NetBIOS name for the virtual server. The server created is available within the Windows Network as this name. In the domain, there must be a machine account with this name in the domain that has not been used by some other machine or virtual server.

[MARKETING, EXE] are the names of the domains that are visible to the virtual server named **VS1**. The virtual server's own domain is always included in the visible domain list that is displayed. By default, the visible domain list includes all domains that have been defined for the for the SpinServer.

ENG is the name of the CIFS domain for this server. This must be the name of a domain created by using the **spinaccess cifs domains new** command.

M is the NetBIOS node type for this virtual server. Valid values are **H**, **M**, **P**, and **B**. If you have a WINS server defined for the domain, use **M**. If you do not have a WINS server defined, use **B**. If you do not enter a value for the **-nodetype** parameter, the node type defaults to **M**.

Note: See RFC 1001 and 1002 for additional information about NetBIOS node types.

No specifies that this server is not to use Kerberos authentication.

Displaying Information about CIFS Servers

Use the **spinaccess cifs servers show** command to display information about CIFS servers. To display general configuration information about all CIFS servers, do not use the parameter. To display information about a specific CIFS server, use the command with the parameter. The command displays the following information:

- Virtual server name
- List of visible domains
- NetBIOS name
- Domain name
- Virtual interface names
- NetBIOS type
- Whether or not Kerberos is enabled
- Kerberos configuration name
- Kerberos Service Principal Name

The **spinaccess cifs servers show** command has the following syntax:

spinaccess cifs servers show [**-vsname** *virtual_server_name*]

The following example displays information about a CIFS server on a virtual server named **VS1**.

```
spinserver::> spinaccess cifs servers show -vsname VS1
Virtual Server                VS1
NetBios Name                  VS1
Domain                        ENG
Visible Domains               CIFSLAND, ENG, SLAB, SUPPORT
Interfaces                    Server2:Data, Server1:Data
Type                          P
Kerberos Enabled              Yes
Kerberos Configuration Name   example.com
Service Principal Name        HOST/SERVER1@EXAMPLE.COM
```

Deleting a CIFS Server

To remove a CIFS server from a system, use the **spinaccess cifs servers delete** command. Deleting the CIFS server from the system does not delete the virtual server. It removes only the ability to access the virtual server via CIFS. Deleting a CIFS virtual server deletes all the CIFS shares associated with that virtual server.

The **spinaccess cifs servers delete** command has the following syntax:

spinaccess cifs servers delete -vsname *virtual_server_name*

Note: Any defined CIFS shares on the server are automatically deleted when you delete the server.

Note: If you delete a CIFS server that is configured to use Kerberos, or is changed to use a different Service Principal Name (SPN), its original service principal is not automatically deleted from or disabled in the Kerberos realm. If you no longer need the service principal, it is recommended that you manually delete or disable the principal. This requires the Kerberos realm administrator's user name and password.

The following example deletes a CIFS server associated with a virtual server named **VS1**:

```
spinserver::> spinaccess cifs servers delete -vsname VS1
```

Forcing a Password Change on a CIFS Server

CIFS virtual servers must be members of a domain. They have Machine Trust Accounts on the Primary Domain Controller. Just like normal user accounts, Machine Trust Accounts have passwords that are known only to the machine using the account. For enhanced security, the password for the machine is randomly generated and cannot be specified by the system administrator.

The SpinServer system automatically changes the Machine Trust Account Passwords once a week. However, you have the option to force an immediate password change for a virtual server if you suspect a security breach or are trying to diagnose a problem. When issuing the **spinaccess cifs servers forcepasswordchange** command, you must specify a virtual server, which contacts its Primary Domain Controller and changes its Machine Trust Account password.

Use the **spinaccess cifs servers forcepasswordchange** command to force a password change for a CIFS server in a system. The command has the following syntax:

```
spinaccess cifs servers forcepasswordchange -vsname
virtual_server_name
```

The following example forces a password change for a CIFS server associated with a virtual server named **VS1**:

```
spinserver::> spinaccess cifs servers \
forcepasswordchange -vsname VS1
```

Resetting the Kerberos Password on a CIFS Server

If the Kerberos configuration on a CIFS server becomes corrupted, use the **spinaccess cifs servers resetpassword** command to reset the server's Kerberos password. The command has the following syntax:

```
spinaccess cifs servers resetpassword -vsname
virtual_server_name -uname admin_user_name -passwd
admin_password
```

Note: The administrator's password appears in clear text on the CLI; however, the GUI masks the password. To protect the security of the password, it is recommended that you use the GUI to issue this command.

The following command example resets the Kerberos password for a CIFS server named **VS1**. The administrative user's name is **admin**, with password **S3cr3t**.

```
spinserver::> spinaccess cifs servers resetpassword \
-vsname VS1 -uname admin -passwd S3cr3t
```

Modifying the List of Visible Domains

The **spinaccess cifs servers modify** command replaces the existing list of visible domains. If you remove a domain from the list of visible domains for a virtual server, any access control list (ACL) entries that specify a user or group from the removed domain no longer has access to the files. The entry is not removed from the ACL, but it is not considered during the access check. The command has the following syntax:

```
spinaccess cifs servers modify -vsname virtual_server_name
-visibledomains [list_of_domains] [-nodetype {H|M|P|B}]
[-kerbprivuser Kerberos_user_name -kerbprivpass
Kerberos_password]
```

The following example modifies a CIFS server such that the **MARKETING** domain is visible to a virtual server named **VS1**. By default, the visible domain list includes all domains that have been defined for the virtual server. In this example, the **EXE** domain is removed from the list of visible domains for the **VS1** server.

```
spinserver::> spinaccess cifs servers modify -vsname VS1 \
-visibledomains MARKETING
```

Managing CIFS Shares

After the necessary CIFS virtual server is created, the virtual server is viewable on the Windows Network, but no data is shared yet. To share data, a CIFS share must be created. The share name must be unique for the virtual server. A CIFS share is a named access point in a VFS that enables CIFS clients to view, browse, and manipulate files on a file server.

To define a CIFS share you create and configure a CIFS share by using the **spinaccess cifs shares new** command. Defining a share determines the directory within the file system that is accessible to Windows clients.

Note: A share is tied to its virtual server and CIFS server and is deleted if either the virtual server or the CIFS server to which it is attached is removed from the system.

Creating and Configuring a CIFS Share

To share data, a CIFS share must be created. Up to 128 shares can be created on a single virtual server. Use the **spinaccess cifs shares new** command to create a CIFS share. When you create a share, you can specify its properties, including the following:

- *Dynamic shares*. In a dynamic share, the share's name and path can be generated by substituting the user's name and domain with the **%u** and **%d** variables, respectively,

specified as the value of the **-path** parameter. For instance, if a dynamic share is defined as **/user/%u**, a user logged on as **bill** sees the path as **/user/bill**. Similarly, if a dynamic share is defined with a name of **%d_%u**, a user logged on as **barbara** from a domain named **FIN** sees the share as **FIN_barbara**. You can create a dynamic share by specifying the **-shareprops** parameter with the **dynpath** and **dynshare** values.

- *Mode-bit inheritance*. With mode-bit inheritance, new files and directories inherit the UNIX mode bits of the parent directory. For instance, if a new file is created in a directory that has a mode-bit setting of 777, then the new file automatically receives a mode-bit setting of 777. On a single CIFS share, you can specify either mode-bit inheritance with the **-shareprops mbinherit** parameter or explicit mode-bit setting with the **-dirmodebits** and **-filemodebits** parameters. You cannot specify both types of mode-bit setting on the same share.
- *Opportunistic locks*. By default, shares use opportunistic locks, also known as client-side caching. Some applications do not work well when oplocks are enabled. In particular, database applications such as Microsoft Access are vulnerable to corruption when oplocks are enabled. An advantage of shares is that a single path can be shared multiple times, with each share having different properties. For instance, if a path named **/dept/finance** contains both a database and other types of files, you can create two shares to it, one with oplocks disabled for safe database access and one with oplocks enabled for client-side caching.

The command has the following syntax:

```
spinaccess cifs shares new -vsname virtual_server_name
-sharename share_name -path directory_path [-shareprops
share_properties] [-dirmodebits directory_mode_bits]
[-filemodebits file_mode_bits] [-remark description]
```

The **-shareprops** parameter specifies any special properties for the share. Multiple values can be specified for a share. Possible values include the following:

- **dynpath**—This specifies that the share's path name is dynamic. For a full dynamic share, specify both the **dynpath** and **dynshare** values for this parameter.
- **dynshare**—This specifies that the share name is dynamic. For a full dynamic share, specify both the **dynpath** and **dynshare** values for this parameter
- **mbinherit**—This specifies that new files and directories inherit the mode bits of the parent directory.
- **oplocks**—This specifies whether the share uses opportunistic locks.

The **-dirmodebits** and **-filemodebits** parameters specify UNIX mode bits for new directories and files, respectively. These mode bits are used only when a directory or file is created. Accessing an existing directory or file through multiple CIFS shares that have different values for these parameters returns consistent permissions and access rights. For instance, assume you have a share named `share1` that has file mode bits of 777 and a share named `share2` that has file mode bits of 755, and that these shares overlap (that is, can access the same directories). If you create a file named `\\server\share1\foo`, the mode bits on that file are 777. If you create a file named `\\server\share2\bar`, the mode bits on that file are 755. If a user accesses the file named `foo` through the path `\\server\share2\foo`, the access permissions are 777; if a user accesses the file named `bar` through the path `\\server\share1\bar`, the access permissions are 755.

The **-remark** parameter specifies a descriptive remark up to 255 characters long. If there is a space in the descriptive remark, the entire string must be quoted (for example, **"This is a share."**).

The following example creates a CIFS share named **SHARE1** on a virtual server named **vs1**. Its directory path is **/development/engineering**. Opportunistic locks are specified on the share, and mode bits are explicitly set as **777** on directories and **775** on files.

```
spinserver::> spinaccess cifs shares new -vsname vs1 \
-sharename SHARE1 -path /development/engineering \
-shareprops oplocks -dirmodebits 777 -filemodebits 775
```

The following example creates a CIFS share named **SHARE2** on a virtual server named **vs1**. Its directory path is **/dept/system**, and mode-bit inheritance is set.

```
spinserver::> spinaccess cifs shares new -vsname vs1 \
-sharename SHARE2 -path /dept/system -shareprops \
mbinherit
```

Displaying Information about CIFS Shares

To review the list of CIFS shares, use the **spinaccess cifs shares show** command. To display general configuration information about all CIFS shares, do not use the parameters. To display information about one specific CIFS share, use the command with the parameters. The command displays the following information:

- Virtual server name
- NetBIOS name of the virtual server
- Share name
- Directory path
- Share properties
- Share remark
- Share status

The command has the following syntax:

```
spinaccess cifs shares show [-vsname virtual_server_name
-sharename share_name]
```

Use this command to display information about a CIFS share named **SYSTEM** on a virtual server named **VS1**.

```
spinserver::> spinaccess cifs shares show -vsname VS1 \
-sharename SYSTEM
Virtual Server        VS1
Share Name            SYSTEM
Server NetBios Name   VS1
Path                  /system
Share Properties      oplocks
Dir Mode Bits         755
File Mode Bits        755
Remark
Status                OK
```

Modifying a CIFS Share

Use the **spinaccess cifs shares modify** command to modify a CIFS share. The command has the following syntax:

```
spinaccess cifs shares modify -vsname virtual_server_name
-sharename share_name [-shareprops share_properties]
[-dirmodebits directory_mode_bits] [-filemodebits
file_mode_bits] [-remark description]
```

The following example modifies a CIFS share named **SHARE1** on a virtual server named **vs1**. Mode bits are explicitly set as **755** on directories and **644** on files.

```
spinserver::> spinaccess cifs shares modify -vsname vs1 \
-sharename SHARE1 -dirmodebits 755 -filemodebits 644
```

Deleting CIFS Shares

To remove a CIFS share from a system, use the **spinaccess cifs shares delete** command. The command has the following syntax:

```
spinaccess cifs shares delete -vsname virtual_server_name
-sharename share_name
```

Note: Deleting CIFS shares does not delete the data that exists in the VFS.

The following example deletes a CIFS share named **ENG_BILL** on a virtual server named **VS1**:

```
spinserver::> spinaccess cifs shares delete -vsname VS1 \
-sharename ENG_BILL
```

Managing CIFS Superusers

Most file systems include a superuser account. Such an account enables system administrators to modify files and their attributes without reference to the usual file permission system. This facility is needed for many management tasks including software installation, recovery from user error (for example, errors resulting in users locking themselves out of their files), and retrieval of files owned by obsolete users.

In the SpinServer system, the granting of superuser privilege is based on the user, not on group membership.

Because each virtual server has an independent user namespace, the privileged superuser and host lists are qualified by the virtual server. This enables the administrator to delegate superuser responsibilities at the virtual server level.

Creating a CIFS Superuser

To grant superuser access, use the **spinaccess superuser cifs new** command to add a CIFS user to the list of CIFS users that are eligible for superuser access. The command has the following syntax:

```
spinaccess superuser cifs new -vsname virtual_server_name
-domainname domain_name -username user_name
```

The following example adds a CIFS user named **tom** to the eligibility list for superuser access in the domain named **ENG** on the virtual server named **VS1**.

```
spinserver::> spinaccess superuser cifs new -vsname VS1 \
-domainname ENG -username tom
```

Displaying Information about CIFS Superusers

To display the list of CIFS users that are eligible for superuser access, use the **spinaccess superuser cifs show** command, as in the following example.

```
   spinserver::> spinaccess superuser cifs show
Virtual Server      Domain Name             User Name

VS1                 ENG                     louise
VS1                 ENG                     myrtle
VS1                 ENG                     tom
```

Deleting CIFS Superusers

To remove a CIFS user from the list of CIFS users that are eligible for superuser access, use the **spinaccess superuser cifs delete** command. The command has the following syntax:

```
spinaccess superuser cifs delete -vsname
virtual_server_name -domainname domain_name -username
user_name
```

The following example removes a CIFS user named **tom** from the list of eligibility for superuser access to a virtual server named **VS1** in a domain named **ENG**.

```
spinserver::> spinaccess superuser cifs delete -vsname \
VS1 -domainname ENG -username tom
```

Creating Privileged IP Addresses

To create an IP address block for superuser access, use the **spinaccess superuser privilegedips new** command. The client must connect from this IP address block to be eligible for superuser access. Privileged IP addresses apply to both NFS and CIFS, although privileged users must be CIFS users. The command has the following syntax:

```
spinaccess superuser privilegedips new -vsname
virtual_server_name -address IP_address -netmask netmask
```

The following example creates a superuser IP address block.

```
spinserver::> spinaccess superuser privilegedips new \
-vsname VS1 -address 192.168.0.0 -netmask 255.255.0.0
```

where

VS1 is the name of the virtual server. Each privileged IP block applies to one and only one virtual server. This parameter specifies the virtual server to which the specified privileged IP address block applies.

192.168.0.0 is the network address that must be matched for the user to be eligible for superuser access.

255.255.0.0 is the network mask that is applied to the client's IP address before it is compared to the network address that determines superuser eligibility.

Displaying Information about CIFS Privileged IP Addresses

To display the superuser IP address block, use the **spinaccess superuser privilegedips show** command. The following example displays all superuser IP address blocks.

```
spinserver::> spinaccess superuser privilegedips show
Virtual Server  Network Address      Hostmask

VS1             10.3.4.241       255.255.255.255
VS1             10.1.0.0         255.255.0.0
```

Deleting CIFS Privileged IP Addresses

To remove an IP address block from the list of network addresses that are eligible for superuser access, use the **spinaccess superuser privilegedips delete** command. The command has the following syntax:

```
spinaccess superuser privilegedips delete -vsname
virtual_server_name -address IP_address -netmask netmask
```

The following example removes an IP address block from the list of network addresses that are eligible for superuser access.

```
spinserver::> spinaccess superuser privilegedips delete \
-vsname VS1 -address 192.168.0.0 -netmask 255.255.0.0
```

where

VS1 is the name given to the virtual server. Each privileged IP block applies to one and only one virtual server. This parameter specifies the virtual server to which the specified privileged IP address block applies.

192.168.0.0 is the network address that must be matched for the user to be eligible for superuser access.

255.255.0.0 is the network mask that is applied to the client's IP address before it is compared to the network address that determines superuser eligibility.

15.0 Managing CIFS-NFS Translations

This chapter discusses setting up file sharing between CIFS and NFS clients.

15.1 CIFS-NFS Translation Concepts

The SpinFS architecture enables concurrent access to files via the NFS and CIFS protocols. Although both protocols' authentication models have the concept of a *principal* (a user or group), they use incompatible databases for storing principal information. CIFS stores user and group information on a Windows Domain Controller (PDC or BDC) by using either LanMan (NTLM) or Kerberos authentication.

NFS stores user and group information in the Network Information System (NIS), with few rules for authentication. NFS supports a single NIS domain per file server, whereas CIFS supports a network of cooperating domains. The SpinAccess CIFS-to-NIS translation provides a level of compatibility between these two models.

The NFS and CIFS internal representations of principal information are incompatible. SpinAccess bases its conversions on user and group names. A typical CIFS user name might be **`ENG\jones`**, where **`ENG`** is the domain or workgroup name and **`jones`** is the login name of a user. An NFS user might be **`jones`**. Note that CIFS names include a domain, where NFS names do not. Because NFS supports only a single domain per file server, the domain name is implied.

SpinAccess translates CIFS principals to NIS principals. It does this by translating a CIFS principal's internal representation into a name, matching the name against a set of rules, and, if a match is found, rewriting the name using a substitution template and converting the rewritten name into NFS' internal principal representation. A similar sequence is used to convert NIS principals into CIFS principals when sending replies to CIFS clients.

Note: Only CIFS requests are subject to translation. NFS principals are the canonical representation.

SpinAccess Conversion Rules

The SpinServer keeps an NIS domain and a set of conversion rules for each virtual server. Each rule consists of two pieces: a *pattern* and a *replacement*. Conversions start at the beginning of the appropriate list and perform a substitution based on the first matching rule. The pattern is a UNIX-style regular expression. The replacement is a string containing escape sequences representing subexpressions from the pattern, as in the UNIX **sed** program.

For example, the following rule converts the **jones** user in the **ENG** domain into the **jones** NIS user.

Pattern	Replacement
ENG\\jones	**jones**

Note that the backslash is a special character in regular expressions and must be escaped with another backslash.

The **^** (caret), **_** (underscore), and **&** (ampersand) characters can now be used as prefixes for digits in replacement patterns. These characters specify uppercase, lowercase, and initial-case transformations, respectively. For instance:

- If the initial pattern is **(.*)** and the replacement pattern is **\1**, then the string `jOe` is mapped to `jOe` (no change).
- If the initial pattern is **(.*)** and the replacement pattern is **_1**, then the string `jOe` is mapped to `joe`.
- If the initial pattern is **(.*)** and the replacement pattern is **\^1**, then the string `jOe` is mapped to `JOE`.
- If the initial pattern is **(.*)** and the replacement pattern is **\&1**, then the string `jOe` is mapped to `Joe`.

If the character following a **_** (backslash-underscore), **\^** (backslash-caret), or **\&** (backslash-ampersand) sequence is not a digit, then the character following the backslash is used verbatim.

The following example converts any CIFS user in the domain named **ENG** into a user with the same name in NIS.

Pattern	Replacement
ENG\\(.*)	**\1**

The double backslash (****) matches a single backslash. The parentheses denote a subexpression but do not match any characters themselves. The period matches any single character. The asterisk matches zero or more of the previous expressions. In this example, you are matching **ENG** followed by one or more of any character. In the replacement, **\1** refers to whatever the first subexpression matched. Assuming the CIFS user **ENG\jones**, the replacement evaluates to **jones**; that is, the portion of the name following **ENG**.

Note: If you are using the CLI, you must delimit all regular expressions with double quotation marks (**"**). For instance, to enter the regular expression **(.+)** in the CLI, type **"(.+)"** at the command prompt. Quotation marks are not required in the GUI.

For further information about regular expressions, see your UNIX system administration documentation, the online UNIX documentation for **sed** or **regex**, or *Mastering Regular Expressions*, published by O'Reilly and Associates.

CIFS-to-NFS Translation Guidelines

For CIFS-to-NFS translation, credentials supplied by a client are subject to translation before being processed by the SpinFS file system. File attributes (file owner and group) are subject to translation going in both directions. ACL entries are subject to translation of CIFS to NFS and NFS to CIFS.

File protection decisions take place on the SpinFS file system after any translations have occurred. All SpinServer users and groups are qualified by a domain. Domain zero is defined to represent the NIS namespace for the current virtual server. Other domain values map to one of the known CIFS domains.

Each virtual server is configured with a NIS domain. When a CIFS primary group cannot be mapped, or when NFS receives the attributes of a file whose domain is nonzero, the NIS **nobody** user or group (UID or GID **-2**) is substituted.

The translation process between CIFS and NFS proceeds as follows:

- When a credential is received from an NFS client, the SpinServer uses the NFS user and group, in domain **0**.
- When a credential is received from a CIFS client:
  - If both the user ID (UID) and primary group are translated, translations are used.
  - If only the user translates, the SpinServer uses the translated user and replaces the primary group with the NIS **nobody** (**-2**) group in the NFS domain (domain **0**).
  - If the user does not translate, the SpinServer uses the CIFS user and group. The SpinServer uses the CIFS user and group with a nonzero domain. The nonzero domain corresponds to the appropriate CIFS domain.
  - For each ACL entry, the SpinServer uses a translation if one is available; otherwise it sends a CIFS principal; that is, a nonzero domain.
- When file attributes are received, destined for a CIFS client:
  - If the file's owning domain is nonzero, the SpinServer assumes the user and group are already from a CIFS domain and does not attempt a mapping.

  - If the file's owning user and group map to CIFS, the SpinServer uses the mappings.
  - If the file's owning user or group does not have a mapping, the SpinServer translates them to local IDs that refer to the server.
  - For users and groups in ACLs, the SpinServer translates them if possible; otherwise it translates them to local IDs that refer to the server.
- When file attributes are received, destined for an NFS client:
  - If the domain is zero, the SpinServer passes the user and group unchanged.
  - If NFS sees a domain other than zero, it sets the user to **`nobody`** (**`-2`**) and the group to **`nobody`** (**`-2`**).

Rules for CIFS-to-NFS Mappings

The following recommendations and rules apply to CIFS-to-NFS mappings.

- A file's attributes specify a single domain, shared by the owning user and group.Thus, the file's user and group must come from the same domain.
- NFS credentials can never map to nonzero SpinFS domains. Thus, an NFS user can never get an untranslatable CIFS user's file access permissions.
- There is no facility for adjusting file ownerships after the translation rules have been adjusted. Changes to the translation rules can cause preexisting files now owned by invalid users—that is, CIFS users that are now translated to NIS users—to be inaccessible. For this reason, it is important to set up any desired translations before users begin creating files.
- Generally, all CIFS users are members of the *`your_domain`***`\Domain Users`** group. It is convenient to create an NIS group (for instance, **`cifs_user`**) that maps to and from **`Domain Users`**.
- If an NFS user is logged in as UID **`0`** (root) without Kerberos authentication, the user is mapped to **`-2`**/**`-2`** (**`nobody`**/**`nobody`**). Because Kerberos-enabled clients must be explicitly set up to map to root, this does not affect Kerberos-enabled clients.

15.2 CIFS-NFS Translation Tasks

To set up file sharing between CIFS and NFS clients, you must define the following:

- NIS domains and NFS servers
- CIFS-to-NFS user and group mappings

Managing NIS Domains

NIS provides simple name network lookup service and user authentication service. Its purpose is to provide information that needs to be available to the entire network, such as login names, passwords, home directories, and group information.

The SpinServer system's virtual servers must access NIS domains so that an NFS user's name can be identified from the listed user IDs, and vice versa. This process is required to accomplish CIFS-to-NFS user mappings. Your first task is to specify the NIS domains to be used.

Specifying NIS Domains

Use the **spinaccess nfs domains new** command to specify the NIS domains to be used by identifying their domain names and the NIS servers to be used for the domain.The NIS server option for this command specifies a square-bracketed, comma-separated list of NIS servers that handle NIS requests for the specified domain. If the option is omitted, a broadcast search is used to look for NIS servers for the domain. The command has the following syntax:

```
spinaccess nfs domains new -domainname NIS_domain_name
-nisservers [NIS_IP_addresses]
```

The following example specifies an NIS domain named **acmebricks.com** with NIS servers that have the IP addresses **10.2.2.1** and **10.3.3.1**.

```
spinserver::> spinaccess nfs domains new -domainname \
acmebricks.com -nisservers [10.2.2.1, 10.3.3.1]
```

Displaying Information about NIS Domains

After specifying the NIS domains for the network, use the **spinaccess nfs domains show** command to display and review the configuration information for the NIS domains. The command has the following syntax:

```
spinaccess nfs domains show [-domainname domain_name]
```

The following information about the NIS domains is displayed by the system: NIS domain name, communication type, and NIS domain server name or names.

To display general configuration information about all NIS domains, do not use the **-domainname** parameter. To display information about one specific NIS domain, use the command with the parameter.

The following example displays configuration information about an NIS domain named **acmebricks.com**:

```
spinserver::> spinaccess nfs domains show -domainname \
acmebricks.com
NIS Domain Name      acmebricks.com
Communication Type   Server
```

```
Domain Servers       10.2.2.1, 10.3.3.1
```

Modifying NIS Domains

Use the **spinaccess nfs domains modify** command to modify NIS domains. The command has the following syntax:

spinaccess nfs domains modify -domainname *NIS_domain_name* **-nisservers** *NIS_IP_address*

The following command modifies the list of NIS servers for an NIS domain named **acmebricks.com**.

```
spinserver::> spinaccess nfs domains modify -domainname \
acmebricks.com -nisservers [10.5.1.0, 10.5.2.0]
```

Deleting NIS Domains

Use the **spinaccess nfs domains delete** command to delete NIS domains. The command has the following syntax:

spinaccess nfs domains delete -domainname *domain_name*

The following example deletes an NIS domain named **acmebricks.com**.

```
spinserver::> spinaccess nfs domains delete -domainname \
acmebricks.com
```

Managing NFS Servers

You have specified the NIS domains to be used by the SpinServer system's virtual servers and the NIS domains now exist in the NFS domains table. Your next task is to attach the NIS domains to the virtual servers so that an NFS user's name can be identified from the listed user IDs, and vice versa.

Configuring an NFS Server's NIS Domains

To attach the NIS domains to virtual servers, use the **spinaccess nfs servers new** command. The command has the following syntax:

spinaccess nfs servers new -vsname *virtual_server_name* **-domainname** *NIS_domain_name* **-kerbrealms [***Kerberos_realms***]**

Note: Only one NIS domain can be bound to a virtual server at a time.

The following example attaches an NIS domain to a virtual server:

```
spinserver::> spinaccess nfs servers new -vsname VS1 \
-domainname acmebricks.com -kerbrealms [eng,finance]
```

where

VS1 is the name of the virtual server.

acmebricks.com is the NIS domain name.

[eng,finance] is the list of Kerberos realms that is visible to this virtual server. Multiple values must be separated by commas and enclosed in square brackets. Kerberos realms in this list must have been previously configured by using the **spinaccess kerberos realm** commands. See "Managing Security" on page 207 for information about Kerberos.

Displaying Information about an NFS Server's NIS Domains

To display and review the configuration information for NIS domains, use the **spinaccess nfs servers show** command. The command has the following syntax:

spinaccess nfs servers show [**-vsname** *virtual_server_name*]

The following information is displayed by the system: virtual server name and NIS domain name. To display configuration information about all NIS domains attached to virtual servers, do not use the parameter. To display information about a specific NIS domain, use the command with the **-vsname** parameter.

The following displays configuration information about the NIS domain attached to a virtual server named **VS1**.

```
spinserver::> spinaccess nfs servers show -vsname VS1
Virtual Server    VS1
NIS Domain Name   acmebricks.com
Kerberos Realms
```

Modifying Mappings Between Virtual Servers and NIS Servers

Use the **spinaccess nfs servers modify** command to modify mappings between virtual servers and their NIS servers. The command has the following syntax:

spinaccess nfs servers modify -vsname *virtual_server_name* **-domainname** *NIS_domain_name* **-kerbrealms [***Kerberos_realms***]**

The following example modifies the NIS domain named **acmebricks.com** attached to a virtual server named **VS1**. The list of Kerberos realms is modified to include **ENG** and **SALES**.

```
spinserver::> spinaccess nfs servers modify -vsname VS1 \
-domainname acmebricks.com -kerbrealms [ENG,SALES]
```

Deleting an NFS Server

To remove a virtual server's associations with NIS servers and NFS Kerberos realms, use the **spinaccess nfs servers delete** command. The command has the following syntax:

```
spinaccess nfs servers delete -vsname virtual_server_name
```

The following example removes the associations of a virtual server named **VS1** with all NIS servers and NFS Kerberos realms.

```
spinserver::> spinaccess nfs servers delete -vsname VS1
```

See "Managing Security" on page 207 for information on NFS Kerberos realms.

Managing CIFS-to-NFS Mappings

You have specified the NIS domains to be used by the SpinServer system's virtual servers and the NIS domains now exist in the NFS domains' table. The NIS domains have been attached to the virtual servers and you have created CIFS servers and shares. Your last task is to map CIFS to NFS users.

CIFS-to-NFS mappings are processed in the order of their positions, with the lowest numbered position (Position 1) being processed first. If the pattern is not matched fully, the next mapping is processed (Position 2), and so on. After a pattern matches fully, no more mappings are processed.

For further information about regular expressions, see *Mastering Regular Expressions* published by O'Reilly and Associates, Inc. at http://www.oreilly.com/catalog/regex.

Additional information can be obtained from a UNIX prompt by running **info regex**. For further information and examples, run **man 7 regex** at a UNIX prompt.

Creating CIFS-to-NFS Mappings

Use the **spinaccess cifs-nfs new** command to configure new CIFS-to-NFS mappings. When a new mapping is added, it is added at the top or the last position in the list of mappings to be displayed. The CIFS and NFS patterns must match the entire user or group name for the system to match the pattern. To be fully translated, each file and directory to be translated must match both a group and a user-name mapping. The command has the following syntax:

```
spinaccess cifs-nfs new -vsname virtual_server_name -type
{USER | GROUP} -cifspattern CIFS_pattern
-cifspatternreplacement CIFS_pattern_replacement
-nfspattern NFS_pattern -nfspatternreplacement
NFS_pattern_replacement
```

Note: When you create CIFS-to-NFS mappings, those CIFS clients that have open connections at the time the new mappings are created must log out and log back in to see the new mappings.

See the *Administration Reference* for additional and detailed information and examples of CIFS-to-NFS mappings.

The following example maps every CIFS user in the domain **ENG** to users in the NIS domain bound on the virtual server **VS1**.

```
spinserver::> spinaccess cifs-nfs new -vsname VS1 -type \
USER -cifspattern "ENG\\(.*)" -cifspatternreplacement \
"\1" -nfspattern "(.*)" -nfspatternreplacement "ENG\\\1"
```

where

VS1 is the name of the virtual server on which the mapping occurs.

USER specifies that this is a user-name mapping.

"ENG\\(.*)" is the regular expression search pattern that is used to match against incoming CIFS user or groups names. This CIFS mapping must directly correlate to the NFS-to-CIFS mapping defined by the **-nfspattern** and **-nfspatternreplacement** parameters.

"\1" is the replacement when the CIFS pattern matches the user name. This replacement is the equivalent user name for NFS. Regular-expression back references can be used to reference matched subexpressions in the CIFS pattern.

"(.*)" is the regular expression search pattern that is used to match against incoming NFS user or groups names. This NFS mapping must directly correlate to the CIFS-to-NFS mapping defined by the **-cifspattern** and **-cifspatternreplacement** parameters.

"ENG\\\1" is the replacement when the NFS pattern matches the user name. This replacement should be the equivalent user name for CIFS. Regular-expression back references can be used to reference matched subexpressions in the NFS pattern. This NFS mapping must directly correlate to the CIFS-to-NFS mapping defined by the **-cifspattern** and **-cifspatternreplacement** parameters.

Changing the Order of the Mapping Position

CIFS-to-NFS mappings are processed in the order of their positions, with the lowest numbered position (Position 1) being processed first. If the pattern is not matched fully, the next mapping is processed (Position 2), and so on. After a pattern matches fully, no more mappings are processed.

To change the order of the position of the mappings, use the following commands.

To Decrease the Position of the Mapping

To decrease the position of the mapping, that is, move it closer to the top of the list, use the **spinaccess cifs-nfs decreaseposition** command, as in the following example:

```
spinserver::> spinaccess cifs-nfs decreaseposition \
-position 6
```

where

6 is the position in the list of mappings to be decreased one position, to Position 5, on the list.

To Increase the Position of the Mapping

To increase the position of the mapping, that is, move it closer to the bottom of the list, use the **spinaccess cifs-nfs increaseposition** command, as in the following example:

```
spinserver::> spinaccess cifs-nfs increaseposition \
-position 6
```

where

6 is the position in the list of mappings to be increased one position, to Position 7, on the list.

To Move the Position to the Top of the List

To move the position of the mapping to the top of the list (Position 1), use the **spinaccess cifs-nfs movefirst** command, as in the following example:

```
spinserver::> spinaccess cifs-nfs movefirst -position 6
```

where

6 is the position in the list of mappings to be moved to the top of the list (the first position).

To Move the Position to the Bottom of the List

To move the position of the mapping to the bottom of the list (the last position), use the **spinaccess cifs-nfs movelast** command, as in the following example:

```
spinserver::> spinaccess cifs-nfs movelast -position 6
```

where

6 is the position in the list of mappings to be moved to the bottom of the list (the last position).

Displaying CIFS-to-NFS Mappings

After setting up the CIFS-to-NFS mappings for the network, use the **spinaccess cifs-nfs show** command to display and review the configuration information for the mappings. The command has the following syntax:

spinaccess cifs-nfs show [**-position** *position*]

The following information about the CIFS-to-NFS mappings is displayed by the system when no parameters are given:

- Position
- Virtual server name
- Group mapping
- CIFS pattern
- NFS pattern

The replacements are displayed by the system when the **-position** parameter is provided.

To display general configuration information about all mappings, do not use the parameter. To display information about one specific mapping, use the command with the parameter.

The following example displays mapping information about all positions

```
spinserver::> spinaccess cifs-nfs show
Pos  Virtual Server  Group?         CIFS pattern             NFS pattern

1    VS1            false          ENG\\(.+)                      (.+)
2    VS1            true           ENG\\Domain Users              cifs_user
3    VS1            true           ENG\\Systest                   systest
```

The following command displays mapping information about Position **1**.

```
spinserver::> spinaccess cifs-nfs show -position 1
Position                  1
Virtual Server            VS1
Group Mapping?            false
CIFS pattern              ENG\\(.+)
CIFS pattern replacement  \1
NFS pattern               (.+)
NFS pattern replacement   ENG\\\1
```

The following command displays mapping information about Position **2**.

```
spinserver::> spinaccess cifs-nfs show -position 2
Position                  2
Virtual Server            faceoff
Group Mapping?            true
CIFS pattern              ENG\\Domain Users
CIFS pattern replacement  cifs_user
NFS pattern               cifs_user
NFS pattern replacement   ENG\\Domain Users
```

The following example displays configuration information for the CIFS-to-NFS mapping at position **6**:

```
spinserver::> spinaccess cifs-nfs show -position 6
Position                  6
Virtual Server            VS1
Group Mapping?            true
CIFS pattern              ENG\\Systest
CIFS pattern replacement  systest
NFS pattern               systest
NFS pattern replacement   ENG\\Systest
```

Modifying CIFS-to-NFS Mappings

To change the configuration of CIFS-to-NFS mappings, use the **spinaccess cifs-nfs modify** command. The command has the following syntax:

```
spinaccess cifs-nfs modify -position mapping_position
-type {USER | GROUP} -cifspattern CIFS_pattern
-cifspatternreplacement CIFS_pattern_replacement
-nfspattern NFS_pattern -nfspatternreplacement
NFS_pattern_replacement
```

Note: The command modifies the entry at the position you specify and modifies only the parameters you pass in.

The CIFS and NFS patterns must match the entire user or group name for the system to match the pattern.

The following example modifies the CIFS-to-NFS mapping at position 4.

```
spinserver::> spinaccess cifs-nfs modify -position 4 \
-type USER -cifspattern "ENG\\(.*)" \
-cifspatternreplacement "\1" -nfspattern "(.*)" \
-nfspatternreplacement "ENG\\\1"
```

where

4 is the position to be modified.

USER specifies that this is a user-name mapping.

"ENG\\(.*)" is the regular expression search pattern that is used to match against incoming CIFS user or group names. This CIFS mapping must directly correlate to the NFS to CIFS mapping defined by the **-nfspattern** and **-nfspatternreplacement** parameters.

"\1" is the replacement when the CIFS pattern matches the user name. This replacement must be the equivalent user name for NFS. Regular-expression back references can be used to reference matched subexpressions in the CIFS pattern.

"(.*)" is the regular-expression search pattern that is used to match against incoming NFS user or groups names. This NFS mapping must directly correlate to the CIFS-to-NFS mapping defined by the **-cifspattern** and **-cifspatternreplacement** parameters.

"ENG\\\1" is the replacement when the NFS pattern matches the user name. This replacement must be the equivalent user name for CIFS. Regular-expression back references can be used to reference matched subexpressions in the NFS pattern. This NFS mapping must directly correlate to the CIFS-to-NFS mapping defined by the **-cifspattern** and **-cifspatternreplacement** parameters.

Deleting CIFS-to-NFS Mappings

Use the **spinaccess cifs-nfs delete** command to remove CIFS-to-NFS mappings from the system. The command has the following syntax:

spinaccess cifs-nfs delete -position *position*

The following example deletes the CIFS-to-NFS mapping at position **6**:

```
spinserver::> spinaccess cifs-nfs delete -position 6
```

16.0 Managing Security

This chapter discusses the use of Kerberos authentication and export rule sets to provide security for data that is managed by the SpinServer system.

16.1 Security Concepts

This section discusses the following security concepts:

- "Kerberos" on page 207 discusses Kerberos authentication.
- "Export Rule Sets" on page 208 provides information about export rule sets.
- "SpinAuth" on page 209 discusses the SpinAuth authentication mechanism.

Kerberos

Kerberos is a network authentication protocol developed at the Massachusetts Institute of Technology. Kerberos uses encryption keys to provide authentication for client/server applications. Authentication provides verification of a user's or process's identity to a server. In the SpinServer environment, Kerberos is used to provide authentication between virtual servers and NFS and CIFS clients.

Any entity (user or process) that can access Kerberos is given a Kerberos *principal*. A principal is a string that lists the following information:

- *Primary*—The name of the user or process; for instance, **`igor`**.
- *Instance*—A host name or other information that qualifies the primary; for instance, **`labServer`**.
- *Realm*—A logical network served by a single Kerberos database and one or more *Key Distribution Centers* (KDCs). A KDC is a server that issues Kerberos tickets. An example realm name is **`EXAMPLE.COM`**. Kerberos realms are listed in uppercase.

A Kerberos principal combines the primary, instance, and realm in the form *primary*/*instance*@*REALM*; for instance, **`igor/labServer@EXAMPLE.COM`**.

For more information about Kerberos, see http://web.mit.edu/kerberos/www/.

Export Rule Sets

Export rule sets enable you to restrict access to virtual file systems (VFSs) to clients that match specific IP addresses and minimum authentication strengths. Each VFS is associated with exactly one rule set. Each rule set is identified by a unique name and a unique numeric ID. A SpinCluster can contain up to 1,024 rule sets. Each cluster has at least one rule set with ID 0, which contains no rules. This rule set cannot be deleted, although it can be renamed or modified.

A newly installed or upgraded cluster contains only the default rule set. Each VFS on a new or upgraded cluster is associated with no rule set, and the default rule set is applied to it; however, if viewed with the **`filestorage vfs show-access`** command, the associated rule set is displayed as `none`. In a new cluster, the default rule set is associated with new VFSs, but you can also explicitly associate new VFSs with the rule set named **`none`**. The primary difference between allowing a new VFS to have the default rule set applied to it and applying the rule set named **`none`** to it is the inheritance or overriding of rule sets when the VFS is mounted or has its export rule set reset. See “Mounting a Virtual File System” on page 160 and “Resetting the Export Rule Set for a VFS” on page 169 for more information.

Rule sets consist of individual rules. A rule set can contain a large number of rules (approximately 4,000). Each rule specifies the number of mask bits in the client IP address that must be matched for that rule to apply to a particular client request. The rule also specifies the authentication levels that are required for both read-only and read-write operations. To have any access to a VFS, matching clients must authenticate at least as strongly as the authentication level specified by the read-only rule. To have write access to the VFS, matching clients must authenticate at least as strongly as the authentication level specified by the read-write rule. If a client makes an access request that is not permitted by the applicable rule set, the request fails with a permission-denied message. If a client IP address does not match any rule in the VFS’s rule set, then access is denied, except in the special case in which the rule set is empty (contains zero rules). If a rule set is empty, then all accesses are implicitly permitted.

When using export rule sets, you can specify a netgroup name instead of an IP address and mask bits to match clients to an export rule. A netgroup is a named collection of arbitrary IP addresses that is stored in a NIS map.

Rules are evaluated in the order in which they appear in the rule set. You can reorder rules in a rule set.

You can modify a rule set dynamically on a running SpinServer system; however, changes can take up to two minutes to propagate and take effect.

Export rule sets are not specific to any one virtual server; however, because each virtual server has an independent NIS domain and the set of IP addresses that a netgroup matches is dependent on NIS, each netgroup-based rule can match different clients on different virtual servers that have different NIS domains.

SpinAuth

SpinAuth is a lightweight authentication mechanism that improves the security of NFS requests, which are otherwise protected only by the AUTH_SYS mechanism. The AUTH_SYS mechanism, which is standard with NFS, does not provide secure authentication of client requests; NFS Kerberos provides strong authentication of client requests but has few available client programs. SpinAuth enables users to authenticate by using Kerberos and pass the authenticated identity to a SpinServer, which then caches the credentials for future requests from the users' clients.

The lifetime of the SpinAuth credential is the same as the lifetime of the Kerberos credential.

16.2 Security Tasks

This section provides the following information:

- "Configuring an NFS-Kerberos Environment" on page 209 describes enabling the SpinServer system to use Kerberos authentication with the NFS protocol.
- "Configuring a CIFS-Kerberos Environment" on page 210 describes enabling the SpinServer system to use Kerberos authentication with the CIFS protocol.
- "Configuring Export Rule Sets" on page 217 describes setting up export rule sets for VFSs.
- "Configuring SpinAuth" on page 222 describes setting up SpinAuth for authentication.

Configuring an NFS-Kerberos Environment

To enable the SpinServer system to use Kerberos authentication with the NFS protocol, perform the following steps:

1. Define a NIS domain in which Kerberos users are defined. See "Specifying NIS Domains" on page 197 for information.

2. Create a Kerberos realm configuration. See "Creating a Kerberos Realm Configuration" on page 210 for information.

3. Enable a virtual server to use Kerberos authentication by using the **`-kerbrealms`** parameter of the **`spinaccess nfs servers new`** or **`spinaccess nfs servers modify`** command. See "Managing NFS Servers" on page 198 for more information.

4. Optionally, you can require NFS clients to use Kerberos authentication with a virtual server by specifying the **`-minauth`** parameter of the **`vservers servers new`** or **`vservers servers modify`** command. See "Creating a Virtual Server" on page 147 for more information.

5. Create mappings between Kerberos principal names and NIS names. See “Creating and Managing Kerberos-NIS Mappings” on page 213 for information.

6. Bind the virtual interface to a Kerberos-enabled virtual server by using the **`vservers servers bindif`** command. See “Binding and Unbinding Virtual Interfaces” on page 149 for more information.

7. Enable the SpinServer interface to use Kerberos. See “Enabling Kerberos on an Interface” on page 215 for information.

Note: If a Kerberos-related command fails, the failure is typically caused either by incorrect entry of case-sensitive data or by clock skew (if the clocks on the SpinServer and the KDC server are more than 300 seconds apart). If a Kerberos-related command fails, it is recommended that you check both of these possible factors. When checking for clock skew, ensure that the time-zone and date settings are correct on all machines. If the failure results from clock skew, it is recommended that you adjust the clocks on one or more servers.

Note: If you create a virtual server for a new Kerberos realm, you can see Kerberos clock-skew errors. These errors do not necessarily originate from the SpinServer; they can originate from a disagreement between the SpinServer and the Kerberos server (for a Microsoft Kerberos server, this is the domain controller or Active Directory server), which has its own clock-skew setting. To prevent this situation from occurring, synchronize the clocks of the SpinServer and the Kerberos server, and ensure that the SpinServer and Kerberos server use the same time-zone setting. For information on setting up Network Time Protocol (NTP) to keep clocks synchronized automatically, see “Setting and Maintaining the Time” on page 260.

Configuring a CIFS-Kerberos Environment

To enable the SpinServer system to use Kerberos authentication with the CIFS protocol, perform the following steps:

1. Define the Kerberos realm or realms that are to be used to control access to the SpinServer. See “Creating a Kerberos Realm Configuration” on page 210 for information.

2. Set up a CIFS domain for Kerberos authentication and account management. See “Defining a CIFS Domain” on page 179 for information.

3. Create a CIFS-enabled virtual server in the target domain and realm. See “Creating and Configuring a CIFS Server” on page 182 for information.

Creating a Kerberos Realm Configuration

To create a Kerberos realm configuration, use the **`spinaccess kerberos realm new`** command. The command has the following syntax:

spinaccess kerberos realm new -configname *configuration_name* [**-adaddr** *Active_Directory_address*] [**-clockskew** *clock_skew*] **-kdc** *KDC_address* [**-kdcport** *KDC_port*] **-kdcvendor** {**Microsoft** | **Other**} [**-adminaddr** *admin_server_address*] [**-adminport** *admin_server_port*] **-passwdaddr** *password_server_address* [**-passwdport** *password_server_port*] **-realm** *Kerberos_realm*

where

-configname *configuration_name* specifies the name of the configuration that is to be created.

[**-adaddr** *Active_Directory_address*] specifies the IP address of the Active Directory server. This needs to be specified only if the KDC vendor is specified as **Microsoft**.

[**-clockskew** *clock_skew*] specifies, in seconds, the permissible clock skew. The default is **300** seconds. This value is global across all realms in the SpinServer cluster; do not modify it unless directed to do so by your Kerberos administrator.

-kdc *KDC_address* specifies the IP address of the KDC server.

[**-kdcport** *KDC_port*] specifies the port of the KDC server. The default is **88**; this value is used unless another value is explicitly specified.

-kdcvendor {**Microsoft** | **Other**} specifies whether the KDC vendor is **Microsoft** or **Other**. If **Microsoft** is specified, the **-adaddr** parameter must be specified; if **Other** is specified, the **-adminaddr** and **-adminport** parameters must be specified.

[**-adminaddr** *admin_server_address*] specifies the IP address of the administrative server. This parameter needs to be specified only if the the KDC vendor is specified as **Other**.

[**-adminport** *admin_server_port*] specifies the port number of the administrative server. The default is **749**; this value is used unless another value is explicitly specified. This parameter needs to be specified only if the KDC vendor is specified as **Other**.

-passwdaddr *password_server_address* specifies the IP address of the kpasswd password server.

[**-passwdport** *password_server_port*] specifies the port number of the kpasswd password server. The default is **464**; this value is used unless another value is explicitly specified.

-realm *Kerberos_realm* specifies the name of the Kerberos realm.

The following example creates a Kerberos realm configuration named **nfs-config1**. The KDC server has an address of **10.2.127.0** and uses port **88**. The KDC vendor is **Other**. The administrative server has an address of **10.10.0.1** and uses port **749**. The address of the kpasswd password server is **10.10.0.1** and uses port **464**. The Kerberos realm is **KERBEROS.EXAMPLE.COM**.

```
spinserver::> spinaccess kerberos realm new -configname \
nfs-config1 -kdc 10.2.127.0 -kdcport 88 -kdcvendor Other \
-adminaddr 10.10.0.1 -adminport 749 -passwdaddr \
10.10.0.1 -passwdport 464 -realm KERBEROS.EXAMPLE.COM
```

Displaying Information about a Kerberos Realm Configuration

Use the **spinaccess kerberos realm show** command to display information about one or more Kerberos realm configurations. The syntax of the command follows:

```
spinaccess kerberos realm show [-configname
configuration_name]
```

The following example displays information about a Kerberos realm configuration named **nfs-config1**:

```
spinserver::> spinaccess kerberos realm show -configname \
nfs-config1
Configuration Name             nfs-config1
Active Directory IP Address
Clock Skew
KDC Address                    10.2.127.0
KDC Port                       88
KDC Vendor                     Other
Admin Server Address           10.10.0.1
Admin Server Port              749
Password Server Address        10.10.0.1
Password Server Port           464
Kerberos Realm                 KERBEROS.EXAMPLE.COM
```

Modifying a Kerberos Realm Configuration

To modify a Kerberos realm configuration, use the **spinaccess kerberos realm modify** command. The command has the following syntax:

```
spinaccess kerberos realm modify -configname
configuration_name [-adaddr Active_Directory_address]
[-clockskew clock_skew] [-kdc KDC_address] [-kdcport
KDC_port] [-kdcvendor {Microsoft | Other}] [-adminaddr
admin_server_address] [-adminport admin_server_port]
[-passwdaddr password_server_address] [-passwdport
password_server_port] [-realm Kerberos_realm]
```

The following example modifies a Kerberos realm configuration named **nfs-config1** so that it uses a kpasswd password server at IP address **10.5.0.1** and port number **464**.

```
spinserver::> spinaccess kerberos realm modify \
-configname nfs-config1 -passwdaddr 10.5.0.1 -passwdport \
464
```

Deleting a Kerberos Realm Configuration

To delete a Kerberos realm configuration, use the **spinaccess kerberos realm delete** command. The command has the following syntax:

```
spinaccess kerberos realm delete -configname
configuration_name
```

The following example deletes a configuration named **nfs-config1**:

```
spinserver::> spinaccess kerberos realm delete \
-configname nfs-config1
```

Creating and Managing Kerberos-NIS Mappings

In an NFS environment that uses Kerberos authentication, you need to set up mapping between Kerberos-authenticated identities and NIS names so that NFS recognizes the authenticated names. Mappings are processed in the order in which they appear in the mappings list. To create and manage these Kerberos-NIS mappings, use the **spinaccess kerb-nis** commands.

Note: If you are using the CLI, you must delimit all regular expressions with double quotation marks (**"**). For instance, to enter the regular expression **(.+)** in the CLI, type **"(.+)"** at the command prompt. Quotation marks are not required in the GUI.

Creating a Kerberos-NIS Mapping

Use the **spinaccess kerb-nis new** command to create a new Kerberos-NIS mapping. The syntax of the command follows:

```
spinaccess kerb-nis new -vsname virtual_server_name
-clientpattern client_pattern -clientpatternreplace
client_pattern_replacement
```

The following example creates a Kerberos-NIS mapping on a virtual server named **vs0**. The Kerberos client pattern maps any principal in the Kerberos realm **ENG.EXAMPLE.COM** to the NIS user that corresponds to the principal's base name with any instance names removed. For instance, the principal named **sally/admin@ENG.EXAMPLE.COM** is mapped to **sally**.

```
spinserver::> spinaccess kerb-nis new -vsname vs0 \
-clientpattern "([^@/]+)(/[^@]+)?@ENG.EXAMPLE.COM" \
-clientreplacepattern "\1"
```

Note: To enter a question mark (**?**) in the CLI, you must first escape the character by pressing **Ctrl-V**. You do not need to use the **Ctrl-V** sequence before entering a question mark in the GUI.

Displaying Information about a Kerberos-NIS Mapping

Use the **spinaccess kerb-nis show** command to display information about one or more Kerberos-NIS mappings. The command has the following syntax:

```
spinaccess kerb-nis show [-position mapping_position]
```

The following example displays information about the Kerberos-NIS mapping at position **4**:

```
spinserver::> spinaccess kerb-nis show -position 4
```

Modifying a Kerberos-NIS Mapping

Use the **spinaccess kerb-nis modify** command to modify a Kerberos-NIS mapping. The syntax of the command follows:

```
spinaccess kerb-nis modify -position mapping_position
-clientpattern client_pattern -clientpatternreplace
client_pattern_replacement
```

The following example modifies the Kerberos-NIS mapping at position **3**. The Kerberos client pattern maps any principal in the Kerberos realm **FINANCE.EXAMPLE.COM** to the NIS user that corresponds to the principal's base name with any instance names removed.

```
spinserver::> spinaccess kerb-nis modify -position 3 \
-clientpattern "([^@/]+)(/[^@]+)?@FINANCE.EXAMPLE.COM" \
-clientreplacepattern "\1"
```

Changing the Position of a Kerberos-NIS Mapping

You can use the following commands to change the position of a Kerberos-NIS mapping:

- The **spinaccess kerb-nis movefirst** command moves a mapping to the top of the mappings list.
- The **spinaccess kerb-nis movelast** command moves a mapping to the bottom of the mappings list.
- The **spinaccess kerb-nis decreaseposition** command decreases the position of a mapping; that is, moves it closer to the top of the mappings list.
- The **spinaccess kerb-nis increaseposition** command increases the position of a mapping; that is, moves it closer to the bottom of the mappings list.

All of these commands require the **-position** *mapping_position* parameter to specify the current position of the mapping that is to be moved.

Deleting a Kerberos-NIS Mapping

Use the **spinaccess kerb-nis delete** command to delete a Kerberos-NIS mapping. The syntax of the command follows:

```
spinaccess kerb-nis delete -position mapping_position
```

The following example deletes the Kerberos-NIS mapping at position **5**.

```
spinserver::> spinaccess kerb-nis delete -position 5
```

Enabling Kerberos on an Interface

In an NFS environment, you cannot enable Kerberos on a virtual interface until it has been bound to a Kerberos-enabled virtual server. To enable an interface for Kerberos, use the **spinaccess kerberos ifconfig enablekerberos** command. The command has the following syntax:

```
spinaccess kerberos ifconfig enablekerberos -ssid
spinserver -ifname interface_name -spn
service_principal_name [-uname admin_user_name -passwd
admin_password]
```

where

-ssid *spinserver* specifies the SpinServer that is associated with the interface that is to be enabled for Kerberos.

-ifname *interface_name* specifies the interface that is to be enabled for Kerberos.

-spn *service_principal_name* specifies the service principal name (SPN). This value must be in the form **nfs/***host_name@REALM*, where *host_name* is the fully qualified domain name of the virtual interface that is being enabled to use Kerberos (for example, **machine.data.example.com**) and *REALM* is the name of the Kerberos realm (for instance, **EXAMPLE.COM**). *REALM* must match the value specified by the **-realm** parameter of the **spinaccess kerberos realm new** command. See “Creating a Kerberos Realm Configuration” on page 210 for information about the **spinaccess kerberos realm new** command.

-uname *admin_user_name* specifies an administrative user name for the Kerberos realm. This must be a Kerberos principal that is in the same realm as the principal specified by the **-spn** parameter; this principal must have permission to create or modify the principal specified by the **-spn** parameter. You do not need to enter this parameter if it has previously been specified for the principal name specified by the **-spn** parameter. The value specified by this parameter is used only for the command and is then discarded; no record of the user name is kept.

-passwd *admin_password* specifies the password for the Kerberos administrative user given in the **-uname** parameter. You do not need to enter this parameter if it has previously been specified for the SPN specified by the **-spn** parameter. The value specified by this parameter is used only for the command and is then discarded; no record of the password is kept. The password is echoed to the console if you are using the CLI; however, the password is masked if you use the GUI.

The following example enables Kerberos for an interface named **NFS1** on a SpinServer named **Server1**. The service principal name is **nfs/harpo.data.example.com@EXAMPLE.COM**. The administrative user name is **smith/admin**, which has a password of **0btus3**.

```
spinserver::> spinaccess kerberos ifconfig \
enablekerberos -ssid Server1 -ifname NFS1 -spn \
nfs/harpo.data.example.com@EXAMPLE.COM -uname \
smith/admin -passwd 0btus3
```

Displaying Kerberos Information about an Interface

Use the **spinaccess kerberos ifconfig show** command to display information about one or more Kerberos-enabled interfaces. The command has the following syntax:

```
spinaccess kerberos ifconfig show [-ssid spinserver
-ifname interface_name]
```

The following example displays information about an interface named **NFS1** on a SpinServer named **Server1**:

```
spinserver::> spinaccess kerberos ifconfig show -ssid \
Server1 -ifname NFS1
SpinServer ID            Server1
Name                     NFS1
Port                     serv0
IP Address               10.5.5.1
Netmask                  255.255.0.0
Failover Port
VIF ID                   2
Role                     Server
Service Principal Name   nfs/harpo.data.example.com@EXAMPLE.COM
```

Disabling Kerberos on an Interface

Use the **spinaccess kerberos ifconfig disablekerberos** command to disable Kerberos for an interface. The command has the following syntax:

```
spinaccess kerberos ifconfig disablekerberos -ssid
spinserver -ifname interface_name
```

The following example disables Kerberos for an interface named **NFS1** on a SpinServer named **Server1**:

```
spinserver::> spinaccess kerberos ifconfig \
disablekerberos -ssid Server1 -ifname NFS1
```

Configuring Export Rule Sets

Use the **spinaccess exportrule** commands to set up export rule sets and rules, which enable you to control access to VFSs based on clients' IP addresses and minimum authentication levels, as discussed in "Export Rule Sets" on page 208. You can create rules and rule sets by using the commands detailed in the following section. To apply them to VFSs, use the **filestorage vfs new**, **filestorage vfs modify**, **filestorage vfs mount**, and **filestorage vfs resetexportruleset** commands. See "Managing Virtual File Systems" on page 155 for information about these commands.

Creating a New Rule Set

Use the **spinaccess exportrule new** command to create a new export rule set. After you create a rule set, you can add rules to it, then manage the rules. The rule-set names **inherit** and **none** are reserved; you cannot create a new rule set with either of these names. The command has the following syntax:

spinaccess exportrule new -rulesetname *rule_set_name*

The following example creates an export rule set named **rs1**.

```
spinserver::> spinaccess exportrule new -rulesetname rs1
```

Displaying Information about Rule Sets

Use the **spinaccess exportrule show** command to display information about one or more rule sets. To display information about a specific rule set, run the command with the **-rulesetname** parameter; to display information about all rule sets, run the command with no parameter. Information displayed includes the rule set's name and ID. The command has the following syntax:

spinaccess exportrule show [**-rulesetname** *rule_set_name*]

The following example displays information about all export rule sets.

```
spinserver::> spinaccess exportrule show
     Ruleset Name          Ruleset ID

  default_ruleset               0
  rs1                           1
  rs2                           2
```

Copying a Rule Set

You can copy a rule set by using the **spinaccess exportrule copy** command. This command requires the name of the rule set that is to be copied and the name of the copy. The rule-set names **inherit** and **none** are reserved; these names cannot be used for a copy of a rule set. The command has the following syntax:

```
spinaccess exportrule copy -rulesetname rule_set_name
-newrulesetname new_rule_set_name
```

The following example makes a copy of an export rule set named **rs1**. The copy of the rule set is named **rs1copy**.

```
spinserver::> spinaccess exportrule copy -rulesetname \
rs1 -newrulesetname rs1copy
```

Renaming a Rule Set

You can rename a rule set by using the **spinaccess exportrule rename** command. This command requires the original name and new name of a rule set. The rule-set names **inherit** and **none** are reserved; you cannot rename a rule set with either of these names. The command has the following syntax:

```
spinaccess exportrule rename -rulesetname rule_set_name
-newrulesetname new_rule_set_name
```

The following example renames an export rule set named **rs1copy** to the name **ruleset1**.

```
spinserver::> spinaccess exportrule rename -rulesetname \
rs1copy -newrulesetname ruleset1
```

Deleting a Rule Set

Use the **spinaccess exportrule delete** command to delete a rule set and all of the rules it contains. You cannot delete the default rule set, which has an ID of **0** (zero). The command has the following syntax:

```
spinaccess exportrule delete -rulesetname rule_set_name
```

The following example deletes an export rule set named **rs1**.

```
spinserver::> spinaccess exportrule delete -rulesetname \
rs1
```

Adding a Rule to a Rule Set

Use the **spinaccess exportrule addrule** command to add a rule to an export rule set that was created by using the **spinaccess exportrule new** command. To add a rule, you need to specify the name of the rule set, the base client IP address to which the rule applies, the number of mask bits in the client IP address that the rule matches, the strength of the read-only rule, and the strength of the write rule. The strengths of the read-only and write rules, listed in order of increasing strength, are as follows:

- **any**—A matching client can access the VFS regardless of authentication strength.
- **cifs-ntlm**—A matching client can access the VFS if it is authenticated by CIFS NTLM or a stronger mechanism.
- **cifs-kerberos**—A matching client can access the VFS if it is authenticated by CIFS Kerberos or a stronger mechanism.
- **nfs-kerberos**—A matching client can access the VFS if it is authenticated by NFS Kerberos or a stronger mechanism.
- **never**—Matching clients cannot access the VFS.

Note: Write access is contingent upon read-only access, so specifying a value for the **-rwrule** parameter that is not as strong as the value of the **-rorule** parameter has no effect. For example, if you give the settings **-rorule nfs-kerberos -rwrule any**, CIFS clients cannot access the VFS in any way, even though the value of the **-rwrule** parameter is **any**.

The command has the following syntax:

```
spinaccess exportrule addrule -rulesetname rule_set_name
-clientmatch {client_IP_address | @netgroup_name}
[-netmask network_mask] -rorule {any | cifs-ntlm |
cifs-kerberos | nfs-kerberos | never} -rwrule {any |
cifs-ntlm | cifs-kerberos | nfs-kerberos | never}
```

To specify a client IP address, enter the IP address as the value of the **-clientmatch** parameter and specify a netmask by using the **-netmask** parameter. To specify a netgroup, specify the netgroup as the value of the **-clientmatch** parameter in the form **@***netgroup_name*, and do not specify the **-netmask** parameter.

The following example adds an export rule to the rule set named **rs1**. The rule applies to the client IP address **192.55.0.0**, and the network mask **255.255.128.0** is used to match the rule. The rule permits read-only access to **any** matching clients, but requires that matching clients have **nfs-kerberos** authentication to write files.

```
spinserver::> spinaccess exportrule addrule -rulesetname \
rs1 -clientmatch 192.55.0.0 -netmask 255.255.128.0 \
-rorule any -rwrule nfs-kerberos
```

The following example adds an export rule to the rule set named **rs2**. The rule applies to the netgroup named **group1**. The rule permits both read-only and read-write access to matching clients that have **any** authentication.

```
spinserver::> spinaccess exportrule addrule -rulesetname \
rs2 -clientmatch @group1 -rorule any -rwrule any
```

Listing the Rules in a Rule Set

Use the **spinaccess exportrule listrules** command to list the rules that are contained in a rule set. Information displayed includes the rule's rule set name, index number, client IP address, mask bits, read-only strength, and read-write strength. The command has the following syntax:

spinaccess exportrule listrules -rulesetname *rule_set_name*

The following example displays information about the rules in an export rule set named **rs1**.

```
spinserver::> spinaccess exportrule listrules \
-rulesetname rs1
Ruleset Name   Index     Client IP     Mask  Read Strength     Write
                                       Bits                  Strength

rs1            0         192.55.0.0    16    any             nfs-kerberos
rs1            1         192.56.0.0    16    cifs-ntlm       cifs-kerberos
rs1            2         192.27.0.0    16    any             any
```

Ordering the Rules in a Rule Set

Because netgroups can have overlapping membership sets, ordering of export rules in a rule set is not automatic. You can order rules in a rule set by using the following commands:

- To decrease the position of a rule (that is, move it closer to the top of the list), use the following command:

  spinaccess exportrule decreaseposition -rulesetname *rule_set_name* **-ruleindex** *index_number*

- To increase the position of a rule (that is, move it closer to the bottom of the list), use the following command:

  spinaccess exportrule increaseposition -rulesetname *rule_set_name* **-ruleindex** *index_number*

- To move a rule to the top of the list, use the following command:

  spinaccess exportrule movefirst -rulesetname *rule_set_name* **-ruleindex** *index_number*

- To move a rule to the bottom of the list, use the following command:

  spinaccess exportrule movelast -rulesetname *rule_set_name* **-ruleindex** *index_number*

Rules must be ordered according to the following guidelines:

- You cannot reorder IP address-based rules such that a less-specific rule precedes a

more-specific rule.

- If you have an IP address-based rule with netmask **0.0.0.0** (that is, matches all clients), it cannot be moved from the last position in the rule set, nor can any other rule, either IP address-based or netgroup-based, be placed after it.

If you attempt to move a rule and the move cannot be performed, the SpinServer issues the error message `Unable to move rule at position N`. If you use the **spinaccess exportule movelast** command to move a rule to the last position and there is an IP address-based **0.0.0.0** rule present, the rule is moved to the next-to-last position unless it is already in the next-to-last position, in which case the preceding error message is issued.

For example, assume you have rules for netgroups **@group1**, **@group2**, and **@group3**, and a rule for IP address **0.0.0.0** with netmask **0.0.0.0**, in that order. If you use the **spinaccess exportule movelast** command to move the **@group1** entry, the SpinServer moves it to the position after **@group3** and before the **0.0.0.0**/**0.0.0.0** rule. If you then try to move the **@group3** rule by using the **spinaccess exportrule movelast** or **spinaccess exportrule increaseposition** command, the attempt fails because the rule cannot be moved any higher in the list.

Similarly, assume you have rules for IP addresses and netmasks **10.4.4.120**/**255.255.255.255**, **10.3.0.0**/**255.255.0.0**, **10.5.0.0**/**255.255.0.0**, and **10.1.0.0**/**255.255.0.0**. If you attempt to move the **10.1.0.0**/**255.255.0.0** rule by using the **spinaccess exportrule movefirst** command, the rule is moved to the second position in the list because the rule cannot precede the **10.4.4.120**/**255.255.255.255** rule.

In summary, the **spinaccess exportrule movefirst** and **spinaccess exportrule movelast** commands move the specified rule as close to the top or bottom of the list as possible, within the framework of the ordering rules. If the specified rule cannot be moved at all, the SpinServer issues an error; otherwise, the SpinServer moves the rule to the appropriate place in the list.

Note: If you are working with netgroups and export rule sets, it is recommended that you upgrade all SpinServers in a cluster to SpinFS 2.5. If there are mixed SpinFS versions in a cluster, the SpinServers running earlier versions of SpinFS do not support native netgroups and ignore netgroup-based rules. Additionally, rules and rule sets cannot be uniformly managed on SpinServers that are running earlier versions of SpinFS.

Deleting a Rule from a Rule Set

Use the **spinaccess exportrule deleterule** command to delete a rule from a rule set. To specify a rule, use the index number for the rule displayed by the **spinaccess exportrule listrules** command.

The command has the following syntax:

```
spinaccess exportrule deleterule -rulesetname
rule_set_name -ruleindex rule_index
```

The following example deletes a rule with index number **2** from the export rule set named **rs1**.

```
spinserver::> spinaccess exportrule deleterule \
-rulesetname rs1 -ruleindex 2
```

Configuring SpinAuth

To use SpinAuth, perform the following steps:

1. Create an NFS-Kerberos configuration on the SpinServer system. A brief overview of the procedure follows.
   a. Create a Kerberos realm by using the **spinaccess kerberos realms new** command. See “Creating a Kerberos Realm Configuration” on page 210 for information.
   b. Set up a NIS domain by using the **spinaccess nfs domains new** command. See “Specifying NIS Domains” on page 197 for details.
   c. Associate the Kerberos realm and the NIS domain with a virtual server by using the **spinaccess nfs servers new** command. See “Configuring an NFS Server’s NIS Domains” on page 198 for information.
   d. Set up Kerberos-NIS mappings by using the **spinaccess kerb-nis new** command. See “Creating and Managing Kerberos-NIS Mappings” on page 213 for details.
   e. Enable Kerberos on one or more virtual interfaces by using the **spinaccess kerberos ifconfig enablekerberos** command. See “Enabling Kerberos on an Interface” on page 215 for information.
2. Enable SpinAuth on the virtual server by specifying the **-spinauth true** parameter on the **vservers servers new** command (for a new virtual server) or **vservers servers modify** command (for an existing virtual server).
3. On the client, use the client platform's **kinit** command to log on to the Kerberos realm.
4. On the client, run the **spinauthcli** command. This step assumes that the **spinauthcli** program has been compiled for your system. Network Appliance can supply you with precompiled **spinauthcli** binaries for your client platform upon request; contact your Network Appliance representative for more information. The **spinauthcli** command has the following syntax:

   spinauthcli [**-h**] [**-i**] [**-d**] [**-q**] *spinserver_name*

where

[**-h**] displays help information for the command

[**-i**] queries the credential's lifetime

[**-d**] destroys the credential

[**-q**] specifies that the command runs in quiet mode

spinserver_name specifies the name of the SpinServer on which SpinAuth credentials are to be established

For example, to set up SpinAuth credentials for a client on a SpinServer named **Server1.data.example.com**, run the following command:

```
% spinauthcli Server1.data.example.com
```

17.0 Protecting Against Viruses

This chapter discusses using the SpinAV™ component to protect data on your system from viruses.

17.1 Virus Scanning Concepts

Virus scanning is available on the SpinServer with the SpinAV component, which uses McAfee virus-scanning technology. SpinAV requires a separate license; contact your Network Appliance representative for details.

Virus scanning is enabled on individual VFSs, not on individual SpinServers. For individual virtual servers, you can specify whether virus scanning is enabled for new VFSs on that virtual server.

The SpinAV scanning engine and virus database must be downloaded from the Network Appliance NOW site. A valid user ID and password are required to access the antivirus files on the NOW site. These are provided to you with the SpinAV license.

17.2 Virus Scanning Tasks

This section discusses the setup and maintenance of the SpinAV antivirus component.

Setting up Virus Scanning on Virtual Servers and VFSs

To set up virus scanning on virtual servers and VFSs, use the **`-virusscanrw`** and **`-virusscanro`** parameters on the **`vservers servers new`**, **`vservers servers modify`**, **`filestorage vfs new`**, and **`filestorage vfs modify`** commands. These parameters take values of **`true`** and **`false`** to enable and disable virus scanning on read-write and read-only VFSs, respectively. Setting these parameters on a virtual server sets the virus-scanning defaults for VFSs on that virtual server. Default settings can be overridden by specifying the parameters for individual VFSs.

See “Managing Virtual Servers” on page 147 and “Managing Virtual File Systems” on page 155 for information on the **`vservers`** and **`filestorage vfs`** commands.

Configuring SpinAV for a Cluster

You can work with the SpinAV configuration for a cluster by using the **services antivirus configuration** commands.

Modifying the Antivirus Configuration for a Cluster

The **services antivirus configuration modify** command enables you to modify the antivirus configuration in a SpinServer cluster. The command has the following syntax:

```
services antivirus configuration modify -scannerenabled
{true | false} -repairfiles {true | false} -updatefileurl
FTP_URL -autoupdateenabled {true | false} -schedule
{#-Minutely | hourly | daily | weekly | monthly | yearly}
```

The following example enables virus scanning and automatic repair of infected files. It also enables automatic updates from an FTP site named **ftp://ftp.spinnakernet.com/pub/antivirus/2.5/update.cfg**. Updates occur **weekly**.

```
spinserver::> services antivirus configuration modify \
-scannerenabled true -repairfiles true -updatefileurl \
ftp://ftp.spinnakernet.com/pub/antivirus/2.5/update.cfg \
-autoupdateenabled true -schedule weekly
```

Testing the Download of an Antivirus Configuration File

Use the **services antivirus configuration testurl** command to test a download of the antivirus configuration file from the URL specified by the **-updatefileurl** parameter of the **services antivirus configuration modify** command. The test is run as a background job. The command returns a job ID; use the **jobs show** command to monitor the progress and outcome of the test.

The command has the following syntax:

```
spinserver::> services antivirus configuration testurl
```

Displaying Information about the Antivirus Configuration for a Cluster

The **`services antivirus configuration show`** command displays the following information about the antivirus configuration on a SpinServer cluster:

- Latest scanner level
- Whether antivirus scanning is enabled
- Whether infected files are automatically repaired
- FTP URL for updates
- Whether automatic updates are enabled
- The schedule for automatic updates

Configuring SpinAV on SpinServers

Each SpinServer has its own copy of an antivirus engine and virus-definition database. In SpinAV terminology, a scanner is the combination of an antivirus engine and a virus-definition database. Use the **`services antivirus servers`** commands to work with the SpinAV configuration on individual SpinServers, and the s**`ervices antivirus scanners`** commands to work with scanners.

Displaying Information about Antivirus Servers

The **`services antivirus servers show`** command displays information about the antivirus server on one or more SpinServers. The command has the following syntax:

`services antivirus servers show` [**`-ssid`** *`spinserver`*]

The command displays the following information:

- SpinServer ID
- Scanner status (`enabled` or `disabled`)
- Current scanner level
- Number of downloaded scanners (a scanner is considered to be an antivirus engine and a virus database file)
- Automatic update status

When the **-ssid** parameter is specified to display information about the antivirus scanner on a particular SpinServer, the command displays the following additional information:

- The time the current scanner was set
- Whether an update is currently in progress
- The time of the next automatic update
- The time of the last automatic update
- The last message from the automatic update process
- The number of antivirus scans completed
- The number of jobs in the high-priority queue
- The number of jobs in the low-priority queue
- The number of idle threads

Setting the Current Scanner for a SpinServer

The **services antivirus servers setcurrent** command sets the scanner for a specified SpinServer. The command has the following syntax:

```
services antivirus servers setcurrent -ssid spinserver
-scanner scanner
```

The following example sets scanner **1** as the current scanner on a SpinServer named **Server1**:

```
spinserver::> services antivirus servers setcurrent \
-ssid Server1 -scanner 1
```

Getting the Latest Antivirus Update

The **services antivirus servers getlatestupdate** command downloads the latest antivirus update to a specified SpinServer and sets the updated files as the current scanner level. This command must be run on each SpinServer that requires updated antivirus files. The command has the following syntax:

```
services antivirus servers getlatestupdate -ssid
spinserver
```

The following example retrieves the latest update to a SpinServer named **Server1** and sets the updated files as the current scanner level:

```
spinserver::> services antivirus servers getlatestupdate
-ssid Server1
```

Getting a Specified Antivirus Update

The **services antivirus servers getupdate** command retrieves a specific antivirus update in the form of an antivirus engine and a virus database file to a specified SpinServer, and sets the updated files as the current scanner level. The command has the following syntax:

```
services antivirus servers getupdate -ssid spinserver
-engineurl URL_for_engine -databaseurl URL_for_database
```

The following example retrieves an antivirus engine named **eng-1007.tgz** and a virus database named **dat-1007.tgz** from the FTP site **ftp://ftp.spinnakernet.com/pub/antivirus/2.5** to a SpinServer named **Server1**, then sets the new files as the current scanner level:

```
spinserver::> services antivirus servers getupdate -ssid
Server1 -engineurl
ftp://ftp.spinnakernet.com/pub/antivirus/2.5/eng-1007.tgz
-databaseurl
ftp://ftp.spinnakernet.com/pub/antivirus/2.5/dat-1007.tgz
```

Deleting Scanner Files from a SpinServer

The **services antivirus servers deletescannerfiles** deletes antivirus scanner files from a specified SpinServer. The command has the following syntax:

```
services antivirus servers deletescannerfiles -ssid
spinserver -scanner scanner
```

The following example deletes the files for scanner **7** from the SpinServer named **Server1**:

```
spinserver::> services antivirus servers
deletescannerfiles -ssid Server1 -scanner 7
```

Displaying Information about Scanners

The **services antivirus scanners show** command displays information about antivirus scanners. The command has the following syntax:

```
services antivirus scanners show [-scannerlevel
scanner_level]
```

The command displays the following information:

- Scanner level
- Engine version
- Database version

Deleting a Scanner

The **services antivirus scanners delete** command deletes a scanner level. The command has the following syntax:

```
services antivirus scanners delete -scannerlevel
scanner_level
```

The following example deletes scanner level **3**:

```
spinserver::> services antivirus scanners delete
-scannerlevel 3
```

18.0 Managing Data Backup and Restoration

This chapter discusses setting up data backups and restorations on a SpinServer system. After you set up the SpinServer system, the backups and restorations are initiated and run from a third-party backup system such as VERITAS NetBackup or Legato NetWorker. See your backup system's documentation for information on running backups and restorations.

18.1 Backup and Restoration Concepts

The Network Data Management Protocol (NDMP) is an open standard for controlling backups and restorations of NDMP-compliant file servers from NDMP-compliant data management applications (DMAs). Each NDMP-compliant file server has its own NDMP service, which is controlled by a DMA running on a backup server system. Because SpinServers are NDMP compliant, you do not need to install any DMA-specific client software on the SpinServer. Each SpinServer has an NDMP daemon that acts on NDMP requests.

There are different modes of NDMP: local, remote, and three-way.

- In local mode, the tape device is directly attached to the SpinServer via a Fibre Channel connection. NDMP control information is exchanged by normal TCP/IP conversations between the backup server (running the DMA software) and the SpinServer (running the NDMP server daemon). All data that is backed up is sent directly from the SpinServer to the tape device via its Fibre Channel connection without overloading the network with this potentially large amount of data. Local mode requires that a tape device be attached directly to the specific SpinServer that contains the VFSs that are backed up or restored.
- In remote mode, the tape device is directly attached to a third-party backup server. NDMP control information is exchanged between the backup server and the SpinServer, and all data that is backed up and restored travels across the TCP/IP connection between the backup server and the SpinServer. This enables tape devices to be used for all SpinServers and for any other hosts (Windows, UNIX, etc.) that are having data backed up or restored. It does, however, increase network traffic.
- In three-way mode, the tape device can be attached locally to a SpinServer via a Fibre Channel connection, and any other SpinServers (in the same cluster or different clusters) can have data backed up to and restored from the tape device on the first SpinServer. NDMP control information is exchanged between the backup server and all involved SpinServers. All data that is backed up and restored travels across the same TCP/IP connections that are used by the backup server's NDMP control information; data travels directly from the source SpinServer to the destination SpinServer, and then through the Fibre Channel connection to the tape device. This method also increases

network traffic.

The SpinServer supports the following:

- NDMP, Version 2
- NDMP, Version 3
- NDMP local mode
- NDMP remote mode
- NDMP three-way mode

For information about tape devices and software for backup and restoration that have been tested for use with the SpinServer, see the Network Appliance web site.

The following considerations must be taken into account when managing backup and restore operations for the SpinServer:

- The unit of backup is a virtual file system (VFS). One or more VFSs can be backed up, but individual files and directories cannot be backed up apart from their containing VFS.
- The unit of restoration is a file, directory, VFS, or multiples of these.
- For restoration, the destination VFS must exist prior to starting the restore operation.
- For backing up or restoration, the VFS name must be preceded by a slash. For example, a VFS named `vfs1` must be referenced as `/vfs1`, regardless of where it is mounted.
- Every VFS to be backed up must be listed separately, even if a VFS is mounted within a VFS that is being backed up. So, if `vfs1` is mounted at `/home/vfs1` and `vfs2` is mounted at `/home/vfs1/vfs2`, then backing up `vfs1` backs up all files and directories within `/home/vfs1`, but nothing in `/home/vfs1/vfs2`.
- By default, the SpinServer NDMP server is set up with clear-text authentication disabled and MD5 authentication enabled.

18.2 Backup and Restoration Tasks

Routine management tasks for NDMP services include the following:

- Configuring NDMP on a SpinServer
- Displaying tape devices on a SpinServer
- Configuring the SpinServer with backup and restoration products

Configuring NDMP

To use the NDMP service for the SpinServer, you must first configure it. The NDMP service user name and password that are set during configuration are used by any NDMP-compliant DMA used to control backups and restorations of a SpinServer.

Displaying the NDMP Service

To display information about the NDMP service, use the **services ndmp show** command. The following information is displayed: SpinServer ID, server enabled status, user name, user's password, clear text password enabled status, and IP address of the SpinServer's cluster interface.

To display the NDMP service, type the following command at the prompt.

```
   spinserver::> services ndmp show
SpinServ Server   Username   Password  Clear
   ID    Enabled                       Text
                                      Passwd

Server1  true    root       S3cr3t      false
```

Modifying the NDMP Service

Use the **services ndmp modify** command to modify the following configuration information for the NDMP service: server enabled status, NDMP name, NDMP password, clear text password enabled status, and IP address of the SpinServer's cluster interface. The command has the following syntax:

```
services ndmp modify -ssid spinserver -enable {true |
false} -username user_name -password user_password
-cleartext {true | false}
```

The following example modifies the NDMP service.

```
spinserver::> services ndmp modify -ssid server1 -enable \
true -username ndmp -password 0978 -cleartext false
```

where

server1 is the ID of the SpinServer.

true indicates that the NDMP service is enabled.

ndmp is the NDMP user name. (The NDMP user name is a name you choose.)

0978 is the password for NDMP. (The NDMP password is a password you choose.)

false indicates that the clear text password on the network is disabled.

Displaying Tape Devices

Use the **system hardware tape** command to display information about the tape drives and robots used by the SpinServer.

Tape Drives

The **`system hardware tape drives show`** command displays the NDMP device path of the tape drive, which is needed when configuring your particular backup product. To obtain more detailed information about a specific tape drive, use the **`-devid`** parameter with the device ID for that particular tape drive.

Tape Robots

The **`system hardware tape robots show`** command displays the NDMP device path of the tape robot, which is needed when configuring your particular backup product. To obtain more detailed information about a specific tape robot, use the **`-devid`** parameter with the device ID for that particular tape robot.

Configuring the SpinServer with Backup and Restoration Products

For information about configuring the SpinServer with your backup and restoration product (for instance, VERITAS or Legato), see the documentation for that product.

For information about tape devices for backups and restorations that have been tested and are recommended for use with the SpinServer, see the Network Appliance web site.

19.0 Managing Disaster Recovery

This chapter discusses disaster recovery scenarios for the SpinServer system. A disaster is a situation in which data from one source is temporarily or permanently lost and must be restored from a backup location. An example of a disaster is a power failure on a RAID enclosure; in this case, data that is stored on the enclosure cannot be retrieved until the power is restored. If the data is duplicated on mirrors or tape backups, it can be accessed while the primary RAID enclosure is unavailable.

19.1 Disaster Recovery Concepts

This section discusses the concepts that are associated with disaster recovery, including the mechanisms provided by the SpinServer system to assist with disaster recovery and considerations for designing a system that enables and simplifies disaster recovery.

Disaster Recovery Mechanisms

The SpinServer system provides the following mechanisms for backing up data at the VFS level:

- VFS SpinShots, which are point-in-time snapshots of VFSs that are automatically mounted on the source VFS. SpinShots contain only changed data and are therefore lightweight. SpinShots can be scheduled to occur at regular intervals. SpinShots are stored on the same storage pool as the source VFS; they are therefore inaccessible if the storage pool becomes unavailable. Creation of SpinShots is very fast (almost instantaneous). SpinShots are typically used to provide users with the ability to retrieve their own data in the event of an error. For instance, if a user accidentally deletes a needed file, he can retrieve it himself from a SpinShot, without needing to ask an administrator for help.
- VFS mirrors, which are full copies of VFSs on a different storage pool from the source VFS. The mirror uses the same amount of storage space as the source VFS. A mirror can be scheduled to be replicate (that is, to synchronize its data with the source VFS's data) at regular intervals. Initial creation of a mirror can take some time to complete, depending on the amount of data in the source VFS, but replications are generally very fast. In the event of a disaster, a mirror can be promoted to take over for its source VFS. See "Disaster Recovery Tasks" on page 237 for information.
- Tape backups of entire storage pools. SpinFS supports NDMP-compliant tape backup systems such as VERITAS and Legato. See "Managing Data Backup and Restoration" on page 231 for more information.

The following table lists the differences between the various components of a complete disaster recovery setup.

Table 19-1. Comparison of Backup Types

	SpinShots	Mirrors	Tape backup
Creation speed	**Fast** SpinShots share blocks with the source VFS.	**Medium** Initial creation involves copying all data in the VFS; only changed data is propagated on subsequent replications.	**Slow**
Retrieval speed of single file	**Fast** Users can retrieve their own data.	**Fast** Assumes mirror is mounted at a user-visible location; otherwise, an administrator must intervene.	**N/A** It is usually inefficient to restore a single file from tape.
Retrieval speed of entire VFS	**Medium** You must copy the SpinShot, then promote the copy to read-write status.	**Fast** You can remove the mirror from the group and convert it to read-write access.	**Medium** Dependent on speed of tape drive and amount of data.
Available if primary storage pool fails?	**No** SpinShots cannot be accessed if the primary VFS is offline.	**Yes** Data can be restored from mirror on other storage pool.	**Yes**

Note: The performance of SpinShots and mirrors is dependent on the performance and reliability of the underlying media. For instance, Fibre Channel-backed SCSI disk arrays have better performance and greater mean time between failure (MTBF) for disk drives than do ATA disk arrays.

Preparing for Disaster Recovery

To ensure that you can recover data in the event of a disaster, include the following elements in the logical setup of the SpinServer system. The following scenarios assume a SpinServer system with one virtual server, two storage pools, and a tape backup system. The virtual server includes VFSs that contain user and project data.

- The root VFS of the virtual server is located on one storage pool. The root VFS does not contain any data and is mirrored to the other storage pool. Because the root VFS

holds no data, the mirror uses a very small amount of storage space (a few KB). The root VFS of a virtual server can always be mirrored whether or not you have the optional SpinMirror software. See “Mirroring the Root of a Virtual Server” on page 151 for information about mirroring the root VFS of a virtual server.

- Each user VFS is scheduled to have SpinShots taken every five minutes. SpinShots are located on the same storage pool as the source VFS. If a user accidentally deletes or overwrites a file, she can retrieve it immediately from the VFS’s SpinShot. For information on creating regular SpinShots, see “Scheduling VFS Operations” on page 169.
- Each user VFS is mirrored to another storage pool, with mirror replications scheduled to occur every 15 minutes. If the primary storage pool fails, an administrator can recover the VFSs from the secondary storage pool. Mirroring of VFSs other than the root VFS of a virtual server requires the optional SpinMirror software.

  Each project VFS is also mirrored to another storage pool, with mirror replications scheduled to occur every five minutes. The read-only mirrors are made available to users who do not need to write to the project VFSs. This increases the access speed of project data to all users. Mirroring of VFSs other than the root VFS of a virtual server requires the optional SpinMirror software.

  For information on mirroring VFSs, see “Mirroring a Virtual File System” on page 166. For information on scheduling mirror replications to occur periodically; see “Scheduling VFS Operations” on page 169.
- A tape backup of all data located on the virtual server is made nightly. In the event of the catastrophic loss of data on any of the disk-based storage pools, data can be restored from the tape. However, because data restoration from tape is a long process, it is recommended only as a final recourse. For information on using tape backup and restoration, see “Managing Data Backup and Restoration” on page 231.

19.2 Disaster Recovery Tasks

This section provides commands and procedures that can be used to recover from a catastrophic data loss. Most disaster-recovery commands are provided in the **`tools`** command node. Because the **`tools`** commands can potentially disrupt client access to data or cause data loss if used incorrectly, exercise extreme care when using them.

Promoting a Read-Only VFS to Read-Write Parent

The **`tools filestorage vfs promote`** command promotes a mirror, a SpinShot, or a SpinShot of a mirror to the read-write parent of the VFS family. This command deletes the family’s current parent read-write VFS and any SpinShots that have a data version greater than the data version of the read-only VFS that is being promoted. The command has the following syntax:

```
tools filestorage vfs promote -vfsname VFS_name
```

The following example promotes a VFS mirror named **mirror.1.john** to the parent read-write VFS for the VFS family:

```
spinserver::> tools filestorage vfs promote -vfsname \
mirror.1.john
```

Removing a Mirror from a Group

The **tools filestorage mirror leavegroup** command removes a virtual file system (VFS) mirror from a mirror group. When a mirror is removed from the mirror group, no more manual or automatic updates from the original VFS can occur. The command has the following syntax:

```
tools filestorage mirror leavegroup -sourcevfsname
source_VFS_name -sinkvfsname sink_VFS_name -newvfsname
new_VFS_name
```

The following example removes a VFS mirror named **vfs1_m1**, which is a mirror of the VFS named **vfs1**, from the mirror group. The VFS is named **old.vfs1.mirror** after it leaves the mirror group.

```
spinserver::> tools filestorage mirror leavegroup \
-sourcevfsname vfs1 -sinkvfsname vfs1_m1 -newvfsname \
old.vfs1.mirror
```

Deleting a VFS's Mount Point

The **tools filestorage vfs deletemountpoint** command deletes a VFS's mount point. Use this command if forced deletions and unmountings of VFS leave stray mount points on the parent VFS. The command has the following syntax:

```
tools filestorage vfs deletemountpoint -vfsname VFS_name
-mountpath mount_path
```

The following example deletes the mount point of a VFS named **bill** that is mounted at **/user/bill**:

```
spinserver::> tools filestorage vfs deletemountpoint \
-vfsname bill -mountpath /user/bill
```

Forcibly Deleting a VFS

The **tools filestorage vfs forcedelete** command forces the deletion of a VFS. Use this command in the event that a VFS must be deleted when it is offline or otherwise unavailable. By default, the command attempts to unmount the VFS by deleting its mount point in its parent VFS. The VFS is then deleted whether or not the unmounting operation was successful.

The command has the following syntax:

```
tools filestorage vfs forcedelete -vfsname VFS_name
-unmount {true | false}
```

The **-unmount** parameter specifies whether or not to attempt to unmount the VFS before deleting it. The default is **true**.

The following example forces the deletion of a VFS named **vfs0**. The command first attempts to unmount the VFS.

```
spinserver::> tools filestorage vfs forcedelete -vfsname \
vfs0 -unmount true
```

Forcing the Unmounting of a VFS

The **tools filestorage vfs forceunmount** command forces the unmounting of a VFS. Use this command in the event that the parent VFS is offline or otherwise inaccessible and a VFS needs to be unmounted. The command attempts to unmount the VFS by deleting its mount point in the parent VFS. If that attempt fails, the command clears the VFS's mount attributes, thus enabling the VFS to be mounted elsewhere. If this situation occurs, use the **tools filestorage vfs deletemountpoint** command to delete the VFS's mount point in the parent VFS when the parent VFS becomes available again.

The command has the following syntax:

```
tools filestorage vfs forceunmount -vfsname VFS_name
```

The following example forces the unmounting of a VFS named **vfs0**:

```
spinserver::> tools filestorage vfs forceunmount \
-vfsname vfs0
```

Restoring a Lost VFS

Commands that forcibly change a VFS's status, such as **tools filestorage vfs promote**, **tools filestorage vfs forcedelete**, and **tools filestorage vfs forceunmount**, can potentially leave the VFS in a state in which it is known to the storage pool but not to the management framework. Such a VFS is called a lost VFS. Data on a lost VFS is not available for client access, and normal **filestorage** and **tools filestorage** commands do not work on the VFS. To restore lost VFSs, use the **tools filestorage lostnfound** commands.

Note: The **tools filestorage lostnfound** commands must be run on the SpinServer that owns the storage pool on which the lost VFS or VFSs are located. For instance, if you want to locate all lost VFSs in a cluster, you must log in individually to each of the cluster's SpinServers and run the **tools filestorage lostnfound show** command.

Displaying Information about a Lost VFS

The **`tools filestorage lostnfound show`** command displays the following information about lost VFSs:

- VFS ID
- Storage pool ID
- VFS name
- VFS type (`read-write`, `read-only`, or `spinshot`)
- Mountpoint
- Upstream VFS ID
- Downstream VFS ID
- VFS status (`online` or `offline`)

If the command is run with the ID and storage pool of a specific lost VFS, it displays the following additional information:

- Virtual server ID
- Creation time
- Amount of storage used, in MB
- Whether the VFS is the root VFS of its virtual server (`true` or `false`)

The command has the following syntax:

`tools filestorage lostnfound show` [**`-vfsid`** *`VFS_ID`* **`-spdevid`** *`storage_pool_ID`*]

The following example displays information about a lost VFS with ID **`879625`** on storage pool **`sp19`**:

```
spinserver::> tools filestorage lostnfound show -vfsid \
879625 -spdevid sp19
VFS ID              879625
Storage Pool        sp19
VFS                 lg15_vfs9.879625
Virtual Server ID   18
Type                spinshot
Create Time         07/20/2004 3:57 AM
Mounted From
Used (MB)           58
Upstream VFS ID     0
Downstream VFS ID   0
Root                false
Status              online
```

Claiming a Lost VFS

Claiming a lost VFS restores it to the management framework. To claim a lost VFS, use the **tools filestorage lostnfound claim** command. After you run the command on a lost VFS, the data it contains is available for client access, and you can run other commands in the **filestorage** and **tools filestorage** command nodes on the VFS.

The command has the following syntax:

```
tools filestorage lostnfound claim -vfsid VFS_ID -spdevid
storage_pool_ID
```

The following example restores a lost VFS with ID **879625** on storage pool **sp19** to the management framework:

```
spinserver::> tools filestorage lostnfound claim -vfsid \
879625 -spdevid sp19
```

Deleting a Lost VFS

You can permanently delete a lost VFS by using the **tools filestorage lostnfound delete** command. The command has the following syntax:

```
tools filestorage lostnfound delete -vfsid VFS_ID -spdevid
storage_pool_ID
```

The following example deletes a lost VFS with ID **788624** on storage pool **sp14**:

```
spinserver::> tools filestorage lostnfound delete -vfsid \
788624 -spdevid sp14
```

Making a VFS Read-Only

The **tools filestorage vfs makereadonly** command makes the specified virtual file system read-only. This command can be used to reset the read status of a VFS that has previously been made read-write (for instance, a copy of a mirror VFS that was set to be read-write after the initial copy). The command has the following syntax:

```
tools filestorage vfs makereadonly -vfsname VFS_name
```

The following example makes a read-write VFS named **vfs.wasRW** a read-only VFS:

```
spinserver::> tools filestorage vfs makereadonly \
-vfsname vfs.wasRW
```

Making a VFS Read-Write

The **tools filestorage vfs makereadwrite** command can be used to make certain virtual file systems read-write. VFSs that are by definition read-only, including mirrors and SpinShots, cannot be made read-write. However, full copies of read-only VFSs (for example, a copy of a mirror) can be made read-write. The command has no effect on read-write VFSs. The command has the following syntax:

tools filestorage vfs makereadwrite -vfsname *VFS_name*

The following example makes a read-only VFS named **vfs.wasRO** a read-write VFS:

```
spinserver::> tools filestorage vfs makereadwrite \
-vfsname vfs.wasRO
```

Making a VFS the Root of the Virtual Server

The **tools filestorage vfs makeroot** command makes a VFS the root VFS of the virtual server. The VFS is renamed as **root.***virtual_server_name*. For instance, if you run this command on a VFS named **user** that is located on a virtual server named **VS1**, the VFS is made the root VFS of the virtual server and is given the name **root.VS1**.

The command has the following syntax:

tools filestorage vfs makeroot -vfsname *VFS_name*

The following example makes a VFS named **dept** the root VFS of the virtual server:

```
spinserver::> tools filestorage vfs makeroot -vfsname dept
```

Renaming a VFS

The **tools filestorage vfs rename** command renames a read-write VFS. The read-write VFS cannot have any associated mirrors. When a VFS is renamed, be sure to unmount it and remount it so that its parent VFS refers to it by its new name. It is recommended that you also unmount and remount any VFSs that are mounted under the renamed VFS.

The **-unmount** parameter specifies whether to unmount the VFS before renaming it. The default is **true**. In the event that the parent VFS is offline or otherwise inaccessible when a VFS is renamed, specify a value of **false** for the **-unmount** option.

The command has the following syntax:

tools filestorage vfs rename -vfsname *VFS_name* **-newvfsname** *new_VFS_name* **-unmount** {**true** | **false**}

The following example renames a VFS named **old.vfs** as **new.vfs**. The VFS is unmounted before it is renamed.

```
spinserver::> tools filestorage vfs rename -vfsname \
old.vfs -newvfsname new.vfs -unmount true
```

Deleting a Specific File from a VFS

The **tools filestorage vfs deletefile** command deletes a specified file from a VFS. This is useful if the file is inaccessible from clients or if you do not wish the file to be accessed by clients (for example, if you suspect that the file contains a virus). The command has the following syntax:

```
tools filestorage vfs deletefile -vfsname VFS_name -path
absolute_path_name
```

The following example deletes a file named **/dept/eng/virus.exe** from a VFS named **dept.eng**:

```
spinserver::> tools filestorage vfs deletefile -vfsname
dept.eng -path /dept/eng/virus.exe
```

Deleting a VFS Family

The **tools filestorage vfs deletefamily** command deletes a read-write VFS and all of its SpinShots, mirrors, and SpinShots of mirrors. The command has the following syntax:

```
tools filestorage vfs deletefamily -vfsname VFS_name
```

The following example deletes a VFS named **vfs.parent** and deletes all of its associated SpinShots and mirrors:

```
spinserver::> tools filestorage vfs deletefamily \
-vfsname vfs.parent
```

Taking a VFS Offline

The **tools filestorage vfs setoffline** command takes a VFS offline; that is, makes it inaccessible to client access. Taking a VFS offline makes it easier to perform maintenance and disaster-recovery tasks. The command has the following syntax:

```
tools filestorage vfs setoffline -vfsname VFS_name
```

The following example takes a VFS named **dept.sales** offline:

```
spinserver::> tools filestorage vfs setoffline -vfsname \
dept.sales
```

Putting a VFS Online

After a VFS is taken offline by using the **tools filestorage vfs setoffline** command, use the **tools filestorage vfs setonline** command to put it online. Use this command after the necessary maintenance or disaster-recovery tasks have been performed on the VFS. The command has the following syntax:

```
tools filestorage vfs setonline -vfsname VFS_name
```

The following example puts a VFS named **dept.sales** online:

```
spinserver::> tools filestorage vfs setonline -vfsname \
dept.sales
```

Forcibly Deleting a Storage Pool

If the name of a storage pool persists on the SpinServer system after the storage pool is deleted, use the **tools storage pools forcedelete** command to forcibly delete it. The command has the following syntax:

```
tools storage pools forcedelete -deviceid device_ID
```

The following example forcibly deletes a storage pool with the device ID **sp0**:

```
spinserver::> tools storage pools forcedelete -deviceid \
sp0
```

Forcing a SpinServer to Take Over as Primary

To force a SpinServer to take over as primary for a storage pool in a high-availability (HA) configuration, use the **tools storage pools forcetakeover** command. This command works only if there is currently no primary for the specified storage pool. The syntax of the command follows.

```
tools storage pools forcetakeover -deviceid device_ID
-newprimary new_primary
```

The following example forces a SpinServer named **Server3** to take over as primary for a storage pool named **sp1**.

```
spinserver::> tools storage pools forcetakeover \
-deviceid sp1 -newprimary Server3
```

Forcing a Storage Pool Offline

The **tools storage pools forceoffline** command forcibly takes a storage pool offline. The command has the following syntax:

```
tools storage pools forcetakeover -deviceid device_ID
-newprimary new_primary
```

The following example forces a SpinServer named Server3 to take over as primary for a storage pool named sp1.

```
spinserver::> tools storage pools forcetakeover \
-deviceid sp1 -newprimary Server3
```

Deleting All Data from a Storage Unit

The **tools storage units erase** command to erase all data on a storage unit. This command is useful for deleting data from an evaluation system before redeploying it as a production system. The value of the **-zerofill** parameter specifies whether the storage unit is filled with zeroes; the default value is **false**. The command has the following syntax:

```
tools storage units erase -deviceid device_ID -zerofill
{true | false}
```

The following example deletes all data from a storage unit named su10. The storage unit is not filled with zeroes.

```
tools storage units erase -deviceid su10 -zerofill false
```

Recovering a Mirror Root VFS

To recover a root VFS that is a mirror of the root, perform the following steps:

1. Use the **tools filestorage mirror leavegroup** command to remove the mirror root from the mirror group. Specify a temporary name as the value of the **-newvfsname** parameter.
   ```
   spinserver::> tools filestorage mirror leavegroup \
   -sourcevfsname bad.root.vfs -sinkvfsname \
   good.mirror.root -newvfsname temp.vfs
   ```

2. Use the **tools filestorage vfs forcedelete** command to delete the problematic read-write root VFS:
   ```
   spinserver::> tools filestorage vfs forcedelete -vfsname \
   bad.root.vfs -unmount true
   ```

3. Use the **tools filestorage vfs makeroot** command to make the mirror root VFS the new read-write root VFS:
   ```
   spinserver::> tools filestorage vfs makeroot -vfsname \
   temp.vfs
   ```

4. Unmount all VFSs that were mounted below the deleted root VFS and remount them below the new root VFS. If necessary, use the **tools filestorage vfs forceunmount** command to unmount the subordinate VFSs.

Recovering a Nonmirror Root VFS

To recover a root VFS that is not a mirror of the root, perform the following steps:

1. Use the **tools filestorage vfs forcedelete** command to delete the problematic read-write root VFS:
   ```
   spinserver::> tools filestorage vfs forcedelete -vfsname \
   bad.root.vfs -unmount true
   ```

2. Use the **tools filestorage vfs makeroot** command to make the mirror root VFS the new read-write root VFS:
   ```
   spinserver::> tools filestorage vfs makeroot -vfsname \
   temp.vfs
   ```

3. Unmount all VFSs that were mounted below the deleted root VFS and remount them below the new root VFS. If necessary, use the **tools filestorage vfs forceunmount** command to unmount the subordinate VFSs.

Moving a Storage Pool to a Different SpinServer

To move the physical drives that comprise a storage pool from one SpinServer, perform the following steps. This procedure assumes that the physical drives comprise a single storage pool named **pool0** that is being moved from a SpinServer named **Server1** to a SpinServer named **Server2**, and that both SpinServers are in the same cluster.

1. If Server1 is still active, take the storage pool offline by using the **storage pools setoffline** command:
   ```
   spinserver::> storage pools setoffline -deviceid pool0
   ```

2. Run the **storage pools disconnect** command:
   ```
   spinserver::> storage pools disconnect -deviceid pool0
   ```

3. Physically disconnect the disk enclosure or enclosures from **Server1**.

4. Physically connect the disk enclosure or enclosures to **Server2**.

5. Run the **tools storage pools forceaddpassive** command:
   ```
   spinserver::> tools storage pools forceaddpassive \
   -deviceid pool0 -passivesec Server2 -priority 1
   ```

6. Run the **tools storage pools forcetakeover** command:
   ```
   spinserver::> tools storage pools forcetakeover \
   -deviceid pool0 -newprimary Server2
   ```

7. If desired, remove **Server1** as a secondary for the storage pool by using the **storage pools removesecondary** command:
   ```
   spinserver::> storage pools removesecondary \
   -deviceid pool0 -secondary Server1
   ```

Migrating a VFS to a Different Virtual Server

Virtual servers provide complete data isolation for all VFSs that it serves; a VFS on one virtual server is inaccessible from any other virtual server. If you need to move a VFS and all of its family VFSs (SpinShots and mirrors) from one virtual server to another, use the **tools filestorage vfs migratefamily** command. After the VFS is migrated, it cannot be accessed from its original virtual server, nor can it be used to access its original virtual server from its new virtual server.

The command has the following syntax:

```
tools filestorage vfs migratefamily -vfsname VFS_name
-vserver virtual_server_name -force {true | false}
```

where

-vfsname *VFS_name* specifies the parent VFS that is to be migrated to a different virtual server along with its family VFSs.

-vserver *virtual_server_name* specifies the virtual server to which the parent VFS and its family VFSs are to be migrated.

-force {**true** | **false**} specifies whether the migration continues if an error is encountered. The default is **false**.



Caution: Unless the VFS's original virtual server and its new virtual server are configured identically, with identical NIS domains, CIFS domains, and CIFS-to-NFS mappings, this command can result in extensive problems with permissions on the VFS's files, possibly causing loss of client access to data.

To prevent problems with permissions, NetApp recommends the following procedure for migrating a VFS:

1. Migrate the parent VFS. The following example migrates a VFS family whose parent VFS is named **dept.hr** to a virtual server named **vs2**. The migration is not forced.

   ```
   spinserver::> tools filestorage vfs migratefamily \
   -vfsname dept.hr -vserver vs2 -force false
   ```

2. Mount the migrated VFS in a location that is accessible only to administrators.

3. Specify the use of restricted ACLs on the VFS by running the **filestorage vfs modify** command with the value of the **-restrictedacl** parameter set to **true**.

4. Set the appropriate permissions on all files in the VFS by using **chmod**, **chown**, or other operating-system tools.

5. If CIFS ACLs are to be used on the VFS, re-enable the use of ACLs by running the **filestorage vfs modify** command with the value of the **-restrictedacl** parameter set to **false**.

6. Unmount the VFS from its safe location, then mount it for access to users.

Part IV: System-Management Tasks

This part discusses SpinServer system-management tasks. It includes the following chapters:

Chapter 20. Managing SpinServers

"Managing SpinServers" on page 247 provides information on basic system-management tasks for the SpinServer.

Chapter 21. Managing System Events

"Managing System Events" on page 265 provides information on monitoring SpinServer system events.

Chapter 22. Using Statistics

"Using Statistics" on page 281 provides information on monitoring activity on the SpinServer system by using statistics.

Chapter 23. Using Third-Party Management Utilities

"Using Third-Party Management Utilities" on page 301 provides information on setting up SNMP trap destinations.

20.0 Managing SpinServers

This chapter discusses concepts and tasks for managing the SpinServers in a cluster. The SpinServer provides management utilities for the following components:

- Machine information and identification
- Internal RAID sets
- SpinFS software
- Startup images
- File-system partitions
- Fibre Channel interfaces
- Health monitoring
- Time setting
- Firewall configuration of management and server ports

To control the amount of system resources that are being used, you can also manage long-running jobs, including the VFS copy, delete, move, and mirror operations.

20.1 System Management Concepts

The SpinServer carries identification information that you can set to uniquely identify it in your environment; for instance, you can assign a specific asset tag to it and list its physical location. You can also restart and shut down a SpinServer. Use **`spinservers`** commands to perform these operations.

The SpinServer includes two internal RAID sets that are used to hold and run the SpinFS software. Each RAID set is considered a separate file-system partition, and each can hold a separate software image; you can toggle between the images that are used to start the SpinServer. Additionally, you can upgrade the software on one or both RAID sets and manage the file-system partitions that they contain. Use the **`system`** commands to perform these operations.

Several VFS operations can take a considerable length of time to complete. These include the copy, move, and mirror operations. Long-running jobs consume a large amount of system resources. To control the use of system resources, you can manage these jobs by using the **`jobs`** command node.

20.2 System Management Tasks

Managing SpinServers includes the following tasks:

- Displaying information about a SpinServer
- Changing identification information for a SpinServer
- Restarting a SpinServer
- Shutting down a SpinServer
- Displaying information about a SpinServer's internal RAID sets
- Displaying information about a SpinServer's software
- Upgrading a SpinServer's software
- Managing a SpinServer's startup images
- Managing a SpinServer's internal file-system partitions
- Managing the Fibre Channel interface
- Managing long-running jobs
- Managing health monitoring
- Setting and maintaining the time
- Configuring the firewall settings for the management and server ports

Displaying Information about SpinServers

The **`spinservers show`** command displays information about a specific SpinServer (if the **`-ssid`** parameter is used) or about all SpinServers in the cluster (if no parameter is used). The following information about the SpinServer or SpinServers is displayed:

- SpinServer ID
- Owner
- Contact information
- Serial number
- Asset tag
- Location
- System uptime
- System time
- Time zone

The command has the following syntax:

```
spinservers show [-ssid spinserver]
```

The following example displays information about a SpinServer named **`Server1`**.

```
spinserver::> spinservers show -ssid Server1
SpinServer ID   3kmin10
Owner           acmebricks
Contact Info    it@acmebricks.com
Serial Number   2106-0005-01-33021234
Asset Tag       123-456-789
Location        Central lab
System Uptime   30 days 14 hr 23 min 05 sec
System Time     13:30:00
TimeZone        US/Eastern
```

Modifying a SpinServer

To modify the configuration information for SpinServer, use the **spinservers modify** command. The command has the following syntax:

spinservers modify -ssid *spinserver* [**-ownername** *owner_name*] [**-contact** *contact_information*] [**-assettag** *asset_tag*] [**-location** *location*] [**-tz** *time_zone*] [**-systime** *system_time*]

Important: Do not modify the SpinServer ID or host name. To do so invalidates the failover port designation for the SpinServer and can result in the loss of data.

You can modify the owner, contact information, asset tag, location listing, and time zone. These changes do not affect data storage on the system. You can set the time manually on a SpinServer by using the **-systime** parameter. The argument of the **-systime** parameter must be in the form *YYYY/MM/DD HH:MM:SS*, with all zeroes included.

The following command example modifies the contact information for the SpinServer named **Server1** so that the owner is listed as **acmebricks**, contact is listed as **engineering@acmebricks.com**, the asset tag is listed as **1234-ABCD**, and the location is listed as **3rd_floor_lab**. The time is set to midnight on 1 January 2004.

```
spinserver::> spinservers modify -ssid Server1 \
-ownername acmebricks -contact \
engineering@acmebricks.com -assettag 1234-ABCD -location \
3rd_floor_lab -systime 2004/01/01 00:00:00
```

Restarting a SpinServer

To restart a SpinServer, use the **spinservers reboot** command. The command has the following syntax:

spinservers reboot -ssid *spinserver* **-rebootreason** *reason_for_restart*

The SpinServer prompts you for confirmation after you issue this command. Before restarting the SpinServer, the command saves support data to assist with possible troubleshooting. The restart reason is written to the event log and is displayed in the subsequent startup message.

The following example restarts a SpinServer named **Server0** and gives a restart reason of **'spinfs upgrade'**.

```
spinserver::> spinservers reboot -ssid Server0 \
-rebootreason 'spinfs upgrade'
```

Shutting Down a SpinServer

To shut down a SpinServer, use the **spinservers shutdown** command. The command has the following syntax:

spinservers shutdown -ssid *spinserver* **-shutdownreason** *reason_for_shutdown*

The SpinServer prompts you for confirmation after you issue this command. Before shutting down the SpinServer, the command saves support data to assist with possible troubleshooting. The shutdown reason is written to the event log and is displayed in the subsequent startup message

The following command example shuts down a SpinServer named **Server0** with a reason of **'system maintenance'**.

```
spinserver::> spinservers reboot -ssid Server0 \
-shutdownreason 'system maintenance'
```

Displaying Information about a SpinServer's Internal RAID Sets

The **system raidsets show** command displays information about a SpinServer's internal RAID sets. The command has the following syntax:

system raidsets show [**-ssid** *spinserver* **-devid** *device_ID*]

Use the command without any parameters to display information about both RAID sets, or with the **-ssid** and **-devid** parameters to display information about a specified RAID set. The information for all RAID sets includes the following:

- SpinServer ID
- Device ID
- Status
- Whether the RAID set is read-only
- Whether Disk0 is operational or faulty
- Whether Disk1 is operational or faulty
- Resynchronization status
- Percentage of RAID set used

The display for a specific RAID set includes the following additional information:

- Description
- RAID level
- Number of blocks
- Number of disks
- Number of operating disks
- Disk0 name
- Disk1 name
- Whether a resynchronization is in progress

Note: This command applies only to the SpinServer's internal RAID sets. It does not apply to SpinStor RAID or JBOD disk enclosures that are externally connected to the SpinServer.

The following example displays information about an internal RAID set named **md1** on a SpinServer named **Server1**:

```
spinserver::> system raidsets show -ssid Server1 -devid \
md1
SpinServer ID       Server1
Device ID           md1
Description         Internal RAID Set md1
RAID Level          raid1
Status              active
Read Only           false
Blocks              1052160
Disks               2
Operating Disks     2
Disk0 Name          sda2
Disk0 Operational   true
Disk0 Faulty        false
Disk1 Name          sdb2
Disk1 Operational   true
Disk1 Faulty        false
```

```
Resync In Progress  false
Resync Status
```

Displaying System Software Information

Use the **system software show** command to display information about the software packages installed on the SpinServer system. It displays the following information:

- SpinServer ID
- Package name
- Version number
- Build number
- Installation time
- Summary description

The command has the following syntax:

system software show [**-ssid** *spinserver* **-name** *software_package*]

The following example displays information about all software packages:

```
spinserver::> system software show
SpinServer ID     Pkg Name          Version      Build       Install Time

Server1           spinvldb        2.0.0           32923    11/5/02 2:16 PM
Server1           spinsyslogd     2.0.0           32923    11/5/02 2:16 PM
Server1           modutils        2.4.2           5        3/5/02 10:38 AM
Server1           spinrdb         2.0.0           32923    11/5/02 2:16 PM
Server1           libstdc++       2.96            81       3/5/02 10:38 AM
Server1           spinsnmp        2.0.0           32923    11/5/02 2:16 PM
Server1           spinmgmt        2.0.0           32923    11/5/02 2:17 PM
Server1           spinfs          2.0.0           32923    11/5/02 2:16 PM
Server1           spinndmpd       2.0.0           32923    11/5/02 2:16 PM
Server1           spingcjlib      3.1.0           32923    11/5/02 2:16 PM
Server1           kernel          2.2.17spinos    32923    11/5/02 2:16 PM
Server1           spindiags       2.0.0           32923    11/5/02 2:16 PM
Server1           glibc           2.2.4           19.3     3/5/02 10:38 AM
```

Upgrading Software

To upgrade a SpinServer's base SpinFS software, use the **system rootfilesystem upgradesoftware** command. The command has the following syntax:

system rootfilesystem upgradesoftware -ssid *spinserver* **-name** *file_system_name* **-installurl** {**cdrom** | **file:***file_name* | **ftp:***FTP_site*}

See the *Release Notes* for the latest software release for complete instructions on upgrading to that release.

Forcing a Software Upgrade

To forcibly update a SpinServer's SpinFS software, use the **tools system rootfilesystem forceupgrade** command. The command has the following syntax:

```
tools system rootfilesystem forceupgrade -ssid spinserver
-name file_system_name -installurl {cdrom | file:file_name
| ftp:FTP_site}
```

Use this command only at the direction of support personnel.

Updating VFSs After an Upgrade from SpinFS 1.x to 2.x

As of SpinFS 2.x, `.readwrite` mountpoints are used in VFSs to enable certain SpinFS features. These mountpoints do not exist in SpinFS 1.x. If you upgrade a SpinServer from SpinFS 1.x to 2.x, you must create `.readwrite` mountpoints in each VFS by using the **tools filestorage vfs makereadwritemountpoints** command.

Use this command only with the assistance of support personnel.

The command has the following syntax:

```
tools filestorage vfs makereadwritemountpoints -vfsname
VFS_name
```

The following example creates `.readwrite` mountpoints in a VFS named **vfs1** on a SpinServer that was upgraded from SpinFS 1.x to 2.x:

```
spinserver::> tools filestorage vfs \
makereadwritemountpoints -vfsname vfs1
```

Managing Startup Images

Each SpinServer includes two startup images. Each image can hold a different software distribution. You can display information about the startup images and select which image is used to start the SpinServer. This is useful when upgrading software; for instance, you can upgrade the software on one startup image, then start up from that image to ensure that everything works correctly before applying the upgrade globally.

Displaying Information about Startup Images

To display information about startup images, use the **system bootimages show** command. The command has the following syntax:

```
system bootimages show [-ssid spinserver -label
boot_image]
```

The following example displays information about both startup images on a SpinServer:

```
spinserver::> system bootimages show
SpinServer ID    Label            Path            Device   Current Default

Server1         image1     /boot/vmlinuz          /dev/sda2  false   false
Server1         image2     /boot/vmlinuz          /dev/sda2  true    true
```

Setting the Default Startup Image

To set the default startup image, use the **system bootimages setdefault** command. The command has the following syntax:

system bootimages setdefault -ssid *spinserver* **-label** *boot_image*

The following example sets the default startup image to **image2** on a SpinServer named **Server1**:

```
spinserver::> system bootimages setdefault -ssid Server1 \
-label image2
```

Managing the Root File-System Partitions

Each SpinServer system has two root file-system partitions named ROOT and ALTROOT. The SpinFS software is installed on these partitions. The partitions typically remain synchronized, but they can become unsynchronized following a software upgrade or other major maintenance operation. Use the **system rootfilesystem** commands to work with root file-system partitions.

Displaying Information about Root File-System Partitions

Use the **system rootfilesystem show** command to display information about one or both root file-system partitions on a SpinServer. The command has the following syntax:

system rootfilesystem show [**-ssid** *spinserver* **-name** *file_system_name*]

The following command displays information about the **ROOT** root file-system partition on a SpinServer named **Server1**:

```
spinserver::> system rootfilesystem show -ssid Server1 \
-name ROOT
SpinServer ID           Server1
Name                    ROOT
Device                  /dev/md2
Mount Path              /
Version                 2.0.0
BuildNumber             33414
InstallTime             11/14/2002-00:43:37
Last Sync Time          UNKNOWN
Last Config Sync Time   UNKNOWN
Sync On Reboot          NONE
```

Synchronizing Root File-System Partitions at the Next System Restart

Use the **system rootfilesystem modify** command to set the synchronization value for a SpinServer's root file system at the next system restart. Possible values for synchronization include **NONE**, for no synchronization; **CONFIGURATION**, to synchronize the configuration of the two root file systems; and **FILESYSTEM**, to synchronize the two root file systems. After a system restart, this setting is automatically reset to **NONE**.

The command has the following syntax:

```
system rootfilesystem modify -ssid spinserver -name
file_system_name -synconreboot {NONE | CONFIGURATION |
FILESYSTEM}
```

The following example modifies the root file-system partition named **ROOT** on a SpinServer named **Server1** so that it synchronizes the configuration of the root file system the next time the SpinServer is restarted:

```
spinserver::> system rootfilesystem modify -ssid Server1 \
-name ROOT -synconreboot CONFIGURATION
```

Managing Fibre Channel Interfaces

You can use **system hardware fibrechannel** commands to work with the SpinServer's Fibre Channel interfaces, which connect the SpinServer to FC-AL-based RAID arrays.

Displaying Information about Fibre Channel Interfaces

Use the **system hardware fibrechannel show** command to display information about one or more Fibre Channel channels. Information includes the following:

- SpinServer ID
- Channel number
- Whether or not the channel is present
- Port type (`QL2200` or `QL2312`)
- Topology (`N_Port`, `NL_Port`, `F_Port`, or `FL_Port`) (specific port only)
- Node name
- Port name
- State of the channel (`up` or `down`)
- Current data rate
- Desired channel mode (`LoopOnly`, `LoopPreferred`, `PointToPointOnly`, or `PointToPointPreferred`) (specific port only)

The command has the following syntax:

```
system hardware fibrechannel show [-ssid spinserver
-channel channel_number]
```

The following example displays information about Fibre Channel **0** on the SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel show -ssid \
Server1 -channel 0
SpinServer ID          Server1
Channel                0
Present                true
Port Type              QL2200
Topology               NL_Port
Node Name              200100e08b25fb27
Port Name              200100e08b25fb27
State                  Up
Current Data Rate      1GB
Desired Channel Mode   LoopPreferred
```

Modifying the Data Rate for a Fibre Channel Interface

Use the **system hardware fibrechannel modify** command to change the data rate for a Fibre Channel channel. The command has the following syntax:

```
system hardware fibrechannel modify -ssid spinserver
-channel channel_number -desiredrate data_rate
```

The following example changes the data rate to **AUTO** (automatic detection) for Fibre Channel **0** on the SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel modify -ssid \
Server1 -channel 0 -desiredrate AUTO
```

Setting the Mode for a Fibre Channel Interface

Use the **system hardware fibrechannel setmode** command to set the mode for a Fibre Channel interface. The command has the following syntax:

```
system hardware fibrechannel setmode -ssid spinserver
-channel channel_number -mode mode
```

Possible values for the value of the **-mode** parameter include **LoopOnly**, **LoopPreferred**, **PointToPointOnly**, and **PointToPointPreferred**. The default is **LoopPreferred**.

The following example sets the mode to **LoopOnly** for Fibre Channel **0** on the SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel setmode -ssid \
Server1 -channel 0 -mode LoopOnly
```

Disabling a Fibre Channel Interface

Use the **system hardware fibrechannel takedown** command to disable a Fibre Channel interface. This is useful for maintenance or for diagnosing problems. The command has the following syntax:

```
system hardware fibrechannel takedown -ssid spinserver
-channel channel_number
```

The following example disables Fibre Channel **2** on a SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel takedown \
-ssid Server1 -channel 2
```

Reenabling a Fibre Channel Interface

If a Fibre Channel interface has been disabled, use the **system hardware fibrechannel bringup** command to reenable it. The command has the following syntax:

```
system hardware fibrechannel bringup -ssid spinserver
-channel channel_number
```

The following example reenables Fibre Channel **2** on a SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel bringup -ssid \
Server1 -channel 2
```

Resetting a Fibre Channel Card

To reset a Fibre Channel card and all of its interfaces, use the **system hardware fibrechannel cardreset** command. The command has the following syntax:

```
system hardware fibrechannel cardreset -ssid spinserver
-channel channel_number
```

The following example resets the Fibre Channel card with channel **0** on a SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel cardreset \
-ssid Server1 -channel 0
```

Issuing a LIP Reset to a Fibre Channel Interface

Use the **system hardware fibrechannel lipreset** command to issue a loop initialization primitive (LIP) reset on the Fibre Channel. This is useful for Fibre Channel discovery and for verifying connections between the SpinServer and third-party storage products. The command has the following syntax:

```
system hardware fibrechannel lipreset -ssid spinserver
-channel channel_number
```

The following example issues a LIP reset to Fibre Channel **0** on the SpinServer named **Server1**:

```
spinserver::> system hardware fibrechannel lipreset \
-ssid Server1 -channel 0
```

Managing Long-Running Jobs

You can use the **jobs** command node to manage long-running VFS operations, including copy, delete, move, and mirror. This is useful if a particular job is consuming too many system resources and you want to stop it or pause it until there is less demand on the system.

Displaying Long-Running Jobs

Use the **jobs show** command to display information about one or more long-running jobs. The syntax of the command follows:

```
jobs show [-id job_ID]
```

The following example displays information about all current long-running jobs:

```
spinserver::> jobs show
Job ID     Job Name                     SpinServer ID   Job State   Job Progress
148        delete virtual server: vs1   Server1         finished    deleting
```

Pausing a Long-Running Job

Use the **jobs pause** command to suspend a long-running job. A paused job can be restarted by using the **jobs resume** command. The **jobs pause** command has the following syntax:

```
jobs pause -id job_ID
```

The following example pauses the job with ID **148**:

```
spinserver::> jobs pause -id 148
```

Resuming a Long-Running Job

Use the **jobs resume** command to resume a long-running job that was suspended by using the **jobs pause** command. The **jobs resume** command has the following syntax:

```
jobs resume -id job_ID
```

The following example causes the job with ID **148** to resume:

```
spinserver::> jobs resume -id 148
```

Stopping a Long-Running Job

Use the **jobs kill** command to stop a long-running job. A job that is stopped with this command cannot be restarted. The command has the following syntax:

```
jobs kill -id job_ID
```

The following example stops the job with ID **148**:

```
spinserver::> jobs kill -id 148
```

Managing Health Monitoring

You can set up automatic health monitoring of the SpinServer cluster by using the **services support configure** commands. With automatic health monitoring, periodic email messages about the SpinServer's condition are sent to a specified address.

Modifying the Health-Monitoring Configuration

Use the **services support configure modify** command to modify the health-monitoring configuration for a SpinServer. The command has the following syntax:

```
services support configure modify -enabled {true | false}
-reportperiod {daily | weekly | monthly } -supportemail
email_address
```

Use the **cluster configuration modify** command to specify the mail-from address and outbound mail server for a SpinServer cluster. For more information, see "Modifying the Cluster Configuration" on page 90.

The following example enables health monitoring on a **weekly** basis and emails the report to **support@spinnakernet.com**.

```
spinserver::> services support configure modify -enabled \
true -reportperiod weekly -supportemail \
support@spinnakernet.com
```

Displaying Information about the Health-Monitoring Configuration

The **`services support configure show`** command displays the following information about the health-monitoring configuration of a SpinServer:

- Whether automatic health monitoring is enabled (`true` or `false`)
- The period for health-monitoring reports (`daily`, `weekly`, or `monthly`)
- The email address to which health reports are sent
- The revision number of the current configuration

The following example displays the health-monitoring configuration of a SpinServer:

```
spinserver::> services support configure show
Enabled                  true
Health Report Period     daily
Support Email Address    support@spinnakernet.com
Configuration Revision   1
```

Setting and Maintaining the Time

An accurate time setting is critical for the proper operation of the SpinServer. It is recommended that you use a Network Time Protocol (NTP) server to keep all SpinServers in a cluster set to the correct time.

Manually Setting the Time

To set the time manually on a SpinServer, use the **`-systime`** parameter of the **`spinservers modify`** command, as described in “Modifying a SpinServer” on page 249. The argument of the **`-systime`** parameter must be in the form *YYYY/MM/DD HH:MM:SS*, with all zeroes included. For instance, to set the system time on **`Server1`** to 1:30 p.m. on 1 June 2003, use the following command:

```
spinserver::> spinservers modify -ssid Server1 -systime \
2003/06/01 13:30:00
```

Configuring a SpinServer Cluster to Use NTP

To configure a SpinServer to use NTP, use the **`services ntp`** commands to create a reference to an NTP server and then to modify the SpinServer cluster to use the NTP server.

Creating a Reference to an NTP Server

Use the **`services ntp servers new`** command to create a reference to an NTP server. The command has the following syntax:

```
services ntp servers new -ipaddress IP_address
```

The following example creates a reference to an NTP server with the IP address **10.1.4.1**:

```
spinserver::> services ntp servers new -ipaddress 10.1.4.1
```

Displaying Information about an NTP Server

Use the **services ntp servers show** command to display information about an NTP server or servers. The command has the following syntax:

services ntp servers show [**-ipaddress** *IP_address*]

The following example displays information about the NTP servers that are configured to work with a SpinServer:

```
spinserver::> services ntp servers show
 IP Address
 10.1.4.1
 10.2.4.1
```

Deleting a Reference to an NTP Server

Use the **services ntp servers delete** command to delete a reference to an NTP server. The command has the following syntax:

services ntp servers delete -ipaddress *IP_address*

The following example deletes the reference to an NTP server with the IP address **10.1.4.1**:

```
spinserver::> services ntp servers delete -ipaddress \
10.1.4.1
```

Modifying the NTP Configuration for a Cluster

Use the **services ntp configure modify** command to modify the NTP configuration for a SpinServer cluster. The command has the following syntax:

services ntp configure modify -clustername *cluster_name* **-enabled** {**true** | **false**} **-usepdcs** {**true** | **false**} **-useclus** {**true** | **false**}

where

-clustername *cluster_name* specifies the name of the cluster.

-enabled {**true** | **false**} specifies whether or not to use NTP in the cluster. The default is **true**.

-usepdcs {**true** | **false**} specifies whether or not to use the time setting of the Windows primary domain controller (PDC). The default is **false**.

-useclus {**true** | **false**} specifies whether or not to use SpinServers in the cluster as NTP peers. The default is **false**.

The following example enables the use of NTP in a cluster named **Clus1**. PDC time is not used, and SpinServers in the cluster are not used as peers.

```
spinserver::> services ntp configure modify -clustername \
Clus1 -enabled true -usepdcs false -useclus false
```

Displaying the NTP Configuration for a Cluster

Use the **services ntp configure show** command to display the NTP configuration for a SpinServer cluster. The command displays the following information:

- Cluster name
- Whether or not NTP is enabled for the cluster
- Whether or not Windows primary domain controller (PDC) time is used for the cluster
- Configuration revision number

The command has the following syntax:

```
services ntp configure show [-clustername cluster_name]
```

The following example displays information about the use of NTP in a cluster named **Clus1**.

```
spinserver::> services ntp configure show -clustername \
Clus1
Cluster Name             Clus1
Enabled                  true
Use PDC Time             false
Configuration Revision   0
```

Setting the Firewall Configuration on Ports

Use the **services firewall configure** commands to manage access to the SpinServer's management and server ports.

Modifying the Firewall Configuration

Use the **services firewall configuration modify** command to modify the firewall configuration on the management and server ports. You can specify whether management is available on the server ports (by default, **false**), whether file-serving services are available on the server ports (by default, **true**), and whether file-serving services are available on the management port (by default, **true**). The command has the following syntax:

```
services firewall configuration modify -mgmtonservports
{true | false} -svcsonservports {true | false}
-svcsonmgmtport {true | false}
```

The following command modifies the configuration such that management is not accessible on server ports, services are accessible on server ports, and services are not available on the management port:

```
spinserver::> services firewall configuration modify
-mgmtonservports false -svcsonservports true
-svcsonmgmtport false
```

Displaying Information about Firewall Configuration

The **services firewall configure show** command displays access information for the SpinServer's management and server ports. The command displays the following information:

- Whether management access is enabled on server ports (`true` or `false`)
- Whether service access is enabled on server ports (`true` or `false`)
- Whether service access is enabled on the management port (`true` or `false`)
- The revision number of the current configuration

The following example displays access configuration information for the SpinServer's ports:

```
spinserver::> services firewall configure show
Mgmt On Server Ports       false
Services On Server Ports   true
Services On Mgmt Port      true
Configuration Revision     1
```

21.0 Managing System Events

This chapter discusses monitoring a SpinServer system by using event monitoring and notification, which are provided by the **`event`** command node. It also discusses monitoring the commands that are issued on a SpinServer cluster.

See the *SpinFS 2.5 Messages and Troubleshooting Guide* for a complete list of system messages and troubleshooting steps.

21.1 Event Management Concepts

SpinServer events provide information about all operations that occur on a SpinServer system. The following sections discuss the ways in which SpinServer events are organized.

Monitoring Services

The following monitoring services and their features are supported:

- **Syslog Messages:** The system log (syslog) service receives event messages from the SpinServer kernel and user applications and directs them to a log file.
- **Alarms:** Sensors are defined for various system elements. The values for these sensors are checked periodically. If a value exceeds a preset threshold, a warning is generated.
- **Notifications:** You can configure user subscriptions for receiving event messages. Subscriptions define the messages in which users are interested and the means by which they are notified. You can configure the frequency with which users are notified of an event.
- **Logging:** A log store logs all event messages. The log messages can be displayed or monitored live. You can also use the SpinLogReader utility to view the logs.

Event Information

All events include the following information:

- **Module:** The source of the event, such as Syslog, RAID, SpinAccess, etc. See the *SpinFS 2.5 Messages and Troubleshooting Guide* for a listing of all modules.
- **Name:** The name of the event.
- **Severity:** The level of severity of the event.

Event Filters

Events can be filtered by IDs, modules, and severities. Filters include the following:

- **Particular Event Filter:** The Particular Event Filter filters messages by message ID
- **Group Event Filter:** The Group Event Filter filters messages by modules and severity

Levels of Severity

The levels of severity are defined as follows:

- **EMERGENCY:** The SpinServer system is unusable.
- **ALERT:** Action must be taken immediately.
- **CRITICAL:** A critical condition has occurred.
- **ERROR:** An error condition has occurred.
- **WARNING:** A warning condition has occurred.
- **NOTICE:** A normal but significant condition has occurred.
- **INFORMATIONAL:** Information is generated.
- **DEBUG:** A debug-level message is generated.

Audit Log

The SpinServer audit log lists all of the commands that are issued on a SpinServer cluster. This can be useful for troubleshooting (for instance, whether a command was run incorrectly) and for tracking down possible security problems.

21.2 Event Management Tasks

Routine management tasks for monitoring the operational performance of a SpinServer system's hardware and software functions include the following:

- Displaying the current state of system alarms
- Creating and modifying event filters
- Configuring event-notification handlers and subscriptions
- Monitoring syslog messages

Displaying Event Alarms

The **event alarm show** command displays the current states of alarm variables. Used without the parameter, the following information is displayed: index, instance, variable, status, and current value. Use the **-index** parameter to display information about a specific alarm. Possible status values are as follows:

- OK (The variable is in the normal range.)
- Alarm (The variable is in a state that indicates a problem.)
- ---- (The alarm sensor is not sensing.)

The following example displays the current states of all alarm variables.

```
    spinserver::> event alarm show
Index   Module          Instance             Variable     Status   Current
                                                                    Value

0       alarm    CPU 1 Fan                 sensorValue    OK       4166
1       alarm    CPU 2 Fan                 sensorValue    Alarm    2647
2       alarm    Chassis Intake Fan 1      sensorValue    Alarm    0
3       alarm    Chassis Intake Fan 2      sensorValue    Alarm    0
4       alarm    Chassis Intake Fan 3      sensorValue    Alarm    0
5       alarm    Chassis Intake Fan 4      sensorValue    Alarm    0
6       alarm    Chassis Exhaust Fan 1     sensorValue    Alarm    0
7       alarm    Motherboard Temperature 1 sensorValue    OK       31
8       alarm    Motherboard Temperature 1 sensorValue    Alarm    31
9       alarm    CPU 1 Temperature         sensorValue    Alarm    16
10      alarm    CPU 1 Temperature         sensorValue    OK       16
11      alarm    Motherboard Temperature 2 sensorValue    OK       34
12      alarm    Motherboard Temperature 2 sensorValue    Alarm    34
13      alarm    CPU 2 Temperature         sensorValue    Alarm    164
14      alarm    CPU 2 Temperature         sensorValue    Alarm    164
15      alarm    Power supply monitor      failed         Alarm    false
16      alarm    CPU Core Voltage          sensorValue    Alarm    0.0
17      alarm    3.3 Volt Supply           sensorValue    Alarm    3.268
18      alarm    3.3 Volt Supply           sensorValue    Alarm    3.268
19      alarm    5.0 Volt Supply           sensorValue    Alarm    4.966
20      alarm    5.0 Volt Supply           sensorValue    Alarm    4.966
21      alarm    12.0 Volt Supply          sensorValue    Alarm    11.8125
22      alarm    12.0 Volt Supply          sensorValue    Alarm    11.8125
```

Managing Spin Events

Use the **event spinevents** commands to create a filter for a particular event and to display configuration information for Spin events.

Spin events include notifications of hardware malfunctions such as low backup battery power, mismatched RAID controllers' firmware, failed RAID set initialization, a disabled UPS, and a failed fan. Spin events also include notifications of incorrect configuration information such as NetBIOS name, server name, password, or virtual server masks that are wrong or do not meet naming conventions.

Creating a Spin Events Filter

The **event spinevents createfilter** command specifies an event ID and filter name for a new Spin event filter. The filter is added to the filter table. The syntax of the command follows:

```
event spinevents createfilter -msgid event_ID -filtername
filter_name
```

The following example creates a filter named **raid_criticals** with event ID **82**:

```
spinserver::> event spinevents createfilter -msgid 82 \
-filtername raid_criticals
```

Adding Spin Events to a Filter

To add an event to an existing Spin event filter, use the **event spinevents addtofilter** command. The command has the following syntax:

```
event spinevents addtofilter -msgid event_ID -filtername
filter_name
```

The following example adds Spin event **655480** to a filter named **raid_criticals**:

```
spinserver::> event spinevents addtofilter -msgid 655480 \
-filtername raid_criticals
```

Sending a Test Spin Event

After setting up Spin event filters, you can test them by using the **event spinevents sendtestevent** command to send test events. If you issue the command with the ID of a single event, the command sends a single message for that event. If you issue the command with the ID of an event group, the command sends a message for each event in the group. You can obtain event IDs by using the **event spinevents show** and **event spinevents show-group** commands, as described in the following section.

The command has the following syntax:

```
event spinevents sendtestevent -msgid event_ID
```

The following example sends out test messages for each event in event ID **82**:

```
spinserver::> event spinevents sendtestevent -msgid 82
```

Displaying Spin Events

The **event spinevents show** command displays the following information for Spin events: message ID, module name, event name, event severity, and the number of events for event groups. To display information for a particular event, specify the **-msgid** parameter. The syntax of the command follows:

event spinevents show [**-msgid** *event_ID*]

The following example displays information about Spin event **82**.

```
spinserver::> event spinevents show -msgid 82
Message ID   82
Module       raid
Name         critical_events
Severity     critical
# Events     25
```

To display all of the Spin events that make up an event group, use the **event spinevents show-group** command. The command has the following syntax:

event spinevents show-group [**-msgid** *event_ID*]

The following example displays the events that make up Spin event **82**:

```
spinserver::> event spinevents show-group -msgid 82
Message     Module                      Name                      Severity
   ID

655372    raid          DISK DEAD                               critical
655385    raid          DISK SOFT ID                            critical
655395    raid          LOOPID CONFLICT                         critical
655401    raid          RAIDSET CHECK DISK FAILED               critical
655402    raid          RAIDSET OFFLINE                         critical
655409    raid          RAIDSET BADBLOCK                        critical
655420    raid          UPS FAILED                              critical
655422    raid          FAN FAILED                              critical
655425    raid          POWER FAILED                            critical
655428    raid          ENCHEAT BAD                             critical
655434    raid          ENC ACCESS OFFLINE                      critical
655435    raid          ENC SOFTADDR OCCURRED                   critical
655440    raid          ENCLOSURE SHUTDOWN                      critical
655442    raid          ENCLCOLD FAIL                           critical
655452    raid          INSTALLATION ABORT                      critical
655453    raid          CTL FIRMWARE MISMATCH                   critical
655455    raid          WARM BOOT ERR                           critical
655479    raid          CTL MIRROR TABLE ERR                    critical
655480    raid          CTL STOP EJECTED                        critical
655488    raid          CHN SCSI BUS DEAD                       critical
655490    raid          CHN FIBRE DEAD                          critical
655506    raid          CTL WRONG FIRMWARE                      critical
655510    raid          CTL RESET FAILED                        critical
655511    raid          CTL RESET ATTEMPT FAILED                critical
655512    raid          RS NEW RESET FAILED                     critical
```

Viewing Log Records

The **event log show** command displays the log records for a SpinServer. The following information is displayed: event ID, the time it occurred, the severity of the event, the module, last time the event occurred, and a descriptive message about the event. To view the log records for a specific event on a SpinServer, use the **-eventid** parameter. The syntax of the command follows:

event log show [**-eventid** *event_ID*]

The following example displays information about the event that has the ID **123**.

```
spinserver::> event log show -eventid 123
```

Managing Event Filters

Event filters enable you to specify the warning messages that you want to receive and the information that you want to be sent to a remote SpinServer. Use the **event filters** commands to create, display, and modify event filters.

Creating Event Filters

Use the **event filters new** command to create a new event filter by designating the name and message IDs for the filter.

Note: To create a new filter for a particular event, use the **event spinevents createfilter** command.

The command has the following syntax:

```
event filters new -name filter_name -msgids message_IDs
```

To obtain a full list of message IDs, use the **event spinevents show** command.

The following example creates a new filter named **sev1** that initially includes event ID **82**:

```
spinserver::> event filters new -name sev1 -msgids 82
```

Adding Events to Event Filters

After you have created a filter, you can add events to it by using the **event filters addgrpevents** command. The command has the following syntax:

```
event filters addgrpevents -name filter_name -modules
[list_of_modules] -severity {debug | informational |
notice | warning | error | critical | alert | emergency}
```

The following examples add **alert** and **emergency** events from the **dblade** and **raid** modules to the filter named **sev1**:

```
spinserver::> event filters addgrpevents -name sev1 \
-modules [dblade,raid] -severity emergency

spinserver::> event filters addgrpevents -name sev1 \
-modules [dblade,raid] -severity alert
```

Modifying Event Filters

The **event filters modify** command modifies the message IDs for a specified filter. Whereas the **event filters addgrpevents** enables you to add messages to a filter incrementally, the **event filters modify** command can add messages to the message list or remove messages from the list, but the complete message list must be specified in the command. For instance, if you have a message list that contains messages **34**, **42**, and **58**, you can use the **event filters addgrpevents** command to add message **66**. If you then want to remove message **42** from the list, you can use the **event filters modify** command to specify a message list of **34**, **58**, and **66**.

The command has the following syntax:

```
event filters modify -name filter_name -msgids message_IDs
```

The following example modifies the **sev1** filter so that it includes messages **34**, **42**, **58**, **66**, **74**, **82**, and **90**.

```
spinserver::> event filters modify -name sev1 -msgids \
34,42,58,66,74,82,90
```

Displaying Event Filters

The **event filters show** command displays the filter name and message IDs for one or more event filters. To display information for a specific event filter, specify the **-name** parameter. The syntax of the command follows:

```
event filters show [-name event]
```

The following example displays information about a filter named **sev1**:

```
spinserver::> event filters show -name sev1
Name               Criticals
Spin Message IDs   34, 42, 58, 66, 74, 82, 90, 98, 106, 114, 122,
130, 138, 146, 154, 162, 170, 178, 186, 194, 202, 218, 234, 242,
250, 258, 266, 274, 282, 290, 306, 322, 330
```

The **event filters showeventnames** command displays the filter name, message names, and message IDs for one or more event filters. The command has the following syntax:

```
event filters showeventnames [-name event]
```

The following example displays event-name information about a filter named **sev1**:

```
spinserver::> event filters showeventnames -name sev1
Action "Show Message Names" returned with the following status:
dblade:critical_events(34)
bip:critical_events(42)
dimm:critical_events(58)
vldb:critical_events(66)
cli:critical_events(74)
```

```
raid:critical_events(82)
alarm:critical_events(90)
streamdaemons:critical_events(98)
software:critical_events(106)
vfs:critical_events(114)
spinaccess:critical_events(122)
audit:critical_events(130)
logmsg:critical_events(138)
syslog:critical_events(146)
kernellog:critical_events(154)
jobcontrol:critical_events(162)
rootfilesystem:critical_events(170)
bootimage:critical_events(178)
fcddriver:critical_events(186)
hamgr:critical_events(194)
ivs:critical_events(202)
dbladeincrcopy:critical_events(218)
spinfsClient:critical_events(234)
rdberrormsg:critical_events(242)
vldbClient:critical_events(250)
vodClient:critical_events(258)
net:critical_events(266)
lp:critical_events(274)
spiff:critical_events(282)
ndmp:critical_events(290)
rpc:critical_events(306)
antivirus:critical_events(322)
backup:critical_events(330)
```

Deleting Event Filters

The **`event filters delete`** command deletes an event filter. The command has the following syntax:

```
event filters delete -name filter_name
```

The following example deletes the **`sev1`** filter.

```
spinserver::> event filters delete -name sev1
```

Managing Event Notifications

To manage event handlers, to view the history of transmitted notifications, and to register events to be handled, use the **`events notification`** commands.

There are three types of event handlers: email, snmptrap, and user. To maintain the life cycles of event handlers, you can create, display, modify, and delete handlers.

Creating Event-Notification Handlers

The **event notification handlers new** command creates a handler of event notifications. For email, the mail server address, followed by a list of email addresses, must be used as the argument of the **-parameters** parameter. The command has the following syntax:

```
event notification handlers new -name name -type {email |
snmptrap | user} -parameters parameters
```

The following example creates an event-notification handler named **Server1** of type **email**. The handler sends event notifications to the email addresses **pat@acmebricks.com** and **chris@acmebricks.com** on the mail server **pop1.acmebricks.com**.

```
spinserver::> event notification handlers new -name \
Server1 -type email -parameters pop1.acmebricks.com, \
pat@acmebricks.com, chris@acmebricks.com
```

Displaying Event-Notification Handlers

The **event notification handlers show** command displays configuration information for a list of handlers of event notifications. The following information about event handlers is displayed: name of handler, type of handler (`email` , `snmptrap`, or `user`), and the parameters for the handler. To view the configuration information for a specific event handler, use the **-name** parameter. The command has the following syntax:

```
event notification handlers show [-name name]
```

The following example displays configuration information for an event-notification handler named **Server1**.

```
spinserver::> event notification handlers show -name \
Server1
```

Modifying Event-Notification Handlers

The **event notification handlers modify** command modifies the configuration of a handler of event notifications. For email, the mail server address, followed by a list of email addresses, must be used as the argument of the **-parameters** parameter. The command has the following syntax:

```
event notification handlers modify -name name -parameters
parameters
```

The following example modifies an event-notification handler named **Server1** so that it sends email notification to the **pat@acmebricks.com** address on the **pop1.acmebricks.com** mail server.

```
spinserver::> event notification handlers modify -name \
Server1 -parameters pop1.acmebricks.com, \
pat@acmebricks.com
```

Testing Event-Notification Handlers

The **event notification handlers test** command tests an event-notification handler by sending a test message through the handler. The command has the following syntax:

```
event notification handlers test -name handler_name
```

The following example tests an event-notification handler named **Server1**:

```
spinserver::> event notification handlers test -name \
Server1
```

Deleting Event-Notification Handlers

The **event notification handlers delete** command deletes a handler of event notifications. The command has the following syntax:

```
event notification handlers delete -name name
```

The following example deletes an event-notification handler named **Server1**.

```
spinserver::> event notification handlers delete -name \
Server1
```

Viewing Event-Notification History

The **event notification history show** command displays the history of event notifications for a specific host. The following information is displayed:

- Host name
- Event ID
- The last time the notification was transmitted
- The severity of the event
- The notification message that was transmitted
- The handler that transmitted the notification
- The last time the event occurred

The command has the following syntax:

```
event notification history show [-host spinserver -eventid
event_ID]
```

To view the log records for a specific event on a SpinServer, use the **-host** and **-eventid** parameters.

The following example displays the history of event notifications for the event **123** on the host named **Server10**.

```
spinserver::> event notification history show -host \
Server10 -eventid 123
```

Managing Event-Notification Subscriptions

To register events to be handled, use the **event notification subscriptions** commands. To manage subscriptions to event notifications, you can create, display, modify, and delete subscriptions.

Creating Event-Notification Subscriptions

The **event notification subscriptions new** command specifies the filters, handlers, threshold values, and threshold time periods (in minutes) for a new subscription to event notifications. The command has the following syntax:

```
event notification subscriptions new -filter filter_name
-handler handler_name -thresholdval threshold_value
-thresholdperiod threshold_period
```

The following example creates a new events-notification subscription that consists of the filter named **all_crits** and the handler named **email_admin**. The threshold value is **1** event and the threshold period is **10** minutes. If an event occurs, and if the time since the last transmission is greater than the threshold value, the notification is sent again.

```
spinserver::> event notification subscriptions new \
-filter all_crits -handler email_admin -thresholdval 1 \
-thresholdperiod 10
```

Displaying Event-Notification Subscriptions

The **event notification subscriptions show** command displays the configuration information for notification subscriptions. The following information is displayed:

- Module
- Event name
- Severity level

To display the configuration information for a particular event-notification subscription, use the **-filter** and **-handler** parameters. The command has the following syntax:

```
event notification subscriptions show [-filter filter_name
-handler handler_name]
```

The following example displays the configuration information for an event-notification subscription with a filter named **all_crits** and a handler named **email_admin**:

```
spinserver::> event notification subscriptions show \
-filter all_crits -handler email_admin
```

Modifying Event-Notification Subscriptions

The **event notification subscriptions modify** command modifies the filters, handlers, threshold values, and threshold time periods (in minutes) of a subscription to event notifications. The command has the following syntax:

```
event notification subscriptions modify -filter
filter_name -handler handler_name -thresholdval
threshold_value -thresholdperiod threshold_period
```

The following example modifies an event-notification subscription named **net_all_crits** so that it uses a filter named **all_crits** and a handler named **email_admin**. It has a threshold value of **2** events and a threshold period of **5** minutes.

```
spinserver::> event notification subscriptions modify \
-name net_all_crits -filter all_crits -handler \
email_admin -thresholdval 2 -thresholdperiod 5
```

Deleting Event-Notification Subscriptions

The **event notification subscriptions delete** command deletes a subscription to event notifications. The command has the following syntax:

```
event notification subscriptions delete -filter
filter_name -handler handler_name
```

The following example deletes an event-notification subscription with a filter named **all_crits** and a handler named **email_admin**:

```
spinserver::> event notification subscriptions delete \
-filter all_crits -hanlder email_admin
```

Managing Syslog Events

The **event syslog** commands enable you to configure the Spin event-tracking system to send local SpinServer system events to the syslog daemon running on a remote SpinServer. You can specify the hosts to which the events are sent and filter the events that are to be sent.

Creating Syslog Events

The **event syslog new** command creates a host configuration that determines where the Spin events are sent and the filter that is used. The command has the following syntax:

```
event syslog new -host spinserver -filter filter_name
-facility facility
```

The following example creates a syslog-event configuration in which a SpinServer named **Server10** receives event messages from a filter named **raid_warn**. The facility is specified as **user**.

```
spinserver::> event syslog new -host Server10 -filter \
raid_warn -facility user
```

Displaying Syslog Events

The **event syslog show** command displays a list of host configurations. To display the configuration information for a particular host, use the **-host** and **-filter** parameters. The command has the following syntax:

```
event syslog show [-host spinserver -filter filter_name]
```

The following example displays configuration information for a syslog event host named **Server10** and a filter named **all_crits**:

```
spinserver::> event syslog show -host Server10 -filter \
all_crits
```

Modifying Syslog Events

Use the **event syslog modify** command to modify a syslog-event configuration. The command has the following syntax:

```
event syslog modify -host spinserver -filter filter_name
-facility facility
```

The following example modifies a syslog-event configuration such that a SpinServer named **Server10** receives event messages from a filter named **raid_alerts**:

```
spinserver::> event syslog modify -host Server10 -filter
raid_alerts -facility user
```

Testing Syslog Events

Use the **event syslog test** command to test a syslog-event configuration. The command causes a test message to be routed through the configuration. The command has the following syntax:

```
event syslog test -host spinserver -filter filter_name
```

The following example tests a syslog-event configuration on a SpinServer named **Server10** with a filter named **raid_warn**:

```
spinserver::> event syslog test -host Server10 -filter
raid_warn
```

Deleting Syslog Events

The **event syslog delete** command deletes a host configuration for Spin events. The command has the following syntax:

```
event syslog delete -host spinserver -filter filter_name
```

The following example deletes a host configuration on a SpinServer named **Server10** with a filter named **all_crits**:

```
spinserver::> event syslog delete -host Server10 -filter \
all_crits
```

Viewing the Audit Log

The audit log lists all commands that have been issued to a SpinServer cluster. Use the **system auditlogs show** command to view the audit log. The information displayed includes the user that ran each command, the date and time at which the command was run, and the status of the command (`Success` or `Failure`). Use the command with the **-logid** parameter to view information on a specific command. The syntax of the command follows:

```
system auditlogs show [-logid log_ID]
```

The following example displays all audit-log entries:

```
spinserver::> system auditlogs show

 Log   SpinServer ID             Command Path                      Date
 ID

1    Server1       raid subsystems show            10/7/02 8:45:30 AM EST
2    Server1       spinservers show                10/7/02 8:45:37 AM EST
3    Server1       ports show                      10/7/02 8:45:38 AM EST
4    Server1       ports show                      10/7/02 8:45:40 AM EST
5    Server1       cluster show                    10/7/02 8:45:40 AM EST
6    Server1       storage units show              10/7/02 8:45:43 AM EST
7    Server1      filestorage scheduler task show   10/7/02 8:46:07 AM EST
8    Server1      filestorage scheduler schedule show 10/7/02 8:46:08 AM EST
9    Server1       filestorage scheduler job show    10/7/02 8:46:10 AM EST
-- Type 'q' to quit and any key to continue --
```

The following example displays information about log entry **150**:

```
spinserver::> system auditlogs show -logid 150
Log ID          150
SpinServer ID   Server1
IP Address      10.1.1.150
User            root
Command Path    filestorage scheduler task show
```

```
Command         filestorage scheduler task show
Date            10/9/02 8:32:07 AM EST
Result          Success
```

22.0 Using Statistics

The various **statistics** commands display operational performance and error information for the hardware and software functions of a SpinServer system.

22.1 Statistics Concepts

The **statistics** commands are divided into the following categories:

- **statistics activeclients**, which provides information on client connections
- **statistics diskdrives**, which provides information on disk drives
- **statistics filesystem**, which provides information on file system operations
- **statistics netaccess**, which provides information on CIFS and NFS operations

All of the commands in the **statistics** command node are **show** commands.

22.2 Statistics Tasks

To monitor the operational performance of the SpinServer system's hardware and software functions, use the **statistics** commands to display data.

Displaying Client Statistics

Use the **statistics activeclients show** command to display statistics on client connections. The command displays the following information:

- Connection ID
- SpinServer ID
- Virtual server name
- Client IP address
- File service protocol (NFS or CIFS)
- Client IP protocol (TCP or UDP)

- Time of last use

Specify the **-connid** *connection_ID* and **-ssid** *spinserver* parameters to display statistics for a single connection on a single SpinServer.

Displaying Disk-Drive Statistics

Use the **statistics diskdrives** commands to display statistics on disk drives. With any of these commands, specify the **-deviceid** *device_ID* parameter to display statistics for a single disk drive.

Displaying General Disk-Drive Statistics

Use the **statistics diskdrives show** command to display general statistics about disk drives. The command displays the following information:

- Device ID
- Number of reads
- Number of writes
- Number of active operations
- Number of waiting operations
- Number of soft errors (GUI only)
- Number of hard errors (GUI only)

Displaying Disk-Drive Error Statistics

Use the **statistics diskdrives show-errors** command to display statistics about disk-drive errors. The command displays the following information:

- Device ID
- LinkErrsA—Number of loss-of-synchronization conditions that have occurred on port A and that exceed 100 ms in duration
- LinkErrsB—Number of loss-of-synchronization conditions that have occurred on port B and that exceed 100 ms in duration
- SyncLossesA—Number of short loss-of-synchronization conditions that have occurred on port A
- SyncLossesB—Number of short loss-of-synchronization conditions that have occurred on port B
- TransErrsA—Number of invalid transmission words that have occurred on port A
- TransErrsB—Number of invalid transmission words that have occurred on port B
- CRCErrsA—Number of write-data frames with invalid CRCs that have been received on port A
- CRCErrsB—Number of write-data frames with invalid CRCs that have been received on port B

- LipF7sInitdA—Number of LIP F7 operations that the drive has initialized on port A
- LipF7sRecdA—Number of LIP F7 operations that the drive has received on port A
- LipF7sInitdB—Number of LIP F7 operations that the drive has initialized on port B
- LipF7sRecdB—Number of LIP F7 operations that the drive has received on port B
- LipF8sInitdA—Number of LIP F8 operations that the drive has initialized on port A
- LipF8sRecdA—Number of LIP F8 operations that the drive has received on port A
- LipF8sInitdB—Number of LIP F8 operations that the drive has initialized on port A
- LipF8sRecdB—Number of LIP F8 operations that the drive has received on port B
- Soft errors—Number of soft errors
- Hard errors—Number of hard errors
- Parity errors—Number of parity errors
- Miscellaneous errors—Number of miscellaneous errors
- PFAs detected—Number of PFAs detected

Displaying File-System Statistics

The **`statistics filesystem`** commands display statistics about file system operations. Statistics are available both for general operations and for VFS-specific operations.

Displaying File-System Usage Statistics

Use the **`statistics filesystem usage show`** command to display statistics about file system usage. Specify the **`-ssid`** *`spinserver`* command to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of disk-read operations
- Number of disk-frame read operations
- Number of disk-write operations
- Number of disk-frame write operations
- Percentage of CPU0 used
- Percentage of CPU1 used
- Number of queued operations
- Total number of requests
- Number of read operations
- Number of write operations
- Number of get-attribute operations

- Number of management calls
- Number of fast operations

If you are using the CLI, the **`statistics filesystem usage show-disk`**, **`statistics filesystem usage show-load`**, and **`statistics filesystem usage show-ops`** commands display subsets of this information (about disk usage, system load, and operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Storage-Pool Activity Statistics

Use the **`statistics filesystem pools show`** command to display statistics about storage-pool activity. Specify the **`-ssid`** *`spinserver`* and **`-deviceid`** *`device_ID`* parameters to display statistics about a single storage pool. The command displays the following information:

- SpinServer
- Storage pool
- Number of reads
- Number of writes
- Number of frame reads
- Number of frame writes
- Number of volume creations
- Number of volume deletions
- Number of volume transfer destinations
- Number of volume clones
- Number of blocks
- Number of inodes

If you are using the CLI, the **`statistics filesystem pools show-disk`**, **`statistics filesystem pools show-usage`**, and **`statistics filesystem pools show-volume`** commands display subsets of this information (about disk usage, usage, and volume operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Primary Spin Operations

Use the **`statistics filesystem spinops primary show`** command to display statistics about primary Spin operations. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of get-attribute operations
- Number of get-build-attribute operations
- Number of write operations
- Number of lookup operations
- Number of read-link operations
- Number of read operations
- Number of read-directory operations

If you are using the CLI, the **`statistics filesystem spinops primary show-get`** and **`statistics filesystem spinops primary show-read`** commands display subsets of this information (about get operations and read operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Directory-Related Spin Operations

Use the **`statistics filesystem spinops directory show`** command to display statistics about directory-related Spin operations. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of access operations
- Number of create operations
- Number of remove operations
- Number of set-attribute operations
- Number of file-system status operations
- Number of make-directory operations
- Number of remove-directory operations
- Number of open operations
- Number of rename operations

If you are using the CLI, the **`statistics filesystem spinops directory show-read`** and **`statistics filesystem spinops directory show-write`** commands display subsets of this information (about read operations and write operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Link- and ACL-Related Spin Operations

Use the **`statistics filesystem spinops link show`** command to display statistics about link- and ACL-related Spin operations. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of links
- Number of symbolic links
- Number of locks
- Number of created mount points
- Number of closes
- Number of set-ACL operations
- Number of get-ACL operations

If you are using the CLI, the **`statistics filesystem spinops link show-acl`** and **`statistics filesystem spinops link show-link`** commands display subsets of this information (about ACL operations and link operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Long-Running Spin Operations

Use the **`statistics filesystem spinops remaining show`** command to display statistics about long-running Spin operations. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of watch-request operations
- Number of unlock-all operations
- Number of notify-obsolete operations
- Number of watch-cancel operations
- Number of fetch-lock operations
- Number of stuff-lock operations
- Number of I/O control operations

If you are using the CLI, the **`statistics filesystem spinops remaining show-lock`** and **`statistics filesystem spinops remaining show-watch`** commands display subsets of this information (about lock-related operations and watch-related operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Spin Volume Operations

Use the **`statistics filesystem spinops volops show`** command to display statistics about Spin volume operations. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer. The command displays the following information:

- SpinServer
- Number of volume creations
- Number of volume deletions
- Number of volume clones
- Number of volume set-attribute operations
- Number of volume get-attribute operations

Displaying Statistics about Primary Spin Operations on VFSs

Use the **`statistics filesystem vfs primary show`** command to display statistics about primary Spin operations on virtual file systems (VFSs). Specify the **`-vfsid`** *`VFS_ID`* parameter to display statistics about a single VFS. The command displays the following information:

- VFS name
- Number of get-attribute operations
- Number of get-build-attribute operations
- Number of write operations
- Number of lookup operations
- Number of read-link operations
- Number of read operations
- Number of read-directory operations

If you are using the CLI, the **`statistics filesystem vfs primary show-get`** and **`statistics filesystem vfs primary show-read`** commands display subsets of this information (about get operations and read operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Directory-Related Spin Operations on VFSs

Use the **`statistics filesystem vfs directory show`** command to display statistics about directory-related Spin operations on virtual file systems (VFSs). Specify the **`-vfsid`** *`VFS_ID`* parameter to display statistics about a single VFS. The command displays the following information:

- VFS name
- Number of access operations
- Number of create operations
- Number of remove operations
- Number of set-attribute operations
- Number of file-system status operations
- Number of make-directory operations
- Number of remove-directory operations
- Number of open operations
- Number of rename operations

If you are using the CLI, the **`statistics filesystem vfs directory show-read`** and **`statistics filesystem vfs directory show-write`** commands display subsets of this information (about read operations and write operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Link- and ACL-Related Spin Operations on VFSs

Use the **`statistics filesystem vfs link show`** command to display statistics about link- and ACL-related Spin operations on virtual file systems (VFSs). Specify the **`-vfsid`** *`VFS_ID`* parameter to display statistics about a single VFS. The command displays the following information:

- VFS name
- Number of links
- Number of symbolic links
- Number of locks
- Number of created mount points
- Number of closes
- Number of set-ACL operations
- Number of get-ACL operations

If you are using the CLI, the **`statistics filesystem vfs link show-acl`** and **`statistics filesystem vfs link show-link`** commands display subsets of this information (about ACL operations and link operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Long-Running Spin Operations on VFSs

Use the **`statistics filesystem vfs remaining show`** command to display statistics about long-running Spin operations on virtual file systems (VFSs). Specify the **`-vfsid`** *`VFS_ID`* parameter to display statistics about a single VFS. The command displays the following information:

- VFS name
- Number of watch-request operations
- Number of unlock-all operations
- Number of notify-obsolete operations
- Number of watch-cancel operations
- Number of fetch-locks operations
- Number of stuff-locks operations
- Number of I/O control operations

If you are using the CLI, the **`statistics filesystem vfs remaining show-lock`** and **`statistics filesystem vfs remaining show-watch`** commands display subsets of this information (about lock operations and watch operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Statistics about Spin Volume Operations on VFSs

Use the **`statistics filesystem vfs volops show`** command to display statistics about Spin volume operations on virtual file systems (VFSs). Specify the **`-vfsid`** *`VFS_ID`* parameter to display statistics about a single VFS. The command displays the following information:

- VFS name
- Number of volume-create operations
- Number of volume-delete operations
- Number of volume-clone operations
- Number of volume set-attribute operations
- Number of volume get-attribute operations

Displaying Statistics about Virus Scanning

If you are running SpinAV, use the **`statistics filesystem antivirus show`** command to display the following information about antivirus scanner activity. Specify the **`-ssid`** *`spinserver`* parameter to display statistics about a single SpinServer.

- SpinServer ID
- Number of upcalls
- Number of upcall errors
- Number of reads
- Number of read errors
- Number of writes
- Number of write errors
- Number of renames
- Number of opens
- Number of closes

In the CLI, the **`statistics filesystem antivirus show-rw`**, **`statistics filesystem antivirus show-upcalls`**, and **`statistics filesystem antivirus show-other`** commands can be used to display subsets of information about read-write operations, upcalls, and miscellaneous operations, respectively.

Displaying Network-Access Statistics

Use the **`statistics netaccess`** commands to display information about network-access protocol activity. The commands display information about general network-access activity, CIFS activity, and NFS activity. Specify the **`-ssid`** *`spinserver`* parameter with any of the commands to display statistics about activity on a single SpinServer.

Displaying Summary Statistics for Network Access

Use the **`statistics netaccess summary show`** command to display summary statistics for the network-access protocols. The command displays the following information:

- SpinServer
- Number of CIFS create operations
- Number of CIFS primary file operations
- Number of CIFS secondary file operations
- Number of CIFS read operations
- Number of CIFS write operations
- Number of CIFS authentication operations
- Number of CIFS locking operations
- Number of CIFS SID-cache operations
- Number of CIFS miscellaneous operations
- Number of NFS V2 read operations
- Number of NFS V2 write operations
- Number of NFS V3 read operations
- Number of NFS V3 write operations

If you are using the CLI, the **`statistics netaccess summary show-cifs`** and **`statistics netaccess summary show-nfs`** commands display subsets of this information (about CIFS operations and NFS operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Primary CIFS Statistics

Use the **`statistics netaccess cifs primaryfilesystem show`** command to display primary statistics for the CIFS protocol. The command displays the following information:

- SpinServer
- Number of CheckDirectory operations
- Number of Transaction operations
- Number of FindClose2 operations
- Number of NtCancel operations
- Number of NtTransact operations
- Number of NtTransactSecondary operations
- Number of Transaction2 operations

If you are using the CLI, the **`statistics netaccess cifs primaryfilesystem show-find`** and **`statistics netaccess cifs primaryfilesystem show-transaction`** commands display subsets of this information (about find operations and transactional operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Secondary CIFS Statistics

Use the **`statistics netaccess cifs secondaryfilesystem show`** command to display secondary statistics for the CIFS protocol. The command displays the following information:

- SpinServer
- Number of QueryInfoDisk operations
- Number of QueryInfo operations
- Number of SetInfo operations
- Number of TransactionSecondary operations
- Number of FindNotifyClose operations
- Number of Search operations
- Number of Find operations
- Number of Transaction2Secondary operations
- Number of FindUnique operations
- Number of Ioctl operations
- Number of IoctlSecondary operations

If you are using the CLI, the **`statistics netaccess cifs secondaryfilesystem show-find`**, **`statistics netaccess cifs secondaryfilesystem show-info`**, and **`statistics netaccess cifs secondaryfilesystem show-transaction`** commands display subsets of this information (about find operations, information operations, and transactional operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Read Statistics

Use the **`statistics netaccess cifs read show`** command to display statistics about CIFS read operations. The command displays the following information:

- SpinServer
- Number of Read operations
- Number of ReadRaw operations
- Number of ReadMpx operations
- Number of ReadMpxSecondary operations
- Number of ReadAndX operations

- Number of ReadBulk operations
- Number of LockAndRead operations

If you are using the CLI, the **`statistics netaccess cifs read show-mpx`** and **`statistics netaccess cifs read show-read`** commands display subsets of this information (about multiplex operations and reading operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Write Statistics

Use the **`statistics netaccess cifs write show`** command to display statistics about CIFS write operations. The command displays the following information:

- SpinServer
- Number of Write operations
- Number of WriteAndUnlock operations
- Number of WriteRaw operations
- Number of WriteMpx operations
- Number of WriteMpxComplete operations
- Number of WriteAndClose operations
- Number of WriteAndX operations
- Number of WriteBulk operations
- Number of WriteBulkData operations

If you are using the CLI, the **`statistics netaccess cifs write show-bulk`** and **`statistics netaccess cifs write show-raw`** commands display subsets of this information (about bulk-write operations and raw-write operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Authentication and Mounting Statistics

Use the **`statistics netaccess cifs auth show`** command to display authentication and mounting statistics for the CIFS protocol. The command displays the following information:

- SpinServer
- Number of TreeDisconnect operations
- Number of Negotiate operations
- Number of LogOffAndX operations
- Number of SessionSetupAndX operations
- Number of TreeConnectAndX operations
- Number of TreeConnect operations
- Number of CloseAndTreeDisconnect operations

If you are using the CLI, the **`statistics netaccess cifs auth show-session`** and **`statistics netaccess cifs auth show-tree`** commands display subsets of this information (about session operations and tree-related operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Creation Statistics

Use the **`statistics netaccess cifs creation show`** command to display creation statistics for the CIFS protocol. The command displays the following information:

- SpinServer
- Number of CreateDirectory operations
- Number of DeleteDirectory operations
- Number of Open operations
- Number of Create operations
- Number of Close operations
- Number of Flush operations
- Number of Delete operations
- Number of Rename operations
- Number of CreateNew operations
- Number of NtCreateAndX operations
- Number of Copy operations
- Number of Move operations
- Number of OpenAndX operations

If you are using the CLI, the **`statistics netaccess cifs creation show-change`**, **`statistics netaccess cifs creation show-create`**, and **`statistics netaccess cifs creation show-delete`** commands display subsets of this information (about change operations, creation operations, and deletion operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Operation-Lock Statistics

Use the **`statistics netaccess cifs oplock show`** command to display statistics about CIFS operation locks. The command displays the following information:

- SpinServer
- Number of batch operation locks requested
- Number of exclusive operation locks requested
- Number of Level II operation locks requested
- Number of batch operation locks granted
- Number of exclusive operation locks granted

- Number of Level II operation locks granted
- Number of operation-lock revocations sent
- Number of operation-lock releases received
- Number of suggested breaks to Level II
- Number of suggested breaks to none
- Number of breaks to Level II
- Number of breaks to none

If you are using the CLI, the **`statistics netaccess cifs oplock show-break`**, **`statistics netaccess cifs oplock show-request`**, and **`statistics netaccess cifs oplock show-response`** commands display subsets of this information (about break operations, request operations, and response operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Range-Lock Statistics

Use the **`statistics netaccess cifs rangelock show`** command to display statistics about CIFS range locks. The command displays the following information:

- SpinServer
- Number of LockByteRange operations
- Number of LockingAndX operations
- Number of LckRangeRequest operations
- Number of LckRangeAsyncNeeded operations
- Number of LckRangeAsyncGranted operations
- Number of LckRangeAsyncRefused operations

If you are using the CLI, the **`statistics netaccess cifs rangelock show-async`** and **`statistics netaccess cifs rangelock show-range`** commands display subsets of this information (about asynchronous operations and range operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying CIFS Name-Server Statistics

Use the **`statistics netaccess cifs nameserver show`** command to display statistics about CIFS name-server operations. The command displays the following information:

- SpinServer
- Number of session requests
- Number of name-server requests
- Number of matching requests

Displaying CIFS SID-Cache Statistics

Use the **`statistics netaccess cifs sidcache show`** command to display statistics about the CIFS security identifier (SID) cache. The command displays the following information:

- SpinServer
- Number of SID lookup operations
- Number of SID misses
- Number of SIDs waiting
- Number of SID cache entries
- Number of SIDs discarded because of no entries
- Number of SIDs discarded because of a full bucket

If you are using the CLI, the **`statistics netaccess cifs sidcache show-entries`** and **`statistics netaccess cifs sidcache show-lookup`** commands display subsets of this information (about entries and lookup operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying Miscellaneous CIFS Statistics

Use the **`statistics netaccess cifs miscellaneous show`** command to display miscellaneous CIFS statistics. The command displays the following information:

- SpinServer
- Number of ProcessExit operations
- Number of Seek operations
- Number of Echo operations
- Number of OpenPrintFile operations
- Number of WritePrintFile operations
- Number of ClosePrintFile operations
- Number of GetPrintQueue operations

If you are using the CLI, the **`statistics netaccess cifs miscellaneous show-print`** and **`statistics netaccess cifs miscellaneous show-seek`** commands display subsets of this information (about print operations and seek operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS Mount Statistics

Use the **`statistics netaccess nfs mount`** show command to display statistics about NFS mount operations. The command displays the following information:

- SpinServer
- Number of mounts
- Number of dumps
- Number of unmounts
- Number of unmount-all operations
- Number of exports
- Number of export-all operations
- Number of paths

If you are using the CLI, the **`statistics netaccess nfs mount show-export`** and **`statistics netaccess nfs mount show-mount`** commands display subsets of this information (about export operations and mount operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS Lock-Manager Statistics

Use the **`statistics netaccess nfs lockmgr show`** command to display lock-manager statistics for the network-access protocols. The command displays the following information:

- SpinServer
- Number of test operations
- Number of lock operations
- Number of cancel operations
- Number of unlock operations
- Number of granted operations
- Number of share operations
- Number of unshare operations
- Number of NM lock operations
- Number of free-all operations

If you are using the CLI, the **`statistics netaccess nfs lockmgr show-free`** and **`statistics netaccess nfs lockmgr show-grab`** commands display subsets of this information (about freeing operations and grabbing operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS V2 Read Statistics

Use the **`statistics netaccess nfs v2read show`** command to display statistics about NFS V2 read operations. The command displays the following information:

- SpinServer
- Number of GetAttr operations
- Number of Lookup operations
- Number of ReadLink operations
- Number of Read operations
- Number of ReadDir operations
- Number of StatFs operations
- Number of Access operations
- Number of ReadDirPlus operations
- Number of FsStat operations
- Number of FsInfo operations
- Number of PathConf operations

If you are using the CLI, the **`statistics netaccess nfs v2read show-get`** and **`statistics netaccess nfs v2read show-read`** commands display subsets of this information (about get operations and read operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS V2 Write Statistics

Use the **`statistics netaccess nfs v2write`** show command to display statistics about NFS V2 write operations. The command displays the following information:

- SpinServer
- Number of SetAttr operations
- Number of Write operations
- Number of Create operations
- Number of Remove operations
- Number of Rename operations
- Number of Link operations
- Number of SymLink operations
- Number of Mkdir operations
- Number of Rmdir operations
- Number of Mknod operations
- Number of Commit operations

If you are using the CLI, the **`statistics netaccess nfs v2write show-create`** and **`statistics netaccess nfs v2write show-directory`** commands display subsets of this information (about creation operations and directory-related operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS V3 Read Statistics

Use the **`statistics netaccess nfs v3read`** show command to display statistics about NFS V3 read operations. The command displays the following information:

- SpinServer
- Number of GetAttr operations
- Number of Lookup operations
- Number of ReadLink operations
- Number of Read operations
- Number of ReadDir operations
- Number of StatFs operations
- Number of Access operations
- Number of ReadDirPlus operations
- Number of FsStat operations
- Number of FsInfo operations
- Number of PathConf operations

If you are using the CLI, the **`statistics netaccess nfs v3read show-access`**, **`statistics netaccess nfs v3read show-directory`**, and **`statistics netaccess nfs v3read show-stat`** commands display subsets of this information (about access operations, directory-related operations, and stat operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

Displaying NFS V3 Write Statistics

Use the **`statistics netaccess nfs v3write`** show command to display statistics about NFS V3 write operations. The command displays the following information:

- SpinServer
- Number of SetAttr operations
- Number of Write operations
- Number of Create operations
- Number of Remove operations
- Number of Rename operations
- Number of Link operations

- Number of SymLink operations
- Number of Mkdir operations
- Number of Rmdir operations
- Number of Mknod operations
- Number of Commit operations

If you are using the CLI, the **`statistics netaccess nfs v3write show-create`**, **`statistics netaccess nfs v3write show-directory`**, and **`statistics netaccess nfs v3write show-link`** commands display subsets of this information (about creation operations, directory-related operations, and link-related operations, respectively) to improve legibility. These alternative commands are not reflected in the GUI.

23.0 Using Third-Party Management Utilities

This chapter discusses the use of third-party utilities to manage the SpinServer. The SpinServer supports Simple Network Management Protocol (SNMP) for discovery and delivery of critical events via SNMP traps. The SpinServer supports SNMPv1 (version 1) and SNMPv2c (version 2).

23.1 Third-Party Management Concepts

SNMP is a client/server protocol in which an *agent*, or server, runs on the device being managed (in this case, the SpinServer). The agent relays the status of the device to a *managing entity*, or client, which runs on a separate machine. The SpinServer communicates with SNMP-based management utilities by using SNMP trap destinations, which notify the management utility when specific events occur on the SpinServer.

To use SNMP-based third-party utilities with the SpinServer system, you must first set up SNMP trap destinations. You can then use a Network Management Station (NMS) or other device capable of receiving SNMP traps with the SpinServer system. The NMS or other device is referred to as a *trap destination.*

23.2 Third-Party Management Tasks

To use an SNMP-based management utility, you must set up SNMP trap destinations.

Setting Up SNMP Trap Destinations

Use the **`services snmp trapdest`** commands to set up SNMP trap destinations.

Creating an SNMP Trap Destination

Use the **`services snmp trapdest new`** command to create an SNMP trap destination. The syntax of the command follows:

```
services snmp trapdest new -ipaddress IP_address -trapport
trap_port -community community -snmpversion {SNMPv1 |
SNMPv2c}
```

The following example creates an SNMP trap destination on a Network Management Station (NMS) at IP address **10.127.5.10** that uses port **162** to receive SNMP traps. The trap destination belongs to a community named **public** and uses **SNMPv2c**.

```
spinserver::> services snmp trapdest new -ipaddress \
10.127.5.10 -trapport 162 -community public -snmpversion \
SNMPv2c
```

Displaying Information about an SNMP Trap Destination

Use the **services snmp trapdest show** command to display information about one or more SNMP trap destinations. Use the command with the **-ipaddress** parameter to display information about a specific trap destination, or without the parameter to display information about all trap destinations. The command has the following syntax:

```
services snmp trapdest show [-ipaddress IP_address]
```

The following example displays information about all SNMP trap destinations:

```
spinserver::> services snmp trapdest show
IP Address  Trap Port  Community  SNMP Version
10.10.127.1  162        public     SNMPv2c
10.10.127.2  162        public     SNMPv1
```

Modifying an SNMP Trap Destination

Use the **services snmp trapdest modify** command to modify an SNMP trap destination. The syntax of the command follows:

```
services snmp trapdest modify -ipaddress IP_address
-trapport trap_port -community community -snmpversion
{SNMPv1 | SNMPv2c}
```

The following example modifies an SNMP trap destination on a Network Management Station (NMS) at IP address **10.127.5.10** that uses port **162** to receive SNMP traps. The trap destination belongs to a community named **public** and uses **SNMPv1**.

```
spinserver::> services snmp trapdest modify -ipaddress \
10.127.5.10 -trapport 162 -community public -snmpversion \
SNMPv1
```

Deleting an SNMP Trap Destination

Use the **services snmp trapdest delete** command to delete an SNMP trap destination. The command has the following syntax:

```
services snmp trapdest delete -ipaddress IP_address
```

The following example deletes an SNMP trap destination at IP address **10.127.5.10**:

```
spinserver::> services snmp trapdest delete -ipaddress \
10.127.5.10
```

Part V: Troubleshooting

This part discusses troubleshooting the SpinServer. It includes the following chapters:

Chapter 24. Advanced Administrative Tools

"Advanced Administrative Tools" on page 307 discussed advanced tools that you can use to administer the SpinServer system.

Chapter 25. Troubleshooting the SpinServer System

"Troubleshooting the SpinServer System" on page 315 provides basic information on troubleshooting.

24.0 Advanced Administrative Tools

This chapter discusses advanced tools that you can use to administer the SpinServer. The tools in this section require that you access the SpinServer with a Linux shell. The tools discussed here are not accessible via the SpinManager CLI or GUI. Use these tools only if you are familiar with Linux/UNIX shells.

The following advanced tools are available:

- The **spinsh** command, as discussed in "The spinsh Command" on page 307.
- The **buildexports** command, as discussed in "The buildexports Command" on page 308.
- Pluggable authentication modules (PAM), as discussed in "Pluggable Authentication Modules" on page 309.
- The **vfsdump** and **vfsrestore** commands, as discussed in "The vfsdump and vfsrestore Commands" on page 311.

24.1 The spinsh Command

You can access SpinManager CLI commands in a shell on the SpinServer by using the **spinsh** command, both interactively and in scripts. The command has the following syntax:

```
# /bin/spinsh [-e] [-h] [-t] [-v] [-y] [-c command]
```

where

-e specifies that the **spinsh** command abort if any of the commands issued under it fail. This option is used only in batch mode.

-h displays command usage.

-t displays the execution time, in seconds, of the command after the command is run.

-v displays all fields in **show** commands.

-y disables any confirmation messages associated with commands that are run under the **spinsh** command.

-c *command* specifies a single CLI command to run, after which the **spinsh** command exits.

24.2 The buildexports Command

The **buildexports** command is available for advanced management of export rule sets. See “Managing Security” on page 207 for additional information about managing export rule sets. The command reads one or more export file lists and creates or updates export lists based on the specifications located in the file.

The command has the following syntax:

```
# /usr/sbin/buildexports [-s server] [-hvV] [-g
netgroup_file_name ...] rule_set_file_name...
```

where

-s *server* specifies the server on which the command is to run. The default is `localhost`.

-h displays command usage.

-v specifies verbose output.

-V displays the version of the program.

-g *netgroup_file_name* specifies the name of a group definition file. The file must be in the standard Solaris **netgroup(4)** format, which is the same format displayed by the **ypcat -k netgroup** command. If groups are not included in the rule set file specified by the *rule_set_file_name* parameter, you do not need to use this option. You can specify multiple **-g** *netgroup_file_name* parameters to use multiple netgroup files.

rule_set_file_name specifies the name of a text file that contains export-list descriptions. You can specify multiple arguments to use multiple rule-set files. Blank lines and white space in the file are ignored, and the pound character (`#`) indicates a comment to the end of the current line. The file must contain lines of the following two forms:

```
ruleset name # Begins a rule set definition for name
rule host_or_group mask_bits RO_rule RW_rule # Adds a rule
```

In the `rule` line, the rule is specified as follows:

- *host_or_group* specifies either a host (as a host name or IP address) or a netgroup prefixed with the at character (`@`). For instance, the expression `@mygroup` specifies a group called `mygroup`. If a netgroup is specified, a netgroup file name must be specified by using the **-g** parameter.
- *mask_bits* specifies the number of mask bits in the rule. Possible values range from `0` to `32`.
- *RO_rule* specifies the read-only rule. Possible values include **any**, **cifs-ntlm**, **cifs-kerberos**, **nfs-kerberos**, and **never**. See “Adding a Rule to a Rule Set” on page 219 for a description of these values.

- *RW_rule* specifies the read-write rule. Possible values include **any**, **cifs-ntlm**, **cifs-kerberos**, **nfs-kerberos**, and **never**. See "Adding a Rule to a Rule Set" on page 219 for a description of these values.

The following is an example of a rule-set file:

```
ruleset englab-only
rule @englab 16 any any  # read-write for hosts in @englab
rule 0.0.0.0 0 any never # rest of world read-only

ruleset webserver-config
rule webmaster-pc.it.example.com 32 any nfs-kerberos
# no access for anyone else
```

When the **buildexports** command is used to create export rule sets, each `rule` line in the rule set file or files is expanded into enough actual rules to cover all of the IP addresses that are needed to represent the host (for instance, multihomed hosts by name) or the netgroup (which can include multiple hosts). If a single `rule` line expands to multiple IP addresses, the mask bits, read-only rule, and read-write rule apply to all of the rules that are created for the expanded set of IP addresses.

24.3 Pluggable Authentication Modules

The SpinServer uses pluggable authentication modules (PAM) to provide mappings from user and group names to IDs. This enables you to alter the mapping algorithms by providing your own PAM modules. The mappings are used in the following contexts:

- Translation of user names to and from the IDs in the SpinManager CLI and GUI for the **-usertype**, **-user**, **-group**, and **-domain** parameters of the following commands:
  - **filestorage vfs modify**
  - **filestorage vfs new**
  - **filestorage vfs show**
  - **vservers servers modify**
  - **vservers servers new**
  - **vservers servers show**
- Translation of user names to and from the IDs for CIFS servers when CIFS-to-NFS mappings are specified. When a CIFS user ID is mapped to an NFS UID, the CIFS ID is passed through the regular-expression mappings; the result of that mapping is looked up in PAM. When an NFS UID is mapped to a CIFS user ID, the NFS UID is looked up in PAM to get a name, and the name is passed through the regular-expression mappings to obtain the CIFS user ID.
- Translation of NFS-Kerberos principal names to NFS UIDs. The NFS-Kerberos client principal name is passed through the regular-expression mappings; the result of that mapping is looked up in PAM to get the UID. This is a one-way mapping; the UID is never translated back to an NFS-Kerberos principal name.

Note: PAM does not affect NFS-only service; standard NFS UIDs are not translated or passed through PAM for authentication.

SpinServer PAM modules must be custom written for translating names to IDs. Third-party PAM modules written for other purposes are not sufficient for use on the SpinServer. For more information about the availability for PAM modules or for details about writing your own PAM modules for use on the SpinServer, contact Network Appliance Technical Support. The default PAM module is named spin_file.so; it is configured by setting up NIS domains for a virtual server through the SpinManager interface, and does not require the following steps to enable it.

Important: The following steps must be performed correctly and identically on each SpinServer in a cluster. Performing any of the steps incorrectly results in inconsistent mappings from SpinServer to SpinServer, which can result in problems such as the inability to access data.

To enable PAM on a SpinServer, perform the following steps:

1. Log onto the SpinServer as root.

2. Change to the `/etc/pam.d` directory:
   `#` **`cd /etc/pam.d`**

3. Locate the file named `spinauth.default`. This file controls the PAM configuration for any virtual server that does not have its own configuration file. To create a PAM configuration file for a single virtual server, copy the `spinauth.default` file to a file named `spinauth.vs`*N*, where *N* is the ID (not name) of the virtual server whose configuration you want to edit. For instance, the configuration file for a virtual server with the ID 2 is named `spinauth.vs2`. You can obtain the ID for a virtual server by using the **`vservers servers show`** command, as described in "Displaying Information about Virtual Servers" on page 148.

4. Open the appropriate `spinauth` file in a text editor.

5. The second line of the file reads as follows:
   ```
   auth required    /lib/security/spin_nis.so
   ```
   Edit the line as follows, replacing `spin_nis.so` with `spin_file.so`:
   ```
   auth required    /lib/security/spin_file.so
   ```

6. Change to the `/var/spinauth` directory:
   `#` **`cd /var/spinauth`**
   By default, this directory is empty.

7. Create password and group files for the file specified in Step 4. The files must be in the standard UNIX password and group file format. To create the files from NIS, run the following commands:
   # **LD_PRELOAD=/usr/spinmgmt/lib/bindresvport.so ypcat -d **
   domain_name **passwd.byname >** *passwd_file_name*
   # **LDPRELOAD=/usr/spinmgmt/lib/bindresvport.so ypcat -d **
   domain_name **group.byname >** *group_file_name*
   The password file (*passwd_file_name*) can be named either `passwd.default` or `passwd.vs`*N*, where *N* is the virtual server ID; the `passwd.default` file is used by virtual servers that do not have an associated `passwd.vs`*N* file. Similarly, the group file (*group_file_name*) can be named either `group.default` or `group.vs`*N*; the group.default file is used by virtual servers that do not have an associated `group.vs`*N* file.

8. If needed, edit the password and group files to affect the mappings from names to IDs.

9. Perform Steps 1 through 7 on all other SpinServers in the cluster. Make sure to perform the steps identically on each machine.

24.4 The vfsdump and vfsrestore Commands

The VFS dump and restoration facilities enable you to write the contents of a VFS to standard output or a specified dump file, and then to restore the contents of the VFS. These facilities can be used for the following purposes:

- As part of a custom backup solution
- To copy a VFS from one cluster to another (the SpinCopy mechanism enables copies only within the same cluster)
- To copy a VFS from one server to another when it is inefficient to use SpinCopy (for instance, if a large VFS needs to be copied to a distant server, and only limited bandwidth is available)

The VFS dump and restoration facilities are provided as a set of application programming interfaces (APIs). The **`vfsdump`** and **`vfsrestore`** commands, which are programs that implement these APIs, are installed as compiled executable files on the SpinServer. This section describes the commands, not the APIs. For information on the APIs, contact Network Appliance Technical Support.

Note: The VFS dump and restoration interface does not provide for the backup and restoration of a subset of a VFS's files and directories. If you require this functionality, use the SpinServer's NDMP backup and restoration functionality with a third-party utility such as VERITAS NetBackup or Legato NetWorker.

The vfsdump Command

The **vfsdump** command writes the contents of a VFS to standard output or a specified output file. By default, the command takes a read-only SpinShot of the source VFS and creates a dump from the SpinShot. The command has the following syntax:

```
# /sbin/vfsdump [-h] [-s spinserver] [-l previous_label]
[-n] [-c] [-t timeout_value] [-o [output_file]] VFS_name
```

where

[**-h**] displays syntax and usage information for the command.

[**-s** *spinserver*] specifies the name or IP address of a SpinServer in the cluster in which the source VFS is located.

[**-l** *previous_level*] specifies the label of a previous dump file for an incremental dump.

[**-n**] specifies that, if the source VFS is a read-write VFS, a SpinShot is not first taken of the source VFS. To ensure consistency, a VFS cannot be modified while a dump of it is created. If you specify this parameter, the SpinServer takes the source VFS offline while the dump is created, then sets the VFS online when the dump is complete.

[**-t** *timeout_value*] specifies the connection timeout value, in seconds. The default is **3** seconds.

[**-o** [*output_file*]] specifies the name of a file to which dump output is to be directed. By default, dump output goes to STDOUT.

VFS_name specifies the VFS whose contents are to be dumped.

The following example dumps the contents of a VFS named **user.eric** to a dump file named **/tmp/user.eric.dump**. The VFS is located in a cluster that includes a SpinServer named **Server1**. This is a full dump, not an incremental dump.

```
# vfsdump -s Server1 -o /tmp/user.eric.dump user.eric
```

The vfsrestore Command

The **vfsrestore** command restores the contents of a VFS from a VFS dump. Data can be restored either to a new VFS or to an existing VFS. The command has the following syntax:

```
# /sbin/vfsrestore [-h] [-s spinserver] [-i input_file]
[-f] [-t timeout_value] [-p storage_pool_name] [-v
virtual_server_name] VFS_name
```

where

[**-h**] displays syntax and usage information for the command.

[**-s** *spinserver*] specifies the name or IP address of the SpinServer on which the restored VFS is to be located.

[**-i** *input_file*] specifies the name of a file from which dump data is to be restored. By default, input is read from STDIN.

[**-f**] specifies that data is to be forcibly restored if the specified dump file is out of sequence.

[**-t** *timeout_value*] specifies the connection timeout value, in seconds. The default is **3** seconds.

[**-p** *storage_pool_name*] specifies the name of the storage pool on which the restored VFS is to be created. Specify this parameter and the **-v** parameter only if a new VFS needs to be created to hold the restored data.

[**-v** *virtual_server_name*] specifies the virtual server on which the restored VFS is to be created. Specify this parameter and the **-p** parameter only if a new VFS needs to be created to hold the restored data.

VFS_name specifies the name of the VFS to which dump data is to be restored. If the VFS does not already exist, the command creates the VFS.

Important: An aborted VFS restoration can leave the target VFS in an inconsistent state. If a restoration is aborted, restart the restoration process with the first full dump of the source VFS. After a restoration failure, the damaged VFS is marked as offline and cannot be brought online until it is checked by **spiff**, the SpinFS file-system checking tool. Contact Network Appliance Technical Support for assistance with running **spiff**.

The following example restores the contents of a dump file named **/tmp/user.eric.dump:0:1054295** to a VFS named **user.eric.new** on a SpinServer named **Server2**. The new VFS is created on a storage pool named **pool2** and a virtual server named **vs0**.

```
# vfsrestore -s Server2 -i /tmp/user.eric.dump:0:1054295 \
-p pool2 -v vs0 user.eric.new
```

Dump Files

A dump file can be either a full dump, consisting of all data in a VFS, or an incremental dump, consisting of data that has changed since a previous dump. Each dump file has a unique label in the following format:

dump_name:*lowest_data_version*:*highest_data_version*

where

dump_name is the name of the dump file as specified by the client. This name is typically but not necessarily the name of the VFS whose data is being dumped.

lowest_data_version is the data version of the oldest modification included in the dump.

highest_data_version is the data version of the most recent modification included in the dump.

The following example is the label for a dump of a VFS named `home.root`, whose oldest modification is data version `93571` and whose most recent modification is data version `143209`:

```
home.root:93571:143209
```

The maximum length of the label is 200 characters. If the length of a label exceeds 200 characters, the *dump_name* portion of the label is truncated as needed.

25.0 Troubleshooting the SpinServer System

This chapter describes how to troubleshoot the SpinServer by using the **sensors** commands and system messages. It also describes how to obtain and send support information to Network Appliance Technical Support.

25.1 Displaying Environmental Information

If a SpinServer malfunctions, verify that its physical environment (for instance, power and temperature) is within operational limits. For information on environmental operating limits, see the *Installation Guide* for your hardware platform. The SpinServer includes a number of sensors that track the physical environment; use the **sensors** commands to gather information from the sensors.

If there is an environmental problem with the SpinServer, correct the situation as needed. For instance, if the SpinServer is in a facility in which the ambient temperature is too high, increase the air flow and cooling in the facility. If a SpinServer component such as a fan fails, contact Network Appliance Technical Support.

Displaying Temperature Information

Use the **sensors temp show** command to display temperature information about one or more SpinServers. The command has the following syntax:

sensors temp show [**-ssid** *spinserver* **-key** *sensor_ID*]

The following command example displays warnings and information about the operating temperature of the motherboard and CPU in a SpinServer enclosure.

```
   spinserver::> sensors temp show
Host Name Sensor ID    Sensor Description    Temperature (Celsius)

sserv10   7-10       CPU 1 Temperature         38
sserv10   7-11       Motherboard Temperature 2 40
sserv10   7-12       CPU 2 Temperature         41
sserv10   7-9        Motherboard Temperature 1 42
```

Displaying Fan Information

Use the **sensors fan show** command to display information about the intake, exhaust, and CPU fans in one or more SpinServers. The command has the following syntax:

sensors fan show [**-ssid** *spinserver* **-key** *fan_ID*]

The following command example displays information about the operational performance of the intake, exhaust, and CPU fans in a SpinServer cluster.

```
   spinserver::> sensors fan show
Host Name Fan ID    Fan Description     Speed (RPM)

Server1   7-4    Chassis Intake Fan 2   2250
Server1   7-8    CPU 2 Fan              5113
Server1   7-5    Chassis Intake Fan 3   2250
Server1   7-7    CPU 1 Fan              5152
Server1   7-6    Chassis Intake Fan 4   2250
Server1   7-1    Chassis Exhaust Fan 1  2860
Server1   7-2    Chassis Exhaust Fan 2  0
Server1   7-3    Chassis Intake Fan 1   2250
Server12  7-3    Chassis Intake Fan 1   2250
Server12  7-4    Chassis Intake Fan 2   2109
Server12  7-1    Chassis Exhaust Fan 1  2860
Server12  7-7    CPU 1 Fan              4591
Server12  7-8    CPU 2 Fan              4821
Server12  7-6    Chassis Intake Fan 4   2250
Server12  7-2    Chassis Exhaust Fan 2  0
Server12  7-5    Chassis Intake Fan 3   2250
```

Displaying Power Information

Use the **sensors power show** command to display voltage information about a SpinServer's motherboard and CPU. The command has the following syntax:

sensors power show [**-ssid** *spinserver* **-key** *sensor_ID*]

The following command example displays warnings and information about the power of the motherboards and CPUs in a SpinServer cluster.

```
   spinserver::> sensors power show
Host Name Sensor ID     Sensor Description            Voltage

Server1   33         3.3 Volt Supply          3.3712
Server1   12         12.0 Volt Supply         12.1875
Server1   CPU        CPU Core Voltage         1.4382
Server1   50         5.0 Volt Supply          5.044
Server2   33         3.3 Volt Supply          3.3712
Server2   50         5.0 Volt Supply          5.018
Server2   12         12.0 Volt Supply         12.1875
```

Displaying UPS Information

Use the **sensors ups show** command to display information about a SpinServer's uninterruptible power supply (UPS). The command has the following syntax:

```
sensors ups show [-ssid spinserver]
```

The following example displays information about the operational performance of the UPSs in a SpinServer cluster.

```
   spinserver::> sensors ups show
SpinServer ID Utility   Power Off  Last Battery   Minutes
                Line    Threshold   Change Date     Left
              Voltage

ss1           117       15          05/22/01       73
ss2           117       15          05/22/01       104
```

25.2 Obtaining Crash Information

If a SpinServer fails, support personnel can request crash information to assist with diagnosis and repair. To obtain crash information, use the **system crashdumps show** command. The command has the following syntax:

```
system crashdumps show -ssid spinserver -name file_name
```

The following example displays information about a crash file named **vmdump.1** on a SpinServer named **Server1**:

```
spinserver::> system crashdumps show -ssid Server1 -name \
vmdump.1
SpinServer ID        Server1
FileName             vmdump.1
Build                35372
Creation Date        1/14/03 11:34:11 AM EST
Size                 1077510144
Last Modified        1/14/03 11:35:29 AM EST
Readable             true
Writeable            true
Compression Method   Not Compressed
Encryption Method    Not Encrypted
```

Use this command only with the assistance of support personnel.

Deleting a Crash File

You can delete a crash file by using the **system crashdumps delete** command. The command has the following syntax:

```
system crashdumps delete -ssid spinserver -name file_name
```

The following example deletes a crash file named **vmdump.1** on a SpinServer named **Server1**:

```
spinserver::> system crashdumps delete -ssid Server1 \
-name vmdump.1
```

Use this command only at the direction of support personnel.

25.3 Sending Information to Technical Support

If you contact Network Appliance Technical Support for assistance, support personnel can request that you submit support-related information from one or more SpinServers. The **spinservers sendsupportinfo** command enables you to send support information to Technical Support via email or FTP. Use this command at the direction of support personnel.

The command has the following syntax:

```
spinservers sendsupportinfo -ssid spinserver -type {email
| ftp} -parameters parameters
```

The value of the **-parameters** parameter varies according to the value of the **-type** parameter. If the value of the **-type** parameter is specified as **email**, specify parameters of the form *email_server, email_id1, email_id2*, etc. If the value of the **-type** parameter is specified as **ftp**, specify a parameter of the form *FTP_URL*.

The following example sends support information from a SpinServer named **Server1** to the FTP URL **ftp://ftp.spinnakernet.com/incoming**:

```
spinserver::> spinservers sendsupportinfo -ssid Server1 \
-type ftp -parameters ftp://ftp.spinnakernet.com/incoming
```

25.4 Parsing System Messages

To view system messages, use the methods described in the *SpinFS 2.5 Messages and Troubleshooting Guide*. This document also includes a complete listing of messages that can be generated by a SpinServer system.

If the cause of a problem is not apparent from the system messages, contact Network Appliance Technical Support.

25.5 Miscellaneous Tips

Resolving Problems With Joining a Cluster

If the **`spinservers joincluster`** command fails with the error message `Timed out after 30 waiting for the RDB online indication`, verify that the network is configured and running correctly (for instance, ensure that the Ethernet switch is configured). Use the **`ping`** command to test connectivity to the SpinServer whose cluster is being joined. For the argument of the **`ping`** command, use the IP address specified by the value of the **`spinservers joincluster`** command's **`-clusternodeipaddress`** parameter. If you are using an MTU setting of 9000, specify the **`-s 9000`** parameter on the **`ping`** command to use a 9000-byte packet:

```
# ping [-s 9000] spinserver_IP_address
```

Updating SpinServer Identification Information

After the SpinServer chassis has been replaced, the identification information that is stored in the NetBIOS on the motherboard of the SpinServer is updated.

SpinServer Serial Number

The serial number of a SpinServer is unique to each SpinServer chassis and is stored in the NetBIOS on the motherboard of the SpinServer chassis.

After the field-replaceable units, including the SCSI disk drives, are installed in the replacement chassis and the chassis is mounted in the rack, power up the SpinServer. The SpinFS software checks the serial number for accuracy and automatically deletes the old serial number and writes the new serial number for the replacement chassis into the management software.

To display the new serial number, navigate to the **SpinServer → Show** node in the SpinManager GUI. The new serial number is displayed.

SpinServer Asset Tag

The asset tag for a SpinServer is linked to the serial number. After the SpinServer chassis is replaced, the asset tag is no longer recognized by the system and is deleted from the management software.

For your equipment identification purposes, the previous asset tag or a new asset tag can be used to identify the SpinServer. To add the asset tag, navigate to the **SpinServer → Show** node in the SpinManager GUI. The new serial number is displayed. Enter the asset tag in the **Asset Tag** field, then click **Modify**.

Appendixes

26.0 Glossary

CIFS - The Common Internet File System is a protocol that defines a standard for client systems to request file services from server systems over a network. CIFS is based on the Standard Server Message Block (SMB) protocol, which is widely used by personal computers and workstations running a vast variety of operating systems.

CIM - The Common Information Model is an object-oriented model that provides a conceptual framework for describing management data.

COM 1 - COM 1 is the name of the serial communications port on the SpinServer that is used to access the command line interface of the SpinServer.

COM 2 - COM 2 is the name of the serial communications port on the SpinServer that is used to attach the SpinServer to the UPS.

DIMM - A Dual In-line Memory Module is a small circuit board that holds memory chips.

CLI - The command line interface is the command version of the SpinServer management interface software.

FC - Fibre Channel is a set of standards, completely developed through industry cooperation, for a serial I/O bus capable of transferring data between two ports at up to 100 MB/second. Fibre Channel supports point-to-point, arbitrated loop, and switched topologies.

Gb - Gigabit is a unit of meaure for data transfer rates between two entities on a network. In this context, it refers to one billion bits. Network data transfer rates are often measured in gigabits per second (Gbps).

GB - Gigabyte is a unit of measure for data storage on computer systems. One gigabyte (GB) is equal to 1,024 megabytes (MB).

GBICs - Gigabit Interface Converters are transceivers that convert between electrical signals used by host bus adapters (and similar Fibre Channel and Ethernet devices) and either electrical or optical signals suitable for transmission.

GbE - Gigabit Ethernet is a group of Ethernet standards in which data is transmitted at 1 Gb per second.

GUI - The Graphical User Interface is the graphical browser version of the SpinServer management interface software.

HBA - Host Bus Adapter is an I/O adapter that connects a host I/O bus to a computer's memory system.

HTML - Hypertext Markup Language is the authoring language used to create documents that can be displayed on a web browser page.

ICMP - Internet Control Message Protocol is an extension to the Internet Protocol (IP). ICMP supports packets containing error, control, and informational messages.

inode - Inodes are data structures that contain information about files in UNIX file systems. In the file system where each file resides, each file has an inode and is identified by an inode number (i-number). Inodes provide information on files, including user and group ownership, access mode (read, write, or execute permissions) and type. Inodes are created when a file system is created. There are a set number of inodes, which indicates the maximum number of files the system can hold. A file's inode number can be found by using the **`ls -i`** command.

I/O - Input/Output is used to describe any program, operation or device that transfers data between a computer and a peripheral device.

IP - Internet Protocol specifies the format of packets (datagrams) and the addressing scheme. Most networks combine IP with a higher-level protocol, such as TCP or UDP, which establishes a virtual connection between a destination and a source.

JBOD - Just a Bunch of Disks. A storage subsystem using multiple independent disk drives that are not configured according to the RAID system. The expansion disk enclosure in a SpinServer system is a JBOD enclosure.

Kerberos - A network authentication protocol developed at the Massachusetts Institute of Technology. Kerberos uses encryption keys to provide authentication for client/server applications. Authentication provides verification of a user's or process's identity to a server. See also **principal**, **SPN**.

LED - A light-emitting diode is an electronic device that lights up when electricity is passed through it.

Mb - Megabit is a unit of measure for a volume of data storage. 1 Mb = 1,024 kilobits (Kb). It is also used to describe data transfer rates on a network. In this context, it refers to one million bits. Networks are often measured in megabits per second (Mbps).

MB - Megabyte is a unit of measure for data storage on computer systems. One megabyte is equal to 1,048,576 (2 to the 20th power) bytes of data.

MHz - Megahertz is the unit of measurement of the speed of microprocessors, called the clock speed. One MHz represents one million cycles per second.

NDMP - Network Data Management Protocol defines a common architecture for the way heterogeneous file servers on a network are backed up. The protocol allows the creation of a common agent used by the central back-up application to back up different file servers running different platforms and platform versions.

NFS - Network File System is a client/server application that allows all network users to access shared files stored on computers of different types.

NICs - Network Interface Cards are expansion boards that enable network connections for a PC or server.

NIS - Network Information Service provides a simple network lookup service consisting of databases and processes. Its purpose is to provide information that needs to be available to the entire network, such as login names, passwords, home directories, and group information.

NIS+ - Network Information Service Plus was designed to address security problems that NIS cannot address. (There is no relation between NIS+ and NIS. The commands and the overall structure of NIS+ are different from NIS.) NIS+ increases security by using an additional authentication method. Users have a standard login password, which gives them access to the system, and an additional secure or network password.

NTLM - NetLanMan is an authentication mechanism used by CIFS.

NVRAM - Non-Volatile Random Access Memory, a type of memory that retains its contents when the power is turned off.

PDF - Portable Document Format is a file format developed by Adobe Systems that provides an electronic photocopy of a document. PDF captures formatting information from a variety of desktop publishing applications, making it possible to send documents electronically and have them appear on the recipient's monitor, or in printed form, as the writer intended the documents to appear. To view a file in PDF format, you need Adobe Acrobat Reader, a free application distributed by Adobe Systems.

PDU - Power Distribution Units are appliances that provide power connections to system components with high voltage requirements.

Principal - In Kerberos, a principal is a unique identity to which Kerberos can assign tickets. An example Kerberos principal is **susan/admin@SECURITY.EXAMPLE.COM**.

RAID - Redundant Array of Independent Disks is a disk enclosure with RAID controllers. The SpinServer supports RAID levels 0, 1, 3, 5, and 0+1. A RAID set is a group of disks in which the disks independently operate, the data strip size is no smaller than the exported block size, and parity check data is distributed across the array's disks.

SCSI - Small Computer System Interface is a parallel interface standard used by computer systems for attaching peripheral devices, such as disk drives and printers.

SES - The SCSI Enclosure Services manager reports the operational status of disk drives to the RAID controllers.

SNMP - The Simple Network Management Protocol is a protocol for monitoring and managing devices on a network.

SP - A Storage Pool is the logical view, as it appears to users, of the compartmentalization of physical storage disks.

SPN - In Kerberos, a Service Principal Name is the name by which a client uniquely identifies an instance of a service.

SSH - Secure Shell is a program used to log into another computer over a network to securely execute commands from a remote machine, and to move files from one machine to another. It was developed by SSH Communications Security Ltd.

TCP - Transmission Control Protocol is a transport protocol. Where the UDP protocol handles only packets, TCP enables two hosts to establish a connection and exchange streams of data. TCP is more reliable than UDP, but slower.

U - U specifies one unit of height in an equipment rack. 1 U = 1.75 inches = 4.45 cm.

UDP - User Datagram Protocol is a connectionless protocol that runs on top of IP networks, as does TCP. Unlike TCP, UDP provides very few error recovery services, offering instead a direct way to send and receive datagrams over an IP network. It is used primarily for broadcasting messages over a network. UDP is faster than TCP, but less reliable.

UPS - An Uninterruptible Power Supply is an appliance that provides power security, with battery power, in the event of a power failure.

VFS - A Virtual File System is the primary unit of data storage in the SpinServer system.

WINS - The Windows Internet Naming Service supports network clients and servers running Windows operating systems. WINS uses a distributed database that is automatically updated with the names of computers currently available on a Windows Network and the IP address assigned to each one. The SpinServer has the ability to register its CIFS servers with the WINS servers that are defined in the SpinServer system.