Dell™ PowerVault™ Modular Disk Storage Manager CLI Guide

Notes and Notices

NOTE: A NOTE indicates important information that helps you make better use of your computer.

NOTICE: A NOTICE indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

September 2008

Contents

1 About the Command Line Interface 13

How to Use the Command Line Interface **14**

Usage Notes 15

CLI Commands 16

Command Line Parameters 18

Formatting Considerations **24**

Detailed Error Reporting **24**

Exit Status **25**

Usage Examples **27**

2 About the Script Commands 29

Script Command Structure **30**

Script Command Synopsis **32**

Recurring Syntax Elements **34**

Usage Guidelines **40**

Adding Comments to a Script File **41**

3 Configuring a Storage Array 43

Configuring a Storage Array **44**

Determining What is on Your Storage Array 44

Saving a Configuration to a File 47

Using the Create Virtual Disk Command 48

Using the Auto Configure Command 53

Modifying Your Configuration **55**

Setting the Storage Array Password 56

Setting Up SMTP and SNMP Alerts 56

Setting the RAID Controller Module Clocks 57

Setting the Storage Array Host Type 57

Setting Modification Priority 58

Assigning Global Hot Spares 59

4 Using the Snapshot Feature 61

Using Host Servers to Create an Initial Snapshot Virtual Disk **63**

Creating a Snapshot Virtual Disk **63**

Enabling the Snapshot Virtual Disk Feature 64

Creating a Snapshot Virtual Disk with User-Assigned Physical Disks 64

Preparing Host Servers to Create an Initial Snapshot Virtual Disk 65

Creating the Initial Snapshot Virtual Disk 66

Creating a Snapshot Virtual Disk with Software-Assigned Physical Disks 67

Creating a Snapshot Virtual Disk by Specifying a Number of Physical Disks 69

User-Defined Parameters 70

Names of Snapshot Virtual Disks and Repository Virtual Disks 72

Changing Snapshot Virtual Disk Settings **73**

Stopping and Deleting a Snapshot Virtual Disk **74**

Re-creating the Snapshot Virtual Disk 75

Preparing Host Servers to Re-create a Snapshot Virtual Disk 75

Re-creating a Snapshot Virtual Disk 76

5 Using the Virtual Disk Copy Feature 77

Creating a Virtual Disk Copy **78**

Enabling the Virtual Disk Copy Feature 79

Determining Virtual Disk Copy Candidates 79

Creating a Virtual Disk Copy 79

Preparing Host Servers to Create a Virtual Disk Copy 80

Copying the Virtual Disk 81

Viewing Virtual Disk Copy Properties **82**

Changing Virtual Disk Copy Settings **83**

Recopying a Virtual Disk **84**

Preparing Host Servers to Recopy a Virtual Disk 85

Recopying the Virtual Disk 86

Stopping a Virtual Disk Copy **87**

Removing Copy Pairs **87**

Interaction with Other Features **88**

Storage Partitioning 88

Snapshot Virtual Disks 89

6 Maintaining a Storage Array 91

Routine Maintenance **91**

Running a Media Scan 91

Running a Consistency Check 93

Resetting a RAID Controller Module 94

Enabling RAID Controller Module Data Transfer 94

Resetting Battery Age 94

Removing Persistent Reservations 94

Synchronizing RAID Controller Module Clocks 95

Locating Physical Disks 95

Performance Tuning **95**

Monitoring Performance 96

Changing RAID Levels 97

Changing Segment Size 97

Defragmenting a Disk Group 98

Troubleshooting and Diagnostics **98**

Collecting Physical Disk Data 98

Diagnosing a RAID Controller Module 99

Recovery Operations **100**

Setting RAID Controller Module Operational Mode 100

Changing RAID Controller Module Ownership 101

Initializing a Physical Disk 101

Reconstructing a Physical Disk 102

Initializing a Virtual Disk 102

Redistributing Virtual Disks 103

7 Script Commands 105

Command Formatting Rules **106**

Commands Listed by Function **108**

Disk Group Commands 108

Enclosure Commands 109

Host Topology Commands 109

iSCSI Commands 109

Physical Disk Commands 110

RAID Controller Module Commands 111

Session Command 111

Show String Command 111

Snapshot Commands 111

Storage Array Commands 111

Virtual Disk Commands 113

Virtual Disk Copy Commands 113

Commands Listed Alphabetically **114**

Accept Storage Array Pending Topology 114

Activate Storage Array Firmware 114

Autoconfigure Storage Array 115

Autoconfigure Storage Array Hot Spares 116

Check Disk Consistency 117

Clear Physical Disk Channel Statistics 118

Clear Storage Array Configuration 118

Clear Storage Array Event Log 119

Clear Storage Array Firmware Pending Area . . . 119

Clear Virtual Disk Reservations 120

Create Disk Group 120

Additional Information 121

Create Host 122

Create Host Group 123

Create Host Port 124

Create iSCSI Initiator 125
Create RAID Virtual Disk (Automatic Physical Disk Select) 126
Create RAID Virtual Disk (Free Capacity Base Select) 128
Create RAID Virtual Disk (Manual Physical Disk Select) 130
Create Snapshot Virtual Disk 133
Create Virtual Disk Copy 137
Delete Disk Group 139
Delete Host 140
Delete Host Group 140
Delete Host Port 141
Delete iSCSI Initiator 141
Delete Virtual Disk 142
Diagnose RAID Controller Module 143
Disable Storage Array Feature 146
Download Enclosure Management Module Firmware 146
Download Physical Disk Firmware 147
Download Storage Array Firmware/NVSRAM . . . 148
Download Storage Array NVSRAM 149
Download Storage Array Physical Disk Firmware 150
Enable RAID Controller Module 151
Enable Storage Array Feature Key 151
Recopy Virtual Disk Copy 152
Recover RAID Virtual Disk 153
Additional Information 155
Re-create Snapshot 156
Remove Virtual Disk Copy 158
Remove Virtual Disk LUN Mapping 159
Repair Virtual Disk Consistency 160
Reset RAID Controller Module 160

Reset Storage Array Battery Install Date 161
Reset Storage Array iSCSI Baseline 162
Reset Storage Array SAS PHY Baseline 162
Reset Storage Array Virtual Disk Distribution . . . 162
Revive Disk Group 163
Revive Physical Disk 163
Save Enclosure Log Data 164
Save Physical Disk Channel Fault Isolation Diagnostic Status 164
Syntax 164
Save Physical Disk Log 165
Save RAID Controller Module NVSRAM 165
Save Storage Array Configuration 166
Save Storage Array Events 167
Save Storage Array iSCSI Statistics 168
Save Storage Array Performance Statistics 169
Save Storage Array SAS PHY Counts 169
Save Storage Array State Capture 170
Save Storage Array Support Data 170
Set Controller 170
Additional Information 172
Set Disk Group 173
Set Enclosure Attribute 174
Set Enclosure Identification 175
Set Foreign Physical Disk to Native 176
Set Host 176
Set Host Group 178
Set Host Port 179
Set iSCSI Initiator 179
Set iSCSI Target Properties 180
Set Physical Disk Channel Status 182
Set Physical Disk Hot Spare 182
Set Physical Disk State 183

Set RAID Controller Module 183
Syntax 183
Syntax Element Statement Data 185
Additional Information 187
Set Session 188
Set Snapshot Virtual Disk 189
Set Storage Array 191
Set Storage Array Enclosure Positions 192
Set Storage Array ICMP Response 193
Set Storage Array iSNS Server IPv4 Address . . . 194
Set Storage Array iSNS Server IPv6 Address . . . 195
Set Storage Array iSNS Server Listening Port . . . 195
Set Storage Array iSNS Server Refresh 196
Set Storage Array Learn Cycle 197
Set Storage Array Time 197
Set Unnamed Discovery Session 198
Set Virtual Disk 198
Set Virtual Disk Copy 203
Show Current iSCSI Sessions 203
Show Disk Group 204
Show Host Ports 205
Show Physical Disk 205
Show Physical Disk Channel Statistics 207
Show Physical Disk Download Progress 208
Show RAID Controller Module 208
Show RAID Controller Module NVSRAM 209
Show Storage Array 210
Show Storage Array Autoconfigure 212
Show Storage Array Host Topology 214
Show Storage Array LUN Mappings 214
Show Storage Array Negotiation Defaults 214
Show Storage Array Pending Topology 215
Show Storage Array Unreadable Sectors 215

Show String 216
Show Unconfigured iSCSI Initiators 216
Show Virtual Disk 217
Show Virtual Disk Action Progress 218
Show Virtual Disk Copy 219
Show Virtual Disk Copy Source Candidates 220
Show Virtual Disk Copy Target Candidates 220
Show Disk Group Import Dependencies 220
Show Virtual Disk Performance Statistics 221
Show Virtual Disk Reservations 222
Start Disk Group Blink 223
Start Disk Group Defragment 223
Start Enclosure Blink 223
Start iSCSI DHCP Refresh 224
Start Physical Disk Channel Fault Isolation Diagnostics 224
Syntax 224
Start Physical Disk Blink 226
Start Physical Disk Initialize 226
Start Physical Disk Reconstruction 227
Start Storage Array Blink 227
Start Disk Group Import/Export 227
Start Virtual Disk Initialization 228
Stop Disk Group Blink 229
Stop Enclosure Blink 229
Stop iSCSI Session 229
Syntax 229
Stop Physical Disk Blink 229
Stop Physical Disk Channel Fault Isolation Diagnostics 230
Stop Snapshot 230
Stop Storage Array Blink 231

Stop Storage Array Physical Disk Firmware Download 231
Stop Virtual Disk Copy 231

A Sample Script Files 233

Configuration Script Example 1 . . . . . **233**

Configuration Script Example 2 . . . . . **236**

Index 239

1

About the Command Line Interface

This guide is intended for system administrators, developers, and engineers who need to use the command line interface (CLI) tool and its associated commands and script files. Selected CLI commands perform functions that you can also access from the Modular Disk (MD) Storage Manager, which is the graphical user interface (GUI) to the storage array. See the *User's Guide*, which describes the Storage Manager software that is used to create and manage multiple storage arrays. For additional information, see the hardware and software manuals that shipped with your system.

NOTE: Always check for updates on **support.dell.com** and read the updates first because they often supersede information in other documents.

NOTE: CLI commands do not have interactive warnings for destructive commands.

The command line interface (CLI) is a software tool that enables storage array installers, developers, and engineers to configure and monitor storage arrays. Using the command line interface, you can issue commands from an operating system prompt, such as the Microsoft® Windows® command prompt (`C:\`) or a Linux operating system terminal.

Each command performs a specific action for managing a storage array or returning information about the status of a storage array. You can enter individual commands, or run script files when you need to perform operations more than once (such as installing the same configuration on several storage arrays). A script file can be loaded and run from the command line interface. You can also run commands in an interactive mode. Interactive mode enables you to connect to a specific storage array and rapidly enter a command, determine the effect on the storage array, and then enter a new command.

The command line interface gives you direct access to a script engine utility in the Dell™ PowerVault™ Modular Disk Storage Manager software (MD Storage Manager). The script engine reads the commands, or runs a script file, from the command line and performs the operations instructed by the commands.

You can use the command line interface to perform the following functions:

- Directly access the script engine and run commands in interactive mode or using a script file.
- Create script command batch files to be run on multiple storage arrays when you need to install the same configuration on different storage arrays.
- Run script commands on a storage array directly connected to a host, a storage array connected to a host by an Ethernet, or a combination of both.
- Display configuration information about the storage arrays.
- Add storage arrays to and remove storage arrays from the management domain.
- Perform automatic discovery of all storage arrays attached to the local subnet.
- Add or delete Simple Network Management Protocol (SNMP) trap destinations and email alert notifications.
- Specify the mail server and sender email address or Simple Mail Transport Protocol (SMTP) server for alert notifications.
- Direct the output to a standard command line display or to a named file.

How to Use the Command Line Interface

Using the CLI commands, you can access the script engine, specify which storage array receives the script commands, and set operation environment parameters.

A CLI command consists of the following elements:

- The term **SMcli**
- Storage array identifier
- Parameters
- Script commands

The following syntax is the general form of a CLI command:

```
SMcli storageArray parameters script-commands;
```

SMcli	Invokes the command line interface
storageArray	Host name or IP address of the storage array
parameters	CLI parameters that define the environment and purpose for the command
script-commands	One or more script commands or the name of a script file containing script commands

The script commands are the storage array configuration commands. "About the Script Commands" on page 29 presents an overview of the script commands. "Script Commands" on page 105 provides definitions, syntax, and parameters for the script commands.

Usage Notes

If you enter **SMcli** and a storage array name but do not specify CLI parameters, script commands, or a script file, the command line interface runs in interactive mode. Interactive mode enables you to run individual commands without prefixing the commands with **SMcli**. You can enter a single command, view the results, and enter the next command without typing the complete **SMcli** string. Interactive mode is useful for determining configuration errors and quickly testing configuration changes.

If you enter **SMcli** without any parameters or with an incorrect parameter, the script engine returns usage information.

NOTE: The SMcli command is installed under the client directory of the selected path during a management station install of the MD Storage Manager software.

NOTE: The SMcli command should be a component of the system environment command path.

CLI Commands

This section lists the CLI commands you can use to perform the following functions:

- Identify storage arrays.
- Set passwords.
- Add storage arrays.
- Specify communication parameters.
- Enter individual script configuration commands.
- Specify a file containing script configuration commands.

The following are general forms of the CLI commands, showing the parameters and terminals used in each command. Table 1-1 lists definitions for the parameters shown in the CLI commands.

Table 1-1. Command Name Conventions

Parameter	Definition
a|b	pipe symbol indicating alternative ("a" or "b")
italicized-words	terminals
[...] (square brackets)	zero or one occurrence
{...} (curly brackets)	zero or more occurrences
<...> (angle brackets)	occurrence exceeds maximum limit of 30 characters
(a|b|c)	choose only one of the alternatives
bold	terminals

```
SMcli host-name-or-IP-address [host-name-or-IP-
address] [-c "command; {command2};"]
[-n storage-array-name | -w WWID]
[-o outputfile][-p password][-e][-S]

SMcli host-name-or-IP-address
[host-name-or-IP-address] [-f scriptfile]
[-n storage-array-name | -w WWID]
[-o outputfile] [-p password] [-e] [-S]
```

```
SMcli (-n storage-array-name | -w WWID)
[-c "command; {command2};"]
[-o outputfile][-p password][-e][-S]

SMcli (-n storage-array-name | -w WWID)
[-f scriptfile]
[-o outputfile] [-p password] [-e] [-S]

SMcli (-n storage-array-name | -w WWID)
[-o outputfile][-p password][-e][-S]

SMcli -a email:email-address
[host-name-or-IP-address1
[host-name-or-IP-address2]]
[-n storage-array-name | -w WWID | -h host-name |
-r (host_sa | direct_sa)]
[-I information-to-include][-q frequency][-S]

SMcli -x email:email-address
[host-name-or-IP-address1
[host-name-or-IP-address2]]
[-n storage-array-name | -w WWID | -h host-name |
-r (host_sa | direct_sa)] [-S]

SMcli (-a | -x) trap:community,
host-name-or-IP-address [host-name-or-IP-address1
[host-name-or-IP-address2]]
[-n storage-array-name | -w WWID | -h host-name |
-r (host_sa | direct_sa)] [-S]

SMcli -d [-w][-i][-s][-v][-S]

SMcli -m host-name-or-IP-address -F email-address
[-g contactInfoFile][-S]

SMcli -A [host-name-or-IP-address
[host-name-or-IP-address]] [-S]

SMcli -X (-n storage-array-name | -w WWID |
-h host-name)

SMcli -?
```

Command Line Parameters

Table 1-2. Command Line Parameters

Parameter	Definition
host-name-or-IP-address	Specify either the host name or the Internet Protocol (IP) address of an in-band managed storage array (IPv4 or iPv6) or an out-of-band managed storage array (IPv4 only). • If you manage a storage array by using a host connected directly to the storage array (in-band storage management), you must use the *-n* parameter if more than one storage array is connected to the host. • If you manage a storage array through an Ethernet connection (out-of-band storage management), you must specify the **host-name-or-IP-address** of the redundant array of independent disks (RAID) controller modules. • If you have previously configured a storage array in the graphical user interface (GUI) of the MD Storage Manager, you can specify the storage array by its user-supplied name by using the *-n* parameter.
-A	Use to add a storage array to the configuration files. If you do not follow the -A parameter with a **host-name-or-IP-address**, automatic discovery scans the local subnet for storage arrays.
-a	Use to add an SNMP trap destination or an email address alert destination. • When adding an SNMP trap destination, the SNMP community is automatically defined as the community name for the trap and the **host** is the IP address or Domain Name Server (DNS) host name of the system to which the trap should be sent. • When adding an email address for an alert destination, the **email-address** is the email address to which to send the alert message.

Table 1-2. Command Line Parameters *(continued)*

Parameter	Definition
-c	Use to indicate that you are entering one or more script commands to run on the specified storage array. Terminate each command by using a semicolon (;). You cannot place more than one **-c** parameter on the same command line. You can include more than one script command after the **-c** parameter.
-d	Use to display the contents of the script configuration file.
-e	Use to disable syntax checking when executing the current CLI command.
-F (uppercase)	Use to specify the email address from which all alerts will be sent.
-f (lowercase)	Use to specify a file name containing script commands intended to run on the specified storage array. This parameter is similar to the *-c* parameter in that both are intended for running script commands. The *-c* parameter allows you to execute individual script commands. The **-f** parameter allows you to execute script commands contained in a file. **NOTE:** By default, any errors encountered when running the script commands in a file are ignored, and the file continues to run. To override this behavior, use the **set session errorAction=stop** command in the script file.

Table 1-2. Command Line Parameters *(continued)*

Parameter	Definition
-g	Use to specify an ASCII file that contains email sender contact information to include in all email alert notifications. The CLI assumes the ASCII file is text only, without delimiters or any expected format. A typical file contains the following information: • Name • Title • Company • Phone • Pager **NOTE:** You can use any file name that your operating system supports. You must not use **userdata.txt**. Some operating systems reserve **userdata.txt** for system information.
-h	Use with the *-a* and *-x* parameters to specify the host name that is running the SNMP agent to which the storage array is connected.
-I	Use to specify the type of information to be included in the email alert notifications. The following are valid information arguments: • **eventOnly** — Only event information is included in the email. • **profile** — Event and array profile information is included in the email. • **supportBundle** — Event and support bundle information is included in the email. **NOTE:** You can enter only one information argument each time you execute the command. If you want all of the information, you must run the command three times.
-i	Use with the *-d* parameter to display the IP address of the known storage arrays.
-m	Use to specify the host name or IP address of the email server from which to send email alert notifications.

Table 1-2. Command Line Parameters *(continued)*

Parameter	Definition
-n	Use to specify the name of the storage array on which to run the script commands. This name is optional when you use **host-name-or-IP-address**; however, if you are using the in-band method for managing the storage array, you must use the **-n** parameter if more than one storage array is connected to the host at the specified address. The storage array name is required when **host-name-or-IP-address** is not used; however, the name of the storage array configured for use in the MD Storage Manager GUI (that is, listed in the configuration file) must not be a duplicate name of any other configured storage array.
-o	Use with the *-c* or *-f* parameter to specify a file name for all output text that is a result of running the script commands.
-p	Use to specify the password for the storage array on which to run commands. A password is not necessary under the following conditions: • A password has not been set on the storage array. • The password is specified in a script file that is running. • The storage array password is specified by using the *-c* parameter and the **set session password=***password* command.

Table 1-2. Command Line Parameters *(continued)*

Parameter	Definition
-q	Use to specify how frequently to include additional profile or support bundle information in the email alert notifications. An email alert notification that contains at least the basic event information is always generated for every critical event. If you set the *-I* parameter to **eventOnly**, the only valid argument for *-q* is **everyEvent**. If you set the *-I* parameter to either **profile** or **supportBundle**, this information is included with the emails with the frequency specified by the *-q* parameter. Valid frequency arguments are: • **everyEvent** — Information is returned with every email alert notification. • **2** — Information is returned no more than once every two hours. • **4** — Information is returned no more than once every four hours. • **8** — Information is returned no more than once every eight hours. • **12** — Information is returned no more than once every 12 hours. • **24** — Information is returned no more than once every 24 hours.
-r	Use with the *-a* or *-x* parameter to specify the name of a management station. The name of a management station can be either **direct_sa** (out-of-band storage array) or **host_sa** (in-band storage arrays [host-agent]). The *-r* parameter enables you to set or change the alert notifications for all storage arrays under each management station.

Table 1-2. Command Line Parameters *(continued)*

Parameter	Definition
-S (uppercase)	Use to suppress the informational messages describing command progress that appear when running script commands. (Suppressing informational messages is also called *silent mode*.) This parameter suppresses the following messages: • Performance syntax check • Syntax check complete • Executing script • Script execution complete • SMcli completed successfully
-s (lowercase)	Use with the *-d* parameter to display the alert settings in the configuration file.
-v	Use with the *-d* parameter to display the current global status of the known devices in the storage array configuration file. (The configuration file lists all of the devices in a storage array configuration and the relationship between the devices. Use the configuration file to reconstruct a storage array.)
-X (uppercase)	Use to delete a storage array from the configuration file. (The configuration file lists all of the devices in a storage array configuration and the relationship between the devices. Use the configuration file to reconstruct a storage array.)
-x (lowercase)	Use to remove an SNMP trap destination or an email address alert destination. The **community** is the SNMP community name for the trap, and the **host** is the IP address or DNS host name of the system to which you want the trap sent.
-?	Use this parameter to display usage information about the CLI commands.

Formatting Considerations

Quotation marks (" ") used as part of a name or label require special consideration when you run the CLI and script commands on a Microsoft® Windows® operating system. The following explains the use of quotation marks in names while running CLI and script commands on Windows.

When quotation marks (" ") are part of an argument, you must insert a backslash (\) before each quotation mark character unless you are in interactive mode. For example:

```
-c "set storageArray userLabel=\"Engineering\";"
```

where **Engineering** is the storage array name.

You cannot use quotation marks (" ") as part of a character string (also called *string literal*) within a script command. For example, you cannot enter the following string to set the storage array name to "**Finance**"**Array**:

```
-c "set storageArray userLabel=
\"\"Finance\"Array\";"
```

On a Linux operating system, the delimiters around names or labels are single quotation marks (‘ ’). The Linux versions of the previous examples are:

```
-c ‘set storageArray userLabel="Engineering";’
```

Detailed Error Reporting

Error data collected from an error encountered by the CLI is written to a file. Detailed error reporting under the CLI works as follows:

- If the CLI must abnormally end execution or abort script command execution, error data is collected and saved before the CLI aborts.
- The CLI automatically saves the error data by writing the data to a file with a standard name.
- The CLI does not have any provisions to avoid overwriting an existing version of the file containing error data.

For error processing, errors appear as two types:

- Parameter or syntax errors you might enter
- Exceptions that occur as a result of an operational error

When the CLI encounters either type of error, it writes information describing the error directly to the command line and sets a return code. Depending on the return code, the CLI might also write additional information about which parameter caused the error. The CLI also writes information about what command syntax was expected to help you identify any syntax errors you might have entered.

When an exception occurs while executing a command, the CLI automatically saves the error information to a file named **excprpt.txt**. The CLI attempts to place **excprpt.txt** in the directory specified by the system property **devmgr.datadir**, which by default is the "client/data" directory under the main installation directory in Windows and the **/var/opt/SM** directory in Linux. If for any reason the CLI cannot place the file in the **devmgr.datadir**-specified directory, the CLI saves the **excprpt.txt** file in the same directory from which the CLI is running. You cannot change the file name or location. The **excprpt.txt** file is overwritten every time an exception occurs. To save the information in the **excprpt.txt** file, you must to copy the information to a new file or directory.

Exit Status

After you run a CLI command or a CLI and script command, status is displayed that indicates the success of the operation defined by the command. The status values are shown in Table 1-3.

Table 1-3. Exit Status

Status Value	Meaning
0	The command terminated without an error.
1	The command terminated with an error. Error information is also displayed.
2	The script file does not exist.
3	An error occurred while opening an output file.
4	A storage array is not at the specified address.
5	Addresses specify different storage arrays.
6	A storage array name does not exist for the host agent connected.
7	The storage array name was not at the specified address.

Table 1-3. Exit Status *(continued)*

Status Value	Meaning
8	The storage array name was not in the configuration file.
10	A management class does not exist for the storage array.
11	A storage array was not found in the configuration file.
12	An internal error occurred.
13	Invalid script syntax was found.
14	The RAID controller module was unable to communicate with the storage array.
15	A duplicate argument was entered.
16	An execution error occurred.
17	A host was not at the specified address.
18	The World Wide Identifier (WWID) was not in the configuration file.
19	The WWID was not at the address.
20	An unknown IP address was specified.
21	The event monitor configuration file was corrupted.
22	The storage array was unable to communicate with the event monitor.
23	The RAID controller module was unable to write alert settings.
24	The wrong management station was specified.
25	The command was not available.
26	The device was not in the configuration file.
27	An error occurred while updating the configuration file.
28	An unknown host error occurred.
29	The sender contact information file was not found.
30	The sender contact information file could not be read.
31	The **userdata.txt** file exists.
32	An invalid *-I* value in the email alert notification was specified.
33	An invalid *-f* value in the email alert notification was specified.

Usage Examples

The following examples show how to enter CLI commands on a command line. The examples show the syntax, form, and, in some examples, script commands. Examples are shown for both Windows and Linux operating systems. The usage for the *-c* parameter varies depending on your operating system. On Windows operating systems, put quotation marks (" ") around the script command following the **-c** parameter. On Linux operating systems, put single quotation marks (‘ ’) around the script command following the *-c* parameter.

NOTE: See "Script Commands" on page 105 for descriptions of the script commands used in the following examples.

This example shows how to change the name of a storage array. The original name of the storage array is **Payroll_Array**. The new name is **Finance_Array**.

Windows:

```
SMcli -n "Payroll_Array" -c "set storageArray
userLabel=\"Finance_Array\";"
```

Linux:

```
SMcli -n ‘Payroll_Array’ -c ‘set storageArray
userLabel="Finance_Array";’
```

This example shows how to delete an existing virtual disk and create a new virtual disk on a storage array. The existing virtual disk name is **Stocks_<_Bonds**. The new virtual disk name is **Finance**. The RAID controller module host names are **finance1** and **finance2**. The storage array is protected and requires the password **TestArray**.

Windows:

```
SMcli finance1 finance2 -c "set session password=
\"TestArray\"; delete virtualDisk [\"Stocks_<_
Bonds\"]; create virtualDisk physicalDiskCount[3]
raidLevel=5 capacity=10 GB userLabel=\"Finance\";
show storageArray healthStatus;"
```

Linux:

```
SMcli finance1 finance2 -c ‘set session password=
"TestArray"; delete virtualDisk
["Stocks_<_Bonds"]; create virtualDisk
```

```
physicalDiskCount[3] raidLevel=5 capacity=10 GB
userLabel="Finance"; show storageArray
healthStatus;'
```

This example shows how to run commands in a script file named **scriptfile.scr** on a storage array named **Example**. The *-e* parameter runs the file without checking syntax. Executing an **SMcli** command without checking syntax enables the file to run more quickly; however, the **SMcli** command may not execute correctly if the syntax is incorrect.

```
SMcli -n Example -f scriptfile.scr -e
```

This example shows how to run commands in a script file named **scriptfile.scr** on a storage array named **Example**. In this example, the storage array is protected by the password **My_Array**. Output, as a result of commands in the script file, goes to file **output.txt**.

Windows:

```
SMcli -n Example -f scriptfile.scr -p "My_Array" -
o output.txt
```

Linux:

```
SMcli -n Example -f scriptfile.scr -p 'My_Array' -
o output.txt
```

This example shows how to display all storage arrays that are currently discovered in the current configuration. The command in this example returns the host name of each storage array.

```
SMcli -d
```

If you want to know the IP address of each storage array in the configuration, add the *-i* parameter to the command.

```
SMcli -d -i
```

2

About the Script Commands

You can use the script commands to configure and manage a storage array. The script commands are distinct from the command line interface (CLI) commands; however, you enter the script commands using the command line interface. You can enter individual script commands, or run a file of script commands. When entering an individual script command, include it as part of a CLI command. When running a file of script commands, include the file name as part of a CLI command. The script commands are processed by a script engine that performs the following functions:

- Verifies command syntax
- Interprets the commands
- Converts the commands to the appropriate protocol-compliant commands, which is, in turn, run by the RAID controller module
- Passes the commands to the storage array

At the storage array, the redundant array of independent disks (RAID) controller modules in the storage array runs the script commands.

The script engine and script commands support the storage array configuration and management operations listed in Table 2-1.

Table 2-1. Configuration and Management Operations

Operation	Activities
Virtual disk, disk group configuration	Creating, deleting, and setting priority; labeling; setting physical disk composition when creating virtual disks; setting segment size; and setting media scan control
Physical disk configuration	Configuring the hot spare
RAID controller module configuration	Defining virtual disk ownership, changing mode settings, defining network settings, and setting host port IDs

Table 2-1. Configuration and Management Operations *(continued)*

Operation	Activities
General storage array configuration	Resetting a configuration to defaults, labeling, checking the health status, setting the time of day, clearing the Major Event Log, and setting the media scan rate
NVSRAM configuration	Downloading and modifying the user configuration region at the bit and byte level, displaying nonvolatile static random access memory (NVSRAM) values
Product identification	Retrieving the enclosure profile display data
Battery management	Setting the battery installation date
Firmware management	Downloading RAID controller module, enclosure management module (EMM), and physical disk firmware

Script Command Structure

All script commands have the following structure:

```
command operand-data {statement-data}
```

where **command** identifies the action to be performed, *operand-data* represents the storage array component to configure or manage (such as a RAID controller module, physical disk, or disk group), and *statement-data* is what you want to do to the component (such as, specifying the RAID level or availability of a disk group).

The general form of the syntax for *operand-data* is as follows:

```
(object-type | allobject-types | [qualifier]
(object-type [identifier] {object-type
[identifier]} | object-types [identifier-list]))
```

An *operand-data* object can be identified four ways:

- The object types and object qualifiers
- The *all* parameter

- Brackets
- A list of identifiers

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Use an object type when the command is not referencing a specific object. The *all* parameter means all objects of the specified type in the storage array (for example, *allVirtualDisks*).

To perform a command on a specific object, use brackets to identify the object (for example, **virtualDisk[engineering]**). Specify a subset of objects with a list of identifiers in brackets (for example, **virtualDisks[sales engineering marketing]**). In a list of identifiers, use a blank space as the delimiter. A qualifier is necessary if you want to include additional information to describe the objects.

Table 2-2 lists the object type and identifiers associated with the object types.

Table 2-2. Object Types and Identifiers

Object Type	Identifier
controller	0 or 1
physicalDisk	Enclosure ID and the slot ID
physicalDiskChannel	Physical disk channel identifier
host	User label
hostChannel	Host channel identifier
hostGroup	User label
hostPort	User label
snapshot	Virtual disk user label
storageArray	Not applicable
enclosure	Enclosure ID
virtualDisk	Virtual disk user label or the World Wide Identifier (WWID) for the virtual disk (**set** command only)

Table 2-2. Object Types and Identifiers *(continued)*

Object Type	Identifier
virtualDiskCopy	Target virtual disk and, optionally, the source virtual disk user labels
diskGroup	Virtual disk group number

Statement data is in the form of **attribute=value** (such as **raidLevel=5**), an **attribute name** (such as **batteryInstallDate**), or an **operation name** (such as **consistencyCheck**).

Script Command Synopsis

Because you can use the script commands to define and manage the different aspects of a storage array (such as host topology, physical disk configuration, RAID controller module configuration, virtual disk definitions, and disk group definitions), the actual number of commands is extensive. The commands, however, fall into general categories that are reused when you apply the commands to the different aspects of a storage array.

Table 2-3 lists the general form of the script commands and provides a definition of each command.

Table 2-3. General Form of the Script Commands

Command Syntax	Description
activate *object* {*statement-data*}	Sets up the environment so that an operation can take place or performs the operation if the environment is already correctly set up.
autoConfigure *storageArray*{*statement-data*}	Automatically creates a configuration based on parameters specified in the command.
check *object* {*statement-data*}	Starts a synchronous operation to report on errors in the object.
clear *object* {*statement-data*}	Discards the contents of some attribute of an object. This is a destructive operation that cannot be reversed.
create *object* {*statement-data*}	Creates an object of the specified type.

Table 2-3. General Form of the Script Commands *(continued)*

Command Syntax	Description
deactivate *object* {*statement-data*}	Removes the environment for an operation.
delete *object*	Deletes a previously created object.
diagnose *object* {*statement-data*}	Runs a test and displays the results.
disable *object* {*statement-data*}	Prevents a feature from operating.
download *object* {*statement-data*}	Transfers data to the storage array or hardware associated with the storage array.
enable *object* {*statement-data*}	Allows a feature to operate.
recopy *object* {*statement-data*}	Restarts a virtual disk copy operation by using an existing virtual disk copy pair. You can change attributes before the operation is restarted.
recover *object* {*statement-data*}	Re-creates an object from saved configuration data and the statement attributes (similar to the **create** command.)
recreate *object* {*statement-data*}	Restarts a snapshot operation using an existing snapshot virtual disk. You can change attributes before the operation is restarted.
remove *object* {*statement-data*}	Removes a relationship from between objects.
repair *object* {*statement-data*}	Repairs errors found by the `check` command.
reset *object* {*statement-data*}	Returns the hardware or object to an initial state.
resume *object*	Starts a suspended operation. The operation begins where it left off when suspended.
revive *object*	Forces the object from the Failed to the Optimal state. Use only as part of an error recovery procedure.
save *object* {*statement-data*}	Writes information about the object to a file.
set *object* {*statement-data*}	Changes object attributes. All changes are completed when the command returns.

Table 2-3. General Form of the Script Commands *(continued)*

Command Syntax	Description
show *object* {*statement-data*}	Displays information about the object.
start *object* {*statement-data*}	Starts an asynchronous operation. You can stop some operations after they have started. You can query the progress of some operations.
stop *object* {*statement-data*}	Stops an asynchronous operation.
suspend *object* {*statement-data*}	Suspends an operation. You can then restart the suspended operation, and it continues from the point at which it was suspended.

Recurring Syntax Elements

Recurring syntax elements are a general category of variables and parameters you can use in one or more script commands. The recurring syntax is used in the general definitions of the script commands that are listed in "Script Commands" on page 105. Table 2-4 lists the recurring syntax and the syntax values that you can use with the syntax.

Table 2-4. Recurring Syntax Elements

Recurring Syntax	Syntax Value
raid-level	(0 | 1 | 5| 6)
snapshot-repository-raid-level	(1 | 5 | 6)
capacity-spec	*integer-literal* [KB | MB | GB | TB | Bytes]
segment-size-spec	integer-literal
boolean	(TRUE | FALSE)
user-label	string-literal
user-label-list	*user-label* {*user-label*}
create-raid-vol-attr-value-list	*create-raid-virtual disk-attribute-value-pair* {*create-raid-virtual disk-attribute-value-pair*}

Table 2-4. Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Value
create-raid-virtual disk-attribute-value-pair	capacity=*capacity-spec* | owner=(0 | 1) | segmentSize=*integer-literal*
RAID controller module-enclosureId	(0–99)
slot-id	(0–31)
port-id	(0–127)
physical disk-spec	enclosureID, slotID
physical disk-spec-list	*physical disk-spec* {*physical disk-spec*}
enclosure-id-list	*enclosureID* {*enclosureID*}
hex-literal	0x hexadecimal-literal
virtual disk-group-number	integer-literal
filename	string-literal
error-action	(stop | continue)
physical disk-channel-identifier	(1 | 2)
physical disk-channel-identifier-list	*physical disk-channel-identifier* {*physical disk-channel-identifier*}
host-channel-identifier	(01 | 02 | 11 | 12)
physical disk-type	(Serial Attached SCSI [SAS] | Serial Advanced Technology Attachment [SATA])
feature-identifier	(snapshot | virtualDiskCopy)
repository-spec	instance-based-repository-spec | count-based-repository-spec
ethernet-port-options	**IPV4Address**=*ipv4-address*| **IPV4ConfigurationMethod**=[*(static* | *dhcp)*] | **IPV4GatewayIP**=*ipv4-address*| **IPV4SubnetMask**=*ipv4-address*|

Table 2-4. Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Value
iscsi-host-port-options	IPV4Address=*ipv4-address* | IPV6LocalAddress=*ipv6-address* | IPV6RoutableAddress=*ipv6-address* | IPV6RouterAddress=*ipv6-address* | enableIPV4= *boolean* | enableIPV6=*boolean* | enableIPV4Vlan=*boolean* | enableIPV6Vlan=*boolean* | enableIPV6Priority=*boolean* | enableIPV6Priority=*boolean* | IPV4ConfigurationMethod=(*static* | *dhcp*) | IPV6ConfigurationMethod=(*static* | *auto*) | IPV4GatewayIP= *ipv4-address* | IPV6HopLimit=*integer* | IPV6NdDetectDuplicateAddress= *integer* | IPV6NdReachableTime=*integer* | IPV6NdRetransmitTime=*integer* | IPV6NdTimeOut=*integer* | IPV4Priority=*integer* | IPV6Priority=*integer* | IPV4SubnetMask=*ipv4-address* | IPV4VlanID=*integer* | IPV6VlanID=*integer* | maxFramePayload=*integer* | tcpListeningPort=*tcp-port-id*| **NOTE:** You must set the enableIPV4 parameter or the enableIPV6 parameter to TRUE to ensure that the specific IPV4 or IPV6 setting is applied. **NOTE:** The IPV6 address space is 128 bits. It is represented by eight 16-bit hexadecimal blocks separated by colons. You may drop leading zeros, and use a double colon to represent consecutive blocks of zeroes.

Table 2-4. Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Value
instance-based-repository-spec	repositoryRAIDLevel=*repository-raid-level* repositoryPhysicalDisks=(*physical disk-spec-list*) [enclosureLossProtect=boolean] repositoryDiskGroup=virtual-disk-group-number [freeCapacityArea=integer-literal] Specify repositoryRAIDLevel with repositoryPhysicalDisks. Do not specify RAID level or physical disks with a disk group. Do not set **enclosureLossProtect** when specifying a disk group. **NOTE:** For enclosure loss protection to work, each physical disk in a disk group must be on a separate enclosure. If you set **enclosureLossProtect=TRUE** and have selected more than one physical disk from any one enclosure, the storage array returns an error. If you set **enclosureLossProtect=FALSE**, the storage array performs operations, but the disk group you create might not have enclosure loss protection. **NOTE:** To determine if a free capacity area exists, issue the **show diskGroup** command.
count-based-repository-spec	repositoryRAIDLevel=*repository-raid-level* repositoryPhysicalDiskCount=*integer-literal* [physicalDiskType=*physical disk-type*] [enclosureLossProtect=*boolean*]
WWID	string-literal. For hostPort identifiers this is a 16-digit hex number without any colon delimiters.

Table 2-4. Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Value
nvsram-offset	hexadecimal-literal
host-type	string-literal | integer-literal
nvsram-byte-setting	nvsram-value (0x hexadecimal | integer-literal)
nvsram-bit-setting	nvsram-mask, nvsram-value (0x hexadecimal, 0x hexadecimal | integer-literal)
ipv4-address	(0–255).(0–255).(0–255).(0–255)
ipv6-address	(0-FFFF):(0-FFFF):(0-FFFF): (0-FFFF):(0-FFFF):(0-FFFF): (0-FFFF):(0-FFF)
autoconfigure-vols-attr-value-list	*autoconfigure-vols-attr-value-pair* {*autoconfigure-vols-attr-value-pair*}
autoconfigure-vols-attr-value-pair	physicalDiskType=*physical disk-type*| raidLevel=*raid-level* | diskGroupWidth=*integer-literal* | diskGroupCount=*integer-literal* | virtualDisksPerGroupCount=*integer-literal*| hotSpareCount=*integer-literal* | segmentSize=*segment-size-spec* **NOTE:** The *physicalDiskType* parameter is not required if only one type of physical disk is in the storage array. If you use the *physicalDiskType* parameter, you must also use the *hotSpareCount* and *diskGroupWidth* parameters. If you do not use the *physicalDiskType* parameter, the configuration defaults to SAS physical disks. **NOTE:** The *virtualDisksPerGroupCount* parameter is the number of equal-capacity virtual disks per disk group.

Table 2-4. Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Value
create-virtual-disk-copy-attr-value-list	create-virtual-disk-copy-attr-value-pair {*create-virtual-disk-copy-attr-value-pair*}
create-virtual-disk-copy-attr-value-pair	copyPriority=highest | high | medium | low | lowest | targetReadOnlyEnabled=boolean
recover-raid-virtual-disk-attr-value-list	recover-raid-virtual-disk-attr-value-pair {*recover-raid-virtual-disk-attr-value-pair*}
recover-raid-virtual-disk-attr-value-pair	owner=(0 | 1)

Table 2-5. Range of Values for Recurring Syntax Elements

Recurring Syntax	Syntax Values
IPV4Priority	0 to 7
IPV4VlanID	1 to 4094
IPV6Priority	0 to 7
IPV6VlanID	1 to 4094
IPV6HopLimit	0 to 255 (default value is 64)
IPV6NdDetectDuplicateAddress	0 to 256
IPV6NdReachableTime	0 to 65535 (default value is 30000 milliseconds)
IPV6RetransmitTime	0 to 65535 (default value is 1000 milliseconds)
IPV6NDTimeOut	0 to 65535 (default value is 3000 milliseconds)

Table 2-5. Range of Values for Recurring Syntax Elements *(continued)*

Recurring Syntax	Syntax Values
maxFramePayload	1500 **NOTE:** The maxFramePayload parameter is shared between IPv4 and IPv6. The payload portion of a standard Ethernet frame is set at 1500 bytes, and a jumbo Ethernet frame is set at 9000 bytes. When using jumbo frames, make sure that all of the devices contained in the network path can handle the larger frame size.
tcpListeningPort (tcp-port-id)	3260, or 49,152 to 65,536 The default value is 3260.

Usage Guidelines

The following list provides guidelines for writing script commands on the command line:

- You must end all commands with a semicolon (;).
- You can enter more than one command on a line, but you must separate each command with a semicolon (;).
- You must separate each base command and its associated primary and secondary parameters with a space.
- The script engine is case sensitive.
- You can add comments to your scripts to make it easier for you and future users to understand the purpose of the script commands. (For information on how to add comments, see "Adding Comments to a Script File" on page 41.)

NOTE: While the CLI and script commands are not case sensitive, user labels (such as for virtual disk, hosts, or host ports) are case sensitive. If you try to map to an object identified by a user label, you must enter the user label exactly as it is defined, or the CLI and script commands will fail.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

NOTE: The *capacity* parameter returns an error if you specify a value greater than or equal to 10 without a space separating the numeric value and its unit of measure. (For example, `10GB` will return an error, but `10 GB` will not return an error).

Adding Comments to a Script File

You can add comments to a script file in three ways.

1. The script engine interprets as a comment any text typed after two forward slashes (//) until an end-of-line character is reached. If the script engine does not find an end-of-line character in the script after processing a comment, an error message is displayed, and the script operation is terminated. This error commonly occurs when a comment is placed at the end of a script and you did not press the Enter key.

   ```
   // Deletes the existing configuration.
   clear storageArray Configuration;
   ```

2. The script engine interprets any text typed between / * and * / as a comment. If the script engine does not find both a beginning and ending comment notation, an error message is displayed, and the script operation is terminated.

   ```
   /* Deletes the existing configuration */
   clear storageArray Configuration;
   ```

3. Use the **show** statement to embed comments in a script file that you want to display while the script file is running. Enclose the text you want to display in quotation marks (" ").

   ```
   show "Deletes the existing configuration";
   clear storageArray Configuration;
   ```

3

Configuring a Storage Array

This chapter explains how to run script commands from the command line to create a virtual disk from a group of physical disks and how to configure a redundant array of independent disks (RAID) storage array. This chapter assumes that you understand basic RAID concepts and terminology. Before configuring the storage array, become familiar with the concepts of physical disks, disk groups, virtual disks, host groups, hosts, and RAID controller modules. Additional information about configuring a storage array and related definitions is in the online help, the *Installation Guide*, the *MD Storage Manager User's Guide*, and the *Owner's Manual*.

Configuring a RAID storage array requires caution and planning to ensure that you define the correct RAID level and configuration for your storage array. The main purpose in configuring a storage array is to create virtual disks addressable by the hosts from a collection of physical disks. The commands described in this chapter enable you to set up and run a RAID storage array. Additional commands are also available to provide more control and flexibility. Many of these commands, however, require a deeper understanding of the firmware as well as various structures that need to be mapped. Use all of the command line interface (CLI) commands and script commands with caution.

The following sections in this chapter show some, but not all, of the CLI and script commands. The purpose of showing these commands is to explain how you can use the commands to configure a storage array. The presentation in this chapter does not explain all possible usage and syntax for the commands. (For complete definitions of the commands, including syntax, parameters, and usage notes, see "Script Commands" on page 105.)

This chapter contains examples of CLI and script command usage. The command syntax used in the examples is for a host running a Microsoft® Windows® operating system. As part of the examples, the complete C:\ prompt and DOS path for the commands are shown. Depending on your operating system, the prompt and path construct will vary.

For most commands, the syntax is the same for all Windows and Linux operating systems, as well as for a script file. Windows operating systems, however, have an additional requirement when entering names in a

command. On Windows, you must enclose the name between two backslashes (\) in addition to other delimiters. For example, the following name is used in a command that runs under Windows:

```
[\"Engineering\"]
```

For a Linux system when used in a script file, the name appears as:

```
["Engineering"]
```

Configuring a Storage Array

When you configure a storage array, you can maximize data availability by ensuring that data is quickly accessible while maintaining the highest level of data protection possible. The speed at which a host can access data is affected by the disk group RAID level and the segment size settings. Data protection is determined by the RAID level, hardware redundancy (such as global hot spares), and software redundancy (such as the Snapshot feature).

In general, you configure a storage array by defining the following entities:

- A disk group and associated RAID level
- The virtual disks
- Which hosts have access to the virtual disks

This section explains how to use the script commands to create a configuration from an array of physical disks.

Determining What is on Your Storage Array

Even when you create a configuration on a previously unconfigured storage array, you still need to determine the hardware and software features that must be included with the storage array. When you configure a storage array with an existing configuration, you must ensure that your new configuration does not inadvertently alter the existing configuration, unless you are reconfiguring the entire storage array. For example, to create a new disk group on unassigned physical disks, you must determine which physical disks are available. The commands described in this section enable you to determine the components and features in your storage array.

The **show storageArray** command returns the following general information about the components and properties of the storage array:

- A detailed profile of the components and features in the storage array
- The battery age
- The default host type (which is the current host type)
- Other available host types
- The hot spare locations
- The identifiers for enabled features
- The logical and physical component profiles
- The time to which both RAID controller modules are set
- The RAID controller module that currently owns each virtual disk in the storage array

To return the most information about the storage array, run the **show storageArray** command with the *profile* parameter. The following is an example of the complete CLI and script command running on Windows:

```
client>smcli 123.45.67.89 -c "show storageArray
profile;"
```

This example identifies the storage array by the dummy IP address 123.45.67.89. You can also identify the storage array by name.

The **show storageArray profile** command returns detailed information about the storage array. The information is presented in several screens on a display. You might need to increase the size of your display buffer to see all of the information. Because this information is so detailed, you might want to save the output to a file. To save the output to a file, enter the command as shown in the following example:

```
client>smcli 123.45.67.89 -c "show storageArray
profile;" -o c:\folder\storageArrayprofile.txt
```

In this example, the name **folder** is the folder in which you choose to place the profile file, and **storageArrayprofile.txt** is the name of the file. You can choose any folder and any file name.

NOTICE: When you write information to a file, the script engine does not check to determine if the file name already exists. If you choose the name of a file that already exists, the script engine writes over the information in the file without warning.

When you save the information to a file, you can use the information as a record of your configuration and as an aid during recovery.

To return a brief list of the storage array features and components, use the *summary* parameter. The command is similar to the following example:

```
client>smcli 123.45.67.89 -c "show storageArray
summary;"
```

The summary information is also returned as the first section of information when you use the *profile* parameter.

The following **show** commands return information about the specific components of a storage array. The information returned by each of these commands is the same as the information returned by the **show storageArray profile** command, but is constrained to the specific component. (The following commands are not complete commands. For information about a command, see the referenced section next to the command.)

- show controller ("RAID Controller Module Commands" on page 111)
- show physicalDisks ("Show Physical Disk" on page 205)
- show physicalDiskchannels stats ("Show Physical Disk Channel Statistics" on page 207)
- show storageArray hostTopology ("Show Storage Array Host Topology" on page 214)
- show storageArray lunmappings ("Show Storage Array LUN Mappings" on page 214)
- show allVirtualDisks ("Show Virtual Disk" on page 217)
- show diskGroup ("Show Disk Group" on page 204)
- show virtualDisk reservations ("Show Virtual Disk Reservations" on page 222)

The following commands also return information about a storage array:

- show storageArray autoConfigure ("Show Storage Array Autoconfigure" on page 212)
- show controller NVSRAM ("Show RAID Controller Module NVSRAM" on page 209)
- show storageArray unreadableSectors ("Show Storage Array Unreadable Sectors" on page 215)
- show virtualDiskCopy sourceCandidates ("Show Virtual Disk Copy Source Candidates" on page 220)
- show virtualDiskCopy targetCandidates ("Show Virtual Disk Copy Target Candidates" on page 220)
- show virtualDisk performanceStat ("Show Disk Group Import Dependencies" on page 220)

For descriptions of the **show** commands, including examples of the information returned by each command, see "Script Commands" on page 105. Other commands can help you learn about your storage array. To see a list of the commands, see "Commands Listed by Function" on page 108. These commands are organized by the storage array activities that the commands support. (Examples include virtual disk commands, host commands, enclosure commands, and others).

Saving a Configuration to a File

NOTICE: When you write information to a file, the script engine does not check to determine if the file name already exists. If you choose the name of a file that already exists, the script engine writes over the information in the file without warning.

After you have created a new configuration or if you want to copy an existing configuration for use on other storage arrays, you can save the configuration to a file. To save the configuration, use the **save storageArray configuration** command. Saving the configuration creates a script file that you can run on the command line. The following syntax is the general form of the command:

```
save storageArray configuration file="filename"
[(allconfig | globalSettings=(TRUE | FALSE)) |
virtualDiskConfigAndSettings=(TRUE | FALSE) |
hostTopology=(TRUE | FALSE) | lunMappings=(TRUE |
FALSE)]
```

You can choose to save the entire configuration or specific configuration features. The command for setting this parameter value looks like the following example:

```
client>smcli 123.45.67.89 -c "save storageArray
configuration file=
\"c:\folder\\storageArrayconfig1.scr\";"
```

In this example, the name **folder** is the folder in which you choose to place the configuration file, and **storageArrayconfig1.scr** is the name of the file. Choose any folder and any file name. MD Storage Manager uses the file extension **.scr** when it creates the configuration file.

Using the Create Virtual Disk Command

The **create virtualDisk** command enables you to create new virtual disks in the storage array in three ways:

- Create a new virtual disk while simultaneously creating a new disk group to which you assign the physical disks.
- Create a new virtual disk while simultaneously creating a new disk group to which the MD Storage Manager software assigns the physical disks.
- Create a new virtual disk in an existing disk group.

You must have unassigned physical disks in the disk group. You do not need to assign the entire capacity of the disk group to a virtual disk.

Creating Virtual Disks with User-Assigned Physical Disks

When you create a new virtual disk and assign the physical disks to use, the MD Storage Manager software creates a new disk group. The RAID controller module firmware assigns a disk group number to the new disk group. The following syntax is the general form of the command:

```
create virtualDisk physicalDisks=
(enclosureID0,slotID0...enclosureIDn,slotIDn)
raidLevel=0 | 1 | 5 | 6)userLabel=
"virtualDiskName" [capacity=virtualDiskCapacity
owner=(0 | 1) segmentSize=segmentSizeValue]
[enclosureLossProtect=(TRUE | FALSE)]
```

NOTE: The *capacity*, *owner*, *segmentSize*, and *enclosureLossProtect* parameters are optional. You can use one or all of the optional parameters as needed to help define your configuration. You do not, however, need to use any optional parameters.

The *userLabel* parameter is the name to give to the virtual disk. The virtual disk name can be any combination of alphanumeric characters, hyphens, and underscores. The maximum length of the virtual disk name is 30 characters. Spaces are not allowed. You must put quotation marks (" ") around the virtual disk name.

The *physicalDisks* parameter is a list of the physical disks that you want to use for the disk group. Enter the enclosure ID and slot ID of each physical disk that you want to use. Put parentheses around the list. Separate the enclosure ID and slot ID of a physical disk by a comma. Separate each enclosure ID and slot ID pair by a space. For example:

```
(0,0 0,1 0,2 0,3 0,4)
```

The *capacity* parameter defines the size of the virtual disk. You do not have to assign the entire capacity of the physical disks to the virtual disk. You can later assign any unused space to another virtual disk.

The *owner* parameter defines the RAID controller module to which you want to assign the virtual disk. If you do not specify a RAID controller module, the RAID controller module firmware determines the owner of the virtual disk.

The *segmentSize* parameter is the same as described for the **autoConfigure storageArray** command. See "Using the Auto Configure Command" on page 53.

The *enclosureLossProtect* parameter turns on or turns off enclosure loss protection for the disk group. (For a description of how enclosure loss protection works, see "Enclosure Loss Protection" on page 52.)

Example of Creating Virtual Disks with User-Assigned Physical Disks

```
client>smcli 123.45.67.89 -c "create virtualDisk
physicalDisks=(0,0 0,1 0,2) raidLevel=5 userLabel=
\"Engineering_1\" capacity=20 GB owner=0;"
```

NOTE: The *capacity* parameter returns an error if you specify a value greater than or equal to 10 without a space separating the numeric value and its unit of measure. (For example, `10GB` will return an error, but `10 GB` will not return an error).

The command in this example automatically creates a new disk group and a virtual disk with the name **Engineering_1**. The disk group will have a RAID level of 5 (RAID 5). The command uses three physical disks to construct the disk group. The virtual disk created has a capacity of 20 GB. If each physical disk has a capacity of 73 GB, the total capacity of the disk group is 219 GB. Because only 20 GB are assigned to the virtual disk, 199 GB remain available for other virtual disks that you can later add to this disk group. The segment size for each virtual disk is 64 KB. Hot spares have not been created for this new disk group. You must create hot spares after running this command.

Creating Virtual Disks with Software-Assigned Physical Disks

You can let the MD Storage Manager software assign the physical disks when you create the virtual disk. To have the software assign the physical disks, you need only specify the number of physical disks to use. The MD Storage Manager software then chooses the physical disks on which the virtual disk is created. The RAID controller module firmware assigns a disk group number to the new disk group. The following syntax is the general form for the command:

```
create virtualDisk physicalDiskCount=
numberOfPhysicalDisks raidLevel=(0 | 1 | 5 | 6)
userLabel="virtualDiskName" [physicalDiskType=
(SAS | SATA)] [capacity=virtualDiskCapacity |
owner=(0 | 1) | segmentSize=segmentSizeValue]
[enclosureLossProtect=(TRUE | FALSE)])
```

NOTE: The *physicalDiskType*, *capacity*, *owner*, *segmentSize*, and *enclosureLossProtect* parameters are optional. You can use one or all of the optional parameters as needed to help define your configuration. You do not, however, need to use any optional parameters.

This command is similar to the previous **create virtualDisk** command, which allows the user to assign the physical disks. This version of the command requires only the number and the type of physical disks to use in the disk group. You do not need to enter a list of physical disks. All other parameters are the same. Enclosure loss protection is performed differently when MD Storage Manager assigns the physical disks as opposed to when a user assigns the physical disks. (For an explanation of the difference, see "Enclosure Loss Protection" on page 52.)

Example of Creating Virtual Disks with Software-Assigned Physical Disks

```
client>smcli 123.45.67.89 -c "create virtualDisk
physicalDiskCount=3 raidLevel=5 userLabel=
\"Engineering_1"\ capacity=20 GB owner=0
segmentSize=64;"
```

The command in this example creates the same virtual disk as the previous **create virtualDisk** command, however, in this case the user does not know which physical disks are assigned to this disk group.

Creating Virtual Disks in an Existing Disk Group

To add a new virtual disk to an existing disk group, use the following command:

```
create virtualDisk DiskGroup=diskGroupNumber
userLabel="virtualDiskName" [freeCapacityArea=
freeCapacityIndexNumber | capacity=
virtualDiskCapacity | owner=(0 | 1) | segmentSize=
segmentSizeValue]
```

NOTE: The *freeCapacityArea*, *capacity*, *owner*, and *segmentSize* parameters are optional. You can use one or all optional parameters as needed to help define your configuration, though you do not need to use any of them.

The *diskGroup* parameter is the number of the disk group in which you want to create a new virtual disk. If you do not know the disk group numbers on the storage array, you can use the **show allVirtualDisks summary** command. This command displays a list of the virtual disks and the disk groups to which the virtual disks belong.

The *userLabel* parameter is the name you want to give to the virtual disk. The virtual disk name can be any combination of alphanumeric characters, hyphens, and underscores. The maximum length of the virtual disk name is 30 characters. You must enclose the virtual disk name with quotation marks (" ").

The *freeCapacityArea* parameter defines the free capacity area to use for the virtual disk. If a disk group has several free capacity areas, you can use this parameter to identify which free capacity area to use for virtual disk creation. You do not have to assign the entire capacity of the physical disks to the virtual disk. Assign any unused space to another virtual disk at another time.

The *userLabel*, *capacity*, *owner*, and *segmentSize* parameters are the same as in the previous versions of the **create virtualDisk** command.

Enclosure Loss Protection

The *enclosureLossProtect* parameter is a boolean switch that turns enclosure loss protection on or off. To work properly, each physical disk in a virtual disk group must be in a separate enclosure. Enclosure loss protection is set under the following conditions:

- You assign the physical disks.
- The RAID controller module assigns the physical disks.

The following table shows possible results for the *enclosureLossProtect* parameter. The results depend on whether you assign the physical disks or the RAID controller module assigns the physical disks.

Method	enclosureLossProtect= TRUE	enclosureLossProtect=FALSE
You assign the physical disks.	If you select more than one physical disk from any one enclosure, the storage array returns an error.	The storage array performs the operation, but the created disk group does not have enclosure loss protection.
The RAID controller module firmware assigns the physical disks.	The storage array posts an error if the RAID controller module firmware cannot provide physical disks to ensure that the new disk group has enclosure loss protection.	The storage array performs the operation even if it means that the disk group might not have enclosure loss protection.

The *enclosureLossProtect* parameter is not valid when creating virtual disks on existing disk groups.

Using the Auto Configure Command

The **autoConfigure storageArray** command creates the disk groups on a storage array, the virtual disks in the disk groups, and the hot spares for the storage array. When you use the **autoConfigure storageArray** command, define the following parameters:

- Type of physical disks (Serial Attached SCSI [SAS] or Serial Advanced Technology Attachment [SATA])
- RAID level
- Number of physical disks in a disk group
- Number of disk groups
- Number of virtual disks in each disk group
- Number of hot spares
- Size of each segment on the physical disks

After defining these parameters, the MD Storage Manager automatically creates the disk groups, virtual disks, and hot spares. The RAID controller modules assign disk group and virtual disk numbers as they are created. After MD Storage Manager creates the initial configuration, you can use the **set virtualDisk** command to define virtual disk labels.

Before running the **autoConfigure storageArray** command, run the **show storageArray autoConfigure** command. The **show storageArray autoConfigure** command returns a list of parameter values that MD Storage Manager will use to create a storage array. Change any of the parameter values by entering new values for the parameters when you run the **autoConfigure storageArray** command. If you are satisfied with the parameter values that the **show storageArray autoConfiguration** command returns, run the **autoConfigure storageArray** command without new parameter values.

The following syntax is the general form of **autoConfigure storageArray** command:

```
autoConfigure storageArray [physicalDiskType=
(SAS | SATA) raidLevel=(0 | 1 | 5 | 6) |
diskGroupWidth=numberOfPhysicalDisks |
diskGroupCount=numberOfDiskGroups |
virtualDisksPerGroupCount=
numberOfVirtualDisksPerGroup | hotSpareCount=
numberOfHotspares | segmentSize=segmentSizeValue]
```

NOTE: All parameters are optional. You can use one or all of the parameters as needed to define your configuration.

When you use the **autoConfigure storageArray** command without specifying the number of disk groups, the firmware determines how many virtual disks and disk groups to create. The firmware creates one disk group and one virtual disk up to the maximum number that the storage array can support. When you specify the number of disk groups, the firmware creates only that number of disk groups. When you create more than one disk group, all of the disk groups have the same number of physical disks and the same number of virtual disks.

- The *diskGroupWidth* parameter defines the number of unassigned physical disks wanted for each new disk group.
- The *diskGroupCount* parameter defines the number of new disk groups wanted in the storage array.
- The *virtualDisksPerGroupCount* parameter defines the number of virtual disks wanted in each disk group.
- The *hotSpareCount* parameter defines the number of hot spares wanted in each disk group.
- The *segmentSize* parameter defines the amount of data in kilobytes that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. The smallest units of storage are data blocks. Each data block stores 512 bytes of data. The size of a segment determines how many data blocks that it contains. An 8-KB segment holds 16 data blocks. A 64-KB segment holds 128 data blocks.

  Valid values for the segment size are 8, 16, 32, 64, 128, 256, and 512.

  When you enter a value for the segment size, the value is checked against the supported values provided by the RAID controller module at run time. If the value you enter is not valid, the RAID controller module returns a list of valid values.

If the virtual disk is for a single user with large I/O requests (such as multimedia), performance is maximized when a single I/O request can be serviced with a single data stripe. A data stripe is the segment size multiplied by the number of physical disks in the disk group that are used for data storage. In this environment, multiple physical disks are used for the same request, but each physical disk is accessed only once.

For optimal performance in a multi-user database or file system storage environment, set the segment size to minimize the number of physical disks needed to satisfy an I/O request. Using a single physical disk for a single request leaves other physical disks available to simultaneously service other requests.

After you have finished creating the disk groups and virtual disks by using the **autoConfigure storageArray** command, you can further define the properties of the virtual disks in a configuration using the **set virtualDisk** command. (See "Modifying Your Configuration" on page 55.)

Example of the Auto Configuration Command

```
client>smcli 123.45.67.89 -c "autoConfigure
storageArray physicalDiskType=SAS raidLevel=5
diskGroupWidth=8 diskGroupCount=3
virtualDisksPerGroupCount=4 hotSpareCount=2
segmentSize=8;"
```

The command in this example creates a storage array configuration that uses SAS physical disks set to RAID level 5. Three disk groups are created. Each disk group consists of eight physical disks configured into four virtual disks. The storage array has two hot spares, and segment size for each virtual disk is 8 KB.

Modifying Your Configuration

After creating your initial configuration, modify the properties of the configuration to ensure that it meets your requirements for data storage. Use the following commands to modify the properties of your configuration:

- **autoConfigure storageArray**
- **create virtualDisk**

Use the **set** commands to modify a storage array configuration. This section explains how to modify the following properties:

- Storage array password
- Simple Mail Transport Protocol (SMTP) and Simple Network Management Protocol (SNMP) alerts
- RAID controller module clocks

- Storage array host type
- Global hot spares

NOTE: Before modifying your configuration, save a copy of your current configuration to a file (see "Saving a Configuration to a File" on page 47). If you have problems with your modifications, you can use the information in the file to restore your previous configuration.

Setting the Storage Array Password

The **set storageArray** command enables you to define a password for a storage array. The following syntax is the general form of the command:

```
set storageArray password="password"
```

The *password* parameter defines a password for the storage array. Passwords provide added security to a storage array to reduce the possibility of implementing destructive commands.

NOTE: CLI commands do not have interactive warnings for destructive commands.

NOTICE: Implementing destructive commands can cause serious damage, including data loss.

Unless you define a password for the storage array, anyone can run all of the script commands. A password protects the storage array from any command that the RAID controller modules consider destructive. A destructive command is any command that can change the state of the storage array, such as virtual disk creation, reset, delete, rename, or change. If you have more than one storage array in a storage configuration, each array has a separate password. Passwords can have a maximum length of 30 characters. You must put quotation marks (" ") around the password. The following example shows how to use the **set storageArray** command to define a password:

```
client>smcli 123.45.67.89 -c "set storageArray
password=\"1a2b3c4d5e"\;"
```

Setting Up SMTP and SNMP Alerts

The storage array can be set up to send automatic email alert messages to specified email addresses when specific events occur. View the current alert configuration settings using the following command:

```
SMcli -d -i -s -w -v -S
```

By default, all alert configuration settings are **None**.

The following example shows how to set the mail server IP and the sender address configurations for SMTP alerts:

```
SMcli -m 123.45.67.89 -F
MyStorageArrayEvent@MyCompany.com
```

or

```
SMcli -m MyCompany.com -F
MyStorageArrayEvent@MyCompany.com
```

An example of a command to set the email alert destination and specify that only event information is to be sent is:

```
SMcli -a email:MyCompanySupport@MyCompany.com
123.45.67.89 -I eventOnly
```

The following example shows how to set the SNMP trap alert configuration. In this example, the trap destination is 123.45.67.891. The storage array is 123.45.67.892, and the community name is **public**.

```
SMcli -a trap:public, 123.45.67.891 123.45.67.892
```

Setting the RAID Controller Module Clocks

To synchronize the clocks on the RAID controller modules with the host, use the **set storageArray time** command. Running this command helps ensure that event timestamps written by RAID controller modules to the Major Event Log (MEL) match event timestamps written to the host log files. The RAID controller modules remain available during synchronization. An example of the command is:

```
client>smcli 123.45.67.89 -c "set storageArray
time;"
```

Setting the Storage Array Host Type

The **set storageArray** command enables you to define the default host type. The following syntax is the general form of the command:

```
set storageArray defaultHostType=(hostTypeName |
hostTypeIdentifier)
```

The *defaultHostType* parameter defines how the RAID controller modules communicate with the operating system on undefined hosts connected to the storage array. This parameter defines the host type only for storage array data I/O activities; it does not define the host type for the management station. The operating system can be Windows or Linux. For example, if you set the *defaultHostType* to Linux, the RAID controller module communicates with any undefined host if the undefined host is running Linux. Typically, you need to change the host type only when you are setting up the storage array. The only time you might need to use this parameter is if you need to change how the storage array behaves relative to the hosts.

Before you can define the default host type, you need to determine what host types are connected to the storage array. To return information about host types connected to the storage array, you can use the **show storageArray** command with the *defaultHostType* parameter or *hostTypeTable* parameter. This command returns a list of the host types with which the RAID controller modules can communicate; it does not return a list of the hosts. The following examples show how to use the *defaultHostType* parameter and the *hostTypeTable* parameter:

```
client>smcli 123.45.67.89 -c "show storageArray
defaultHostType;"

client>smcli 123.45.67.89 -c "show storageArray
hostTypeTable;"
```

The following example shows how to define a specific default host type:

```
client>smcli 123.45.67.89 -c "set storageArray
defaultHostType=11;"
```

The value 11 is the host type index value from the host type table.

Setting Modification Priority

Modification priority defines how much processing time is allocated for virtual disk modification operations. Time allocated for virtual disk modification operations affects system performance. Increases in virtual disk modification priority can reduce read/write performance. Operations affected by modification priority include:

- Copyback
- Reconstruction

- Initialization
- Changing segment size
- Defragmentation of a disk group
- Adding free capacity to a disk group
- Changing the RAID level of a disk group

The lowest priority rate favors system performance, but the modification operation takes longer. The highest priority rate favors the modification operation, but the system performance might be degraded.

The **set virtualDisk** command enables you to define the modification priority for a virtual disk. The following syntax is the general form of the command:

```
set (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks [virtualDiskName1
... virtualDiskNamen] | virtualDisk <wwid> |
accessVirtualDisk) modificationPriority=(highest |
high | medium | low | lowest)
```

The following example shows how to use this command to set the modification priority for virtual disks named **Engineering 1** and **Engineering 2**:

```
client>smcli 123.45.67.89 -c "set virtualDisks
[\"Engineering_1\" \"Engineering_2\"]
modificationPriority=lowest;"
```

The modification rate is set to **lowest** so that system performance is not significantly reduced by modification operations.

Assigning Global Hot Spares

Hot spare physical disks can replace any failed physical disk in the storage array. The hot spare must be the same type of physical disk as the physical disk that failed and must have capacity greater than or equal to any physical disk that can fail. If a hot spare is smaller than a failed physical disk, the hot spare cannot be used to rebuild the data from the failed physical disk. Hot spares are available only for RAID levels 1 or 5.

You can assign or unassign global hot spares by using the **set physicalDisk** command. To use this command, you must perform these steps:

1. Identify the location of the physical disks by enclosure ID and slot ID.
2. Set the *hotSpare* parameter to **TRUE** to enable the hot spare or **FALSE** to disable an existing hot spare.

The following syntax is the general form of the command:

```
set (physicalDisk [enclosureID,slotID] |
physicalDisks [enclosureID0,slotID0 ...
enclosureIDn,slotIDn] hotSpare=(TRUE | FALSE)
```

The following example shows how to use this command to set hot spare physical disks:

```
client>smcli 123.45.67.89 -c "set physicalDisks
[0,2 0,3] hotSpare=TRUE;"
```

Enter the enclosure ID and slot ID of each physical disk that you want to use. You must put brackets ([ ]) around the list. Separate the enclosure ID and slot ID of a physical disk by a comma. Separate each enclosure ID and slot ID pair by a space.

4

Using the Snapshot Feature

This chapter describes how the Snapshot feature works, lists the snapshot script commands, and explains how to use the commands to create snapshot virtual disks. Additional information about the Snapshot feature and related definitions is available in the online help, the *Installation Guide*, the *MD Storage Manager User's Guide*, and the *Owner's Manual*.

The Snapshot feature creates a snapshot virtual disk that you can use as a backup of your data. A snapshot virtual disk is a logical point-in-time image of a standard virtual disk. Because it is not a physical copy, a snapshot virtual disk is created more quickly than a physical copy and requires less physical disk space. Typically, you create a snapshot virtual disk so that an application, such as a backup application, can access the snapshot virtual disk. The application reads the data while the source virtual disk remains online and user accessible. You can also create several snapshot virtual disks of a source virtual disk and write data to the snapshot virtual disks to perform testing and analysis.

NOTE: If you ordered Premium Features for the Snapshot Virtual Disks, you received a Premium Features Activation card shipped in the same box as your Dell PowerVault™ MD storage array. Follow the directions on the card to obtain a key file and to enable the feature. For more information, see "Premium Feature — Snapshot Virtual Disks" in the *User's Guide*.

Snapshot virtual disks allow you to perform the following tasks:

- Create a complete image of the data on a source virtual disk at a particular point in time.
- Use only a small amount of disk space.
- Provide quick, frequent, nondisruptive backups; or test new versions of a database system without affecting actual data.
- Provide for snapshot virtual disks to be read, written, and copied.
- Use the same availability characteristics of the source virtual disk (such as redundant array of independent disks (RAID) protection and redundant path failover).

- Map the snapshot virtual disk and make it accessible to any host on a storage area network. You can make snapshot data available to secondary hosts for read and write access by mapping the snapshot to the hosts.
- Create up to four snapshots per virtual disk.

  NOTE: The maximum number of snapshot virtual disks is one-half of the total number of virtual disks supported by the RAID controller module.
- Increase the capacity of a snapshot virtual disk.

Table 4-1 lists the components that comprise a snapshot virtual disk and briefly describes what they do.

Component	Description
Source virtual disk	Standard virtual disk from which the snapshot is created
Snapshot virtual disk	Point-in-time image of a standard virtual disk
Snapshot repository virtual disk	Virtual disk that contains snapshot metadata and copy-on-write data for a particular snapshot virtual disk

Table 4-1 lists the snapshot virtual disk commands and brief descriptions of what the commands do.

Table 4-1. Snapshot Virtual Disk Commands

Command	Description
create snapshotVirtualDisk	Creates a snapshot virtual disk.
re-create snapshot	Starts a fresh copy-on-write operation by using an existing snapshot virtual disk.
set (snapshotVirtualDisk)	Defines the properties for a snapshot virtual disk and enables you to rename a snapshot virtual disk.
stop snapshot	Stops a copy-on-write operation.

Using Host Servers to Create an Initial Snapshot Virtual Disk

NOTICE: Before using the Snapshot Virtual Disks Premium Feature in a Microsoft® Windows® clustered configuration, you must first map the snapshot virtual disk to the cluster node that owns the source virtual disk. This ensures that the cluster nodes correctly recognize the snapshot virtual disk.

If you map the snapshot virtual disk to the node that does not own the source virtual disk before the snapshot enabling process is completed, the operating system may fail to correctly identify the snapshot virtual disk. This can result in data loss on the source virtual disk or an inaccessible snapshot.

For details on mapping the snapshot virtual disk to the secondary node, refer to the *Dell PowerEdge™ Cluster SE600W Systems Installation and Troubleshooting Guide* on **support.dell.com**

NOTE: You can create concurrent snapshots of a source virtual disk on both the source disk group and on another disk group.

Before creating a Snapshot Virtual Disk, note the following:

- The following types of virtual disks are not valid source virtual disks: snapshot repository virtual disks, snapshot virtual disks, target virtual disks that are participating in a virtual disk copy.
- You cannot create a snapshot of a virtual disk that contains unreadable sectors.
- You must satisfy the requirements of your host operating system for creating snapshot virtual disks. Failure to meet the requirements of your host operating system results in an inaccurate point-in-time image of the source virtual disk or the target virtual disk in a virtual disk copy.

Creating a Snapshot Virtual Disk

The **create snapshotVirtualDisk** command provides three methods for defining the physical disks for your snapshot repository virtual disk:

- Define each physical disk for the snapshot repository virtual disk by enclosure ID and slot ID.
- Define a disk group in which the snapshot repository virtual disk resides. Optionally define the capacity of the repository virtual disk.

- Define the number of physical disks, but not specific physical disks, for the repository virtual disk.

When using the **create snapshotVirtualDisk** command to create a snapshot virtual disk, the standard virtual disk name for the source virtual disk is the minimum information required. When you provide only the standard virtual disk name, the storage management software provides default values for the other required property parameters for a snapshot virtual disk.

NOTE: In some cases, depending on the host operating system and any virtual disk manager software in use, the software prevents you from mapping the same host to both a source virtual disk and its associated snapshot virtual disk.

An error message appears in the command line when the utility cannot distinguish between the following:

- Source virtual disk and snapshot virtual disk (for example, if the snapshot virtual disk has been removed)
- Standard virtual disk and virtual disk copy (for example, if the virtual disk copy has been removed)

If you are running a Linux operating system, run the **hot_add** utility to register the snapshot virtual disk with the host operating system.

NOTE: The hot_add utility is not available for Windows.

Enabling the Snapshot Virtual Disk Feature

The first step in creating a snapshot virtual disk is to make sure the feature is enabled on the storage array. You need a feature key to enable the feature. The command for enabling the feature key file is:

```
enable storageArray feature file="filename"
```

where the *file* parameter is the complete file path and file name of a valid feature key file. Enclose the file path and file name in quotation marks (" "). Valid file names for feature key files usually end with **.key** extension.

Creating a Snapshot Virtual Disk with User-Assigned Physical Disks

Creating a snapshot virtual disk by assigning the physical disks allows you to choose from the available physical disks when defining your storage array configuration. When you choose the physical disks for your snapshot virtual disk, you automatically create a new disk group. You can specify which physical disks to use and the RAID level for the new disk group.

Preparing Host Servers to Create an Initial Snapshot Virtual Disk

NOTICE: Before you create a new point-in-time image of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer®, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk(s) in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the Snapshot.

Before creating a snapshot virtual disk, the server has to be in the proper state. To ensure that the host server is properly prepared to create a snapshot virtual disk, you can either use an application to carry out this task, or you can perform the following steps:

1. Stop all I/O activity to the source.
2. Using your Windows system, flush the cache to the source. At the host prompt, type

   `SMrepassist -f <filename-identifier>`

   and press <Enter>. See "SMrepassist Utility" in the *User's Guide* for more information.
3. Remove the drive letter(s) of the source in Windows or unmount the virtual drive(s) in Linux to help guarantee a stable copy of the drive for the Snapshot. If this is not done, the snapshot operation will report that it has completed successfully, but the snapshot data will not be updated properly.

   NOTE: Verify that the virtual disk has a status of Optimal or Disabled by clicking the Summary tab and then clicking the Disk Groups & Virtual Disks link.
4. Follow any additional instructions for your operating system. Failure to follow these additional instructions can create unusable snapshot virtual disks.

NOTE: If your operating system requires additional instructions, you can find those instructions in your operating system documentation.

If you want to use a snapshot regularly, such as for backups, use the Disable Snapshot and Re-create Snapshot options to reuse the snapshot. Disabling and re-creating snapshots preserves the existing virtual disk-to-host mappings to the snapshot virtual disk.

After your server has been prepared, see "Creating the Initial Snapshot Virtual Disk" on page 66.

Creating the Initial Snapshot Virtual Disk

After first preparing the host server(s) as specified in the preceding procedure, use the following examples to make a virtual disk snapshot.

The following syntax is the general form of the command to create a snapshot virtual disk:

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryRAIDLevel=(0 |
1 | 5) (repositoryPhysicalDisks=
(enclosureID0,slotID0 ... enclosureIDn,slotIDn)
userLabel="snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot)] [enclosureLossProtect=(TRUE | FALSE)]
```

NOTE: Use one or all of the optional parameters as needed to help define your configuration. You do not, however, need to use any optional parameters.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Create an Initial Snapshot Virtual Disk" on page 65. The following example shows a command in which users assign the physical disks:

```
client>smcli 123.45.67.89 -c "create
snapshotVirtualDisk sourceVirtualDisk=
\"Mars_Spirit_4\" repositoryRAIDLevel=5
repositoryPhysicalDisks=(1,1 1,2 1,3 1,4 1,5);"
```

The command in this example creates a new snapshot of the source virtual disk **Mars_Spirit_4**. The snapshot repository virtual disk consists of five physical disks that form a new disk group. The new disk group has a RAID level of 5. This command also takes a snapshot of the source virtual disk, starting the copy-on-write operation.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Create an Initial Snapshot Virtual Disk" on page 65. The following example is the script file version of the command:

```
create snapshotVirtualDisk sourceVirtualDisk=
"Mars_Spirit_4" repositoryRAIDLevel=5
repositoryPhysicalDisks=(1,1 1,2 1,3 1,4 1,5);
```

A minimal version of this command might look like the following example:

```
client>smcli 123.45.67.89 -c "create
snapshotVirtualDisk sourceVirtualDisk=
\"Mars_Spirit_4\";"
```

The command in this example creates a new snapshot for the source virtual disk **Mars_Spirit_4**. The repository virtual disk is created in the same disk group as the source virtual disk, which means that the repository virtual disk has the same RAID level as the source virtual disk. This command starts the copy-on-write operation.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Create an Initial Snapshot Virtual Disk" on page 65. The following example is the script file version of the command:

```
create snapshotVirtualDisk sourceVirtualDisk=
"Mars_Spirit_4";
```

Creating a Snapshot Virtual Disk with Software-Assigned Physical Disks

This version of the **create snapshotVirtualDisk** command lets you choose an existing disk group in which to place the snapshot repository virtual disk. The storage management software determines which physical disks to use. You can also define how much space to assign to the repository virtual disk. Because you are using an existing disk group, the RAID level for the snapshot virtual disk defaults to the RAID level of the disk group in which you place it. You cannot define the RAID level for the snapshot virtual disk. The general syntax for this command is:

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryDiskGroup=
diskGroupNumber freeCapacityArea=
freeCapacityIndexNumber userLabel=
```

```
"snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot)] [enclosureLossProtect=(TRUE |
FALSE)]
```

NOTE: Use one or all of the optional parameters as needed to define your configuration. It is not necessary to use any optional parameters.

The following example is a command in which software assigns the physical disks:

```
client>smcli 123.45.67.89 -c "create
snapshotVirtualDisk sourceVirtualDisk=
\"Mars_Spirit_4\" repositoryDiskGroup=2
freeCapacityArea=2;"
```

The command in this example creates a new snapshot repository virtual disk in disk group 2. The source virtual disk is **Mars_Spirit_4**. The size of the snapshot repository is 4 GB. This command also takes a snapshot of the source virtual disk, which starts the copy-on-write operation.

Define the capacity of a snapshot repository virtual disk as any percentage of the size of the source virtual disk. A value of 20 percent is a good compromise between size and speed. In the previous example, the size of the snapshot repository is set to 4 GB. The underlying assumption is that the source virtual disk size is 20 GB (0.2 x 20 GB = 4 GB).

The following example is the script file version of the command:

```
create snapshotVirtualDisk sourceVirtualDisk=
"Mars_Spirit_4" repositoryDiskGroup=2
freeCapacityArea=2;
```

Creating a Snapshot Virtual Disk by Specifying a Number of Physical Disks

With this version of the **create snapshotVirtualDisk** command, you must specify the number of physical disks and the RAID level for the snapshot repository virtual disk. This version of the **create snapshotVirtualDisk** command creates a new disk group. You must have physical disks in the storage array that are not assigned to a disk group for this command to work:

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryRAIDLevel=(0 |
1 | 5 | 6) repositoryPhysicalDiskCount=
numberOfPhysicalDisks physicalDiskType=(SAS |
SATA) userLabel="snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot)] [enclosureLossProtect=(TRUE |
FALSE)]
```

NOTE: Use one or all optional parameters as needed to define your configuration. It is not necessary to use any optional parameters.

The following example is a command in which users specify the number of physical disks:

```
client>smcli 123.45.67.89 -c "create
snapshotVirtualDisk sourceVirtualDisk=
\"Mars_Spirit_4\" repositoryRAIDLevel=5
repositoryPhysicalDiskCount=3;"
```

The command in this example creates a new snapshot repository virtual disk that consists of three physical disks. The three physical disks comprise a new disk group with a RAID level of 5. This command also takes a snapshot of the source virtual disk, which starts the copy-on-write operation.

The following example is the script file version of the command:

```
create snapshotVirtualDisk sourceVirtualDisk=
"Mars_Spirit_4" repositoryRAIDLevel=5
repositoryPhysicalDiskCount=3;
```

User-Defined Parameters

Parameters for the **create snapshotVirtualDisk** command enable you to define the snapshot virtual disk to suit the requirements of your storage array. Table 4-2 lists the parameters and descriptions of what the parameters do.

Table 4-2. Snapshot Virtual Disk Parameters

Parameter	Description
physicalDiskType	Specifies the type of physical disk to use for the snapshot repository virtual disk. The choice is either Serial Attached SCSI (SAS) or Serial Advanced Technology Attachment (SATA). This parameter works only with the count-based repository method of defining a snapshot virtual disk.
repositoryDiskGroup	Specifies the disk group in which to build the snapshot virtual disk. Default builds the snapshot repository virtual disk in the same disk group as the source virtual disk.
freeCapacityArea	Specifies the amount of storage space to use for the snapshot repository virtual disk. Free storage space is defined in units of bytes, kilobytes, megabytes, or gigabytes.
userLabel	Specifies the name to give to the snapshot virtual disk. If you do not choose a name for the snapshot virtual disk, the RAID controller modules create a default name using the source virtual disk name. For example, if the source virtual disk name is **Mars_Spirit_4** and it does not have a snapshot virtual disk, the default snapshot virtual disk name is **Mars_Spirit_4-1**. If the source virtual disk already has $n - 1$ number of snapshot virtual disks, the default name is **Mars_Spirit_4-***n*.

Table 4-2. Snapshot Virtual Disk Parameters *(continued)*

Parameter	Description
repositoryUserLabel	Specifies the name to give to the snapshot repository virtual disk. If you do not choose a name for the snapshot repository virtual disk, the RAID controller modules create a default name using the source virtual disk name. For example, if the source virtual disk name is **Mars_Spirit_4** and it does not have an associated snapshot repository virtual disk, the default snapshot repository virtual disk name is **Mars_Spirit_4-R1**. If the source virtual disk already has *n* – 1 number of snapshot repository virtual disks, the default name is **Mars_Spirit_4-R***n*.
warningThresholdPercent	Specifies how full to allow the snapshot repository virtual disk to get before sending a warning that the snapshot repository virtual disk is close to capacity. The warning value is a percentage of the total capacity of the snapshot repository virtual disk. The default value is 50, which represents 50 percent of total capacity. (Change this value using the **set snapshotVirtualDisk** command.)
repositoryPercentOfSource	Specifies the size of the snapshot repository virtual disk as a percentage of the source virtual disk size. The default value is 20, which represents 20 percent of the source virtual disk size.
repositoryFullPolicy	Specifies how snapshot processing continues if the snapshot repository virtual disk is full. You can choose to fail writes to the source virtual disk (**failSourceWrites**) or fail writes to the snapshot virtual disk (**failSnapShot**). The default value is **failSnapShot**.

The following example of the **create snapshotVirtualDisk** command includes user-defined parameters:

```
client>smcli 123.45.67.89 -c "create
snapshotVirtualDisk sourceVirtualDisk=
\"Mars_Spirit_4\" repositoryRAIDLevel=5
repositoryPhysicalDiskCount=5 physicalDiskType=
SAS userLabel=\"Mars_Spirit_4_snap1\"
```

```
repositoryUserLabel=\"Mars_Spirit_4_rep1\"
warningThresholdPercent=75
repositoryPercentOfSource=40
repositoryFullPolicy=failSnapShot;"
```

The following example is the script file version of the command:

```
create snapshotVirtualDisk sourceVirtualDisk=
"Mars_Spirit_4" repositoryRAIDLevel=5
repositoryPhysicalDiskCount=5 physicalDiskType=
SAS userLabel="Mars_Spirit_4_snap1"
repositoryUserLabel="Mars_Spirit_4_rep1"
warningThresholdPercent=75
repositoryPercentOfSource=40
repositoryFullPolicy=failSnapShot;
```

NOTE: In the previous examples, the names for the snapshot virtual disk and repository virtual disk are defined by the user. If you do not choose to create names for the snapshot virtual disks or the repository virtual disks, the RAID controller modules provide default names. (See "Names of Snapshot Virtual Disks and Repository Virtual Disks" on page 72 for an explanation of naming conventions.)

Names of Snapshot Virtual Disks and Repository Virtual Disks

The names of snapshot virtual disks and repository virtual disks can be any combination of alphanumeric characters, hyphens, and underscores. The maximum length of the virtual disk names is 30 characters. You must enclose the name in quotation marks. The character string cannot contain a new line. Make sure that you use unique names or the RAID controller module firmware returns an error.

One technique for naming the snapshot virtual disk and the repository virtual disk is to add a hyphenated suffix to the original name of the source virtual disk. The suffix distinguishes between the snapshot virtual disk and the repository virtual disk. For example, if you have a source virtual disk with a name **Engineering Data**, the snapshot virtual disk can have a name **Engineering Data-S1**. The repository virtual disk can have a name of **Engineering Data-R1**.

If you do not choose a unique name for the either the snapshot virtual disk or repository virtual disk, the RAID controller modules create a default name by using the name of the source virtual disk. For example, if the name of the source virtual disk is **aaa** and it does not have a snapshot virtual disk, then the

default name is **aaa-1**. If the source virtual disk already has *n* – 1 number of snapshot virtual disks, then the default name is **aaa-n**. Similarly, if the name of the source virtual disk is **aaa** and it does not have a repository virtual disk, then the default repository virtual disk name is **aaa-R1**. If the source virtual disk already has *n* – 1 number of repository virtual disks, then the default name is **aaa-R***n*.

In the examples from the previous section, the user-defined name of the snapshot virtual disk was **Mars_Spirit_4_snap1**. The user-defined name of the repository virtual disk was **Mars_Spirit_4_rep1**. The default name provided by the RAID controller module for the snapshot virtual disk would be **Mars_Spirit_4-1**. The default name provided by the RAID controller module for the repository virtual disk would be **Mars_Spirit_4-R1**.

Changing Snapshot Virtual Disk Settings

The **set (snapshot) virtualDisk** command enables you to change the property settings for a snapshot virtual disk. Using this command, you can change the following parameters:

- Name of the snapshot virtual disk
- Warning threshold percent
- Repository full policy

The following example shows the command to change the name of a snapshot virtual disk:

```
client>smcli 123.45.67.89 -c "set virtualDisk
[\"Mars_Spirit_4-1\"] userLabel=\"Mars_Odyssey_3-
2\";"
```

The following example is the script file version of the command:

```
set virtualDisk ["Mars_Spirit_4-1"] userLabel=
"Mars_Odyssey_3-2";
```

When you change the warning threshold percent and repository full policy, you can apply the changes to one or several snapshot virtual disks. The following example uses the **set (snapshot) virtualDisk** command to change these properties on more than one snapshot virtual disk:

```
client>smcli 123.45.67.89 -c "set virtualDisks
[\"Mars_Spirit_4-1\" \"Mars_Spirit_4-2\"
\"Mars_Spirit_4-3\"] warningThresholdPercent=50
repositoryFullPolicy=failSourceWrites;"
```

The following example is the script file version of the command:

```
set virtualDisks ["Mars_Spirit_4-1"
"Mars_Spirit_4-2" "Mars_Spirit_4-3"]
warningThresholdPercent=50 repositoryFullPolicy=
failSourceWrites;
```

Stopping and Deleting a Snapshot Virtual Disk

When you create a snapshot virtual disk, copy-on-write immediately starts running. As long as a snapshot virtual disk is enabled, storage array performance is affected by the copy-on-write operations to the associated snapshot repository virtual disk. If you no longer want copy-on-write operations to run, you can use the **stop snapshot virtualDisk** command to stop the copy-on-write operations. When you stop a snapshot virtual disk, the snapshot virtual disk and the repository virtual disk are still defined for the source virtual disk; only copy-on-write has stopped. The following example stops a snapshot virtual disk:

```
client>smcli 123.45.67.89 -c "stop snapshot
virtualDisks [\"Mars_Spirit_4-2\" \"Mars_Spirit_4-
3\"];"
```

The following example is the script file version of the command:

```
stop snapshot virtualDisks ["Mars_Spirit_4-2"
"Mars_Spirit_4-3"];
```

When you stop the copy-on-write operations for a specific snapshot virtual disk, only that snapshot virtual disk is disabled. All other snapshot virtual disks remain in operation.

Re-creating the Snapshot Virtual Disk

To restart a copy-on-write operation, use the **recreate snapshot virtualDisk** command. This command starts a fresh copy-on-write operation using an existing snapshot virtual disk. When you restart a snapshot virtual disk, the snapshot virtual disk must have either an Optimal or a Disabled state. The following conditions then occur:

- All copy-on-write data previously on the snapshot repository virtual disk is deleted.
- Snapshot virtual disk and snapshot repository virtual disk parameters remain the same as the previously disabled snapshot virtual disk and snapshot repository virtual disk. You can also change the *userLabel*, *warningThresholdPercent*, and *repositoryFullPolicy* parameters when you restart the snapshot virtual disk.
- The original names for the snapshot repository virtual disk are retained.

Preparing Host Servers to Re-create a Snapshot Virtual Disk

NOTICE: Before you create a new point-in-time image of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk and snapshot virtual disk to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the Snapshot.

Before re-creating a snapshot virtual disk, both the server and the associated virtual disk you are re-creating have to be in the proper state. To ensure that the host server is properly prepared to re-create a snapshot virtual disk, you can either use an application to carry out this task, or you can perform the following steps:

1. Stop all I/O activity to the source and snapshot virtual disk (if mounted).
2. Using your Windows system, flush the cache to both the source and the snapshot virtual disk (if mounted). At the host prompt, type

   `SMrepassist -f <filename-identifier>`

   and press <Enter>. See "SMrepassist Utility" in the *User's Guide* for more information.

3 Remove the drive letter(s) of the source and (if mounted) snapshot virtual disk in Windows or unmount the virtual drive(s) in Linux to help guarantee a stable copy of the drive for the Snapshot. If this is not done, the snapshot operation will report that it has completed successfully, but the snapshot data will not be updated properly.

4 Follow any additional instructions for your operating system. Failure to follow these additional instructions can create unusable snapshot virtual disks.

NOTE: If your operating system requires additional instructions, you can find those instructions in your operating system documentation.

After your server has been prepared, see "Re-creating the Snapshot Virtual Disk" on page 75 to re-create the snapshot virtual disk.

Re-creating a Snapshot Virtual Disk

After first preparing the host server(s) as specified in the preceding procedure, use the following examples to re-create a virtual disk snapshot.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Re-create a Snapshot Virtual Disk" on page 75. The following example shows the command to restart a snapshot virtual disk:

```
client>smcli 123.45.67.89 -c "recreate snapshot
virtualDisks [\"Mars_Spirit_4-2\" \"Mars_Spirit_4-
3\"];"
```

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Re-create a Snapshot Virtual Disk" on page 75. The following example is the script file version of the command:

```
recreate snapshot virtualDisks ["Mars_Spirit_4-2"
"Mars_Spirit_4-3"];
```

If you do not intend to use a snapshot virtual disk again, you can delete the snapshot virtual disk using the **delete virtualDisk** command. When you delete a snapshot virtual disk, the associated snapshot repository virtual disk is also deleted.

5

Using the Virtual Disk Copy Feature

This chapter describes how the Virtual Disk Copy feature works, lists the script commands for Virtual Disk Copy, and explains how to use the commands to create and run Virtual Disk Copy. Additional information about Virtual Disk Copy and related definitions is available in the online help, the *Installation Guide*, the *MD Storage Manager User's Guide*, and the *Owner's Manual*.

NOTE: If you ordered Premium Features for Virtual Disk Copy, you received a Premium Features Activation card shipped in the same box as your Dell PowerVault™ MD storage array. Follow the directions on the card to obtain a key file and to enable the feature. For more information, see "Premium Feature — Virtual Disk Copy" in the *User's Guide*.

The Virtual Disk Copy feature enables you to copy data from one virtual disk (the source) to another virtual disk (the target) in a single storage array. You can use this feature to perform the following functions:

- Back up data.
- Copy data from disk groups that use smaller capacity physical disks to disk groups using larger capacity physical disks.
- Restore snapshot virtual disk data to the associated source virtual disk.

NOTE: The preferred method is to perform a Virtual Disk Copy from a Snapshot Virtual Disk. This allows the original virtual disk used in the Snapshot operation to remain in full use while the Snapshot of this virtual disk is used as the source for the virtual disk copy operation.

NOTE: The Virtual Disk Copy for any Virtual Disk cannot be mounted on the same host as the source Virtual Disk. The Microsoft® Windows® operating system does not allow assigning a drive letter to the Virtual Disk Copy.

Table 5-1 lists the Virtual Disk Copy commands and briefly describes what the commands do.

Table 5-1. Virtual Disk Copy Commands

Command	Description
create virtualDiskCopy	Creates a virtual disk copy and starts the virtual disk copy operation.
disable storageArray feature=virtualDiskCopy	Turns off the current virtual disk copy operation.
enable storageArray feature	Activates the Virtual Disk Copy feature.
recopy virtualDiskCopy	Re-initiates a virtual disk copy operation by using an existing virtual disk copy pair.
remove virtualDiskCopy	Removes a virtual disk copy pair.
set virtualDiskCopy	Defines the properties for a virtual disk copy pair.
show virtualDiskCopy	Returns information about virtual disk copy operations. You can retrieve information about a specific virtual disk copy pair, or all virtual disk copy pairs in the storage array.
show virtualDiskCopy sourceCandidates	Returns information about the candidate virtual disks that you can use as the source for a virtual disk copy operation.
show virtualDiskCopy targetCandidates	Returns information about the candidate virtual disks that you can use as the target for a virtual disk copy operation.
stop virtualDiskCopy	Stops a virtual disk copy operation.

Creating a Virtual Disk Copy

Before creating a virtual disk copy, ensure that a suitable target virtual disk exists on the storage array, or create a new target virtual disk specifically for the virtual disk copy. The target virtual disk must have a capacity equal to or greater than the source virtual disk.

You can have a maximum of eight virtual disk copies in progress at one time. Any virtual disk copy greater than eight has a status of Pending until one of the virtual disk copies with a status of In Progress completes.

The following steps show the general process for creating a virtual disk copy:

1. Enable the Virtual Disk Copy feature.
2. Determine candidates for a virtual disk copy.
3. Create the target virtual disk and source virtual disk for a virtual disk copy.

Enabling the Virtual Disk Copy Feature

The first step in creating a virtual disk copy is to make sure the feature is enabled on the storage array. You need a feature key to enable the feature. To enable the feature key file, use the command:

```
enable storageArray feature file="filename"
```

where the **file** parameter is the complete file path and file name of a valid feature key file. Enclose the file path and file name in quotation marks (" "). Valid file names for feature key files usually end with a **.key** extension.

Determining Virtual Disk Copy Candidates

All virtual disks might not be available for use in virtual disk copy operations. To determine which candidate virtual disks on the storage array can be used as a source virtual disk, use the **show virtualDiskCopy sourceCandidates** command. To determine which candidate virtual disks on the storage array can be used as a target virtual disk, use the **show virtualDiskCopy targetCandidates c**ommand. These commands return a list of the expansion enclosure, slot, and capacity information for source virtual disk and target virtual disk candidates. You can use the **show virtualDiskCopy sourceCandidates** and the **show virtualDiskCopy targetCandidates** commands only after you have enabled the virtual disk copy feature.

Creating a Virtual Disk Copy

NOTICE: A virtual disk copy overwrites data on the target virtual disk. Ensure that you no longer need the data or have backed up the data on the target virtual disk before starting a virtual disk copy.

When you create a virtual disk copy, you must define which virtual disks to use for the source virtual disk and target virtual disks. Define the source virtual disk and target virtual disk by the name of each virtual disk. You can

also define the copy priority and choose whether you want the target virtual disk to be write enabled or read only after the data is copied from the source virtual disk.

Preparing Host Servers to Create a Virtual Disk Copy

NOTICE: Before you create a new copy of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk (and, if applicable, the target disk) to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk(s) in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the virtual disk copy.

Before creating a virtual disk copy, both the server and the associated virtual disk you are copying have to be in the proper state. To ensure that the host server is properly prepared to create a virtual disk copy, you can either use an application to carry out this task, or you can perform the following steps:

1. Stop all I/O activity to the source and target virtual disk.
2. Using your Windows system, flush the cache to both the source and the target virtual disk (if mounted). At the host prompt, type

   `SMrepassist -f <filename-identifier>`

   and press <Enter>. See "SMrepassist Utility" in the *User's Guide* for more information.
3. Remove the drive letter(s) of the source and (if mounted) virtual disk in Windows or unmount the virtual drive(s) in Linux to help guarantee a stable copy of the drive for the virtual disk. If this is not done, the copy operation will report that it has completed successfully, but the copied data will not be updated properly.
4. Follow any additional instructions for your operating system. Failure to follow these additional instructions can create unusable virtual disk copies.

   NOTE: If your operating system requires additional instructions, you can find those instructions in your operating system documentation.

After your server has been prepared, see "Copying the Virtual Disk" on page 81 to copy the virtual disk.

Copying the Virtual Disk

After first preparing the host server(s) as specified in the preceding procedure, use the following examples to make a virtual disk copy.

The following syntax is the general form of the command:

```
create virtualDiskCopy source="sourceName" target=
"targetName" [copyPriority=(highest | high |
medium | low | lowest) targetReadOnlyEnabled=(TRUE
| FALSE)]
```

NOTE: Use one or both of the optional parameters as needed to help define your configuration. It is not necessary to use any optional parameters.

Once the virtual disk copy has started, the source virtual disk will be read only to all I/O activity. Any write attempts to the source virtual disk will fail until the operation completes.

Once the virtual disk copy operation is completed register the target virtual disk with the OS to be used by performing the following steps:

- Enable write permission on the target virtual disk by either removing the Virtual Disk Copy Pair or explicitly setting write permission.
  - In Windows, assign a drive letter to the virtual disk.
  - In Linux, mount the virtual disk.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Create a Virtual Disk Copy" on page 80. The **create virtualDiskCopy** command might look like the following example:

```
client>smcli 123.45.67.89 -c "create
virtualDiskcopy source=\"Jaba_Hut\" target=
\"Obi_1\" copyPriority=medium
targetreadonlyenabled=true"
```

The command in this example copies the data from the source virtual disk named **Jaba_Hut** to the target virtual disk named **Obi_1**. Setting the copy priority to **medium** provides a compromise between the following storage array operations:

- The speed with which the data is copied from the source virtual disk to the target virtual disk
- The amount of processing resource required for data transfers to other virtual disks in the storage array

Setting the *targetReadOnlyEnabled* parameter to **TRUE** means that write requests cannot be made to the target virtual disk. This setting also ensures that the data on the target virtual disk remains unaltered.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Create a Virtual Disk Copy" on page 80. The following example is the script file version of the command:

```
create virtualDiskcopy source="Jaba_Hut" target=
"Obi_1" copyPriority=medium targetreadonlyenabled=
true;
```

After the virtual disk copy operation is completed, the target virtual disk automatically becomes read-only to the hosts. Any write requests to the target virtual disk are rejected, unless you disable the read-only attribute. Use the **set virtualDiskCopy** command to disable the read-only attribute.

Viewing Virtual Disk Copy Properties

Using the **show virtualDiskCopy** command, you can view information about one or more selected source virtual disks or target virtual disks. This command returns the following information:

- The virtual disk role (target or source)
- The copy status
- The start timestamp
- The completion timestamp
- The virtual disk copy priority
- The read-only attribute setting for the target virtual disk
- The source virtual disk World Wide Identifier (WWID) or the target virtual disk WWID

A virtual disk can be a source virtual disk for one virtual disk copy and a target virtual disk for another virtual disk copy. If a virtual disk participates in more than one virtual disk copy, the details are repeated for each associated copy pair.

The following syntax is the general form of the command:

```
show virtualDiskCopy (allVirtualDisks | source
[sourceName] | target [targetName])
```

The following example shows a command that returns information about a virtual disk used for a virtual disk copy:

```
client>smcli 123.45.67.89 -c "show virtualDiskCopy
source [\"Jaba_Hut\"];"
```

The command in the preceding example requests information about the source virtual disk **Jaba_Hut**. If you want information about all virtual disks, use the *allVirtualDisks* parameter. You can also request information about a specific target virtual disk.

The following example is the script file version of the command:

```
show virtualDiskCopy source ["Jaba_Hut"];
```

Changing Virtual Disk Copy Settings

The **set virtualDiskCopy** command enables you to change the property settings for a virtual disk copy pair. Using this command, you can change the following items:

- Copy priority
- Read/write permission for the target virtual disk

Copy priority has five relative settings, which range from highest to lowest. The highest priority supports the virtual disk copy, but I/O activity might be affected. The lowest priority supports I/O activity, but the virtual disk copy takes longer. You can change the copy priority at three different times in the operation:

- Before the virtual disk copy begins
- While the virtual disk copy has a status of In Progress
- After the virtual disk copy has completed re-creating a virtual disk copy using the **recopy virtualDiskCopy** command

When you create a virtual disk copy pair and after the original virtual disk copy has completed, the target virtual disk is automatically defined as read-only to the hosts. The read-only status of the target virtual disk ensures that the copied data on the target virtual disk is not corrupted by additional writes

to the target virtual disk after the virtual disk copy is created. Maintain the read-only status when the following conditions apply:

- You are using the target virtual disk for backup purposes
- You are copying data from one disk group to a larger disk group for greater accessibility
- You are planning to use the data on the target virtual disk to copy back to the source virtual disk in case of a disabled or failed snapshot virtual disk

At other times you might want to write additional data to the target virtual disk. You can use the **set virtualDiskCopy** command to reset the read/write permission for the target virtual disk.

NOTE: If you enabled host writes to the target virtual disk, read and write requests are rejected while the virtual disk copy has a status of In Progress, Pending, or Failed.

The following syntax is the general form of the command:

```
set virtualDiskCopy target [targetName] [source
[sourceName]] copyPriority=(highest | high |
medium | low | lowest) targetReadOnlyEnabled=(TRUE
| FALSE)
```

NOTE: Use one or both of the parameters as needed to help define your configuration. It is not necessary to use either parameter.

The following example shows how to change parameters using the **set virtualDiskCopy** command:

```
client>smcli 123.45.67.89 -c "set virtualDiskcopy
target [\"Obi_1\"] copyPriority=highest
targetreadonlyenabled=false;"
```

The following example is the script file version of the command:

```
set virtualDiskcopy target ["Obi_1"] copyPriority=
highest targetreadonlyenabled=false;
```

Recopying a Virtual Disk

NOTICE: The **recopy virtualDiskCopy** command overwrites existing data on the target virtual disk and makes the target virtual disk read-only to hosts. The **recopy virtualDiskCopy** command fails all snapshot virtual disks associated with the target virtual disk, if any exist.

Using the **recopy virtualDiskCopy** command, you can create a new virtual disk copy for a previously defined copy pair that has a status of Stopped, Failed, or Completed. Use the **recopy virtualDiskCopy** command to create backups of the target virtual disk, then copy the backup to tape for off-site storage. When using the **recopy virtualDiskCopy** command to make a backup, you cannot write to source while the recopy is running. The recopy might take a long time.

When you run the **recopy virtualDiskCopy c**ommand, the data on the source virtual disk is copied in its entirety to the target virtual disk.

Reset the copy priority for the recopy operation by using the **recopy virtualDiskCopy** command. The higher priorities allocate storage array resources to the virtual disk copy at the expense of storage array performance.

Preparing Host Servers to Recopy a Virtual Disk

NOTICE: Before you create a new copy of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk (and, if applicable, the target disk) to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk(s) in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the virtual disk copy.

Before creating a new virtual disk copy for an existing copy pair, both the server and the associated virtual disk you are recopying have to be in the proper state. To ensure that the host server is properly prepared to create a virtual disk recopy, you can either use an application to carry out this task, or you can perform the following steps:

1. Stop all I/O activity to the source and target virtual disk.
2. Using your Windows system, flush the cache to both the source and the target virtual disk (if mounted). At the host prompt, type

   `SMrepassist -f <filename-identifier>`

   and press <Enter>. See "SMrepassist Utility" in the *User's Guide* for more information.

3 Remove the drive letter(s) of the source and (if mounted) virtual disk in Windows or unmount the virtual drive(s) in Linux to help guarantee a stable copy of the drive for the virtual disk. If this is not done, the copy operation will report that it has completed successfully, but the copied data will not be updated properly.

4 Follow any additional instructions for your operating system. Failure to follow these additional instructions can create unusable virtual disk copies.

NOTE: If your operating system requires additional instructions, you can find those instructions in your operating system documentation.

After your server has been prepared, see "Recopying the Virtual Disk" on page 86 to recopy the virtual disk.

Recopying the Virtual Disk

After first preparing the host server(s) as specified in the preceding procedure, use the following examples to make a virtual disk copy.

The following syntax is the general form of the command:

```
recopy virtualDiskCopy target [targetName] [source
[sourceName] copyPriority=(highest | high | medium
| low | lowest) targetReadOnlyEnabled=(TRUE |
FALSE)]
```

NOTE: Use one or all of the optional parameters as needed to help define your configuration. It is not necessary to use any optional parameters.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Recopy a Virtual Disk" on page 85. The following example shows a command that changes the copy priority:

```
client>smcli 123.45.67.89 -c "recopy
virtualDiskCopy target [\"Obi_1\"] copyPriority=
highest;"
```

The command in this example copies data from the source virtual disk associated with the target virtual disk **Obi_1** to the target virtual disk again. The copy priority is set to the highest value to complete the virtual disk copy as quickly as possible. The underlying consideration for using this command is that you have already created the virtual disk copy pair. When you create a virtual disk copy pair, you automatically created one virtual disk copy. Using

this command, you are copying the data from the source virtual disk to the target virtual disk. You are making this copy because the data on the source virtual disk changed since the previous copy was made.

Refer to steps 1 through 4 in the preceding section, "Preparing Host Servers to Recopy a Virtual Disk" on page 85. The following example is the script file version of the command:

```
recopy virtualDiskCopy target ["Obi_1"]
copyPriority=highest;
```

Stopping a Virtual Disk Copy

The **stop virtualDiskCopy** command enables you to stop a virtual disk copy that has a status of In Progress, Pending, or Failed. After you stop a virtual disk copy, you can use the **recopy virtualDiskCopy** command to create a new virtual disk copy using the original virtual disk copy pair. All mapped hosts will have write access to the source virtual disk.

The following syntax is the general form of the command:

```
stop virtualDiskCopy target [targetName] [source
[sourceName]]
```

The following example shows a command to stop a virtual disk copy operation:

```
client>smcli 123.45.67.89 -c "stop virtualDiskCopy
target [\"Obi_1\"];"
```

The following example is the script file version of the command:

```
stop virtualDiskCopy target ["Obi_1"];
```

Removing Copy Pairs

The **remove virtualDiskCopy** command enables you to remove a virtual disk copy pair from the storage array configuration. All virtual disk copy information for the source virtual disk and target virtual disk is removed from the storage array configuration. The data on the source virtual disk or target virtual disk is not deleted. Removing a virtual disk copy from the storage array configuration also removes the read-only attribute for the target virtual disk.

NOTICE: If the virtual disk copy has a status of In Progress, you must stop the virtual disk copy before you can remove the virtual disk copy pair from the storage array configuration.

The following syntax is the general form of the command:

```
remove virtualDiskCopy target [targetName] [source
[sourceName]]
```

The following example shows a command to remove a virtual disk copy pair:

```
client>smcli 123.45.67.89 -c "remove
virtualDiskCopy target [\"Obi_1\"];"
```

The following example is the script file version of the command:

```
remove virtualDiskCopy target ["Obi_1"];
```

Interaction with Other Features

You can run the Virtual Disk Copy feature while running the following features:

- Storage Partitioning
- Snapshot Virtual Disks

When running the Virtual Disk Copy feature with other features, you must take the requirements of other features into consideration to ensure you set up a stable storage array configuration.

You can also run the Virtual Disk Copy feature while running Dynamic Virtual Disk Expansion.

Storage Partitioning

Storage partitioning enables hosts to share access to virtual disks in a storage array. You create a storage partition when you define the following storage array assignments:

- A host
- A host group
- Virtual disk-to-logical unit number (LUN) mapping

The virtual disk-to-LUN mapping enables you to define which host group or host has access to a particular virtual disk in the storage array.

After you create a virtual disk copy, the target virtual disk automatically becomes read-only to hosts to ensure that the data is preserved. Hosts that have been mapped to a target virtual disk do not have write access to the virtual disk, and any attempt to write to the read-only target virtual disk results in a host I/O error.

If you want hosts to have write access to the data on the target virtual disk, use the **set virtualDiskCopy** command to disable the read-only attribute for the target virtual disk.

Snapshot Virtual Disks

A snapshot virtual disk is a point-in-time image of a virtual disk. It is typically created so that an application, such as a backup, can access the snapshot virtual disk and read the data while the source virtual disk remains online and accessible to hosts.

NOTICE: Before using the source virtual disk of a snapshot virtual disk as your target, you must disable all snapshot virtual disks associated with the source virtual disk. By disabling the snapshot virtual disks, you avoid altering the snapshot data if the source virtual disk is changed.

Creating a snapshot virtual disk automatically creates a snapshot repository virtual disk. The snapshot repository virtual disk stores information about the data that has changed since the snapshot virtual disk was created. Snapshot repository virtual disks cannot be selected as a source virtual disk or target virtual disk in a virtual disk copy.

The virtual disk for which the point-in-time image is created is the source virtual disk and must be a standard virtual disk in the storage array.

You can select snapshot virtual disks as the source virtual disk for a virtual disk copy. Selecting a snapshot virtual disk is a good use of this feature, because it enables complete backups without significant impact to the storage array I/O. However, some I/O processing resources are lost to the copy operation.

The Snapshot Virtual Disk feature can be used with the Virtual Disk Copy feature to back up data on the same storage array and to restore the data on the snapshot virtual disk back to its original source virtual disk.

6

Maintaining a Storage Array

Maintenance covers a broad spectrum of activities. Its goal is to keep a storage array operational and available to all hosts. This chapter provides descriptions of command line interface (CLI) and script commands that you can use to perform storage array maintenance. The commands are organized into four sections:

- Routine maintenance
- Performance tuning
- Troubleshooting and diagnostics
- Recovery operations

The organization is not a rigid approach, and you can use the commands as appropriate for your storage array. The commands listed in this chapter do not cover the entire array of commands you can use for maintenance. Other commands, particularly the **set** commands, can provide diagnostic or maintenance capabilities.

Routine Maintenance

Routine maintenance involves those tasks you might perform periodically to ensure that the storage array is running as well as possible or to detect conditions before they become problems.

Running a Media Scan

Media scan provides a method of detecting physical disk media errors before they are found during a normal read from or write to the physical disks. Any errors detected are reported to the Major Event Log (MEL). Media scan provides an early indication of a potential drive failure and reduces the possibility of encountering a media error during host operations. A media

scan is performed as a background operation and scans all data and consistency information in defined user virtual disks. A media scan runs on all virtual disks in the storage array with the following conditions:

- An Optimal status
- No modification operations in progress
- Media scan enabled
- Errors detected during a scan of a user virtual disk are reported to the MEL and handled as:
  - Unrecovered media error — The physical disk could not read the requested data on its first attempt or on any subsequent retries. For virtual disks with redundancy protection, the data could not be reconstructed from the redundant copy. The error is not corrected but it is reported to the MEL.
  - Reconstructed media error — The physical disk could not read the requested data on its first attempt or on any subsequent retries. The data is reconstructed from the redundant copy, rewritten to the drive, verified, and the error is reported to the MEL.
  - Recovered media error — The physical disk could not read the requested data on its first attempt. The result of this action is that the data is rewritten to the physical disk and verified. The error is reported to the MEL.
  - Consistency mismatches — Consistency errors are found, and a media error is forced on the block stripe so that it is found when the physical disk is scanned again. If consistency is repaired, this forced media error is removed. The result of this action is that the first ten consistency mismatches found on a virtual disk are reported to the MEL.
  - Unfixable error — The data could not be read and consistency information could not be used to regenerate it. For example, consistency information cannot be used to reconstruct data on a degraded virtual disk. The result of this action is that the error is reported to the MEL.

The script command set provides two commands to define media scan properties:

- **set virtualDisks**
- **set storageArray**

  The **set virtualDisk** command enables a media scan for the virtual disk. The following syntax is the general form of the command:

  ```
  set (allVirtualDisks | virtualDisk
  [virtualDiskName] | virtualDisks [virtualDiskName1
  ... virtualDiskNamen] | virtualDisk <wwid>)
  mediaScanEnabled=(TRUE | FALSE)
  ```

  The **set storageArray** command defines how frequently a media scan is run on a storage array. The following syntax is the general form of the command:

  ```
  set storageArray mediaScanRate=(disabled | 1-30)
  ```

Running a Consistency Check

Consistency checks are performed when media scans are run, if consistency check is enabled on the virtual disk. (See "Running a Media Scan" on page 91 for an explanation about setting up and running media scans.) During a consistency check, all data blocks in a virtual disk are scanned, and deteriorated data is corrected. The method of correction depends on the redundant array of independent disks (RAID) levels:

- RAID 5 and RAID 6 virtual disks — Consistency is checked and repaired.
- RAID 1 virtual disks — The data is compared between the mirrored physical disks, and data inconsistencies are repaired.
- RAID 0 virtual disks — No redundancy exists.

Before attempting a consistency check, you must enable the process with the **set virtualDisk** command, which uses the following general form:

```
set (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks [virtualDiskName1
... virtualDiskNamen] | virtualDisk <wwid>)
consistencyCheckEnabled=(TRUE | FALSE)
```

Resetting a RAID Controller Module

NOTICE: When you reset a RAID controller module, the RAID controller module is not available for I/O operations until the reset is complete. If a host is using virtual disks owned by the RAID controller module being reset, the I/O directed to the RAID controller module is rejected. Before resetting the RAID controller module, ensure that a multipath driver is installed on all hosts using these virtual disks. If a multipath driver is not installed, the virtual disks will not be available.

Resetting a RAID controller module is the same as rebooting the RAID controller module processors. To reset a RAID controller module, run the following command:

```
reset controller [(0 | 1)]
```

Enabling RAID Controller Module Data Transfer

At times, a RAID controller module might become quiescent while running diagnostics. If this occurs, the RAID controller module might become unresponsive. To revive a RAID controller module, run the following command:

```
enable controller [(0 | 1)] dataTransfer
```

Resetting Battery Age

NOTE: A smart battery module does not require the battery age to be reset.

After replacing the batteries in the storage array, you must reset the age of the battery, either for an entire storage array or one battery in a specific RAID controller module. To reset the age to zero days, run the following command:

```
reset storageArray batteryInstallDate [controller=
(0 | 1)]
```

Removing Persistent Reservations

Persistent reservations preserve virtual disk registrations and prevent hosts, other than the host defined for the virtual disk, from accessing the virtual disk. You must remove persistent reservations before you perform the following changes to your configuration:

- Change or delete logical unit number (LUN) mappings on a virtual disk holding a reservation.
- Delete virtual disk groups or virtual disks that have any reservations.

To determine which virtual disks have reservations, run the following command:

```
show (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks [virtualDiskName1
... virtualDiskNamen]) reservations
```

To clear persistent virtual disk reservations, run the following command:

```
clear (allVirtualDisks |
virtualDisk[virtualDiskName] | virtualDisks
[virtualDiskName1" ... "virtualDiskNamen])
reservations
```

Synchronizing RAID Controller Module Clocks

To synchronize the clocks on both RAID controller modules in a storage array with the host clock, run the following command:

```
set storageArray time
```

Locating Physical Disks

At times, you might need to locate a specific physical disk. In very large storage array configurations, this can sometimes be awkward. If you need to locate a specific physical disk, turn on the indicator LED on the front of the physical disk. To turn on the indicator LED on a physical disk, run the following command:

```
start physicalDisk [enclosureID,slotID] blink
```

To turn off the indicator LED after locating the physical disk, run the following command:

```
stop physicalDisk blink
```

Performance Tuning

Over time, as a storage array exchanges data between the hosts and physical disks, its performance can degrade. Monitor the performance of a storage array and make adjustments to the storage array operational settings to improve performance.

Monitoring Performance

Monitor the performance of a storage array by using the **save storageArray performanceStats** command. This command saves performance information to a file that you can review to determine how well the storage array is running. Table 6-1 lists the performance information saved to the file.

Table 6-1. Storage Array Performance Information

Type of Information	Description
Devices	Devices are: • RAID Controller Modules — The RAID controller module in slot 0 or 1 and a list of the virtual disks owned by the RAID controller module • Virtual Disk — A list of the virtual disk names • Storage Array Totals — A list of the totals for both RAID controller modules in an active-active RAID controller module pair, regardless if one, both, or neither are selected for monitoring
Total I/Os	Number of total I/Os performed since the storage array was started
Read Percentage	Percentage of total I/Os that are read operations (calculate the write percentage by subtracting the read percentage from 100 percent)
Cache Hit Percentage	Percentage of reads that are fulfilled by data from the cache rather than requiring an actual read from a physical disk
Current KB/second	Current transfer rate in kilobytes per second (current means the number of kilobytes per second since the last time the polling interval elapsed, causing an update to occur)
Maximum KB/second	Highest data transfer value achieved in the current kilobyte-per-second statistic block
Current IO/second	Current number of I/Os per second (current means the number of I/Os per second since the last time the polling interval elapsed, causing an update to occur)
Maximum IO/second	Highest number of I/Os achieved in the current I/O-per-second statistic block

The general form of the command is:

```
save storageArray performanceStats file="filename"
```

where *file* is the name of the file in which you want to save the performance statistics. You can use any file name your operating system can support. The default file type is **.csv**. The performance information is saved as a comma-delimited file.

Before using the **save storageArray performanceStats** command, run the **set session performanceMonitorInterval** and **set session performanceMonitorIterations** commands to specify how often statistics are collected.

Changing RAID Levels

When creating a disk group, define the RAID level for the virtual disks in that group. You can later change the RAID level to improve performance or provide more secure protection for your data. To change the RAID level, run the following command:

```
set diskGroup [diskGroupNumber] raidLevel=
(0|1|5|6)
```

where *diskGroupNumber* is the number of the disk group for which to change the RAID level.

Changing Segment Size

When creating a new virtual disk, define the segment size for that virtual disk. You can later change the segment size to optimize performance. In a multi-user database or file system storage environment, set your segment size to minimize the number of physical disks needed to satisfy an I/O request. Use larger values for the segment size. Using a single physical disk for a single request leaves other disks available to simultaneously service other requests. If the virtual disk is in a single-user large I/O environment, performance is maximized when a single I/O request is serviced with a single data stripe; use smaller values for the segment size. To change the segment size, run the following command:

```
set virtualDisk ([virtualDiskName] | <wwid>)
segmentSize=segmentSizeValue
```

where *segmentSizeValue* is the new segment size you want to set. Valid segment size values are 8, 16, 32, 64, 128, 256, and 512. You can identify the virtual disk by name or World Wide Identifier (WWID) (see "Set Virtual Disk" on page 198).

Defragmenting a Disk Group

When you defragment a disk group, you consolidate the free capacity in the disk group into one contiguous area. Defragmentation does not change the way in which the data is stored on the virtual disks. As an example, consider a disk group with five virtual disks. If you delete virtual disks 1 and 3, your disk group is configured in the following manner:

space, virtual disk 2, space, virtual disk 4, virtual disk 5, original unused space

When you defragment this group, the space (free capacity) is consolidated into one contiguous location after the virtual disks. After being defragmented, the disk group is:

virtual disk 2, virtual disk 4, virtual disk 5, consolidated unused space

To defragment a disk group, run the following command:

```
start diskGroup [diskGroupNumber] defragment
```

where *diskGroupNumber* is the identifier for the disk group.

NOTE: Defragmenting a disk group starts a long-running operation.

Troubleshooting and Diagnostics

If a storage array exhibits abnormal operation or failures, you can use the commands described in this section to determine the cause of the problems.

Collecting Physical Disk Data

To gather information about all the physical disks in a storage array, run the **save allPhysicalDisks** command. This command collects sense data from all the physical disks in a storage array and saves the data to a file. The sense data consists of statistical information maintained by each of the physical disks in the storage array.

Diagnosing a RAID Controller Module

The **diagnose controller** command's *testID* parameter takes the following options, which you can use to verify that a RAID controller module is functioning correctly:

- **1**— Reads the test
- **2**— Performs a data loop-back test
- **3**— Writes the test

The read test initiates a **read** command as it would be sent over an I/O data path. The read test compares data with a known, specific data pattern, checking for data integrity and errors. If the **read** command is unsuccessful or the data compared is not correct, the RAID controller module is considered to be in error and is placed offline.

Run the data loopback test only on RAID controller modules that have connections between the RAID controller module and the physical disks. The test passes data through each RAID controller module physical disk-side channel out onto the loop and back again. Enough data is transferred to determine error conditions on the channel. If the test fails on any channel, this status is saved so that it can be returned if all other tests pass.

The write test initiates a **write** command as it would be sent over an I/O data path to the diagnostics region on a specified physical disk. This diagnostics region is then read and compared to a specific data pattern. If the write fails or the data compared is not correct, the RAID controller module is considered to be in error, and it is failed and placed offline.

For best results, run all three tests at initial installation. Also, run the tests any time you make changes to the storage array or to components connected to the storage array (such as hubs, switches, and host adapters).

A custom data pattern file called **diagnosticsDataPattern.dpf** is included on the **Utility** directory of the installation CD. You can modify this file, but the file must have the following properties to work correctly for the tests:

- The file values must be entered in hexadecimal format (00 to FF) with only one space between the values.
- The file must be no larger than 64 bytes in size. Smaller files can be used, but larger files can cause an error.

The test results contain a generic, overall status message and a set of specific test results. Each test result contains the following information:

- Test (read/write/data loopback)
- Port (read/write)
- Level (internal/external)
- Status (pass/fail)

Events are written to the MEL when diagnostics are started and when testing is completed. These events help you to evaluate whether diagnostics testing was successful or failed and the reason for the failure.

Recovery Operations

Recovery operations involve replacing failed RAID controller modules and physical disks, restoring data, and restoring the storage array to operation.

Setting RAID Controller Module Operational Mode

A RAID controller module has three operational modes:

- Online
- Offline
- Service

NOTICE: Placing a RAID controller module offline can cause loss of data.

Placing a RAID controller module online sets it to the Optimal state and makes it active and available for I/O operations. Placing a RAID controller module offline makes it unavailable for I/O operations and moves its disk groups to the other RAID controller module if failover protection is enabled.

Taking a RAID controller module offline can seriously impact data integrity and storage array operation.

If you take a RAID controller module offline, the second RAID controller module in the pair takes over. Disk groups and their associated virtual disks that were assigned to the offline RAID controller module are automatically reassigned to the remaining RAID controller module.

NOTICE: Place a RAID controller module in Service mode only under the direction of Technical Support.

Use Service mode when you want to perform an operation, such as replacing a RAID controller module. Placing a RAID controller module in Service mode makes it unavailable for I/O operations. Placing a RAID controller module in Service mode also moves the disk groups from the RAID controller module to the second RAID controller module without affecting the disk groups' preferred path. Moving disk groups might significantly reduce performance. The disk groups are automatically transferred back to the preferred RAID controller module when it is placed back online.

NOTICE: A multipath driver is required on all hosts and is the only supported configuration. If the multipath driver is not installed, the virtual disks will not be accessible.

Before you place a RAID controller module in Service mode, ensure that a multipath driver is installed on all hosts using these virtual disks.

To change the operational mode of a RAID controller module, run the following command:

```
set controller [(0 | 1)] availability=(online |
offline | serviceMode)
```

Changing RAID Controller Module Ownership

You can change which RAID controller module owns a virtual disk by using the **set virtualDisk** command. The following syntax is the general form of the command:

```
set (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks [virtualDiskName1
... virtualDiskNamen] | virtualDisk <wwid>) owner=
(0 | 1)
```

Initializing a Physical Disk

NOTICE: When you initialize a physical disk, all data on the physical disk is lost.

You must initialize a physical disk when you have moved physical disks that were previously part of a disk group from one storage array to another. If you do not move the entire set of physical disks, the disk group and virtual disk information on the physical disks that you move is incomplete. Each physical disk that you move contains only part of the information defined for the

virtual disk and disk group. To be able to reuse the physical disks to create a new disk group and virtual disk, you must erase all old information from the physical disks by initializing the physical disk.

When you initialize a physical disk, all old disk group and virtual disk information is erased, and the physical disk is returned to an unassigned state. Returning a physical disk to an unassigned state adds unconfigured capacity to a storage array. You can use this capacity to create additional disk groups and virtual disks.

To initialize a physical disk, run the following command:

```
start physicalDisk [enclosureID,slotID] initialize
```

where *enclosureID* and *slotID* are the identifiers for the physical disk.

Reconstructing a Physical Disk

If two or more physical disks in a disk group have failed, the virtual disk shows a status of Failed. All of the virtual disks in the disk group are no longer operating. To return the disk group to an Optimal status, you must replace the failed physical disks. After replacing the physical disks, reconstruct the data on physical disks. The reconstructed data is the data as it would appear on the failed physical disks.

To reconstruct a physical disk, run the following command:

```
start physicalDisk [enclosureID,slotID]
reconstruct
```

where *enclosureID* and *slotID* are the identifiers for the physical disk.

NOTE: You can use this command only when the physical disk is assigned to a RAID 1, 5, or 6 disk group.

Initializing a Virtual Disk

NOTICE: When you initialize a virtual disk, all data on the virtual disk and all information about the virtual disk are destroyed.

A virtual disk is automatically initialized when you first create it. If the virtual disk starts exhibiting failures, you might be required to re-initialize the virtual disk to correct the failure condition.

The initialization process cannot be cancelled once it has begun. This option cannot be used if any modification operations are in progress on the virtual disk or disk group. To initialize a virtual disk, run the following command:

```
start virtualDisk [virtualDiskName] initialize
```

where *virtualDiskName* is the identifier for the virtual disk.

Redistributing Virtual Disks

Redistributing virtual disks returns the virtual disks to their preferred RAID controller module owners. The preferred RAID controller module ownership of a virtual disk or disk group is the RAID controller module of an active-active pair that is designated to own the virtual disks. The preferred owner for a virtual disk is initially designated when the virtual disk is created. If the preferred RAID controller module is being replaced or undergoing a firmware download, ownership of the virtual disks is automatically shifted to the second RAID controller module. The second RAID controller module becomes the current owner of the virtual disks. This change is considered to be a routine ownership change and is reported in the MEL.

NOTICE: Ensure that a multipath driver is installed, or the virtual disks will not be accessible.

To redistribute virtual disks to their preferred RAID controller modules, run the following command:

```
reset storageArray virtualDiskDistribution
```

NOTE: You cannot run this command if all virtual disks are currently owned by their preferred RAID controller module or if the storage array does not have defined virtual disks.

Under some host operating systems, you must reconfigure the multipath host driver. You might also need to make operating system modifications to recognize the new I/O path to the virtual disk.

7

Script Commands

This chapter describes the script commands used to configure, monitor, and maintain a storage array. This chapter is organized in four sections:

- "Command Formatting Rules" on page 106 lists general formatting rules that apply to the command syntax.
- "Commands Listed by Function" on page 108 lists the commands by functional activity:
  - Disk group
  - Enclosure
  - Host topology
  - Physical disk
  - Redundant array of independent disks (RAID) controller module
  - Session
  - Show string
  - Snapshot
  - Storage array
  - Virtual disk
  - Virtual disk copy
- "Commands Listed Alphabetically" on page 114 lists the commands alphabetically and, for each command, includes command name, syntax, and parameters.

NOTICE: Commands entered using the command line interface (CLI) are capable of damaging a configuration and causing loss of data if not used properly. Command operations are performed as soon as you run the commands. Some commands can immediately delete configurations or data. Before using the command line interface, make sure you have backed up all data, and save the current configuration so that you can reinstall it if the changes you make do not work.

Command Formatting Rules

This section describes the general rules for formatting a script command and how the command syntax is presented in the following command descriptions. Syntax unique to a specific command is explained in the notes at the end of each command description.

- The script commands are not case sensitive. Type the commands in lowercase, uppercase, or mixed case. (In the following command descriptions, mixed case is used as an aid to reading the command names and understanding the purpose of the command.)
- You must enter spaces in the commands as they are shown in the command descriptions.
- Brackets are used in two ways:
  - As part of the command syntax
  - To indicate which parameters are optional

    The description of each parameter tells you when you must put brackets around a parameter value.
- Parentheses shown in the command syntax enclose specific choices for a parameter. That is, if you want to use the parameter, you must use one of the values shown in the parentheses. Generally, you do not include parentheses in a command. In some instances, however, you must put parentheses around a list. For example, you must put parentheses around a list of enclosure ID values and slot ID values. The description of each parameter tells you if you must put parentheses around a parameter value.
- Vertical bars in a command indicate *or* and separate the valid entries for the parameter. For example, the syntax for the *raidLevel* parameter in the command description appears as follows:

  ```
  raidLevel=(0 | 1 | 5 | 6)
  ```

  To use the *raidLevel* parameter to set a RAID level of 5, enter:

  ```
  raidLevel=5
  ```

- When you specify physical disk locations by using enclosure ID values and slot ID values, separate the ID values with a comma. If you enter more than one set of ID values, separate each set of values by a space. Put parentheses around the set of values. For example:

  ```
  (0,0 0,1 0,2 0,3 1,0 1,1 1,2 1,3)
  ```

- Italicized terms in the command indicate a value or information that you need to provide. For example, when you encounter the italicized term:

  numberOfPhysicalDisks

  replace the italicized term with a value for the number of physical disks that you want to include with the command.

- You can use any combination of alphanumeric characters, hyphens, and underscores for the names of the following components:
  - Storage arrays
  - Host groups
  - Hosts
  - Disk groups
  - Virtual disks
  - Host bus adapter (HBA) host ports

    Names can have a maximum of 30 characters. If the label contains multiple words, underscores, or hyphens, you must put quotation marks around the name. In some usages you must also put brackets around the name. The description of each parameter tells you if you must put quotation marks or brackets around a parameter value. The character string cannot contain a new line. You must use unique names or the RAID controller module firmware returns an error.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

On Microsoft® Windows®, you must put a backslash (\) before and after the name in addition to other delimiters. For example, the

following name is used in a command running under a Windows operating system:

```
[\"Engineering"\]
```

For Linux, and when used in a script file, the name appears as the following:

```
["Engineering"]
```

When you enter the World Wide Identifier (WWID) of an HBA host port, some usages require quotation marks around the WWID. In other uses, you must put angle brackets (< >) around the WWID. The description of the *WWID* parameter tells you if you must put quotation marks or angle brackets around the WWID.

- Script commands must end with a semicolon (;). You can enter more than one script command on the command line each time you enter a CLI command.

Commands Listed by Function

This section presents the commands organized into groups related to physical, logical, and operational features of the storage array.

Disk Group Commands

"Create Disk Group" on page 120

"Delete Disk Group" on page 139

"Revive Disk Group" on page 163

"Set Disk Group" on page 173

"Show Disk Group" on page 204

"Start Disk Group Blink" on page 223

"Start Disk Group Defragment" on page 223

"Stop Disk Group Blink" on page 229

"Show Disk Group Import Dependencies" on page 220

"Start Disk Group Import/Export" on page 227

Enclosure Commands

"Download Enclosure Management Module Firmware" on page 146

"Save Enclosure Log Data" on page 164

"Set Enclosure Attribute" on page 174

"Set Enclosure Identification" on page 175

"Start Enclosure Blink" on page 223

"Stop Enclosure Blink" on page 229

Host Topology Commands

"Create Host" on page 122

"Create Host Group" on page 123

"Create Host Port" on page 124

"Delete Host" on page 140

"Delete Host Group" on page 140

"Delete Host Port" on page 141

"Set Host" on page 176

"Set Host Group" on page 178

"Set Host Port" on page 179

"Show Host Ports" on page 205

iSCSI Commands

"iSCSI Commands" on page 109

"Delete iSCSI Initiator" on page 141

"Reset Storage Array iSCSI Baseline" on page 162

"Save Storage Array iSCSI Statistics" on page 168

"Set Controller" on page 170

"Set Host" on page 176

"Set Host Port" on page 179

"Set iSCSI Initiator" on page 179

"Set iSCSI Target Properties" on page 180

"Set Storage Array ICMP Response" on page 193

"Set Storage Array iSNS Server IPv4 Address" on page 194

"Set Storage Array iSNS Server IPv6 Address" on page 195

"Set Storage Array iSNS Server Refresh" on page 196

"Set Unnamed Discovery Session" on page 198

"Show Current iSCSI Sessions" on page 203

"Show Storage Array Negotiation Defaults" on page 214

"Show Unconfigured iSCSI Initiators" on page 216

"Start iSCSI DHCP Refresh" on page 224

"Stop iSCSI Session" on page 229

Physical Disk Commands

"Set Foreign Physical Disk to Native" on page 176

"Clear Physical Disk Channel Statistics" on page 118

"Download Physical Disk Firmware" on page 147

"Revive Physical Disk" on page 163

"Save Physical Disk Channel Fault Isolation Diagnostic Status" on page 164

"Set Physical Disk Channel Status" on page 182

"Set Physical Disk Hot Spare" on page 182

"Set Physical Disk State" on page 183

"Show Physical Disk" on page 205

"Show Physical Disk Channel Statistics" on page 207

"Show Physical Disk Download Progress" on page 208

"Start Physical Disk Channel Fault Isolation Diagnostics" on page 224

"Start Physical Disk Blink" on page 226

"Start Physical Disk Initialize" on page 226

"Start Physical Disk Reconstruction" on page 227

"Stop Physical Disk Blink" on page 229

"Stop Physical Disk Channel Fault Isolation Diagnostics" on page 230

RAID Controller Module Commands

"Diagnose RAID Controller Module" on page 143

"Enable RAID Controller Module" on page 151

"Reset RAID Controller Module" on page 160

"Save RAID Controller Module NVSRAM" on page 165

"Set Controller" on page 170

"Set RAID Controller Module" on page 183

"Show RAID Controller Module" on page 208

"Show RAID Controller Module NVSRAM" on page 209

Session Command

"Set Session" on page 188

Show String Command

"Show String" on page 216

Snapshot Commands

"Create Snapshot Virtual Disk" on page 133

"Set Snapshot Virtual Disk" on page 189

"Stop Snapshot" on page 230

Storage Array Commands

"Accept Storage Array Pending Topology" on page 114

"Activate Storage Array Firmware" on page 114

"Autoconfigure Storage Array" on page 115

"Autoconfigure Storage Array Hot Spares" on page 116
"Clear Storage Array Configuration" on page 118
"Clear Storage Array Event Log" on page 119
"Clear Storage Array Firmware Pending Area" on page 119
"Disable Storage Array Feature" on page 146
"Download Storage Array Firmware/NVSRAM" on page 148
"Download Storage Array Physical Disk Firmware" on page 150
"Download Storage Array NVSRAM" on page 149
"Enable Storage Array Feature Key" on page 151
"Reset Storage Array Battery Install Date" on page 161
"Reset Storage Array Virtual Disk Distribution" on page 162
"Save Storage Array Configuration" on page 166
"Save Storage Array Events" on page 167
"Save Storage Array Performance Statistics" on page 169
"Save Storage Array SAS PHY Counts" on page 169
"Save Storage Array State Capture" on page 170
"Save Storage Array Support Data" on page 170
"Set Storage Array" on page 191
"Set Storage Array Learn Cycle" on page 197
"Set Storage Array Enclosure Positions" on page 192
"Show Storage Array" on page 210
"Show Storage Array Autoconfigure" on page 212
"Show Storage Array Host Topology" on page 214
"Show Storage Array LUN Mappings" on page 214
"Show Storage Array Pending Topology" on page 215
"Show Storage Array Unreadable Sectors" on page 215

"Start Storage Array Blink" on page 227

"Stop Storage Array Blink" on page 231

"Stop Storage Array Physical Disk Firmware Download" on page 231

Virtual Disk Commands

"Check Disk Consistency" on page 117

"Clear Virtual Disk Reservations" on page 120

"Create RAID Virtual Disk (Automatic Physical Disk Select)" on page 126

"Create RAID Virtual Disk (Free Capacity Base Select)" on page 128

"Create RAID Virtual Disk (Manual Physical Disk Select)" on page 130

"Delete Virtual Disk" on page 142

"Recover RAID Virtual Disk" on page 153

"Remove Virtual Disk LUN Mapping" on page 159

"Repair Virtual Disk Consistency" on page 160

"Set Virtual Disk" on page 198

"Show Virtual Disk" on page 217

"Show Virtual Disk Action Progress" on page 218

"Show Virtual Disk Reservations" on page 222

"Start Virtual Disk Initialization" on page 228

Virtual Disk Copy Commands

"Create Virtual Disk Copy" on page 137

"Recopy Virtual Disk Copy" on page 152

"Remove Virtual Disk Copy" on page 158

"Set Virtual Disk Copy" on page 203

"Show Virtual Disk Copy" on page 219

"Show Virtual Disk Copy Source Candidates" on page 220

"Show Virtual Disk Copy Target Candidates" on page 220

"Stop Virtual Disk Copy" on page 231

Commands Listed Alphabetically

Following are the script commands listed alphabetically.

Accept Storage Array Pending Topology

This command configures all or part of the pending host topology discovered by the **show storageArray pendingTopology** command.

Syntax

```
accept storageArray pendingTopology (allHosts |
host "hostName" | hosts ("hostName1" ...
"hostNamen")
```

Parameters

Parameter	Description
allHosts	Selects all hosts identified by **show storageArray pendingTopology**.
host or hosts	Name of the host to include in the storage array topology. You can enter more than one host name. You must put quotation marks (" ") around the host name.

Activate Storage Array Firmware

This command activates firmware previously downloaded to the pending configuration area in the RAID controller modules in the storage array.

Syntax

```
activate storageArray firmware
```

Parameters

None

Autoconfigure Storage Array

This command automatically configures a storage array. Before entering the **autoConfigure storageArray** command, enter the **show storageArray autoConfiguration** command. The **show storageArray autoConfiguration** command returns configuration information in the form of a list of valid physical disk types, RAID levels, virtual disk information, and hot spare information. (This list corresponds to the parameters for the **autoConfigure storageArray** command.)

The RAID controller modules audit the storage array and then determine the highest RAID level that the storage array can support and the most efficient virtual disk definition for the RAID level. If the configuration described by the returned list is acceptable, enter the **autoConfigure storageArray** command without any parameters. To modify the configuration, change a single parameter or all of the parameters to meet your configuration requirements. After entering the **autoConfigure storageArray** command, the RAID controller modules set up the storage array using either the default parameters or those you selected.

Syntax

```
autoConfigure storageArray [physicalDiskType=
(SAS | SATA) raidLevel=(0 | 1 | 5 | 6)
diskGroupWidth=numberOfPhysicalDisks
diskGroupCount=numberOfDiskGroups
virtualDisksPerGroupCount=
numberOfVirtualDisksPerGroup hotSpareCount=
numberOfHotspares segmentSize=segmentSizeValue]
```

Parameters

Parameter	Description
physicalDiskType	Type of physical disks to use for the storage array. Valid physical disk types are Serial Attached SCSI (SAS) or Serial Advanced Technology Attachment (SATA). This parameter is not required if only one type of physical disk is in the storage array.

Parameter	Description
raidLevel	RAID level of the disk group that contains the physical disks in the storage array. Valid RAID levels are **0**, **1**, **5** or **6**.
diskGroupWidth	Number of physical disks in a disk group in the storage array. For information about the number of physical disks that you can use in a disk group, see "Enclosure Loss Protection" on page 52.
diskGroupCount	Number of disk groups in the storage array. Use integer values.
virtualDisksPerGroupCount	Number of equal-capacity virtual disks per disk group. Use integer values.
hotSpareCount	Number of hot spares in the storage array. Use integer values. For information about hot spares, see "Assigning Global Hot Spares" on page 59.
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are **8**, **16**, **32**, **64**, **128**, **256**, or **512**. For information about the *segmentSize* parameter, see "Using the Auto Configure Command" on page 53.

Autoconfigure Storage Array Hot Spares

This command automatically defines and configures the hot spares in a storage array. You can run this command at any time. This command provides the best hot spare coverage for a storage array.

Syntax

```
autoConfigure storageArray hotSpares
```

Parameters

None.

NOTE: When you run the **autoconfigure storageArray hotSpares** command, the RAID controller module firmware determines the number of hot spares to create based on the total number and type of physical disks in the storage array.

Check Disk Consistency

This command checks a virtual disk for consistency and media errors, and writes the results of the check to a file.

Syntax

```
check virtualDisk [virtualDiskName] consistency
[consistencyErrorFile=filename] [mediaErrorFile=
filename] [priority=(highest | high | medium | low
| lowest)] [verbose=(TRUE|FALSE)]
```

Parameters

Parameter	Description
virtualDisk	Name of the specific virtual disk to check consistency. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.
consistencyErrorFile	Name of the file in which to save the consistency error information. You must put quotation marks (" ") around the file name.
mediaErrorFile	Name of the file in which to save the media error information. You must put quotation marks (" ") around the file name.

Parameter	Description
priority	Specifies the priority that the consistency check has relative to host I/O activity. Valid entries are highest, high, medium, low, or lowest.
verbose	Captures progress details, such as percent complete, and shows the information while virtual disk consistency is being repaired. To capture progress details, set this parameter to **TRUE**. To prevent capturing progress details, set this parameter to **FALSE**.

Clear Physical Disk Channel Statistics

This command resets the statistics for all physical disk channels.

Syntax

```
clear allPhysicalDiskChannels stats
```

Parameters

None.

Clear Storage Array Configuration

This command clears the entire configuration from the RAID controller modules in a storage array. Information that defines all disk groups, virtual disks, and hot spares is deleted. Use this command to create a new configuration on a storage array that already has a configuration defined.

NOTICE: As soon as you run this command, the existing storage array becomes unresponsive. You must remove and re-add the storage array to resume communication with the host. To remove an unresponsive storage array, access the Enterprise Management Window, and click **Remove** on the Modular Disk Storage Manager toolbar. To re-add the storage array, access the Enterprise Management Window, click **New** in the Modular Disk Storage Manager toolbar, and enter the appropriate IP address.

Syntax

```
clear storageArray configuration (all |
volumeGroups)
```

Parameters

If you do not enter a parameter, this command removes all configuration information for the storage array, except for the information related to security and identification.

Parameter	Description
all	The setting to remove the entire configuration of the storage array, including security and identity information. Removing all configuration information returns the storage array to its initial state.
volumeGroups	The setting to remove the virtual disk configuration and the disk group configuration. The rest of the configuration stays intact.

Clear Storage Array Event Log

This command clears the Major Event Log (MEL) for the storage array by deleting the data in the MEL buffer.

NOTICE: As soon as you run this command, the existing MEL in the storage array is deleted.

Syntax

```
clear storageArray eventLog
```

Parameters

None.

Clear Storage Array Firmware Pending Area

This command deletes a previously downloaded firmware image or nonvolatile static random access memory (NVSRAM) values from the pending area buffer.

NOTICE: As soon as you run this command, the contents of the existing pending area in the storage array are deleted.

Syntax

```
clear storageArray firmwarePendingArea
```

Parameters

None.

Clear Virtual Disk Reservations

This command clears persistent virtual disk reservations.

Syntax

```
clear (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks [virtualDiskName1
... virtualDiskNamen]) reservations
```

Parameters

Parameter	Description
allVirtualDisks	Clears reservations on all virtual disks in the storage array.
virtualDisk or virtualDisks	Name of the specific virtual disk for which to clear reservations. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Create Disk Group

This command creates either a free-capacity disk group or one virtual disk on a set of unassigned physical disks.

Syntax

```
create diskGroup physicalDisks=(trayID1,slotID1
... trayIDn,slotIDn) raidLevel=(0 | 1 | 5 | 6)
userlabel=userlabel
[enclosureLossProtect=(true | false)]
```

Parameters

Parameter	Description
userLabel	The name that you want to use for the new disk group. Enclose the name in double quotation marks (" ").
physicalDisks	The physical disks that you want to assign to the virtual disk that you want to create. Specify the tray ID and slot ID for each physical disk that you assign to the virtual disk. Tray ID values are **0** to **99**. Slot ID values are 0 to 31. Enclose the tray ID values and the slot ID values in parentheses.
raidLevel	The RAID level of the disk group that contains the virtual disk. Valid values are **0**, **1**, **5**, or **6**.
enclosureLossProtect	The setting to enforce enclosure loss protection when you create the disk group. To enforce enclosure loss protection, set this parameter to **true**. The default value is **false**.

Additional Information

physicalDisks

The *physicalDisks* parameter lets you choose the number of physical disks that you want to use in the disk group. If you choose this option, you do not need to specify the physical disks by tray ID and slot ID. The RAID controller modules choose the specific physical disks to use for the disk group. If you do not specify a capacity by using the *capacity* parameter, all of the physical disk capacity that is available in the disk group is used. If you do not specify capacity units, **bytes** is used as the default value.

Enclosure Loss Protection

For enclosure loss protection to work, each physical disk in a disk group must be in a separate enclosure. If you set the *enclosureLossProtect* parameter to **true** and have selected more than one physical disk from any one enclosure, the storage array returns an error. If you set the *enclosureLossProtect* parameter to **false**, the storage array performs operations, but the disk group that you create might not have enclosure loss protection. Enclosure loss protection is not valid when you create virtual disks on existing disk groups.

Create Host

This command creates a new host.

NOTE: A host is a system that is attached to the storage array and accesses the virtual disks on the storage array through its HBA host ports. You can define specific virtual disk-to-logical unit number (LUN) mappings to an individual host or assign the host to a host group that shares access to one or more virtual disks.

Syntax

```
create host userLabel="hostName" [hostGroup=
"hostGroupName"]
```

Parameters

Parameter	Description
userLabel	Name to give the host that you are creating. You must put quotation marks (" ") around the host name.

Parameter	Description
hostGroup	Name of the host group in which to create a new host. You must put quotation marks (" ") around the host group name. (If a host group does not exist, you can create a new host group by using the **create hostGroup** command.) **NOTE:** A host group is an optional topological element defined to designate a collection of hosts that share access to the same virtual disks. The host group is a logical entity. Define a host group only if you have two or more hosts that share access to the same virtual disks. If you do not specify a host group in which to place the host you are creating, the newly defined host belongs to the default host group.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Example

```
-c create host userLabel= \"job2900\";"
```

Create Host Group

This command creates a new host group.

NOTE: A host group is an optional topological element that you can define to designate a collection of hosts that share access to the same virtual disks. The host group is a logical entity. Define a host group only if you have two or more hosts that can share access to the same virtual disks.

Syntax

```
create hostGroup userLabel="hostGroupName"
```

Parameters

Parameter	Description
userLabel	Name to give the host group that you are creating. You must put quotation marks (" ") around the host group name.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Create Host Port

This command creates a new HBA host port identification. The identification is a software value that represents the physical HBA host port to the RAID controller module. Without the host port identification, the RAID controller module cannot receive instructions or data from the host port.

Syntax

```
create hostPort identifier="wwid" userLabel=
"portLabel" host="hostName"
```

Parameters

Parameter	Description
identifier	WWID of the HBA host port. You must put quotation marks (" ") around the WWID.
userLabel	Name to give the new HBA host port. You must put quotation marks (" ") around the port label.

Parameter	Description
host	Name of the host for which you are defining an HBA host port. You must put quotation marks (" ") around the host name. **NOTE:** An HBA host port is a physical connection on a host adapter that resides within a host system. An HBA host port provides host access to the virtual disks in a storage array. If the host bus adapter has only one physical connection (one host port), the terms *host port* and *host bus adapter* are synonymous

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Create iSCSI Initiator

This command creates a new iSCSI initiator object.

Syntax

```
create iscsiInitiator iscsiName = iSCSI-ID
userLabel = name host = host-name [chapSecret =
password]
```

Parameters

Parameter	Description
`iscsiName`	The default identifier of the iSCSI initiator.
`userLabel`	The name that you want to use for the iSCSI initiator. Enclose the name in double quotation marks (" ")
`host`	The name of the host in which the iSCSI initiator is installed.
`chapSecret`	The password that you want to use to authenticate a peer connection.

NOTE: Challenge Handshake Authentication Protocol (CHAP) is a protocol that authenticates the peer of a connection. CHAP is based upon the peers sharing a "secret." A secret is a security key that is similar to a password. Use `chapSecret` only for initiators requiring mutual authentication.

Create RAID Virtual Disk (Automatic Physical Disk Select)

This command creates a disk group across the storage array physical disks, and a new virtual disk in the disk group. The RAID controller modules in the storage array choose the physical disks to include in the virtual disk.

Syntax

```
create virtualDisk physicalDiskCount=
numberOfPhysicalDisks
raidLevel=0 | 1 | 5 | 6 userLabel=
"virtualDiskName" [physicalDiskType=(SAS | SATA)
capacity=virtualdiskCapacity owner=(0 | 1)
segmentSize=segmentSizeValue
enclosureLossProtect=(TRUE | FALSE)]
```

Parameters

Parameter	Description
physicalDiskCount	Number of unassigned physical disks to use in the disk group. **NOTE:** The *physicalDiskCount* parameter enables you to choose the number of physical disks to use in the disk group. You do not need to specify the physical disks by enclosure ID and slot ID. The RAID controller modules choose the specific physical disks to use for the disk group.
raidLevel	RAID level of the disk group that contains the virtual disk. Valid values are 0, 1, 5 or 6.
userLabel	Name to give to the new virtual disk. You must put quotation marks (" ") around the new virtual disk name.

Parameter	Description
physicalDiskType	Specifies the type of physical disk to use in the virtual disk. You cannot mix physical disk types in the virtual disk. Valid physical disk types are SAS or SATA.
capacity	The size of the virtual disk that you are adding to the storage array. Size is defined in units of bytes, kilobytes, megabytes, or gigabytes. **NOTE:** A space must be added between the last digit and the size (MB, GB, or KB) for values greater than 9.
owner	The RAID controller module that owns the virtual disk. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. If you do not specify an owner, the RAID controller module firmware determines the owner. **NOTE:** The *owner* parameter defines which RAID controller module owns the virtual disk. If you do not specify a capacity, all of the available physical disk capacity in the disk group is used. If you do not specify capacity units, bytes are used as the default units.

Parameter	Description
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, **16**, **32**, **64**, 128, **256**, or **512**. For information about the *segmentSize* parameter, see "Using the Auto Configure Command" on page 53.
enclosureLossProtect	Specifies that enclosure loss protection is enforced when creating the disk group. To enforce enclosure loss protection, set this parameter to **TRUE**. The default setting is **FALSE**. For information about the *enclosureLossProtect* parameter, see "Enclosure Loss Protection" on page 52.

Create RAID Virtual Disk (Free Capacity Base Select)

This command creates a virtual disk in the free space of a disk group.

Syntax

```
create virtualDisk diskGroup=diskGroupNumber
userLabel="virtualDiskName" [freeCapacityArea=
freeCapacityIndexNumber capacity=
virtualDiskCapacity owner=(0 | 1) segmentSize=
segmentSizeValue]
```

Parameters

Parameter	Description
diskGroup	Sequence number of the disk group in which to create the new virtual disk. (To determine the sequence numbers of the disk groups in your storage array, enter the **show storageArray Profile** command.)

Parameter	Description
userLabel	Name for the new virtual disk. You must put quotation marks (" ") around the new virtual disk name. **NOTE:** You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Spaces are not allowed. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.
freeCapacityArea	Index number of the free space in an existing disk group to use to create the new virtual disk. Free capacity is defined as the free capacity between existing virtual disks in a disk group. For example, a disk group might have the following areas: virtual disk 1, free capacity, virtual disk 2, free capacity, virtual disk 3, free capacity. To use the free capacity following virtual disk 2, specify: `freeCapacityArea=2` Use the **show diskGroup** command to determine if the free capacity area exists.
capacity	Size of the virtual disk that you are adding to the storage array. Size is defined in units of bytes, kilobytes, megabytes, or gigabytes. **NOTE:** If you do not specify a capacity, all of the available capacity in the free capacity area of the disk group is used. If you do not specify capacity units, bytes are used as the default units. A space must be added between the last digit and the size (MB, GB, or KB) for values greater than 9.

Parameter	Description
owner	RAID controller module that owns the virtual disk. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. If you do not specify an owner, the RAID controller module firmware determines the owner. **NOTE:** The *owner* parameter defines which RAID controller module owns the virtual disk. The preferred RAID controller module ownership of a virtual disk is the RAID controller module that currently owns the disk group.
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, 16, 32, 64, 128, 256, or 512. For information about the *segmentSize* parameter, see "Using the Auto Configure Command" on page 53.

Create RAID Virtual Disk (Manual Physical Disk Select)

This command creates a new disk group and virtual disk, and enables you to specify the physical disks for the virtual disk.

NOTE: You cannot use mixed physical disk types in the same disk group and virtual disk. This command fails if you specify different types of physical disks for the RAID virtual disk.

Syntax

```
create virtualDisk physicalDisks=
(enclosureID0,slotID0...enclosureIDn,slotIDn)
raidLevel=0 | 1 | 5 | 6 userLabel="virtualDiskName"
```

```
[capacity=virtualDiskCapacity owner=(0 | 1)
segmentSize=segmentSizeValue
enclosureLossProtect=(TRUE | FALSE)]
```

Parameters

Parameter	Description
physicalDisks	Specifies the physical disks to assign to the created virtual disk. Specify the enclosure ID and slot ID for each unassigned physical disk to assign to the virtual disk. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put parentheses around the enclosure ID values and the slot ID values.
raidLevel	RAID level of the disk group that contains the virtual disk. Valid values are 0, 1, 5, or 6. **NOTE:** If you set the *raidLevel* parameter to RAID 1, the RAID controller module firmware takes the list of physical disks and pairs them using the following algorithm: Data physical disk = X Consistency physical disk = $N/2 + X$ where X goes from 1 to $N/2$ and N is the number of physical disks in the list. The following data show an example of six physical disks and their mirror pairs. **Data** — **Consistency** 1 — $N/2 + 1 = 4$ 2 — $N/2 + 1 = 5$ 3 — $N/2 + 1 = 6$
userLabel	Name that you want to give the new virtual disk. You must put quotation marks (" ") around the new virtual disk name. **NOTE:** You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Spaces are not allowed. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Parameter	Description
capacity	Size of the virtual disk that you are adding to the storage array. Size is defined in units of bytes, kilobytes, megabytes, gigabytes, or terabytes. **NOTE:** If you do not specify a capacity, all physical disk capacity available in the disk group is used. If you do not specify capacity units, bytes are used as the default. A space must be added between the last digit and the size (MB, GB, or KB) for values greater than 9.
owner	RAID controller module that owns the virtual disk. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. If you do not specify an owner, the RAID controller module firmware determines the owner. **NOTE:** The *owner* parameter defines which RAID controller module owns the virtual disk. The preferred owner of a virtual disk is the RAID controller module that currently owns the disk group.
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, **16**, **32**, **64**, 128, **256**, or **512**. For information about the *segmentSize* parameter, see "Using the Auto Configure Command" on page 53.
enclosureLossProtect	Specifies that enclosure loss protection is enforced when creating the repository. To enforce enclosure loss protection, set this parameter to **TRUE**. The default setting is **FALSE**. For information about the *enclosureLossProtect* parameter, see "Enclosure Loss Protection" on page 52.

Create Snapshot Virtual Disk

This command creates a snapshot virtual disk. When you use this command, you can define the snapshot virtual disk in one of three ways:

- User-defined physical disks
- User-defined disk group
- User-defined number of physical disks for the snapshot virtual disk

If you choose to define a number of physical disks, the RAID controller module firmware chooses which physical disks to use for the snapshot virtual disk.

NOTE: Refer to "Preparing Host Servers to Create an Initial Snapshot Virtual Disk" on page 65.

NOTICE: Before you create a new point-in-time image of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows® Internet Explorer®, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the Snapshot.

Syntax (User-Defined Physical Disks)

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryRAIDLevel=
0 | 1 | 5 | 6 repositoryPhysicalDisks=
(enclosureID0,slotID0 ... enclosureIDn,slotIDn)
userLabel="snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot) enclosureLossProtect=(TRUE | FALSE)]
```

Syntax (User-Defined Disk Group)

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryDiskGroup=
diskGroupNumber freeCapacityArea=
```

```
freeCapacityIndexNumber userLabel=
"snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot) enclosureLossProtect=(TRUE | FALSE)]
```

Syntax (User-Defined Number of Physical Disks)

```
create snapshotVirtualDisk sourceVirtualDisk=
"sourceVirtualDiskName" [repositoryRAIDLevel=
0 | 1 | 5 | 6 repositoryPhysicalDiskCount=
numberOfPhysicalDisks
physicalDiskType=(SAS | SATA) userLabel=
"snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryPercentOfSource=percentValue
repositoryUserLabel="repositoryName"
repositoryFullPolicy=(failSourceWrites |
failSnapShot) enclosureLossProtect=(TRUE | FALSE)]
```

Parameters

Parameter	Description
sourceVirtualDisk	Name of the source virtual disk from which to take a snapshot. You must put quotation marks (" ") around the source virtual disk name.
repositoryRAIDLevel	RAID level for the repository virtual disk. Valid values are 0, 1, 5, or 6.
repositoryPhysicalDisks	Specifies the physical disks to assign to the repository. Specify the enclosure ID and slot ID for each physical disk assigned to the virtual disk. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put parentheses around the list of repository physical disks.

Parameter	Description
repositoryPhysicalDiskCount	Number of unassigned physical disks to use for the repository virtual disk.
physicalDiskType	Type of physical disks to use for the repository virtual disk. Valid physical disk types are SAS or SATA.
repositoryDiskGroup	Sequence number of the disk group where the repository virtual disk is located.
freeCapacityArea	The index number of the free space in an existing disk group to use to create the snapshot repository virtual disk. Free capacity is defined as the free capacity between existing virtual disks in a disk group. For example, a disk group might have the following areas: virtual disk 1, free capacity, virtual disk 2, free capacity, virtual disk 3, free capacity. To use the free capacity following virtual disk 2, you specify: `freeCapacityArea=2` Use the **show diskGroup** command to determine if free capacity area exists. **NOTE:** If you do not specify the unconfigured or free space, the repository virtual disk is placed in the same disk group as the source virtual disk. If the disk group where the source virtual disk resides does not have enough space, this command fails.
userLabel	The name to give the snapshot virtual disk. You must put quotation marks (" ") around the snapshot virtual disk name.
warningThresholdPercent	The percentage of repository capacity at which you receive a warning that the repository is nearing full. Use integer values. For example, a value of 70 means 70 percent. The default value is 50.

Parameter	Description
repositoryPercentOfSource	The size of the repository virtual disk as a percentage of the source virtual disk. Use integer values. For example, a value of 40 means 40 percent. The default value is 20.
repositoryUserLabel	The name to give to the repository virtual disk. You must put quotation marks (" ") around the repository virtual disk name.
repositoryFullPolicy	Specifies how snapshot processing continues if the repository is full. You can choose to fail writes to the source virtual disk (**failSourceWrites**) or fail the snapshot virtual disk (**failSnapShot**). The default value is **failSnapShot**.
enclosureLossProtect	Specifies that enclosure loss protection is enforced when creating the repository. To enforce enclosure loss protection, set this parameter to **TRUE**. The default setting is **FALSE**. For information about the *enclosureLossProtect* parameter, see "Enclosure Loss Protection" on page 52.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

NOTE: One technique for naming the snapshot virtual disk and the repository virtual disk is to add a hyphenated suffix to the original name of the source virtual disk. The suffix distinguishes between the snapshot virtual disk and the repository virtual disk. For example, if you have a source virtual disk with a name of **Engineering Data**, the snapshot virtual disk can have a name of **Engineering Data-S1**, and the repository virtual disk can have a name of **Engineering Data-R1**.

NOTE: If you do not choose a name for the either the snapshot virtual disk or repository virtual disk, the RAID controller modules create a default name using the source virtual disk name. For example, if the source virtual disk name is **aaa** and does not have a snapshot virtual disk, the default snapshot virtual disk name is **aaa** – **1**. If the source virtual disk already has *n* – 1 number of snapshot virtual disks, the default name is **aaa** – ***n***. If the source virtual disk name is **aaa** and the source virtual disk does not have a repository virtual disk, the default repository virtual disk name is **aaa** – **R1**. If the source virtual disk already has *n* – 1 number of repository virtual disks, the default name is **aaa** – **R*n***.

Create Virtual Disk Copy

This command creates a virtual disk copy and starts the virtual disk copy operation.

NOTE: Refer to "Preparing Host Servers to Create a Virtual Disk Copy" on page 80.

NOTICE: Before you create a new copy of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk and (if applicable, the target disk) to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk(s) in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the virtual copy.

NOTE: You can have a maximum of eight virtual disk copies in progress at one time. If you try to create more than eight virtual disk copies at one time, the RAID controller modules return a status of Pending until one of the virtual disk copies that is in progress finishes and returns a status of Complete.

Syntax

```
create virtualDiskCopy source="sourceName" target=
"targetName" [copyPriority=(highest | high |
medium | low | lowest) targetReadOnlyEnabled=(TRUE
| FALSE)]
```

Parameters

Parameter	Description
source	Name of an existing virtual disk to use as the source virtual disk. You must put quotation marks (" ") around the source virtual disk name. **NOTE:** You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.
target	Name of an existing virtual disk to use as the target virtual disk. You must put quotation marks (" ") around the target virtual disk name. **NOTE:** You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Parameter	Description
copyPriority	Specifies the priority that a virtual disk copy has relative to host I/O activity. Valid entries are highest, high, medium, low, or lowest. **NOTE:** CopyPriority defines the amount of system resources used to copy the data between the source virtual disk and target virtual disk of a virtual disk-copy pair. If you select the highest priority level, the virtual disk copy uses the most system resources to perform virtual disk copy, which decreases performance for host data transfers.
targetReadOnlyEnabled	Specifies whether the target virtual disk is write enabled or read only. To be able to write to the target virtual disk, set this parameter to **FALSE**. To prevent writing to the target virtual disk, set this parameter to **TRUE**.

Delete Disk Group

This command deletes an entire disk group and its associated virtual disks.

NOTICE: All data in the disk group is lost as soon as you run this command.

Syntax

```
delete diskGroup [diskGroupNumber]
```

Parameters

Parameter	Description
diskGroup	Number of the disk group to delete. You must put brackets ([ ]) around the disk group number.

Delete Host

This command deletes a host.

Syntax

```
delete host [hostName]
```

Parameters

Parameter	Description
host	Name of the host to delete. You must put brackets ([ ]) around the host name. If the host name has special characters, you must also put quotation marks (" ") around the host name. **NOTE:** A host is a system that is attached to the storage array and accesses the virtual disks on the storage array through its HBA host ports.

Delete Host Group

This command deletes a host group.

NOTICE: This command deletes all of the host definitions in the host group.

Syntax

```
delete hostGroup [hostGroupName]
```

Parameters

Parameter	Description
hostGroup	Name of the host group to delete. You must put brackets ([ ]) around the host group name. If the name of the host group has special characters, you must also put quotation marks (" ") around the host group name. **NOTE:** A host group is an optional topological element that is a collection of hosts that share access to the same virtual disks. The host group is a logical entity.

Delete Host Port

This command deletes an HBA host port identification. The identification is a software value that represents the physical HBA host port to the RAID controller module. By deleting the identification, the RAID controller module no longer recognizes instructions and data from the HBA host port.

Syntax

```
delete hostPort [hostPortName]
```

Parameters

Parameter	Description
hostPort	Name of the HBA host port to delete. You must put brackets ([ ]) around the name of the HBA host port. **NOTE:** An HBA host port is a physical connection on a host bus adapter that resides within a host system. An HBA host port provides a host access to the virtual disks in a storage array. If the host bus adapter has only one physical connection (one host port), the terms HBA host port and host bus adapter are synonymous.

Example

```
-c "delete host [\"job2900\"];"
```

Delete iSCSI Initiator

This command deletes a specific iSCSI initiator object.

Syntax

```
delete iscsiInitiator ([iSCSI-ID | name])
```

Parameters

Parameters	Description
iSCSI-ID	The identifier of the iSCSI initiator that you want to delete. Enclose the name in double quotation marks (" ").
name	The name of the iSCSI initiator that you want to delete. Enclose the name in double quotation marks (" ").

Example

```
-c "delete iscsiInitiator [\"job29002\"];"
```

Delete Virtual Disk

This command deletes one or more standard virtual disks or snapshot and snapshot repository virtual disks.

NOTICE: All of the data in the virtual disk is lost as soon as you run this command.

Syntax

```
delete (allVirtualDisks | virtualDisk
[virtualDiskName] | virtualDisks
[virtualDiskName1... virtualDiskNameN])
```

Parameters

Parameter	Description
allVirtualDisks	Deletes all virtual disks in a storage array. **NOTE:** Using the *allVirtualDisks* parameter deletes virtual disks until all are removed or until an error is encountered. If an error is encountered, this command does not attempt to delete the remaining virtual disks.

Parameter	Description
virtualDisk or virtualDisks	Name of the virtual disk to delete. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.

Diagnose RAID Controller Module

This command runs diagnostic tests on the RAID controller module.

The *testID* parameter takes the following options, which you can use to verify that a RAID controller module is functioning correctly:

- 1 — Reads the test
- 2 — Performs a data loop-back test
- 3 — Writes the test

The diagnostic tests consist of loop-back tests in which data is written to physical disks and read from the physical disks.

Syntax

```
diagnose controller [(0 | 1)]
loopbackPhysicalDiskChannel=(allchannels | (1 | 2
)) testID=(1 | 2 | 3 | discreteLines)
[patternFile="filename"]
```

Parameters

Parameter	Description
controller	RAID controller module on which to run the diagnostic tests. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier. If you do not specify a RAID controller module, the storage management software returns a syntax error.
loopbackPhysicalDiskChannel	Physical disk channel on which to run the diagnostic tests. You can choose to run the diagnostics on all channels or select a specific channel on which to run diagnostics. Valid physical disk channel values are 1 or 2. **NOTE:** When you run a data loopback test, you can optionally specify a file that contains a data pattern. If you do not specify a file, the firmware provides a default pattern.

Parameter	Description

testID	Identifier for the diagnostic test to run. The identifier and corresponding tests are:
	1 — Reads the test
	2 — Performs a data loop-back test
	3 — Writes the test
	discreteLines — Discrete lines diagnostic test
	NOTE: Discrete lines are control and status lines connected between two RAID controller modules in a RAID controller. The discrete lines test enables each RAID controller module to verify that control signal transitions can be observed at the control inputs of the alternate RAID controller module. The discrete line test automatically runs after each power cycle or RAID controller module reset. You can run the discrete lines diagnostic test after you have replaced a component that failed the initial discrete lines diagnostic test. When the test runs successfully, the following message is shown:
	`The controller discrete lines successfully passed the diagnostic test. No failures were detected.`
	If the test fails, the following message is shown:
	`One or more controller discrete lines failed the diagnostic test.`
	If the CLI cannot run the test, the CLI returns Error 270, which means the diagnostic test could not start or complete.

Parameter	Description
patternFile	Name of a file that contains a data pattern to use as test data. You must put quotation marks (" ") around the data pattern file name.

Disable Storage Array Feature

This command disables a storage array feature. Issue the **show storageArray** command to display a list of the feature identifiers for all enabled features in the storage array.

Syntax

```
disable storageArray feature=(snapshot |
virtualDiskCopy)
```

Parameters

None.

Download Enclosure Management Module Firmware

This command downloads firmware for the enclosure management module (EMM).

Syntax

```
download (allEnclosures | enclosure [enclosureID])
firmware file="filename"
```

Parameters

Parameter	Description
enclosure	Identifies the enclosure to which to load new firmware. Enclosure ID values are 0 to 99. You must put brackets ([ ]) around the enclosure ID value. **NOTE:** You can use the following parameters: (1) the *allEnclosures* parameter, which downloads new firmware to all of the EMMs in the storage array, and (2) the *enclosure* parameter, which downloads new firmware to a specific EMM. If you need to download new firmware to more than one EMM, but not all EMMs, you must enter this command for each enclosure.
file	File path and file name of the file that contains the firmware image. You must put quotation marks (" ") around the firmware image file path and file name.

Download Physical Disk Firmware

This command downloads a firmware image to a physical disk.

NOTICE: Downloading physical disk firmware incorrectly can result in damage to the physical disks or loss of data.

Before attempting to download physical disk firmware, you must take the following precautions:

1. Stop all I/O activity to the storage array before downloading the firmware image.
2. Ensure the firmware image file is compatible with the physical disk enclosure. If you download a file that is not compatible with the selected physical disk enclosure, the enclosure might become unusable.
3. Do not make any configuration changes to the storage array while downloading the physical disk firmware. Attempting to make a configuration change can cause the firmware download to fail and make the selected physical disks unusable.

You can use this command to test the firmware on one physical disk before installing the firmware on all of the physical disks in a storage array. (Use the **download storageArray physicalDiskFirmware** command to download firmware on all of the physical disks in the storage array.) This command blocks all I/O activity until the download finishes or fails. The download returns one of the following statuses: Successful, Unsuccessful With Reason, or Never Attempted With Reason.

Syntax

```
download physicalDisk [enclosureID,slotID]
firmware file="filename"
```

Parameters

Parameter	Description
physicalDisk	Physical disk to which to download the firmware image. Specify the enclosure ID and slot ID for the physical disk. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and slot ID value.
filename	File path and file name of the file that contains the firmware image. You must put quotation marks (" ") around the firmware image file path and file name.

Download Storage Array Firmware/NVSRAM

This command downloads firmware and, optionally, NVSRAM values for the RAID controller module in the storage array. If you want to download only NVSRAM values, use the **downLoad storageArray NVSRAM** command.

Syntax

```
download storageArray firmware [, NVSRAM ] file=
"filename" [, "NVSRAM-filename"] [downgrade=(TRUE
| FALSE)] [activateNow=(TRUE | FALSE)]
```

Parameters

Parameter	Description
NVSRAM	Specifies that you want to download a file with NVSRAM values when you download a firmware file. You must not put brackets around this parameter. Include a comma after the term **firmware**.
file	File path and name of the file that contains the firmware. Valid file names must end with a **.dlp** extension. You must put quotation marks (" ") around the file name.
NVSRAM-filename	File path and name of the file that contains the NVSRAM values. Valid file names must end with a **.dlp** extension. You must put quotation marks (" ") around the NVSRAM file name. You must include a comma after the firmware file name.
downgrade	Specifies that you are loading firmware that is a previous version. The default value is **FALSE**. Set downgrade to **TRUE** if you want to download an earlier version of firmware.
activateNow	Activates the firmware and NVSRAM images. The default value is **TRUE**. If you set **activateNow** to **FALSE**, you must use the **activate storageArray firmware** command to activate the firmware and NVSRAM values at a later time.

Download Storage Array NVSRAM

This command downloads NVSRAM values for the storage array RAID controller module.

Syntax

```
download storageArray NVSRAM file="filename"
```

Parameters

Parameter	Description
file	File path and name of the file that contains the NVSRAM values. Valid file names must end with a **.dlp** extension. You must put quotation marks (" ") around the file name.

Download Storage Array Physical Disk Firmware

This command downloads firmware images to all physical disks in the storage array.

Syntax

```
download storageArray physicalDiskFirmware file=
"filename" [file="filename2"...file="filenamen"]
```

Parameters

Parameter	Description
file	File path and name of the file that contains the firmware image. You must put quotation marks (" ") around the firmware image file path and file name.

NOTE: When you run this command, you can download more than one firmware image file to the physical disks in a storage array. The number of firmware image files you can download depends on the storage array. The storage management software returns an error if you attempt to download more firmware image files than the storage array can accept.

NOTE: You can schedule downloads for multiple physical disks at the same time, including multiple physical disks in a redundant disk group. Each firmware image file contains information about the physical disk types on which the image runs. The specified firmware images can be downloaded only to a compatible physical disk. Use the **download physicalDisk firmware** command to download an image to a specific physical disk.

NOTE: The **download storageArray physicalDiskFirmware** command blocks all I/O activity until a download attempt has been made for each candidate physical disk or you issue the **stop storageArray downloadPhysicalDiskFirmware** command. When the **download storageArray physicalDiskFirmware** command finishes downloading the firmware image, each candidate physical disk is displayed showing the download status for each physical disk. One of the following download status messages is shown: **Successful**, **Unsuccessful With Reason**, or **Never Attempted With Reason**.

Enable RAID Controller Module

This command revives a RAID controller module that quiesces while running diagnostics.

Syntax

```
enable controller [(0 | 1)] dataTransfer
```

Parameters

Parameter	Description
controller	RAID controller module that you want to revive. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier. If you do not specify a RAID controller module, the storage management software returns a syntax error.

Enable Storage Array Feature Key

This command enables a feature using a feature key file.

Syntax

```
enable storageArray feature file="filename"
```

Parameters

Parameter	Description
file	File path and file name of a valid feature key file. Valid file names for feature key files must end with a **.key** extension. You must put quotation marks (" ") around the file path and file name.

Recopy Virtual Disk Copy

This command reinitiates a virtual disk copy operation by using an existing virtual disk copy pair.

NOTE: Refer to "Preparing Host Servers to Recopy a Virtual Disk" on page 85.

NOTICE: Before you create a new copy of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk and (if applicable, the target disk) to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows Internet Explorer, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk(s) in Windows or unmounting the vitual drive in Linux will help to guarantee a stable copy of the drive for the virtual copy.

Syntax

```
recopy virtualDiskCopy target [targetName] [source
[sourceName]] [copyPriority=(highest | high |
medium | low | lowest) targetReadOnlyEnabled=(TRUE
| FALSE)]
```

Parameters

Parameter	Description
target	Name of the target virtual disk for which to reinitiate a virtual disk copy operation. You must put brackets ([ ]) around the target virtual disk name. If the target virtual disk name has special characters, you must also put quotation marks (" ") around the target virtual disk name.
source	Name of the source virtual disk for which to reinitiate a virtual disk copy operation. You must put brackets ([ ]) around the source virtual disk name. If the source virtual disk name has special characters, you must also put quotation marks (" ") around the source virtual disk name.
copyPriority	Specifies the priority that the virtual disk copy has relative to host I/O activity. Valid entries are highest, high, medium, low, or lowest. **NOTE:** CopyPriority defines the amount of system resources used to copy the data between the source and target virtual disks of a virtual disk copy pair. If you select the highest priority level, the virtual disk copy uses the most system resources to perform virtual disk copy, which decreases performance for host data transfers.
targetReadOnlyEnabled	Specifies whether the target virtual disk is write enabled or read only. To be able to write to the target virtual disk, set this parameter to **FALSE**. To prevent writing to the target virtual disk, set this parameter to **TRUE**.

Recover RAID Virtual Disk

This command creates a RAID virtual disk with the given properties without initializing any of the user data areas on the disks. Parameter values are derived from the Recovery Profile data file for the storage array.

Syntax

```
recover virtualDisk (physicalDisk=(trayID,slotID)
|
physicalDisks=(trayID1,slotID1 ...
trayIDn,slotIDn) |
```

```
diskGroup=diskGroupNumber) [newVolumeGroup=
VolumeGroupName]
userLabel="virtualDiskName" capacity=
virtualDiskCapacity
offset=offsetValue raidLevel=(0 | 1 | 5 | 6)
segmentSize=segmentSizeValue [owner=(0 | 1)
cacheReadPrefetch=(TRUE | FALSE)]
```

Parameters

Parameter	Description
physicalDisk or physicalDisks	Specifies the physical disks to assign to the virtual disk that you want to create. Specify the tray ID and slot ID for each physical disk that you assign to the virtual disk. Tray ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the tray ID value and the slot ID value.
newVolumeGroup	This parameter enables the user to specify a name for a new volume group that is automatically created by the RAID controller module.
userLabel	Name to give the new virtual disk. Enclose the new virtual disk name in double quotation marks (" ").
capacity	Size of the virtual disk that you are adding to the storage array. Size is defined in units of bytes.
offset	Number of blocks from the beginning of the disk group to the beginning of the referenced virtual disk (1 block is equal to 512 bytes).
raidLevel	RAID level of the disk group that contains the physical disks. Valid values are **0**, **1**, **5**, or **6**.
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, **16**, **32**, **64**, 128, **256**, or **512**. For information about the *segmentSize* parameter, see "Using the Auto Configure Command" on page 53.

Parameter	Description
owner	RAID controller module that owns the virtual disk. Valid RAID controller module identifiers are **0** or **1**, where **0** is the RAID controller module in Slot **0** and **1** is the RAID controller module in Slot **1**. If you do not specify an owner, the RAID controller module firmware determines the owner. For information about the *owner* parameter, see "Creating Virtual Disks with User-Assigned Physical Disks" on page 48.
cacheReadPrefetch	The setting to turn on or turn off *cacheReadPrefetch*. To turn off *cacheReadPrefetch*, set this parameter to **FALSE**. To turn on *cacheReadPrefetch* set this parameter to **TRUE**.

Additional Information

You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Names can have a maximum of 30 characters. The *owner* parameter defines which RAID controller module owns the volume. The preferred controller ownership of a volume is the RAID controller module that currently owns the disk group.

Segment Size

The size of a segment determines how many data blocks that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Each data block stores 512 bytes of data. A data block is the smallest unit of storage. The size of a segment determines how many data blocks that it contains. For example, an 8-KB segment holds 16 data blocks. A 64-KB segment holds 128 data blocks.

When you enter a value for the segment size, the value is checked against the supported values that are provided by the RAID controller module at run time. If the value that you entered is not valid, the RAID controller module returns a list of valid values. Using a single physical disk for a single request leaves other physical disks available to simultaneously service other requests.

If the virtual disk is in an environment where a single user is transferring large units of data (such as multimedia), performance is maximized when a single data transfer request is serviced with a single data stripe (a data stripe is the segment size that is multiplied by the number of physical disks in the volume

group that are used for data transfers). In this case, multiple physical disks are used for the same request, but each physical disk is accessed only once. For optimal performance in a multiuser database or file system storage environment, set your segment size to minimize the number of physical disks that are required to satisfy a data transfer request.

CacheReadPrefetch

Cache read prefetch lets the RAID controller module copy additional data blocks into cache while the RAID controller module reads and copies data blocks that are requested by the host from disk into cache. This action increases the chance that a future request for data can be fulfilled from cache. Cache read prefetch is important for multimedia applications that use sequential data transfers. The configuration settings for the storage array that you use determine the number of additional data blocks that the RAID controller module reads into cache.

Valid values for the *cacheReadPrefetch* parameter are **TRUE** or **FALSE**.

Re-create Snapshot

This command starts a new copy-on-write operation by using an existing snapshot virtual disk. You can re-create a single snapshot virtual disk or re-create multiple virtual disks.

NOTE: Refer to "Preparing Host Servers to Re-create a Snapshot Virtual Disk" on page 75.

NOTICE: Before you create a new point-in-time image of a source virtual disk, stop any data access (I/O) activity or suspend data transfer to the source virtual disk to ensure that you capture an accurate point-in-time image of the source virtual disk. Close all applications, including Windows® Internet Explorer®, to make sure all I/O activity has stopped.

NOTE: Removing the drive letter of the associated virtual disk in Windows or unmounting the virtual drive in Linux will help to guarantee a stable copy of the drive for the Snapshot.

Syntax

```
recreate snapshot (virtualDisk [virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen]) [userLabel=
"snapshotVirtualDiskName"
```

```
warningThresholdPercent=percentValue
repositoryFullPolicy=(failSourceWrites |
failSnapShot)]
```

Parameters

Parameter	Description
virtualDisk or virtualDisks	Name of the specific virtual disk for which to start a fresh copy-on-write operation. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.
userLabel	Name of the snapshot virtual disk. You must put quotation marks (" ") around the snapshot virtual disk name. If you enter more than one snapshot virtual disk name, this command fails.
warningThresholdPercent	Percentage of repository capacity at which you receive a warning that the repository is nearing full. Use integer values. For example, a value of 70 means 70 percent. The default value is 50 percent. **NOTE:** If warningThresholdPercent is not specified, the previously set value is used.
repositoryFullPolicy	Specifies how snapshot processing continues if the repository is full. You can choose to fail writes to the source virtual disk (**failSourceWrites**) or fail writes to the snapshot virtual disk (**failSnapShot**). The default value is **failSnapShot.** **NOTE:** If repositoryFullPolicy is not specified, the previously set value is used.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

NOTE: If the *warningThresholdPercent* parameter or the *repositoryFullPolicy* parameter is not specified, the previously set values are used. If you use the optional parameters, the re-create Snapshot will process each snapshot virtual disk separately. If the same userLabel (name) is specified for more than one virtual disk, the command fails. If no optional parameters are specified, the re-create Snapshot will process the specified snapshot virtual disks as a batch, provided a validation check of all of the virtual disks runs successfully. If successful, the snapshots start the re-creation process and all of the affected virtual disks (snapshot, source, and repository) are quiesced until the process is complete.

NOTE: If no optional parameters are specified, the **recreate snapshot** command will process the specified snapshot virtual disks as a batch, provided a validation check of all of the virtual disks runs successfully. If successful, the snapshots start the re-creation process and all of the affected virtual disks (snapshot, source, and repository) are quiesced until the process is complete.

NOTE: The Microsoft Virtual Shadow Copy Service (VSS) provider allows the recreation of multiple snapshots simultaneously.

Remove Virtual Disk Copy

This command removes a virtual disk copy pair.

Syntax

```
remove virtualDiskCopy target [targetName] [source
[sourceName]]
```

Parameters

Parameter	Description
target	Name of the target virtual disk to remove. You must put brackets ([ ]) around the target virtual disk name. If the target virtual disk name has special characters, you must also put quotation marks (" ") around the target virtual disk name.
source	Name of the source virtual disk to remove. You must put brackets ([ ]) around the source virtual disk name. If the source virtual disk name has special characters, you must also put quotation marks (" ") around the source virtual disk name.

Remove Virtual Disk LUN Mapping

This command removes the logical unit number (LUN) mapping.

Syntax

```
remove (allVirtualDisks | virtualDisk
["virtualDiskName"] |
virtualDisks ["virtualDiskName1" ...
"virtualDiskNamen"] | accessVirtualDisk)
lunMapping (host="hostName" | hostGroup=
"hostGroupName")
```

Parameters

Parameter	Description
allVirtualDisks	Removes the LUN mapping from all virtual disks.
virtualDisk or virtualDisks	Name of the specific virtual disk to remove from the LUN mapping. You can enter more than one virtual disk name. You must put quotation marks (" ") and brackets ([ ]) around the virtual disk name. The virtual disk name and quotation marks must be inside the brackets.
accessVirtualDisk	Removes the access virtual disk. **NOTICE:** The host agent uses the access virtual disks to communicate in-band with a storage array. If you remove an access virtual disk mapping for a storage array from a host that has an agent running on it, the storage management software is no longer able to manage the storage array through the in-band agent.
host	Name of the host to which the virtual disk is mapped. You must put quotation marks (" ") around the host name.
hostGroup	Name of the host group that contains the host to which the virtual disk is mapped. You must put quotation marks (" ") around the host group name.

NOTE: You must use the *host* and *hostGroup* parameters when specifying a nonaccess virtual disk or an access virtual disk. The script engine ignores the *host* or *hostGroup* parameters when you use the *allVirtualDisks* or *virtualDisks* parameters.

Repair Virtual Disk Consistency

This command repairs the consistency errors on a virtual disk.

Syntax

```
repair virtualDisk [virtualDiskName] consistency
consistencyErrorFile=filename [verbose=(TRUE |
FALSE)]
```

Parameters

Parameter	Description
virtualDisk	Name of the specific virtual disk for which to repair consistency. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.
consistencyErrorFile	Name of the file that contains the consistency error information you use to repair the errors. You must put quotation marks (" ") around the file name.
verbose	Captures progress details, such as percent complete, and shows the progress detail information as virtual disk consistency is being repaired. To capture progress details, set this parameter to **TRUE**. To prevent capturing progress details, set this parameter to **FALSE**.

Reset RAID Controller Module

This command resets a RAID controller module.

NOTE: When you reset a RAID controller module, the RAID controller module is not available for I/O operations until the reset is complete. If a host is using virtual disks owned by the RAID controller module being reset, the I/O directed to the RAID controller module is rejected. Before resetting the RAID controller module, either verify that the virtual disks owned by the RAID controller module are not in use or ensure a multipath driver is installed on all hosts using these virtual disks.

Syntax

```
reset controller [(0 | 1)]
```

Parameters

Parameter	Description
controller	RAID controller module to reset. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier. If you do not specify an owner, the RAID controller module firmware returns a syntax error.

Reset Storage Array Battery Install Date

This command resets the age of the batteries in a storage array to zero days. You can reset the batteries for an entire storage array or just the battery for a specific RAID controller module or in a specific battery pack.

Syntax

```
reset storageArray batteryInstallDate controller=
(0 | 1)
```

Parameters

Parameter	Description
controller	Specifies the RAID controller module that contains the battery for which to reset the age. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left, and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. Use the *controller* parameter only for RAID controller modules with batteries.

NOTE: If you do not specify a RAID controller module, the age is reset for the storage array battery or both RAID controller module batteries. If you specify a RAID controller module, then the age for only that RAID controller module battery is reset.

Reset Storage Array iSCSI Baseline

This command resets the iSCSI baseline for the storage array to 0.

Syntax

```
reset storageArray iscsiStatsBaseline
```

Parameters

None.

NOTE: This command resets the baseline to 0 for both RAID controller modules in the storage array. The purpose of resetting both of the RAID controller module baselines is to help ensure that the counts are synchronized between the RAID controller modules. If one RAID controller module resets but the second RAID controller module does not reset, the host is informed that the RAID controller modules are out of synchronization. The host is informed by the time stamps that are reported with the statistics.

Example

```
-c "reset storageArray iscsiStatsBaseline;"
```

Reset Storage Array SAS PHY Baseline

This command resets the SAS PHY baseline for all SAS devices in a storage array.

Syntax

```
reset storageArray SASPHYBaseline
```

Parameters

None.

Example

```
-c "delete host [\"job2900\"];"
```

Reset Storage Array Virtual Disk Distribution

This command reassigns (moves) all virtual disks to their preferred RAID controller module.

Syntax

```
reset storageArray virtualDiskDistribution
```

Parameters

None.

NOTICE: Ensure that the multipath driver is running before you use this command, or the virtual disk will not be accessible.

NOTE: Under certain host operating system environments, you might be required to reconfigure the multipath host physical disk. You might also need to make operating system modifications to recognize the new I/O path to the virtual disks.

Revive Disk Group

This command forces the specified disk group and associated failed physical disks to the Optimal state. All physical disks assigned to the disk group must be installed before you attempt to run this command.

NOTICE: Correct use of this command depends on the data configuration on all of the physical disks in the disk group. Never attempt to revive a physical disk unless supervised by a Customer or Technical Support representative.

NOTICE: Do not attempt to run this command o a disk group that is in the Degraded state. Running this command on a disk group that is in the Degraded state can cause loss of access to the data on the physical disks in the disk group.

Syntax

```
revive diskGroup [diskGroupNumber]
```

Parameters

Parameter	Description
diskGroup	Number of the disk group to be set to the Optimal state. You must put brackets ([ ]) around the disk group number.

Revive Physical Disk

This command forces the specified physical disk to the Optimal state.

NOTICE: Correct use of this command depends on the data configuration on all physical disks in the disk group. Never attempt to revive a physical disk unless supervised by a Technical Support representative.

Syntax

```
revive physicalDisk [enclosureID,slotID]
```

Parameters

Parameter	Description
physicalDisk	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and the slot ID value.

Save Enclosure Log Data

This command saves the log data maintained by the EMM in all enclosures in a storage array to a file.

Syntax

```
save allEnclosures logFile="filename"
```

Parameters

Parameter	Description
logFile	Name of the file to which to save the EMM log data. You must put quotation marks (" ") around the file name.

Save Physical Disk Channel Fault Isolation Diagnostic Status

This command saves the physical disk channel fault isolation diagnostic data that is returned from the **start physical disk channel fault isolation diagnostics** command. You can save the diagnostic data to a file as standard text or as XML.

See "Start Physical Disk Channel Fault Isolation Diagnostics" on page 224 for more information.

Syntax

```
save physicalDiskChannel[(0 | 1)] faultDiagnostics
file="filename"
```

Parameters

Parameter	Description
file	The name of the file in which you are storing the results of the fault isolation diagnostics test on the drive channel. Enclose the name in double quotation marks (" ").

NOTE: A file extension is not automatically appended to the saved file. You must specify the applicable format suffix for the file. If you specify a file extension of .txt, then the output will be in a text file format. If you specify a file extension of .xml, then the output will be in an XML file format.

Save Physical Disk Log

This command saves the log sense data to a file. Log sense data is maintained by the storage array for each physical disk.

Syntax

```
save allPhysicalDisks logFile="filename"
```

Parameters

Parameter	Description
logFile	Name of the file to which to write the log sense data. You must put quotation marks (" ") around the file name.

Save RAID Controller Module NVSRAM

This command saves a copy of the RAID controller module NVSRAM values to a file. This command saves all regions.

Syntax

```
save controller [(0 | 1)] NVSRAM file="filename"
```

Parameters

Parameter	Description
controller	RAID controller module with the NVSRAM values to save. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier.
file	Name of the file in which to save the values. You must put quotation marks (" ") around the file name.

Save Storage Array Configuration

This command creates a script file to use to create the current storage array virtual disk configuration.

Syntax

```
save storageArray configuration file="filename"
[(allConfig | globalSettings=(TRUE | FALSE)
virtualDiskConfigAndSettings=(TRUE | FALSE)
hostTopology=(TRUE | FALSE)
lunMappings=(TRUE | FALSE))]
```

Parameters

Parameter	Description
file	Name of the file that contains the configuration values. You must put quotation marks (" ") around the file name.
allConfig	Saves all of the configuration values to the file. (If you choose this parameter, all of the configuration parameters are set to **TRUE**.)
globalSettings	Saves the global settings to the file. To save the global settings, set this parameter to TRUE. To prevent saving the global settings, set this parameter to **FALSE**. The default value is **TRUE**.

Parameter	Description
virtualDiskConfigAndSettings	Saves the virtual disk configuration settings and all of the global settings to the file. To save the virtual disk configuration and global settings, set this parameter to **TRUE**. To prevent saving the virtual disk configuration and global settings, set this parameter to **FALSE**. The default value is **TRUE**.
hostTopology	Saves the host topology to the file. To save the host topology, set this parameter to **TRUE**. To prevent saving the host topology, set this parameter to **FALSE**. The default value is **FALSE**.
lunMappings	Saves the LUN mapping to the file. To save the LUN mapping, set this parameter to **TRUE**. To prevent saving the LUN mapping, set this parameter to **FALSE**. The default value is **FALSE**.

NOTE: When you use this command, you can specify any combination of the parameters for global setting, virtual disk configuration setting, host topology, or LUN mapping. To enter all settings, use the *allConfig* parameter. The parameters are optional; you do not have to enter any parameters.

Save Storage Array Events

This command saves events from the Major Event Log (MEL) to a file. You can save either all the events or only the critical events.

Syntax

```
save storageArray (allEvents | criticalEvents)
file="filename" [count=numberOfEvents]
```

Parameters

Parameter	Description
allEvents | criticalEvents	Specifies whether to save all events (**allEvents**) or only the critical events (**criticalEvents**).
file	Name of the file to which to save the events. You must put quotation marks (" ") around the file name.

Parameter	Description
count	Specifies the number of events or critical events to save to a file. If you do not enter a value for the count, all events or critical events are saved to the file. If you enter a value for the count, only that number of events or critical events (starting with the last event entered) are saved to the file. Use integer values.

Save Storage Array iSCSI Statistics

This command saves the storage array iSCSI performance statistics to a file. The following statistics are saved to the file:

- Statistics related to the physical Ethernet port
- Statistics related to the TCP protocol
- Statistics related to the IP protocol

Syntax

```
save storageArray iscsiStatistics [raw | baseline]
file="filename"
```

Parameters

Parameter	Description
raw	This parameter defines that the statistics collected are all statistics from the RAID controller module start-of-day. Enclose the parameter in square brackets ([ ]).
baseline	This parameter defines that the statistics collected are all statistics from the time the RAID controller modules were reset to zero using the **reset storageArray iscsiStatsBaseline** command. Enclose the parameter in square brackets ([ ]).
file	The name of the file to which you want to save the performance statistics. Enclose the file name in double quotation marks (" ").

NOTE: If you have not reset the iSCSI baseline statistics since the RAID controller module start-of-day, the time at the start-of-day is the default baseline time.

NOTE: This command does not automatically append a file extension to the new file. You must specify the file extension when you enter the file name.

Example

```
-c "save storageArray iscsiStatistics [raw] file =
\"testfile\";"
```

Save Storage Array Performance Statistics

This command saves the performance statistics to a file. Before you use this command, issue the **set session performanceMonitorInterval** and **set session performanceMonitorIterations** commands to specify how often statistics are collected.

Syntax

```
save storageArray performanceStats file="filename"
```

Parameters

Parameter	Description
file	Name of the file to which to save the performance statistics. You must put quotation marks (" ") around the file name.

Save Storage Array SAS PHY Counts

This command saves the storage array SAS PHY counters to a file.

Syntax

```
save storageArray SASPHYCounts file="filename"
```

Parameters

Parameter	Description
file	Name of the file to which to save the storage array SAS PHY counters. You must put quotation marks (" ") around the file name.

Save Storage Array State Capture

This command saves the state capture to a file.

Syntax

```
save storageArray stateCapture file="filename"
```

Parameters

Parameter	Description
file	Name of the file to which to save the state capture. You must put quotation marks (" ") around the file name.

Save Storage Array Support Data

This command saves the storage array support-related information to a file.

Syntax

```
save storageArray supportData file="filename"
```

Parameters

Parameter	Description
file	Name of the file to which to save the storage array support-related data. You must put quotation marks (" ") around the file name.

Set Controller

This command defines the attributes for the RAID controller modules.

Syntax

```
set controller [(0 | 1)]
availability=(online | offline | serviceMode)
ethernetPort [1] = ethernet-port-options
globalNVSRAMByte [nvsramOffset]=
(nvsramByteSetting | nvsramBitSetting) |
hostNVSRAMByte [hostType, nvsramOffset]=
```

```
(nvsramByteSetting | nvsramBitSetting) |
iscsiHostPort [(1 | 2)] = iscsi-host-port-options
rloginEnabled=(TRUE | FALSE)
```

Parameters

Parameter	Description
controller	This parameter is the RAID controller module for which you want to define properties. Valid identifiers for the RAID controller module are **0** or **1**, where **0** is the RAID controller module on the left and **1** is the RAID controller module on the right when viewed from the rear of the RAID enclosure. You must put square brackets ([ ]) around the identifier for the RAID controller module. If you do not specify a RAID controller module, the firmware for the RAID controller module returns a syntax error.
availability	This parameter sets the mode for the RAID controller module to **online**, **offline**, or **serviceMode**.
ethernetPort	This parameter defines the attributes (options) for the management Ethernet ports. Refer to Table 2-4 for a list of the attributes that you can set. Valid Ethernet port identifiers are **1** or **2**. You must put square brackets ([ ]) around the Ethernet port identifier.
globalNVSRAMByte	This parameter modifies a portion of the RAID controller module NVSRAM. Specify the region to be modified using the starting byte offset within the region, and the size and value of the new data to be stored into NVSRAM.
hostNVSRAMByte	This parameter updates the NVSRAM for the host specific region. Specifies the host index for the specific host, the starting offset within the region, the number of bytes, and the values to be written.
iscsiHostPort	This parameter defines the attributes (options) for the host Ethernet ports. Refer to Table 2-5 for a list of the attributes that you can set. Valid Ethernet port identifiers are **1** or **2**. You must put square brackets ([ ]) around the *Ethernet port* identifier.

Parameter	Description
rloginEnabled	This parameter defines whether the remote login feature is turned on or turned off. To turn on the remote login feature, set this parameter to **TRUE**. To turn off the remote login feature, set this parameter to **FALSE**.

NOTE: When you use this command, you can specify one or more of the parameters. You do not, however, need to use all of the parameters.

NOTE: Setting *availability* to **serviceMode** causes the alternate RAID controller module to take ownership of all virtual disks. The specified RAID controller module no longer has any virtual disks and refuses to take ownership of any more virtual disks. Service mode is persistent across reset and power cycles until the *availability* parameter is set to **online**.

NOTE: Use the **show controller NVSRAM** command to display parts or all of the NVSRAM.

Additional Information

The *maxFramePayload* option is shared between IPv4 and IPv6. The payload portion of a standard Ethernet frame is set at **1500**, and a jumbo Ethernet frame is set at **9000**. When using jumbo frames, all of the devices that are in the network path should be capable of handling the larger frame size.

You must set the *enableIPv4* parameter or the *enableIPv6* parameter to **TRUE** to make sure that the specific IPv4 setting or the specific IPv6 setting is applied.

When the *duplexMode* parameter is set to **TRUE**, the selected Ethernet port is set to full duplex. The default value is half duplex (the *duplexMode* parameter is set to **FALSE**).

The *portSpeed* parameter is expressed as megabits per second (Mb/s).

The IPv6 address space is 128 bits. It is represented by eight 16-bit hexadecimal blocks separated by colons. You may drop leading zeros, and you may use a double colon to represent consecutive blocks of zeroes.

The default value for the *IPv6HopLimit* parameter is **64**.

The default value for the *IPv6NdReachableTime* parameter is **30000** milliseconds.

The default value for the*IPv6NdRetransmitTime* parameter is 1000 milliseconds.

The default value for the *IPv6NdTimeOut* parameter is 30000 milliseconds.

The default port value for the *tcpListeningPort* parameter is 3260.

Examples

```
-c "set controller [0] iscsiHostPort[0]
IPV6LocalAddress=
FE80:0000:0000:0000:0214:22FF:FEFF:EFA9 enableIPV6=
TRUE;"

-c "set controller [0] iscsiHostPort[0]
IPV6ConfigurationMethod=auto enableIPV6=TRUE;"

-c "set controller [0] availability=online;"

-c "set controller [0] ethernetPort[1] IPV4Address=
192.168.0.101;"

-c "set controller [0] iscsiHostPort[1]
IPV4SubnetMask=255.255.255.0 enableIPV4;"

-c "set controller [0] iscsiHostPort[1]
IPV4GatewayIP=192.168.0.1 enableIPV4;"
```

Set Disk Group

This command defines the properties for a disk group.

Syntax

```
set diskGroup [diskGroupNumber] addPhysicalDisks=
(trayID1,slotID1

... trayIDn,slotIDn) raidLevel=(0 | 1 | 5 | 6)
owner=(0 | 1)

availability=(online | offline)
```

Parameters

Parameter	Description
diskGroup	Sequence number of the disk group for which to set properties. You must put brackets ([ ]) around the disk group number.
addPhysicalDisks	Identifies the physical disk by tray and slot location to include in the disk group. Tray ID values are 0 to 99. Slot ID values are 0 to 31. You must put parentheses around the tray ID values and the slot ID values.
raidLevel	RAID level for the disk group. Valid values are **0**, **1**, **5**, or **6**.
owner	RAID controller module that owns the disk group. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. Use this parameter only if you want to change the disk group owner.
availability	Setting for the disk group, which is either online or offline.

NOTE: When using this command, you can specify one or more of the parameters. You do not, however, need to use all of the parameters. Using two many parameters can cause host I/O errors or result in internal controller reboots because the time period ends before the disk group definition is set.

NOTE: The **addPhysicalDisks** or **raidLevel** operations cannot be stopped once they have been initiated.

NOTE: The time required to complete the **addPhysicalDisks** or **raidLevel** operations is dependent on the number and size of the disks used.

Set Enclosure Attribute

This command sets user-defined attributes for an enclosure.

Syntax

```
set enclosure [enclosureID] (chassisName |
assetTag)= "userID"
```

Parameters

Parameter	Description
enclosure	Identifies a specific enclosure for which to set the attribute. Enclosure ID values are 0 to 99. You must put brackets ([ ]) around the enclosure ID value.
chassisName	Chassis name or number to give the new enclosure. Chassis names can be any combination of alphanumeric characters with a maximum length of 32 characters. Alphabetical characters can be uppercase or lowercase. You can also use the underscore character (_) and the hyphen (-) character. You cannot use spaces in a chassis name. You must put quotation marks (" ") around the chassis name.
assetTag	Asset tag name or number to give the new enclosure. Asset tags can be any combination of alphanumeric characters with a maximum length of ten characters. Alphabetical characters can be uppercase or lowercase. You can also use the underscore character (_) and the hyphen (-) character. You cannot use spaces in an asset tag name. You must put quotation marks (" ") around the asset tag name.

Set Enclosure Identification

This command sets the ID of an enclosure in a storage array.

Syntax

```
set enclosure ["Service Tag"] id=enclosureID
```

Parameters

Parameter	Description
enclosure	Service tag of the RAID enclosure or the expansion enclosure for which you are setting the enclosure ID. You must put quotation marks (" ") around the Service Tag.
id	Specifies the value for the RAID enclosure or expansion enclosure ID. Valid values are 00 through 99. You do not need to put parentheses around the enclosure ID value.

Set Foreign Physical Disk to Native

This command incorporates foreign physical disks that have not been imported into the storage array configuration through normal means. This operation is for emergency recovery use only. Use this statement only when one or more physical disks were added after the configuration adoption process completed.

NOTICE: Using this command for purposes other than what is stated above might result in data corruption or data loss without notification.

Syntax

```
set (physicalDisk [trayID,slotID] |
allPhysicalDisks) nativeState
```

Parameters

Parameter	Description
physicalDisk	The tray and the slot where the physical disk resides. Tray ID values are 0 to 99. Slot ID values are 0 to 31. Enclose the tray ID values and the slot ID values in square brackets ([ ]).
allPhysicalDisks	Selects all the physical disks.

Set Host

This command assigns a host to a host group or moves a host to a different host group. You can also create a new host group and assign the host to the new host group with this command. The actions performed by this command depend on whether the host has individual virtual disk-to-LUN mappings or does not have individual virtual disk-to-LUN mappings.

Syntax

```
set host [hostName]
hostGroup=("hostGroupName" | none | defaultGroup)
userLabel="newHostName"
hostType=(hostTypeIndexLabel |
hostTypeIndexNumber)
```

Parameters

Parameter	Description
host	The name of the host that you want to assign to a host group. Enclose the host name in square brackets ([ ]). If the host name has special characters, you must also enclose the host name in double quotation marks (" ").
hostGroup	The host group to which you want to assign the host. (The following table defines how the command runs if the host does or does not have individual virtual disk-to-LUN mappings.) Enclose the host group name in double quotation marks (" "). The **defaultGroup** is the host group that contains the host to which the virtual disk is mapped.
userLabel	The new host name. Enclose the host name in double quotation marks (" ").
hostType	The index label or number of the host type for the HBA host port. Use the `show storageArray hostTypeTable` command to generate a list of available host type identifiers. If the host type has special characters, enclose it in double quotation marks (" ").

Host Group Parameter	Host Has Individual Virtual Disk-to-LUN Mappings	Host Does Not Have Individual Virtual Disk-to-LUN Mappings
hostGroupName	The host is removed from the present host group and is placed under the new host group defined by *hostGroupName*.	The host is removed from the present host group and is placed under the new host group defined by *hostGroupName*.
none	The host is removed from the host group as an independent partition and is placed under the root node.	The host is removed from the present host group and is placed under the default group.
defaultGroup	The command fails.	The host is removed from the present host group and is placed under the default group.

NOTE: When you use this command, you can specify one or more of the optional parameters. You do not, however, need to use all of the parameters.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Examples

```
-c "set host [job2900] hostGroup=none userLabel=
\"job2900\" hostType=0;"

-c "set host [\"pe2900\"] userLabel=\"pe2901\";"
```

Set Host Group

This command renames a host group.

Syntax

```
set hostGroup [hostGroupName] userLabel=
"newHostGroupName"
```

Parameters

Parameter	Description
hostGroup	Name of the host group to rename. You must put brackets ([ ]) around the host group name. If the host group name has special characters, you must also put quotation marks (" ") around the host group name.
userLabel	New name for the host group. You must put quotation marks (" ") around the host group name.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Set Host Port

This command changes the host type for an HBA host port. You can also change an HBA host port label with this command.

Syntax

```
set hostPort [portLabel] host="hostName"
userLabel="newPortLabel"
```

Parameters

Parameter	Description
hostPort	The name of the HBA host port for which you want to change the host type, or for which you want to create a new name. Enclose the HBA host port name in square brackets ([ ]). If the HBA host port label has special characters, enclose the HBA host port label in double quotation marks (" ").
host	The name of the host to which the HBA host port is connected. Enclose the host name in double quotation marks (" ").
userLabel	The new name that you want to give to the HBA host port. Enclose the new name of the HBA host port in double quotation marks (" ").

NOTE: When you use this command, you can specify one or more of the optional parameters. You do not, however, need to use all of the parameters.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Set iSCSI Initiator

This command sets the attributes for an iSCSI initiator.

Syntax

```
set iscsiInitiator iscsiName = new-iSCSI-ID |
userLabel = new-name | host = new-host-name |
chapSecret = new-password
```

Parameters

Parameter	Description
`iscsiName`	The name of the iSCSI initiator for which you want to set attributes.
`userLabel`	The new name that you want to use for the iSCSI initiator.
`host`	The name of the new host to which the HBA host port is connected. Enclose the host name in double quotation marks (" ").
`chapSecret`	The password that you want to use to authenticate a peer connection.

NOTE: Challenge Handshake Authentication Protocol (CHAP) is a protocol that authenticates the peer of a connection. CHAP is based upon the peers sharing a "secret." A secret is a security key that is similar to a password.

NOTE: Use the *chapSecret* parameter only for initiators requiring mutual authentication.

Examples

```
-c "set iscsiInitiator [\"pe29000\"] userLabel=
\"pe29001\";"

-c "set iscsiInitiator <\"iqn.1991-
05.com.microsoft:svctag-70wnh91\"> userLabel=
\"29000\";"

-c "show iscsiInitiator[\"pe29000\"]
iscsiSessions;"

-c "show iscsiInitiator <\"iqn.1991-
05.com.microsoft:svctag-70wnh91\">
iscsiSessions;"
```

Set iSCSI Target Properties

This command defines properties for an iSCSI target.

Syntax

```
set iscsiTarget authenticationMethod = (none |
chap) | chapSecret = password |
isnsRegistration = (TRUE | FALSE) |
targetAlias = user-label
```

Parameters

Parameter	Description
`authenticationMethod`	The means of authenticating your iSCSI session.
`chapSecret`	The password that you want to use to authenticate a peer connection.
`isnsRegistration`	The means of listing the iSCSI target on the iSNS server. Set the parameter to **TRUE** to list it.
`targetAlias`	The name that you want to use for the target.

NOTE: Challenge Handshake Authentication Protocol (CHAP) is a protocol that authenticates the peer of a connection. CHAP is based upon the peers sharing a "secret." A secret is a security key that is similar to a password.

NOTE: Use the *chapSecret* parameter only for initiators requiring mutual authentication.

NOTE: The *targetAlias* is a descriptive name that you can give to the target to help make it easier to identify. You can use any combination of alphanumeric characters, hyphens, and underscores for the *targetAlias*. The *targetAlias* can have a maximum of 30 characters.

Examples

```
-c "set iscsiTarget <\"iqn.1984-
05.com.dell:powervault.

6001372000f5f0e600000000463b9292\">
authenticationMethod = none;"

-c "set iscsiTarget [\"iscsi2900\"] targetAlias =
\"iscsi2902\";"
```

```
-c "set iscsiTarget <\"iqn.1984-
05.com.dell:powervault.
6001372000f5f0e600000000463b9292\"> targetAlias =
\"iscsi2902\";"
```

Set Physical Disk Channel Status

This command defines how the physical disk channel performs.

Syntax

```
set physicalDiskChannel [( 1 | 2 )] status=
(optimal | degraded)
```

Parameters

Parameter	Description
physicalDiskChannel	Identifier number of the physical disk channel for which to set the status. Valid physical disk channel values are 1 or 2. You must put brackets ([ ]) around the physical disk channel number.
status	Condition of the channel. You can set the channel status to **optimal** or **degraded**. **NOTE:** Use the *optimal* parameter to move a degraded channel back to the Optimal state. Use the *degraded* parameter if the channel is experiencing problems, and the storage array requires additional time for data transfers.

Set Physical Disk Hot Spare

This command assigns or unassigns one or more physical disks as a hot spare.

Syntax

```
set (physicalDisk [enclosureID,slotID] |
physicalDisks [enclosureID0,slotID0 ...
enclosureIDn,slotIDn]) hotSpare=(TRUE | FALSE)
```

Parameters

Parameter	Description
physicalDisk or physicalDisks	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID values and the slot ID values.
hotSpare	Assigns the physical disk as the hot spare. To assign the physical disk as the hot spare, set this parameter to **TRUE**. To remove a hot spare assignment from a physical disk, set this parameter to **FALSE**.

Set Physical Disk State

This command sets a physical disk to the failed state. To return a physical disk to the Optimal state, use the **revive physicalDisk** command.

Syntax

```
set physicalDisk [enclosureID,slotID]
operationalState=failed
```

Parameters

Parameter	Description
physicalDisk	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and the slot ID value.

Set RAID Controller Module

This command defines the attributes for the RAID controller modules.

Syntax

```
set controller [(0 | 1)] availability=(online |
offline |
serviceMode) | ethernetPort [(1 | 2)]=
ethernetPortOptions |
```

```
globalNVSRAMByte [nvsramOffset]=
(nvsramByteSetting |

nvsramBitSetting) | hostNVSRAMByte [hostType,

nvsramOffset]=(nvsramByteSetting |
nvsramBitSetting) |

iscsiHostPort [(1 | 2)]=iscsiHostPortOptions
rloginEnabled=(TRUE

| FALSE) | serviceAllowedIndicator=(on | off)
```

Parameters

Parameter	Description
controller	The RAID controller module for which you want to define properties. Valid identifiers for the RAID controller module are 0 or 1, where 0 is the RAID controller module on the left, and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. Enclose the identifier for the RAID controller module in square brackets ([ ]). If you do not specify a RAID controller module, the firmware for the RAID controller module returns a syntax error.
availability	The mode for the RAID controller module, which you can set to **online**, **offline**, or **serviceMode** (service).
ethernetPort	The attributes (options) for the management Ethernet ports. The entries to support this parameter are listed in "Syntax Element Statement Data" on page 185". Many settings are possible, including setting the IP address, the gateway address, and the subnet mask address.
globalNVSRAMByte	A portion of the RAID controller module NVSRAM. Specify the region to be modified using the starting byte offset within the region, and the size and value of the new data to be stored into NVSRAM.
hostNVSRAMByte	The NVSRAM for the host-specific region. The setting specifies the host index for the specific host, the starting offset within the region, the number of bytes, and the values to be written.

Parameter	Description
iscsiHostPort	The values that support this parameter are listed in "Syntax Element Statement Data" on page 185. Many settings are possible, including setting the IP address, the gateway address, the subnet mask address, the IPv4 priority, and the IPv6 priority.
rloginEnabled	The setting for whether the remote login feature is turned on or turned off. To turn on the remote login feature, set this parameter to **TRUE**. To turn off the remote login feature, set this parameter to **FALSE**.
serviceAllowedIndicator	The setting for whether the Service Action Allowed indicator light is turned on or turned off. To turn on the Service Action Allowed indicator light, set this parameter to **on**. To turn off the Service Action Allowed indicator light, set this parameter to **off**.

Syntax Element Statement Data

The following options are available for the *ethernetPort* parameter.

```
enableIPv4=(TRUE | FALSE) |

enableIPv6=(TRUE | FALSE) |
```

ethernetPort Options

```
IPv6LocalAddress=(0-FFFF):(0-FFFF):(0-FFFF):(0-
FFFF):(0-

FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |

IPv6RoutableAddress=(0-FFFF):(0-FFFF):(0-
FFFF):(0-FFFF):(0-

FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |

IPv6RouterAddress=(0-FFFF):(0-FFFF):(0-FFFF):(0-
FFFF):(0-

FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |

IPv4Address=(0-255).(0-255).(0-255).(0-255) |

IPv4ConfigurationMethod=[(static | dhcp)] |
```

```
IPv4GatewayIP=(0-255).(0-255).(0-255).(0-255) |
IPv4SubnetMask=(0-255).(0-255).(0-255).(0-255) |
duplexMode=(TRUE | FALSE) |
portSpeed=[(autoNegotiate | 10 | 100 | 1000)]
```

iscsiHostPort Options

The following options are available for the *iscsiHostPort* parameter.

```
IPv4Address=(0-255).(0-255).(0-255).(0-255) |
IPv6LocalAddress=(0-FFFF):(0-FFFF):(0-FFFF):(0-
FFFF):(0-
FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |
IPv6RoutableAddress=(0-FFFF):(0-FFFF):(0-
FFFF):(0-FFFF):(0-
FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |
IPv6RouterAddress=(0-FFFF):(0-FFFF):(0-FFFF):(0-
FFFF):(0-
FFFF):(0-FFFF):(0-FFFF):(0-FFFF) |
enableIPv4=(TRUE | FALSE) |
enableIPv6=(TRUE | FALSE) |
```

iscsiHostPort Options

```
enableIPv4Vlan=(TRUE | FALSE) |
enableIPv6Vlan=(TRUE | FALSE) |
enableIPv4Priority=(TRUE | FALSE) |
enableIPv6Priority=(TRUE | FALSE) |
IPv4ConfigurationMethod=(static | dhcp) |
IPv6ConfigurationMethod=(static | auto) |
IPv4GatewayIP=(TRUE | FALSE) |
IPv6HopLimit=[0-255] |
```

```
IPv6NdDetectDuplicateAddress=[0-256] |
IPv6NdReachableTime=[0-65535] |
IPv6NdRetransmitTime=[0-65535] |
IPv6NdTimeOut=[0-65535] |
IPv4Priority=[0-7] |
IPv6Priority=[0-7] |
IPv4SubnetMask=(0-255).(0-255).(0-255).(0-255) |
IPv4VlanId=[1-4094] |
IPv6VlanId=[1-4094] |
maxFramePayload=[frameSize] |
tcpListeningPort=[3260, 49152-65536] |
```

Additional Information

When you use this command, you can specify one or more of the parameters. You do not need to use all of the parameters. Setting the *availability* parameter to **serviceMode** causes the alternate RAID controller module to take ownership of all of the virtual disks. The specified RAID controller module no longer has any virtual disks and refuses to take ownership of any more virtual disks. Service mode is persistent across reset cycles and power cycles until the *availability* parameter is set to online.

Use the **show controller NVSRAM** command to show the NVSRAM information. The **maxFramePayload** option is shared between IPv4 and IPv6. The payload portion of a standard Ethernet frame is set at 1500, and a jumbo Ethernet frame is set at 9000. When using jumbo frames, all of the devices that are in the network path should be capable of handling the larger frame size.

You must set the *enableIPv4* parameter or the *enableIPv6* parameter to **TRUE** to make sure that the specific IPv4 setting or the specific IPv6 setting is applied.

When the *duplexMode* parameter is set to **TRUE**, the selected Ethernet port is set to full duplex. The default value is half duplex (the *duplexMode* parameter is set to **FALSE**).

The *portSpeed* parameter is expressed as megabits per second (Mb/s).

The IPv6 address space is 128 bits. It is represented by eight 16-bit hexadecimal blocks separated by colons. You may drop leading zeros, and you may use a double colon to represent consecutive blocks of zeroes.

The default value for the *IPv6HopLimit* parameter is 64.

The default value for the *IPv6NdReachableTime* parameter is 30000 milliseconds.

The default value for the *IPv6NdRetransmitTime* parameter is 1000 milliseconds.

The default value for the *IPv6NdTimeOut* parameter is 30000 milliseconds.

The default port value for the *tcpListeningPort* parameter is 3260.

Set Session

This command defines how you want the current script engine session to run.

Syntax

```
set session errorAction=(stop | continue)
password="storageArrayPassword"
performanceMonitorInterval=intervalValue
performanceMonitorIterations=iterationValue
```

Parameters

Parameter	Description
errorAction	Specifies how the session responds if an error is encountered during processing. You can choose to stop the session if an error is encountered, or you can continue after encountering an error. The default error action is to stop. (This parameter defines the action for execution errors, not syntax errors. Some error conditions might override the **continue** value.)

Parameter	Description
password	Specifies the password for the storage array. You must put quotation marks (" ") around the password. **NOTE:** Passwords are stored on each storage array in a management domain. If a password was not previously set, you do not need a password. The password can be any combination of alphanumeric characters with a maximum of 30 characters. (You can define a storage array password by using the **set storageArray** command.)
performanceMonitorInterval	Specifies how frequently to gather performance data. Enter an integer value for the polling interval, in seconds, for which you want to capture data. The range of values is 3 to 3600 seconds. The default value is 5 seconds. **NOTE:** The polling interval you specify remains in effect until you end the session. After you end the session, the polling interval returns to the default values.
performanceMonitorIterations	Specifies the number of samples to capture. Enter an integer value. The range of values for samples captured is 1 to 3600. The default value is 1. **NOTE:** The number of iterations you specify remains in effect until you end the session. After you end the session, the number of iterations returns to the default values.

NOTE: When using this command, you can specify one or more of the parameters. You do not, however, need to use all of the parameters.

Set Snapshot Virtual Disk

This command defines the properties for a snapshot virtual disk and enables you to rename a snapshot virtual disk.

Syntax

```
set (virtualDisk [virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen]) userLabel=
```

```
"snapshotVirtualDiskName"
warningThresholdPercent=percentValue
repositoryFullPolicy=(failSourceWrites |
failSnapShot)
```

Parameters

Parameter	Description
virtualDisk or virtualDisks	Name of the specific snapshot virtual disk for which to define properties. (You can enter more than one snapshot virtual disk name by using the *virtualDisks* parameter). You must put quotation marks (" ") and brackets ([ ]) around the snapshot virtual disk name. The snapshot virtual disk name and quotation marks must be inside the brackets.
userLabel	New name to give to a snapshot virtual disk. You must put quotation marks (" ") around the new snapshot virtual disk name.
warningThresholdPercent	Percentage of repository capacity at which a warning is given that the repository is nearing full. Use integer values. For example, a value of 70 means 70 percent. The default value is 50. **NOTE:** You can set this parameter for both the repository virtual disk and snapshot virtual disk.
repositoryFullPolicy	Specifies the desired method of snapshot processing if the repository is full. You can choose to fail writes to the source virtual disk (**failSourceWrites**) or fail writes to the snapshot virtual disk (**failSnapShot**). The default value is **failSnapShot**. **NOTE:** You can set this parameter for both the repository virtual disk and snapshot virtual disk.

NOTE: When using this command, you can specify one or more of the optional parameters. You do not, however, need to use all of the parameters.

NOTE: You can use any combination of alphanumeric characters, hyphens, and underscores for the names. Command names can have a maximum of 30 characters. If you exceed the maximum character limit, replace square brackets ([ ]) with angle brackets (< >) to overcome this limitation.

Set Storage Array

This command defines the properties of the storage array.

Syntax

```
set storageArray cacheBlockSize=
cacheBlockSizeValue failoverAlertDelay=delayValue
mediaScanRate=(disabled | 1-30) |
password="password" |
userLabel="storageArrayName"
```

Parameters

Parameter	Description
cacheBlockSize	Specifies the cache block size used by the RAID controller module for managing the cache. Valid values are **4** (4 KB) or **16** (16 KB), the default value is **4**. **NOTE:** Typically, this parameter should not be changed from the default. The default setting has been set based on best performance for all environments. **NOTE:** When defining cache block sizes, the 4-KB cache block size is best suited for systems that require I/O streams that are typically small and random. The 16-KB cache block size is more useful for systems that require large data transfer, sequential, high bandwidth applications. This parameter defines the cache block size for all virtual disks in the storage array. For redundant configurations, this parameter includes all virtual disks owned by both controllers within the storage array.

Parameter	Description
failoverAlertDelay	Specifies the failover alert delay time in minutes. The valid delay time range is 0 to 60 minutes. The default value is 5 minutes.
mediaScanRate	Specifies the number of days over which the media scan runs. Valid values are: **0**, which disables media scan, or **1** to **30**, where **1** is the fastest scan rate, and **30** is the slowest. **NOTE:** Media scan runs on all virtual disks in the storage array that have an Optimal status, have no modification operations in progress, and have the *mediaScanRate* parameter enabled.
password	Specifies the password for the storage array. You must put quotation marks (" ") around the password. **NOTE:** Passwords are stored on each storage array. The password can be any combination of alphanumeric characters, with a maximum of 30 characters.
`userLabel`	Specifies a name for the storage array. You must put quotation marks (" ") around the storage array name.

NOTE: When using this command, you can specify one or more of the optional parameters. You do not, however, need to use all of the parameters.

Set Storage Array Enclosure Positions

This command defines the position of the enclosures in a storage array. You must include all enclosures in the storage array when you enter this command.

Syntax

```
set storageArray enclosurePositions=(enclosure-id-
list)
```

Parameters

Parameter	Description
enclosurePositions	List of enclosure IDs. The sequence of the module IDs in the list define the positions for the RAID enclosure and expansion enclosures in a storage array. Valid values are 0 to 99. Separate the enclosure ID values with a space, and put parentheses around the list of enclosure IDs.

NOTE: This command defines the position of an enclosure in a storage array by the position of the enclosure ID in the **enclosurePositions** list. For example, if you have a RAID enclosure with an ID set to 84 and expansion enclosures with IDs set to 1 and 12, the **enclosurePositions** sequence (84 1 12) places the RAID enclosure in the first position, expansion enclosure 1 in the second position, and expansion enclosure 12 in the third position. The **enclosurePositions** sequence (1 84 12) places the RAID enclosure in the second position, expansion enclosure 1 in the first position, and expansion enclosure 12 in the third position.

Set Storage Array ICMP Response

This command returns the default values for negotiable settings for sessions and connections, which represent the storage array's starting point for negotiations.

Syntax

```
set storageArray icmpPingResponse = (TRUE | FALSE)
```

Parameter

Parameter	Description
icmpPingResponse	This parameter turns on or turns off Echo Request messages. Set the parameter to **TRUE** to turn on Echo Request messages. Set the parameter to **FALSE** to turn off Echo Request messages.

NOTE: The Internet Control Message Protocol (ICMP) is used by operating systems in a network to send error messages, such as a requested service is not available or that a host or router could not be reached. The ICMP response command sends

ICMP Echo Request messages and receives Echo Response messages to determine if a host is reachable and how long packets take to get to and from that host.

Example

```
-c "set storageArray icmpPingResponse = TRUE;"
```

Set Storage Array iSNS Server IPv4 Address

This command sets the configuration method and address for an IPv4 Internet Storage Name Service (iSNS).

Syntax

```
set storageArray isnsIPV4ConfigurationMethod =
[static | dhcp] isnsIPV4Address = ipv4-address
```

Parameters

Parameter	Description
`isnsIPV4Configuration Method`	The method that you want to use to define the iSNS server configuration. You can enter the IP address for the IPv4 iSNS servers by selecting `static`. For IPv4, you can choose to have a Dynamic Host Configuration Protocol (DHCP) server select the iSNS server IP address by entering `dhcp`. To enable DCHP, you must set the isnsIPV4Address IP address to 0.0.0.0.
`isnsIPV4Address`	The IP address that you want to use for the iSNS server. Use this parameter with the `static` argument for IPv4 configurations. If you choose to have a DHCP server set the IP address for an IPv4 IP iSNS server, you must set the isnsIPV4Address IP address to 0.0.0.0.

NOTE: The DHCP server passes configuration parameters, such as network addresses, to IP nodes. DHCP enables a client to acquire all of the IP configuration parameters that it needs to operate. DHCP enables you to automatically allocate reusable network addresses.

Example

```
-c "set storageArray isnsIPV4ConfigurationMethod =
static isnsIPV4Address = 192.168.0.1;"
```

Set Storage Array iSNS Server IPv6 Address

This command sets the address for an IPv6 Internet Storage Name Service (iSNS).

Syntax

```
set storageArray isnsIPV6Address=ipv6-address
```

Parameter

Parameter	Description
isnsIPV6Address	IPv6address you want to use for the iSNS server

NOTE: The iSNS protocol facilitates the automated discovery, management, and configuration of iSCSI and Fibre Channel devices on a TCP/IP network. The iSNS protocol provides intelligent storage discovery and management services comparable to those found in Fibre Channel networks, allowing a commodity IP network to function in a similar capacity as a storage area network. The iSNS protocol also facilitates a seamless integration of IP and Fibre Channel networks, due to its ability to emulate Fibre Channel fabric services, and manage both iSCSI and Fibre Channel devices.

Set Storage Array iSNS Server Listening Port

This command sets the iSNS server listening port.

Syntax

```
set storageArray isnsListeningPort = integer
```

Parameter

Parameter	Description
`isnsListeningPort`	The IP address that you want to use for the iSNS server listening port. The range of values for the listening port is 49152 to 65535. The default value is 3205.

NOTE: A listening port resides on the database server and is responsible for listening (monitoring) for incoming client connection requests and managing the traffic to the server

NOTE: When a client requests a network session with a server, a listener receives the actual request. If the client information matches the listener information, then the listener grants a connection to the database server.

Example

```
-c "set storageArray isnsListeningPort = 3205;"
```

Set Storage Array iSNS Server Refresh

This command refreshes the network address information for the iSNS server. This command is valid for only IPv4.

Syntax

```
set storageArray isnsServerRefresh
```

Parameters

None.

NOTE: If the DHCP server is not operating at full capability, or if the DHCP server is unresponsive, the refresh operation can take between two and three minutes to complete.

NOTE: The **set storageArray isnsServerRefresh** command returns an error if you did not set the configuration method to DHCP. To set the configuration method to DHCP, use the **set storageArray isnsIPV4ConfigurationMethod** command.

Example

```
-c "start storageArray isnsServerRefresh ;"
```

Set Storage Array Learn Cycle

This command sets the learn cycle for the battery backup unit, which enables the MD Storage Manager Software to predict the remaining battery life. Learn cycles run at set intervals, and they store the results for software analysis.

Syntax

```
set storageArray learnCycleDate

(daysToNextLearnCycle=integer-literal | day=
string-literal) time=HH:MM
```

Parameters

Parameter	Description
daysToNextLearnCycle	Valid values are **0** through **7**, where **0** is immediately and **7** is in seven days. The *daysToNextLearnCycle* parameter takes place up to seven days after the next scheduled learn cycle
day	Valid values include the days of the week (**Sunday**, **Monday**, **Tuesday**, **Wednesday**, **Thursday**, **Friday**, and **Saturday**). Setting the day causes the next learn cycle to be scheduled on the specified day, after the currently scheduled learn cycle.
time	The time in 24-hour format; for example 8:00 a.m. is entered as 08:00. Nine o'clock p.m. is entered as 21:00, and 9:30 p.m. is entered as 21:30.

NOTE: You can set the learn cycle to occur only once during a seven-day period. The time parameter selects a specific time that you want to run the learn cycle. If a value is not entered, the command uses a default value of 00:00 (midnight). If the day and time specified are in the past, the next learn cycle takes place on the next possible day specified.

Set Storage Array Time

This command sets the clocks on both RAID controller modules in a storage array by synchronizing the RAID controller module clocks with the clock of the host from which you issue this command.

Syntax

```
set storageArray time
```

Parameters

None.

Set Unnamed Discovery Session

This command enables the storage array to participate in unnamed discovery sessions.

Syntax

```
set iscsiTarget unnamedDiscoverySession = (TRUE |
FALSE)
```

Parameter

Parameter	Description
`unnamedDiscoverySession`	This parameter turns on or turns off unnamed discovery sessions. Set the parameter to **TRUE** to turn on an unnamed discovery sessions. Set the parameter to **FALSE** to turn off an unnamed discovery sessions.

NOTE: Discovery is the process where initiators determine the targets that are available. Discovery occurs at power-on/initialization and also if the bus topology changes, for example, if an extra device is added.

NOTE: An unnamed discovery session is a discovery session that is established without specifying a target ID in the login request. For unnamed discovery sessions, neither the target ID or the target portal group ID are available to the targets.

Set Virtual Disk

This command defines the properties for a virtual disk. You can use most of the parameters to define properties for one or more virtual disks, however, some of the parameters define properties for only one virtual disk at a time.

The syntax definitions are separated to show which parameters apply to several virtual disks and which apply to only one virtual disk. The syntax for virtual disk mapping is listed separately.

Syntax Applicable to One or More Virtual Disks

```
set (allVirtualDisks | virtualDisk
["virtualDiskName"] |
virtualDisks ["virtualDiskName1" ...
"virtualDiskNamen"] |virtualDisk <wwid>)
mediaScanEnabled=(TRUE | FALSE)
mirrorCacheEnabled=(TRUE | FALSE)
modificationPriority=(highest | high | medium |
low | lowest)owner=(0 | 1)
writeCacheEnabled=(TRUE | FALSE)
cacheReadPrefetch=(TRUE | FALSE)
```

NOTE: Enabling *Write Cache* on a virtual disk generally improves performance for applications with significant Write content (unless the application features a continuous string of Writes. However, *Write Cache* does introduce some risk of data loss in the unlikely event of a controller failure.

Syntax Applicable to Only One Virtual Disk

```
Set (virtualDisk ["virtualDiskname"] | virtualDisk
<wwid>) addCapacity=virtualDiskcapacity
[addPhysicalDisks=(enclosureID0,slotID0 ...
enclosureIDn,slotIDn)] consistencyCheckEnabled=
(True | False) segmentSize=segmentSizeValue
userLabel="virtualDiskName"
```

Syntax Applicable to Virtual Disk Mapping

```
set (virtualDisk ["virtualDiskName"] | virtualDisk
<wwid> | accessVirtualDisk) logicalUnitNumber=LUN
(host="hostName" | hostGroup=("hostGroupName")
```

Parameters

Parameter	Description
allVirtualDisks	Specifies the properties for all virtual disks in the storage array.
virtualDisk or virtualDisks (name)	Specifies the name of the virtual disk for which to define properties. You can enter more than one virtual disk name if you use the **virtualDisks** parameter. You must put quotation marks (" ") and brackets ([ ]) around the virtual disk name. The virtual disk name and quotation marks must be inside the brackets.
virtualDisk (wwid)	Specifies the WWID of the virtual disk for which you are setting properties. You can use the WWID instead of the virtual disk name to identify the virtual disk. You must put angle brackets (< >) around the WWID.
mediaScanEnabled	Turns media scan for the virtual disk on or off. To turn media scan on, set to **TRUE**. To turn media scan off, set to **FALSE**. If media scan is disabled at the storage array level, this parameter has no effect.
mirrorCacheEnabled	Turns the mirror cache on or off. The default setting is **TRUE**. To turn the mirror cache off, set this parameter to **FALSE**. **NOTE:** Data loss can occur if a RAID controller module fails when cache mirroring is set to FALSE on the virtual disks owned by the failed controller.
modificationPriority	Specifies the priority for virtual disk modifications while the storage array is operational. Valid entries are highest, high, medium, low, or lowest. **NOTE:** The *ModificationPriority* parameter defines the amount of system resources used when modifying virtual disk properties. If you select the highest priority level, the virtual disk modification uses the most system resources, which decreases performance for host data transfers.

Parameter	Description
owner	Specifies the RAID controller module that owns the virtual disk. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. Use only if you want to change the virtual disk owner.
writeCacheEnabled	Turns the write cache on or off. To turn on the write cache, set this parameter to **TRUE**. To turn off the write cache, set this parameter to **FALSE**. The default value is **TRUE**.
cacheReadPrefetch	This parameter turns the cache read prefetch on or off. To turn off cache read prefetch, set this parameter to **FALSE**. To turn on cache read prefetch, set this parameter to **TRUE**. The default value is **TRUE**.
addCapacity	Increases storage size (capacity) of the virtual disk for which you are defining properties. Size is defined in units of bytes, kilobytes, megabytes, or gigabytes, or terabytes. The default units are bytes. Virtual Disk capacity expansion increases the size of the logical unit that is exposed by the RAID controller. Refer to your OS documentation for additional information about how the OS can recognize the additional capacity on the virtual disk and increase the size of OS volumes on the virtual disk.
addPhysicalDisks	Adds new physical disks to the virtual disk. Specify the enclosure ID and slot ID for each physical disk that you assign to the virtual disk. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID values and the slot ID values. Use with the *addCapacity* parameter if you must specify additional physical disks to accommodate the new size.
consistencyCheckEnabled	Turns consistency checking during a media scan on or off. To turn consistency checking on, set to **TRUE**. To turn consistency checking off, set to **FALSE**.

Parameter	Description
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, **16**, **32**, **64**, **128**, **256**, or **512**. For information about the **segmentSize** parameter, see "Using the Auto Configure Command" on page 53.
userLabel	Specifies the new name to give an existing virtual disk. You must put quotation marks (" ") around the new virtual disk name.
accessVirtualDisk	Specifies the logical unit number for the access virtual disk. The logical unit number is the only property that you can set for the access virtual disk. **NOTE:** If you specify the *accessVirtualDisk* parameter, the only property that you can set is the *logicalUnitNumber*.
logicalUnitNumber	Defines specific virtual disk-to-LUN mappings to an individual host or assigns the host to a host group.
host	Specifies the name of the host to which the virtual disk is mapped. You must put quotation marks (" ") around the host name.
hostGroup	Specifies the name of the host group to which the virtual disk is mapped. You must put quotation marks (" ") around the host group name.

NOTE: When using this command, you can specify one or more of the parameters. You do not, however, need to use all of the parameters.

NOTE: You can apply the *addCapacity*, *segmentSize*, *userLabel*, and *logicalUnitNumber* parameters to only one virtual disk at a time.

NOTE: Setting the *addCapacity*, *addPhysicalDisks*, or *segmentSize* parameter starts a long-running operation that you cannot stop. These long-running operations are performed in the background and do not prevent you from running other commands. To display the progress of long-running operations, use the **show virtualDisk actionProgress** command.

Set Virtual Disk Copy

This command defines the properties for a virtual disk copy pair.

Syntax

```
set virtualDiskCopy target [targetName] [source
[sourceName]] copyPriority=(highest | high |
medium | low | lowest) targetReadOnlyEnabled=(TRUE
| FALSE)
```

Parameters

Parameter	Description
target	Specifies the name of the target virtual disk for which to define properties. You must put brackets ([ ]) around the target virtual disk name. If the target virtual disk name has special characters, you must also put quotation marks (" ") around the target virtual disk name.
source	Specifies the name of the source virtual disk for which to define properties. You must put brackets ([ ]) around the source virtual disk name. If the source virtual disk name has special characters, you must also put quotation marks (" ") around the source virtual disk name.
copyPriority	Specifies the priority that the virtual disk copy has relative to host I/O activity. Valid entries are highest, high, medium, low, or lowest.
targetReadOnlyEnabled	Specifies whether the target virtual disk is write enabled or read only. To be able to write to the target virtual disk, set to **FALSE**. To prevent writing to the target virtual disk, set to **TRUE**.

NOTE: When using this command, you can specify one or more of the parameters. You do not, however, need to use all of the parameters.

Show Current iSCSI Sessions

This command returns information about an iSCSI session.

Syntax

```
show iscsiInitiator iscsiSessions
[iscsiInitiatorName | iscsiTargetName]
```

Parameters

Parameter	Description
iscsiInitiatorName	The name of the iSCSI initiator for which you want to obtain session information. Enclose the iSCSI initiator name in square brackets ([ ]).
iscsiTargetName	The name of the iSCSI target for which you want to obtain session information. Enclose the iSCSI target name in square brackets ([ ]).

NOTE: If you enter this command without defining any arguments, this command returns information about all iSCSI sessions that are currently running. To limit the information returned, enter a specific iSCSI initiator or a specific iSCSI target. This command then returns information about the session for only the iSCSI initiator or the iSCSI target that you named.

Show Disk Group

This command returns the following information about a disk group:

- Status (online or offline)
- Physical disk type (SAS or SATA)
- Enclosure loss protection (yes or no)
- Current owner (RAID controller module 0 or RAID controller module 1)
- Associated virtual disks and free capacity
- Associated physical disks

NOTE: You can use the free capacity area value when you create a virtual disk based on the free capacity of a disk group. For a description of how to use the free capacity value, see the **create virtualDisk** command on "Create RAID Virtual Disk (Free Capacity Base Select)" on page 128.

Syntax

```
show diskGroup [diskGroupNumber]
```

Parameters

Parameter	Description
diskGroup	Number of the disk group for which to display information. You must put brackets ([ ]) around the disk group number.

Show Host Ports

For all HBA host ports connected to a storage array, this command returns the following information:

- HBA host port identifier
- HBA host port name
- HBA host type

Syntax

```
show allHostPorts
```

Parameters

None.

Example

```
-c "show allHostPorts;"
```

Show Physical Disk

For each physical disk in the storage array, this command returns the following information:

- The total number of physical disks
- The type of physical disk (SAS or SATA)
- Basic physical disk information:
  - Enclosure location and slot location
  - Status
  - Capacity
  - Data transfer rate

  - Product ID
  - Firmware level
- Physical disk channel information:
  - Enclosure location and slot location
  - Preferred channel
  - Redundant channel
- Hot spare coverage
- Details for each physical disk

Depending on the size of the storage array, this information can be several pages long. The physical disk information is also returned when you issue the **show storageArray profile** command.

Syntax

```
show (allPhysicalDisks [physicalDiskType=
(SAS | SATA)] |
physicalDisk [enclosureID,slotID] |
physicalDisks [enclosureID0,slotID0 ...
enclosureIDn,slotIDn])
[summary]
```

Parameters

Parameter	Description
allPhysicalDisks	Returns information about all physical disks in the storage array. **NOTE:** To determine information about the type and location of all physical disks in the storage array, use the *allPhysicalDisks* parameter.
physicalDiskType	Specifies the type of physical disk for which to retrieve information. Valid physical disk types are SAS or SATA. **NOTE:** To determine the information about the SAS or SATA physical disks in the storage array, use the *physicalDiskType* parameter.

Parameter	Description
physicalDisk or physicalDisks	Identifies the enclosure and slot where the physical disk resides. You can enter enclosure IDs and slot IDs for one or several physical disks. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put parentheses around the enclosure ID values and the slot ID values.
summary	Returns the status, capacity, data rate, product ID, and firmware version for the specified physical disks. **NOTE:** To determine the type of physical disk in a specific location, use the *physicalDisk* parameter, and enter the enclosure and slot IDs for the physical disk.

Show Physical Disk Channel Statistics

This command shows cumulative physical disk channel data transfer and error information. If the RAID controller module has automatically degraded a channel, this command also shows interval statistics. When using this command, you can display information about one specific physical disk channel, several physical disk channels, or all physical disk channels.

Syntax

```
show (physicalDiskChannel [(1 | 2 )] |
physicalDiskChannels [(1 | 2 ) ... (1n | 2n )] |
allPhysicalDiskChannels) stats
```

Parameters

Parameter	Description
physicalDiskChannel	Identifier number of the physical disk channel for which to display information. Valid physical disk channel values are 1 or 2. You must put brackets ([ ]) around the physical disk channel values.

Show Physical Disk Download Progress

This command returns the status of firmware downloads for the physical disks targeted by the **download physicalDisk firmware** or **download storageArray physicalDiskFirmware** commands.

Syntax

```
show allPhysicalDisks downloadProgress
```

Parameters

None.

NOTE: When all firmware downloads have successfully completed, this command returns a **Successful** status. If any firmware downloads fail, this command shows the firmware download status of each targeted physical disk. This command returns the status values shown in the following table.

Status	Definition
Successful	Downloads completed without errors.
Not Attempted	Downloads did not start.
Partial Download	Downloads are in progress.
Failed	Downloads completed with errors.

Show RAID Controller Module

For each RAID controller module in a storage array, this command returns the following information:

- Status (Online, Offline)
- Current firmware and NVSRAM configuration
- Pending firmware and NVSRAM configuration configurations (if any)
- Board ID
- Product ID
- Product revision
- Serial number
- Date of manufacture

- Date and time to which the RAID controller module is set
- Associated virtual disks (including preferred owner)
- Ethernet port
- Physical disk interface

Syntax

```
show (allControllers | controller [(0 | 1)])
[summary]
```

Parameters

Parameter	Description
allControllers	Returns information about both RAID controller modules in the storage array.
controller	Returns information about a specific RAID controller module in the storage array. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier. **NOTE:** The detailed RAID controller module information is also returned for the **show storageArray** command.
summary	Returns a concise list of information about both RAID controller modules in the storage array. **NOTE:** When you use the *summary* parameter, the command returns the list of information without the physical disk channel and host channel information.

Show RAID Controller Module NVSRAM

This command returns a list of the NVSRAM byte values for the specified host type. If you do not enter the optional parameters, this command returns a list of all NVSRAM byte values.

Syntax

```
show (allControllers | controller [(0 | 1)])
NVSRAM [hostType=(hostTypeIndexLabel | host=
"hostName")]
```

Parameters

Parameter	Description
allControllers	Returns information about both RAID controller modules in the storage array.
controller	Returns information about a specific RAID controller module in the storage array. Valid RAID controller module identifiers are 0 or 1, where 0 is the RAID controller module on the left and 1 is the RAID controller module on the right when viewed from the rear of the enclosure. You must put brackets ([ ]) around the RAID controller module identifier.
hostType	Specifies the index label or number of the host type. Use the **show storageArray hostTypeTable** command to generate a list of available host type identifiers.
host	Specifies the name of the host connected to the RAID controller modules. You must put quotation marks (" ") around the host name.

NOTE: Use the **show controller NVSRAM** command to display parts or all of the NVSRAM before using the **set controller** command to change NVSRAM values.

Show Storage Array

This command returns configuration information about the storage array. The parameters return lists of values for the components and features in the storage array. You can enter the command with a single parameter or more than one parameter. If you enter the command with no parameters, the entire storage array profile is displayed (which is the same information as if you entered the *profile* parameter).

Syntax

```
show storageArray profile batteryAge connections
defaultHostType healthStatus hostTypeTable
hotSpareCoverage features time
virtualDiskDistribution summary
```

Parameters

Parameter	Description
profile	Displays all properties of the logical and physical components that comprise the storage array. The information returned takes several screens to display. **NOTE:** The *profile* parameter returns detailed information about the storage array. The information covers several screens on a display. You might need to increase the size of your display buffer to see all the information. Because this information is so detailed, you might want to save the output of this parameter to a file. To save the output to a file, enter the **show storageArray** command similar to the following: `smcli 123.45.67.89 -c "show storagearray profile;" -o "c:\\folder\\storagearray profile.txt"` **NOTE:** The previous command syntax is for a host running Windows. The actual syntax varies depending on your operating system.
batteryAge	Displays the status, the age of the battery in days, and the number of days until the battery needs to be replaced.
connections	Displays a list of the drive channel port locations and the drive channel connections.
defaultHostType	Displays the default host type and host type index.
healthStatus	Displays the health, logical properties, and physical component properties of the storage array.
hostTypeTable	Displays a table of all host types known to the RAID controller module. Each row in the table displays a host type index and the platform the index represents.
hotSpareCoverage	Displays information about which virtual disks of the storage array have hot spare coverage and which virtual disks do not.

Parameter	Description
features	Displays a list of the feature identifiers for all enabled features in the storage array.
time	Displays the current time to which both RAID controller modules in the storage array are set.
virtualDiskDistribution	Displays the current RAID controller module owner for each virtual disk in the storage array.
summary	Returns a concise list of information about the storage array configuration.

NOTE: When you save the information to a file, you can use the information as a record of your configuration and as an aid during recovery.

Show Storage Array Autoconfigure

This command shows the default autoconfiguration that the storage array creates if you issue the **autoConfigure storageArray** command. To determine whether the storage array can support specific properties, enter the parameter for the properties when you issue this command. You do not, however, need to enter any parameters for this command to return configuration information. If you do not specify any properties, this command returns the RAID 5 candidates for each physical disk type. If RAID 5 candidates are not available, this command returns candidates for RAID 1 or RAID 0. In order to view the RAID 6 autoConfiguration options, you must specify the raidLevel option. When you specify auto-configuration properties, the RAID controller modules validate that the firmware can support the properties.

Syntax

```
show storageArray autoConfiguration
[physicalDiskType=(SAS | SATA)
raidLevel=(0 | 1 | 5 | 6)
diskGroupWidth=numberOfPhysicalDisks
diskGroupCount=numberOfDiskGroups
virtualDisksPerGroupCount=
numberOfVirtualDisksPerGroup hotSpareCount=
numberOfHotspares
segmentSize=segmentSizeValue]
```

Parameters

Parameter	Description
physicalDiskType	Type of physical disk to use for the storage array. Valid physical disk types are SAS or SATA. The physicalDiskType parameter is not required if only one type of physical disk is in the storage array. This parameter is not required if only one type of physical disk is in the storage array.
raidLevel	RAID level of the disk group that contains the physical disks in the storage array. Valid RAID levels are **0**, **1**, **5** or **6.**
diskGroupWidth	Number of physical disks in a disk group in the storage array. This number depends on the capacity of the physical disks. Integer values are required. For information about the number of physical disks that you can use in a disk group, see "Enclosure Loss Protection" on page 52.
diskGroupCount	Number of disk groups in the storage array. Use integer values.
virtualDisksPerGroupCount	Number of equal-capacity virtual disks per disk group. Use integer values.
hotSpareCount	Number of hot spares desired in the storage array. Use integer values. For information about hot spares, see "Assigning Global Hot Spares" on page 59.
segmentSize	Amount of data (in kilobytes) that the RAID controller module writes on a single physical disk in a virtual disk before writing data on the next physical disk. Valid values are 8, **16**, **32**, **64**, 128, **256**, or **512**. For information about the segmentSize parameter, see "Using the Auto Configure Command" on page 53.

Show Storage Array Host Topology

This command returns storage partition topology, host type labels, and host type index for the host storage array.

Syntax

```
show storageArray hostTopology
```

Parameters

None.

Show Storage Array LUN Mappings

This command returns information from the storage array profile about the storage array LUN mappings. If you run this command with no parameters, this command returns all LUN mappings.

Syntax

```
show storageArray lunMappings [host ["hostName"] |
hostgroup ["hostGroupName"]]
```

Parameters

Parameter	Description
host	Name of a specific host for which to see the LUN mappings. You must put quotation marks (" ") and brackets ([ ]) around the host name. The host name and quotation marks must be inside the brackets.
hostGroup	Name of a specific host group for which to see the LUN mappings. You must put quotation marks (" ") and brackets ([ ]) around the host group name.The host group name and quotation marks must be inside the brackets.

Show Storage Array Negotiation Defaults

This statement returns information about connection-level settings that are subject to initiator-target negotiation.

Syntax

```
show storageArray iscsiNegotiationDefaults
```

Parameters

None.

NOTE: Information returned includes RAID controller module default settings (settings that are the starting point for negotiation), and the current active settings.

Example

```
-c "show storageArray iscsiNegotiationDefaults;"
```

Show Storage Array Pending Topology

This command identifies the hosts and host groups that the storage management software discovered. Use the **accept storageArray pendingTopology** command to create hosts and host groups from the pending topology.

Syntax

```
show storageArray pendingTopology
```

Parameters

None.

Show Storage Array Unreadable Sectors

This command returns a table of the addresses of all unreadable sectors in the storage array. The table is organized with column headings for the following information:

1. Virtual disk user label
2. LUN
3. Accessible by host or host group
4. Date/Time
5. Virtual disk-relative logical block address

   Hex format (0x nnnnnnnn)

6 Physical disk location

Enclosure t, slot s

7 Physical disk-relative logical block address

Hex format (0x nnnnnnnn)

8 Failure Type

The data is sorted first by virtual disk user label and second by the logical block address (LBA). Each entry in the table corresponds to a single sector.

Syntax

```
show storageArray unreadableSectors
```

Parameters

None.

Show String

This command shows a string of text from a script file. This command is similar to the **echo** command in the MS DOS and Linux operating systems.

Syntax

```
show "string"
```

Parameters

None.

NOTE: You must put quotation marks (" ") around the string.

Show Unconfigured iSCSI Initiators

This command returns a list of initiators that have been detected by the storage array but are not yet configured into the storage array topology.

Syntax

```
show storageArray
unconfiguredIscsiInitiators
```

Parameters

None.

Example

```
-c "show storageArray
unconfiguredIscsiInitiators;"
```

Show Virtual Disk

For the virtual disks in a storage array, this command returns the following information:

- Number of virtual disks
- Name
- Status
- Capacity
- RAID level
- Disk group where the virtual disk is located
- Details
  - Virtual disk ID
  - Subsystem ID
  - Physical disk type (SAS or SATA)
  - Enclosure loss protection
  - Preferred owner
  - Current owner
  - Segment size
  - Modification priority
  - Read cache status (enabled, disabled)
  - Write cache status (enabled, disabled)
  - Write cache without batteries status (enabled, disabled)
  - Flush write cache after time
  - Enable background media scan status (enabled, disabled)
  - Media scan with consistency check status (enabled, disabled)

- Snapshot repository virtual disks
- Snapshot virtual disks
- Copies

Syntax

```
show (allVirtualDisks | virtualDisk
[virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen]) [summary]
```

Parameters

Parameter	Description
allVirtualDisks	Returns information about all virtual disks in the storage array.
virtualDisk or virtualDisks	Specifies the name of the specific virtual disk from which to retrieve information. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.
summary	Returns a concise list of information about the virtual disks.

Show Virtual Disk Action Progress

For a long-running operation that is currently running on a virtual disk, this command returns information about the virtual disk action and amount of the long-running operation completed. The completed amount of the long-running operation is shown as a percentage (for example, 25 means that 25 percent of the long-running operation is completed).

Syntax

```
show virtualDisk ["virtualDiskName"]
actionProgress
```

Parameters

Parameter	Description
virtualDisk	Name of the virtual disk running the long-running operation. You must put quotation marks (" ") and brackets ([ ]) around the virtual disk name. The virtual disk name and quotation marks must be inside the brackets.

Show Virtual Disk Copy

This command returns information about virtual disk copy operations. The following information is returned:

- Copy status
- Start timestamp
- Completion timestamp
- Copy priority
- Source virtual disk WWID or target virtual disk WWID
- Read-only attribute setting of the target virtual disk

You can retrieve information about a specific virtual disk copy pair or all virtual disk copy pairs in the storage array.

Syntax

```
show virtualDiskCopy (allVirtualDisks |
source ["sourceName"] |
target ["targetName"])
```

Parameters

Parameter	Description
allVirtualDisks	Returns information about virtual disk copy operations for all virtual disk copy pairs.
source	Name of the source virtual disk about which to retrieve information. You must put quotation marks (" ") and brackets ([ ]) around the source virtual disk name. The source virtual disk name and quotation marks must be inside the brackets.

Parameter	Description
target	Name of the target virtual disk about which to retrieve information. You must put quotation marks (" ") and brackets ([ ]) around the target virtual disk name. The target virtual disk name and quotation marks must be inside the brackets.

Show Virtual Disk Copy Source Candidates

This command returns information about the candidate virtual disks that you can use as the source for a virtual disk copy operation.

Syntax

```
show virtualDiskCopy sourceCandidates
```

Parameters

None.

Show Virtual Disk Copy Target Candidates

This command returns information about the candidate virtual disks that you can use as the target for a virtual disk copy operation.

Syntax

```
show virtualDiskCopy source ["sourceName"]
targetCandidates
```

Parameters

Parameter	Description
source	Name of the source virtual disk for which you are trying to find a candidate target virtual disk. You must put quotation marks (" ") and brackets ([ ]) around the source virtual disk name. The source virtual disk name and quotation marks must be inside the brackets.

Show Disk Group Import Dependencies

This command shows a list of dependencies for the physical disks in a disk group that you want to move from one storage array to a second storage array.

Syntax

```
show diskGroup [diskGroupNumber]
importDependencies [cancelImport=(TRUE | FALSE)]
```

Parameters

Parameter	Description
diskGroup	The number of the disk group for which you want to show information. Enclose the disk group number in square brackets ([ ])
cancelImport	The setting to spin the physical disks back down after the disk group dependencies have been read. To spin down the physical disks, set this parameter to **TRUE**. To let the physical disks stay spinning, set this parameter to **FALSE**.

NOTE: This command returns the dependencies of a specific disk group, which must be in an Exported state or a Forced state. If a decision is made to retain the listed dependencies, then the *cancelImport* parameter can be enforced to spin the physical disks back down.

NOTE: The **show diskGroup importDependencies** command must be run before the **start diskGroup import** command.

Show Virtual Disk Performance Statistics

This command returns information about the performance of the virtual disks in a storage array.

Syntax

```
show (allVirtualDisks | virtualDisk
[virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen] performanceStats
```

Parameters

Parameter	Description
allVirtualDisks	Returns performance statistics for all of the virtual disks in the storage array.
virtualDisk or virtualDisks	Name of the specific virtual disk for which you are retrieving performance statistics. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.

NOTE: Before issuing the **show virtualDisk performanceStat** command, issue the **set session performanceMonitorInterval** and **set session performanceMonitorIterations** commands to define how often you collect the statistics.

Show Virtual Disk Reservations

This command returns information about the virtual disks that have reservations.

Syntax

```
show (allVirtualDisks | virtualDisk
[virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen] reservations
```

Parameters

Parameter	Description
allVirtualDisks	Returns reservation information about all virtual disks in the storage array.
virtualDisk or virtualDisks	Name of the specific virtual disk for which you are retrieving reservation information. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.

Start Disk Group Blink

This command identifies the physical disks that are logically grouped together to form the specified disk group by blinking the indicator LEDs on the physical disks. (Use the **stop diskGroup blink** command to turn off the indicator LEDs on the physical disks.)

Syntax

```
start diskGroup [diskGroupNumber] blink
```

Parameter

Parameter	Description
diskGroup	Number of the disk group to identify. You must put brackets ([ ]) around the disk group number.

Start Disk Group Defragment

This command starts a defragment operation on the specified disk group.

NOTE: Defragmenting a disk group starts a long-running operation that you cannot stop.

Syntax

```
start diskGroup [diskGroupNumber] defragment
```

Parameter

Parameter	Description
diskGroup	Number of the disk group to defragment. You must put brackets ([ ]) around the disk group number.

Start Enclosure Blink

This command identifies an enclosure by turning on the indicator LED. (Use the **stop enclosure blink** command to turn off the indicator LED for the enclosure.)

Syntax

```
start enclosure [enclosureID] blink
```

Parameter

Parameter	Description
enclosure	Identifies the enclosure to blink. Enclosure ID values are 0 to 99. You must put brackets ([ ]) around the enclosure ID value.

Start iSCSI DHCP Refresh

This command initiates a refresh of the DHCP parameters for the iSCSI interface. If the configuration method for the interface is not set to DHCP, the procedure returns an error.

Syntax

```
start storageArray [iscsi-host-port] dhcpRefresh
```

Parameter

Parameter	Description
iscsi-host-port	The identifier of the port on the storage array on which you want to refresh the DHCP parameters. Enclose the iSCSI host port name in square brackets ([ ]).

NOTE: This operation ends the iSCSI connections for the portal and brings down the portal temporarily.

Start Physical Disk Channel Fault Isolation Diagnostics

This command runs the physical disk channel fault isolation diagnostics and stores the results.

Syntax

```
start physicalDiskChannel [(1 | 2)] controller [(0
| 1)]
```

```
faultDiagnostics {testDevices=[all | controller=(0
| 1) |
emms=[trayID1 (left | right), ... trayIDn (left |
right)] |
physicalDisks=[trayID1, slotID1, ... , trayIDn,
slotIDn]] |
dataPattern=(fixed | pseudoRandom) |
patternNumber=[(0xhexadecimal | integerLiteral)] |
maxErrorCount=integer | testIterations=integer |
timeout=timeInterval}
```

Parameters

Parameter	Description
controller	The identifier letter of the RAID controller module that you want to test. Valid RAID controller module identifier values are 0 or 1, where 0 is the RAID controller module on the left, and 1 is the RAID controller module on the right when viewed from the rear of the enclosure.
testDevices	The identifiers of the devices (RAID controller modules, EMMs, or physical disks) that you want to test. You can specify all or enter the specific identifiers for the devices that you want to diagnose.
dataPattern	The method of repeatability that you want to test.
patternNumber	The hexadecimal data pattern you want to use to run the test. This number can be any hexadecimal number between 0000 to FFFF.
maxErrorCount	The number of errors that you want to accept before terminating the test.
testIterations	The number of times that you want to repeat the test.
timeout	The length of time in minutes that you want to run the test.

NOTE: Use the **save physicalDiskChannel faultDiagnostics** command and the **stop physicalDiskChannel faultDiagnostics** command in association with the **start physicalDiskChannel faultDiagnostics** command. These commands are needed to save the diagnostic test results to a file and to stop the diagnostic test.

NOTE: You can stop this command at any time by pressing Ctrl+C.

Start Physical Disk Blink

This command identifies a physical disk by turning on the indicator LED on the physical disk. (Use the **stop physicalDisk blink** command to turn off the indicator LED on the physical disk.)

Syntax

```
start physicalDisk [enclosureID,slotID] blink
```

Parameters

Parameter	Description
physicalDisk	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and slot ID value.

Start Physical Disk Initialize

This command starts physical disk initialization.

NOTICE: As soon as you enter this command, all user data is destroyed.

Syntax

```
start physicalDisk [enclosureID,slotID] initialize
```

Parameters

Parameter	Description
physicalDisk	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and slot ID value.

Start Physical Disk Reconstruction

This command starts reconstructing a physical disk.

Syntax

```
start physicalDisk [enclosureID,slotID]
reconstruct
```

Parameters

Parameter	Description
physicalDisk	Identifies the enclosure and slot where the physical disk resides. Enclosure ID values are 0 to 99. Slot ID values are 0 to 31. You must put brackets ([ ]) around the enclosure ID value and slot ID value.

Start Storage Array Blink

This command identifies a storage array by turning on the indicator LED for the storage array. (Use the **stop storageArray blink** command to turn off the indicator LED for the storage array.)

Syntax

```
start storageArray blink
```

Parameters

None.

Example

```
-c "start storageArray blink ;"
```

Start Disk Group Import/Export

The import command moves a disk group into a Complete state to make a newly introduced disk group available to its new storage array. The disk group must be in an Exported state or a Forced state before you run this command. Upon successfully running the command, the disk group is operational.

The export command prepares a disk group so that it can be moved from one enclosure to another. The disk group is in the Exported state and is unavailable for use once the command has been successfully run.

NOTE: Within the disk group, you cannot move virtual disks that are associated with the premium features from one storage array to another storage array.

Syntax

```
start diskGroup [diskGroupNumber] (import |
export)
```

Parameter

Parameter	Description
diskGroupNumber	The number of the disk group that you want to import. Enclose the disk group number in square brackets ([ ]).

NOTE: Higher-level virtual disks that are specifically related to premium features (Snapshot, Remote Virtual Disk Mirroring, Virtual Disk Copy, host-to-virtual disk mapping, and persistent reservations) are removed as part of the import operation.

NOTE: The **show diskGroup importDependencies** command must be run before the **start diskGroup import** command.

Start Virtual Disk Initialization

This command starts the formatting of a virtual disk in a storage array.

NOTE: Initializing a virtual disk is a long-running operation that you cannot stop.

Syntax

```
start virtualDisk [virtualDiskName] initialize
```

Parameters

Parameter	Description
virtualDisk	Name of the virtual disk for which to start the formatting. You must put brackets ([ ]) around the virtual disk name. If the virtual disk name has special characters, you must also put quotation marks (" ") around the virtual disk name.

Stop Disk Group Blink

This command turns off the indicator LED on the physical disk that were turned on by the **start diskGroup blink** command.

Syntax

```
stop diskGroup blink
```

Parameters

None.

Stop Enclosure Blink

This command turns off the indicator LED on the enclosure that were turned on by the **start enclosure blink** command.

Syntax

```
stop enclosure blink
```

Parameters

None.

Stop iSCSI Session

This command forces the termination of an iSCSI session

Syntax

```
stop storageArray iscsiSession [integer]
```

Parameter

Parameter	Description
integer	The identifier number of the iSCSI session. Enclose the identifier number of the iSCSI session in square brackets ([ ]).

Stop Physical Disk Blink

This command turns off the indicator LED on the physical disk that were turned on by the **start physicalDisk blink** command.

Syntax

```
stop physicalDisk blink
```

Parameters

None.

Stop Physical Disk Channel Fault Isolation Diagnostics

This command stops the physical disk channel fault isolation diagnostics, which stops the **start physicalDiskChannel** fault isolation diagnostics command before it completes. See the "Start Physical Disk Channel Fault Isolation Diagnostics" on page 224.

Syntax

```
stop physicalDiskChannel faultDiagnostics
```

Parameters

None.

Stop Snapshot

This command stops a copy-on-write operation.

Syntax

```
stop snapshot (virtualDisk [virtualDiskName] |
virtualDisks [virtualDiskName1 ...
virtualDiskNamen]
```

Parameters

Parameter	Description
virtualDisk or virtualDisks	Name of the specific virtual disk for which to stop a copy-on-write operation. You can enter more than one virtual disk name. You must put brackets ([ ]) around the virtual disk names. If the virtual disk names have special characters, you must also put quotation marks (" ") around the virtual disk names.

Stop Storage Array Blink

This command turns off the indicator LED on the storage array that were turned on by the **start storageArray blink** command.

Syntax

```
stop storageArray blink
```

Parameters

None.

Example

```
-c "stop storageArray iscsiSession [5];"
```

Stop Storage Array Physical Disk Firmware Download

This command stops a firmware download to the physical disks in a storage array that was started with the **download storageArray physicalDiskFirmware** command. This command does not stop a firmware download that is already in progress to a physical disk; however, this command stops all firmware downloads to physical disks that are waiting for the download.

Syntax

```
stop storageArray physicalDiskFirmwareDownload
```

Parameters

None.

Stop Virtual Disk Copy

This command stops a virtual disk copy operation.

Syntax

```
stop virtualDiskCopy target [targetName] [source
[sourceName]]
```

Parameters

Parameter	Description
target	Name of the target virtual disk for which to stop a virtual disk copy operation. You must put brackets ([ ]) around the target virtual disk name. If the target virtual disk name has special characters, you must also put quotation marks (" ") around the target virtual disk name.
source	Name of the source virtual disk for which to stop a virtual disk copy operation. You must put brackets ([ ]) around the source virtual disk name. If the source virtual disk name has special characters, you must also put quotation marks (" ") around the source virtual disk name.

A

Sample Script Files

This appendix provides sample scripts for configuring a storage array. These examples show how the script commands appear in a complete script file. You can copy these scripts and modify them to create a configuration unique to your storage array.

Create a script file in two ways:

- Use the **save storageArray configuration** command
- Write a script

By using the **save storageArray configuration** command, you can create a file to use to copy an existing configuration from one storage array to other storage arrays. You can also use this file to restore an existing configuration that has become corrupted. You can also copy an existing file to serve as a pattern from which you create a new script file by modifying portions of the original file. The default file extension is **.scr.**

Create a new script file using a text editor, such as Microsoft® Notepad. The maximum line length is 256 characters. The command syntax must conform to the guidelines in "Usage Guidelines" on page 40 and the rules in "Command Formatting Rules" on page 106. When creating a new script file, use any file name and extension that will run on the host operating system.

To run a script file from the command line, enter the following text:

```
client>smcli 123.45.67.89 -f scriptfile.scr;
```

Configuration Script Example 1

This example creates a new virtual disk using the **create virtualDisk** command in the free space of a disk group.

```
Show "Create RAID 5 Virtual Disk 7 on existing
Disk Group 1";

//Create virtual disk on a disk group created by
the create virtual disk command
```

```
//Note: For disk groups that use all available
capacity, the last virtual disk on the disk group
is created using all remaining capacity by
omitting the capacity=virtualDiskCapacity
parameter

create virtualDisk diskGroup=1 raidLevel=5
userLabel="7" owner=0 segmentSize=16 capacity=2GB;

show "Setting additional attributes for
virtualDisk 7";
//Configuration settings that cannot be set during
virtualDisk creation
set virtualDisk["7"] mediaScanEnabled=false;
set virtualDisk["7"] consistencyCheckEnabled=
false;
set virtualDisk["7"] modificationPriority=high;
```

This example shows blank lines between the lines beginning with **Show**, **Create**, //**Note**, and **create**. The blank lines are included in this example only for clarity. Each command is actually written on one line in the script file; however, the size of this page causes the command text to wrap. You might want to include blank lines in your script files to separate blocks of commands or make a comment more outstanding. You can do this by entering two forward slashes (//), which causes the script engine to treat the line as a comment.

The first line of text is the **show string** command. This command displays text bounded by quotation marks (" ") on a monitor screen when the script file runs. In this example, the text **Create RAID 5 Virtual Disk 7 on existing Disk Group 1** serves as a title describing the expected results of running this script file.

The line beginning //**Create** is a comment explaining that the purpose of this script file is to create a new virtual disk by using the **create virtualDisk** command on an existing disk group.

The line beginning //**Note:** is a comment in the script file explaining that the size of the last virtual disk created uses all of the available capacity because the **capacity** parameter is not used.

The command in this example creates a new virtual disk in disk group 1. The virtual disk has a redundant array of independent disks (RAID) level of 5. The virtual disk name (user label) is 7. (Note the quotation marks around the 7. The quotation marks indicate that the information in the marks is a label.) The new virtual disk is assigned to the RAID controller module in slot 0 in the RAID enclosure. The segment size is set to 16.

The following syntax is the general form of the command:

```
create virtualDisk diskGroup=diskGroupNumber
userLabel="virtualDiskName" [freeCapacityArea=
freeCapacityIndexNumber] [capacity=
virtualDiskCapacity | owner=(0 | 1) | segmentSize=
segmentSizeValue]
[enclosureLossProtect=(TRUE | FALSE)]
```

The general form of the command shows the optional parameters in a different sequence than the optional parameters in the example. You can enter optional parameters in any sequence. You must enter the required parameters in the sequence shown in the command descriptions.

The line **show "Setting additional attributes for virtual disk 7"** is another example of using the **show string** command. This command is placed here to tell you that the **create virtualDisk** command ran successfully. In addition, properties that could not be set by the **create virtualDisk** command are now set.

The **set virtualDisk** command parameters are shown on separate lines. You do not need to use separate lines for each parameter. You can enter more than one parameter with the **set virtualDisk** command by leaving a space between the parameters.

By using separate lines, however, you can more clearly see what parameters you are setting and the values to which you are setting the parameters. Blocking the parameters in this manner makes it easier to edit the file or copy specific parameter settings for use in another script file.

Configuration Script Example 2

This example creates a new virtual disk using the **create virtualDisk** command with user-defined physical disks in the storage array.

```
Show "Create RAID 5 Virtual Disk 2 on existing
Disk Group 2";

//This command creates the disk group and the
initial virtual disk on that group.

//Note: For disk groups that use all available
capacity, the last virtual disk on the group is
created using all remaining capacity by omitting
the capacity=virtualDisk creation parameter

create virtualDisk raidLevel=5 userLabel="2"
physicalDisks=[0,1 0,6 1,7 1,3 2,3 2,6] owner=1
segmentSize=16 capacity=2GB;

show "Setting additional attributes for virtual
disk 7";
//Configuration settings that cannot be set during
virtual disk creation
set virtualDisk["7"] mediaScanEnabled=false;
set virtualDisk["7"] consistencyCheckEnabled=
false;
set virtualDisk["7"] modificationPriority=high;
```

The command in this example, like the **create virtualDisk** command in the previous example, creates a new virtual disk. The significant difference between these two examples is that this example shows how you can define specific physical disks to include in the virtual disk. To find out what physical disks are available in a storage array, run the **show storageArray profile** command.

The following syntax is the general form of the **create virtualDisk** command shown in the previous example:

```
create virtualDisk raidLevel=(0 | 1 | 5 | 6)
userLabel="virtualDiskName" physicalDisks=
(enclosureID0,slotID0... enclosureIDn,slotIDn)
[capacity=virtualDiskCapacity | owner=(0 | 1) |
segmentSize=segmentSizeValue]
[enclosureLossProtect=(TRUE | FALSE)]
```

Index

A

activate storage array firmware, 114

adding comments to a script, 41

assigning global hot spares, 59

autoconfigure storage array, 115

autoconfigure storage array hot spares, 116

C

changing
- RAID controller module ownership, 101
- RAID level, 97
- segment size, 97
- Snapshot Virtual Disk settings, 73
- Virtual Disk Copy settings, 83

check virtual disk consistency, 117

clear
- physical disk channel statistics, 118
- storage array configuration, 118
- storage array event log, 119
- storage array firmware pending, 119
- virtual disk reservations, 120

CLI
- commands, 16
- usage examples, 27

clocks, RAID controller module, synchronizing, 95

collecting physical disk data, 98

command formatting rules, 106

command line interface, how to use, 14

command line parameters, 18

commands
- listed alphabetically, 114
- listed by function, 108

commands listed by function, 108

comments, adding, 41

configuration
- script example 1, 233
- script example 2, 236

configure
- autoconfigure storage array, 115
- autoconfigure storage array hot spares, 116
- changing RAID levels, 97
- changing segment size, 97
- create RAID virtual disk, automatic physical disk select, 126

configure *(continued)*
create RAID virtual disk, free capacity base select, 128
create RAID virtual disk, manual physical disk select, 130
storage array, 44
storage partitioning, 88

copying
virtual disk, 81

create
disk group, 120
host, 122
host group, 123
host port, 124
iSCSI initiator, 125
Snapshot Virtual Disk, 63-72
Snapshot virtual disk, 133
virtual disk, 48-52
Virtual Disk Copy, 78, 137

create RAID virtual disk
automatic physical disk select, 126
free capacity base select, 128
manual physical disk select, 130

creating
snapshot virtual disk, 66
Virtual Disk Copy, 79

D

defragmenting a disk group, 98

delete
disk group, 139
host, 140

delete *(continued)*
host group, 140
host port, 141
iSCSI initiator, 141
Snapshot Virtual Disk, 74-75
virtual disk, 142

detailed error reporting, 24

determining
Virtual Disk Copy candidates, 79
what is on your storage array, 44

diagnosing RAID controller module, 99, 143

disable storage array feature, 146

disk group
commands, 108
create disk group, 120
defragmenting, 98, 223
delete disk group, 139
reviving, 163
show, 204
stop blink, 229

download
enclosure management module firmware, 146
physical disk firmware, 147
storage array firmware NVSRAM, 148
storage array NVSRAM, 149
storage array physical disk firmware, 150

E

enable
  RAID controller module, 151
  storage array feature key, 151
  Virtual Disk Copy, 64, 79

enclosure
  commands, 109

enclosure loss protection, 52

enclosure management module firmware
  download, 146

exit status, 25

F

foreign physical disk
  set to native, 176

formatting considerations, 24

H

host
  create host, 122
  delete host, 140
  set host, 176
  show storage array host topology, 214
  topology commands, 109

host group
  create host group, 123
  delete host group, 140
  set host group, 178

host port
  create host port, 124
  delete host port, 141
  set, 179
  show host ports, 205

hot spare
  assigning global hot spares, 59
  set physical disk hot spare, 182

how to use the command line interface, 14

I

initializing
  physical disk, 101
  virtual disk, 102

interaction with other features, 88

ISCSI
  recurring syntax values, 39

iSCSI
  commands, 109

iSCSI initiator
  create iSCSI initiator, 125
  delete iSCSi initiator, 141

L

locating physical disks, 95

M

media scan, running, 91
modifying configuration, 55
monitoring performance, 96

P

partitioning, storage, 88
performance tuning, 95
persistent reservations,
removing, 94
physical disk
commands, 110
download firmware, 147
initializing, 101
locating, 95
reconstructing, 102
reviving, 163
set commands, 182-183
set state, 183
show commands, 205-208
start, 226-227
stop blink, 229

R

RAID controller module
changing ownership, 101
clocks, synchronizing, 95
commands, 111
diagnosing, 99, 143
enable RAID controller
module, 151
RAID controller module
(continued)
enabling RAID controller module
data, 94
reset, 94, 160
saving NVSRAM values, 165
set, 183
setting operational mode, 100
show, 208
RAID level, changing, 97
RAID virtual disk
recover, 153
reconstructing a physical
disk, 102
recopy
virtual disk, 86, 152
recopying virtual disk, 84
recover
RAID Virtual Disk, 153
RAID virtual disk, 153
recovery operations, 100
re-creating
snapshot virtual disk, 76
recurring syntax elements, 34
redistributing virtual disk, 103
redundancy check, running, 93
remove
copy pairs, 87
persistent reservations, 94
virtual disk copy, 158
virtual disk LUN mapping, 159

repair virtual disk
consistency, 160
reset
RAID controller module, 160
storage array battery install
date, 161
storage array virtual disk
distribution, 162
reset a RAID controller
module, 94
reset storage array iSCSI
baseline, 162
reset storage array SAS PHY
baseline, 162
restarting a Snapshot Virtual
Disk, 74-75
revive
disk group, 163
physical disk, 163
routine maintenance, 91
running
media scan, 91
redundancy check, 93

S

save
configuration to a file, 47
enclosure log data, 164
storage array, 166-170
storage array events, 167
storage array performance
statistics, 169
save *(continued)*
storage array SAS PHY
Counts, 169
storage array state capture, 170
storage array support data, 170
save storage array iSCSI
statistics, 168
saving
RAID controller module
NVSRAM values, 165
script command
structure, 30
synopsis, 32
usage guidelines, 40
segment size, changing, 97
session command, 111
Set, 170
set
enclosure id, 175
foreign physical disk to
native, 176
host group, 178
physical disk channel status, 182
physical disk hot spare, 182
physical disk state, 183
RAID controller module, 183
session, 188
Snapshot Virtual Disk, 189
storage array, 191
storage array enclosure
positions, 192
virtual disk, 198
Virtual Disk Copy, 203
set controller, 170

set disk group, 173
set enclosure attribute, 174
set host, 176
set host port, 179
set iSCSI initiator, 179
set iSCSI target properties, 180
set storage array ICMP response, 193
set storage array iSNS server IPv4 address, 194-195
set storage array iSNS server IPv6 address, 195
set storage array iSNS server listening port, 195
set storage array iSNS server refresh, 196
set storage array learn cycle, 197
set storage array time, 197
set unnamed discovery session, 198
setting
  controller clocks, 57
  modification priority, 58
  RAID controller module operational mode, 100
  storage array host type, 57
  storage array password, 56
show
  disk group, 204
  host ports, 205
  physical disk, 205
show *(continued)*
  physical disk channel statistics, 207
  physical disk download progress, 208
  RAID controller module, 208
  RAID controller module NVSRAM, 209
  storage array autoconfigure, 212
  storage array command, 210
  storage array host topology, 214
  storage array LUN mappings, 214
  storage array unreadable sectors, 215
  string, 216
  string command, 111
  unconfigured iSCSI initiators, 216
  virtual disk, 217
  virtual disk action progress, 218
  Virtual Disk Copy, 219
  Virtual Disk Copy source candidates, 220
  Virtual Disk Copy target candidates, 220
  virtual disk reservations, 222
show current iSCSI sessions, 203
show disk group import dependencies, 220
show storage array negotiation defaults, 214
show unconfigured iSCSI initiators, 216
show virtual disk performance statistics, 221

SMcli commands, 16

snapshot
commands, 111
names, 72
virtual disks, 89

Snapshot Virtual Disk
creating, 64-72
deleting, 76
enabling, 64
restarting, 75
stopping, 74-75

snapshot virtual disk
creating, 66
re-creating, 76

start
disk group blink, 223
disk group defragment, 223
enclosure blink, 223
physical disk blink, 226
physical disk initialize, 226
physical disk reconstruction, 227
storage array blink, 227
virtual disk initialization, 228

start disk group import, 227

Start Disk Group Import/Export, 227

start iSCSI DHCP refresh, 224

start physical disk channel fault isolation diagnostics, 224

stop
disk group blink, 229
enclosure blink, 229
physical disk blink, 229

stop *(continued)*
Snapshot, 230
Snapshot Virtual Disk, 74-75
storage array blink, 231
storage array physical disk firmware download, 231
Virtual Disk Copy, 87, 231

stop iSCSI session, 229

stop physical disk channel fault isolation diagnostics, 230

storage array
accept pending topology, 114
commands, 111
download commands, 148-151
save commands, 166-170
set commands, 191-198
show commands, 210-216
show host topology, 214
show LUN mappings, 214
show pending topology, 215
show unreadable sectors, 215
start blink, 227
stop blink, 231

storage array feature
disable, 146

storage partitioning, 88

support.dell.com, 13

synchronizing RAID controller module clocks, 95

syntax element statement data, 185

syntax elements
ISCSI values, 39

T

troubleshooting storage array, 98

U

user-defined parameters, 70

V

virtual disk
  check consistency, 117
  commands, 113
  copying, 81
  creating in an existing disk group, 51
  creating with software-assigned physical disks, 50
  creating with user-assigned physical disks, 48
  initializing, 102
virtual disk *(continued)*
  recopy, 86
  recopy virtual disk, 152
  recopying, 84
  redistributing, 103
  repair consistency, 160
  set, 198
  show commands, 217-222
Virtual Disk Copy
  command, 113
  show, 219
  stop, 231
  viewing properties, 82
virtual disk copy
  remove, 158