



EMC[®] Solutions Enabler Symmetrix[®] Array Controls CLI

Version 7.0

Product Guide

P/N 300-002-940

REV A06

EMC Corporation

Corporate Headquarters:
Hopkinton, MA 01748-9103

1-508-435-1000

www.EMC.com

Copyright © 2005 - 2009 EMC Corporation. All rights reserved.

Published April, 2009

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date regulatory document for your product line, go to the Technical Documentation and Advisories section on EMC Powerlink.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

All other trademarks used herein are the property of their respective owners.

Part I Concepts and Procedures

Chapter 1 Managing Configuration Changes

Introduction	22
Mixed array environments.....	23
Configuration change session overview	24
Making configuration changes safely	28
Impact on I/O	29
Configuration control operations	30
Enginuity support for configuration controls	31
Setting Symmetrix metrics.....	35
Subsystem IDs for mainframe systems	36
Planning system protection configurations	37
Device mirrors.....	37
2-Way mirror protection.....	40
Parity RAID protection	41
SRDF remote protection	42
Device creation strategy	44
Managing devices	46
Copying a device	49
Managing metadevices	51
Creating metadevices.....	52
Meta configuration restrictions	56
Expanding an existing metadevice	56
Removing metamembers.....	57
Dissolving a metadevice.....	57
Configuring thin metadevices	58
Binding a thin metadevice to a pool	58
Data preservation concerns.....	59
Managing spare disks	60
Adding a spare disk	60
Deleting a spare disk.....	60
Viewing existing spares.....	60
Converting devices.....	61
Converting thin devices.....	63
Converting a metadevice.....	64
Changes to mirror protection	65
VDEV configuration restrictions	65
SAVE device configuration restrictions	65

RDF configuration changes	65
Device locking	66
Held devices	66
Setting device attributes	67
Setting device geometry	68
Showing device geometry	69
Setting device identifiers	71
Identifier exclusions	71
Viewing device identifiers	72
Deleting devices	74
Managing device reservations	76
Reserving devices	76
Committing changes on reserved devices	77
Viewing existing reserved devices	78
Releasing reserved devices	79
Configuring Virtual Provisioning	80
Creating DATA devices	81
Creating thin devices (TDEVs)	82
Creating thin pools	84
Thin pool restrictions	84
Setting up Virtual Provisioning	85
Allocating space on a thin device	86
Freeing up unwritten allocated space	86
Setting pool limits	88
Managing device pools	90
Creating a device pool	91
Deleting a device pool	92
Adding devices to a pool	92
Removing devices from a pool	92
Enabling/disabling devices	92
Monitoring a device pool	93
Viewing a device pool	94
Mapping devices	96
Mapping CKD devices	98
Mapping a range of devices	98
Copying the mapping of a range of devices	99
Assigning PAV alias addresses to mapped devices	99
Removing PAV alias addresses from mapped devices	100
Unmapping devices	101
Unmapping a range of devices	101
Configuring dynamic RDF-capable devices	103
Enabling dynamic RDF on the device	103
Setting RDF group attributes	105
Adding/removing RDF mirrors	107
Add/remove mirror restrictions	108
Swapping RA groups	110
Swapping RDF devices	111
Setting port characteristics	112

Chapter 2 Masking Devices with Autoprovisioning Groups

Autoprovisioning Groups	118
What's different?	118
How it works	118
Discovering host HBAs	120

Creating groups and views	121
Creating storage groups	121
Creating port groups.....	122
Creating initiator groups.....	123
Creating a masking view.....	124
Managing masking views.....	126
Deleting masking views	126
Naming groups and views.....	126
Backing up and restoring views.....	126
Managing storage groups.....	128
Adding devices	128
Removing devices.....	129
Deleting storage groups.....	129
Managing port groups	130
Adding ports	130
Removing ports.....	130
Deleting port groups.....	130
Using CHAP authentication	131
Locking down a Fibre Channel ID	131
Managing initiator groups.....	133
Adding initiators	133
Removing initiators.....	134
Deleting initiator groups	134
Setting HBA flags	134
Replacing an HBA	136
Displaying Autoprovisioning information	137
Viewing group details.....	137
Autoprovisioning Groups examples.....	139

Chapter 3 **Device Masking**

Device masking overview	142
How Symmetrix device masking works	142
Host access examples	142
Managing access for unsupported host platforms	144
Device masking controls overview	145
Database device locking	145
Supported topologies	146
Fibre Channel topology	146
Native iSCSI topology.....	147
Device masking VCMDB security enhancement	148
VCMDB management updates.....	148
Initial device masking setup	149
Access control environment setup	149
Device masking configuration	150
Identify configuration components	150
Initialize and update the database	151
Enable authentication	151
Recommendations for activating the configuration.....	151
Discovering host HBAs.....	153
Using alias names	153
How to add and remove masked devices.....	155
Adding devices	155
Using reserved devices	155
Adding device group devices.....	155

Adding devices from a device file.....	156
Adding devices with dynamic LUN addressing.....	156
Removing devices	157
Removing device group devices.....	158
Removing devices using a device file	158
Device masking VCMDB maintenance.....	159
Initializing the database	159
Preventing unauthorized modification of the VCMDB	159
Managing a backup VCMDB file.....	160
VCMDB database types	160
Converting a VCMDB type	161
Restoring a backup VCMDB	161
Restoring a backup and converting its type.....	161
Blocking direct writes to the VCMDB.....	162
How to view the VCMDB	163
Viewing the database	163
Viewing device capacity	165
Viewing the login history table.....	166
Refreshing the database	166
HBA initiator management.....	169
Creating and renaming aliases.....	169
Swapping a new HBA for a failed HBA	169
Deleting HBA associations	170
Fibre Channel-to-host interface management.....	171
Locking down a Fibre Channel ID	171
Setting device LUN visibility	173
Setting the LUN base/offset skip adjustment	173
Setting the HBA port flags.....	175
Setting the heterogeneous host configuration	176

Chapter 4 **Device Masking: iSCSI Setup**

iSCSI authentication overview	180
iSCSI CHAP authentication: Enginuity version 56xx	181
Requirements	181
Setting iSCSI authentication with the SYMCLI	181
Showing authentication information	181
Restoring authentication data and updating the iSCSI driver	182
iSCSI CHAP authentication: Enginuity version 57xx	183
Setting one-way CHAP authentication.....	183
Setting two-way CHAP authentication	184
Initializing an authentication database.....	184
Displaying authentication information	185
Backing up and restoring authentication information	186
iSCSI software driver configuration.....	187
Requirements	187
Installing the iSCSI software initiator	187
Configuring iSCSI with CHAP authentication.....	188
Configuring iSCSI for two-way chap authentication	195
Configuring iSCSI without CHAP authentication.....	196
RADIUS server configuration	201

Chapter 5	Managing Network IPsec	
	Introduction	204
	IPsec standard	204
	IPsec and Symmetrix	205
	SYMCLI IPsec management component	207
	Viewing policy information	208
	Command file scripting for creating/modifying policies	212
	Command file syntax adding/modifying policies	212
	Committing a policy-scripted command file	217
	Deleting a policy-scripted command file	218
 Chapter 6	 Optimizing Array Performance	
	What is Symmetrix Optimizer?	220
	How Optimizer works	220
	Supported environments	221
	Configuration requirements	222
	Dynamic reallocation volumes (DRVs)	222
	Open mirror position requirement	222
	Open configuration lock requirement	222
	How does Symmetrix Optimizer work?	224
	The analysis	224
	Performance metrics	224
	Modeling service time	225
	Finding the best swap	225
	Swap procedure for Engenuity 5874 and higher	226
	Swap procedure for Engenuity 5773 and earlier	226
	Legality of swaps	228
	Impact of swaps	230
	Symmetrix Optimizer operations using SYMCLI	231
	Starting and stopping Optimizer	231
	Showing Optimizer information	232
	Reading Optimizer logs	232
	Committing command file changes	232
	Approving a swap list	233
	Setting time windows	234
	Clearing a time window	235
	Setting swap priority	235
	Performing rollbacks	235
	Setting manual swap lists	235
	Migrating devices	236
	Setting control parameters	238
 Chapter 7	 Enhanced Virtual LUN Technology	
	Overview	240
	What's happening inside the array	241
	Source devices	242
	Target criteria	242
	Using files	242
	Creating virtual LUN migration sessions	243
	Validate	243
	Establish	243
	Migrating to configured space	244
	Migrating to unconfigured space	246

Managing virtual LUN migration sessions	247
Listing sessions.....	247
Querying sessions	248
Verifying sessions	248
Terminating sessions	249
Viewing the audit log	250
Virtual LUN migration examples	251
 Chapter 8	
Performing Double Checksum Operations	
Overview	254
Traditional methods of preventing data corruption.....	254
Data corruption between host and conventional storage	254
Benefits of checking within the Symmetrix array	255
Implementing EMC Double Checksum for Oracle	256
Other checksum operations.....	256
Suppressing default features.....	257
Enabling checksum options.....	257
Verifying checksum is enabled	258
Validating for checksum operations	258
Disabling checksum.....	259
Functionality by Enginuity level	259
Implementing Generic SafeWrite for generic applications	260
Torn pages: Using generic safewrite to protect applications.....	260
Why generic?	260
Where to enable Generic SafeWrite.....	261
Configuring Generic SafeWrite.....	262
How to disable Generic SafeWrite.....	263
Listing Generic SafeWrite devices	264
Performance considerations	264
Syntax and examples	265
 Chapter 9	
Managing Quality of Service	
Overview	268
Setting QoS copy parameters and values	269
Copy pace to certain devices	269
Setting QoS LRU parameters and values.....	270
Using Symmetrix Priority Controls	271
Enabling and disabling priority of service	271
Setting priority of service.....	271
Listing priority of service.....	272
Using dynamic cache partitions.....	273
Enabling cache partitions.....	273
Using analyze mode	273
Empty partition removal.....	274
Creating cache partitions	275
Adding devices to a cache partition.....	275
Modifying a cache partition	276
Removing devices from a cache partition	276
Listing cache partitions	276
QoS examples.....	278

Part 2 Operational Examples

Appendix A Configuration Change Examples

Example 1: Creating devices	282
Example 2: Mapping devices	288
Example 3: Mapping a single device on multiple EA ports	293
Example 4: Mapping a range of devices and adding aliases	299
Example 5: Unmapping devices	304
Example 6: Setting front-end port attributes	308
Example 7: Configuring concurrent RDF devices	313
Example 8: Updating a disk label with emulated device geometry	319
Example 9: Configuring virtual provisioning	327
Create a pool.....	327
Create DATA devices.....	327
Adding DATA devices to the pool	330
Setting pool limits.....	331
Creating thin devices and binding them to the pool.....	332
Binding with preallocate	335
Disabling DATA devices	337
Allocating space for thin devices	338
Freeing space on thin devices	339
Example 10: Automatically creating metadevices	342

Appendix B Virtual LUN Migration Example

Virtual LUN migration example	344
-------------------------------------	-----

Appendix C Updating the Host

Sun Solaris systems	356
HP-UX systems	357
IBM AIX systems	358
HP Tru64 UNIX systems	359
Windows systems	360
Windows 2000	360
Windows 2003	360

Appendix D SIU: Overview and Management

Introduction	362
SIU capabilities.....	362
Privilege requirements.....	362
SIU commands	362
Volume management software.....	363
SFW dynamic disk groups with TimeFinder	363
Configuring a VMware virtual server	365
Viewing detailed information about disks	366
Specifying the output type	366
Mounting and unmounting disks	367
SIU and SYMAPI database.....	367
Disk unmount operations.....	367
Manipulating disk signatures	369
Updating partition tables	370
Flushing cache buffers to disk	371
Device masking operations	372

Using volume flags	374
Performing typical storage operations	375
Synchronizing the BCV	375
Restoring Veritas dynamic disks with TimeFinder	377
Creating a backup using TimeFinder/Snap	378
Creating a backup using TimeFinder/Clone	379
SIU Examples	380
XML EXAMPLES	382

Index

	Title	Page
1	Symmetrix concatenated metadvice two-way mirror scheme	40
2	Symmetrix parity RAID mirror scheme	41
3	Simple SRDF device configuration	42
4	Various SRDF-connected device configurations with BCVs	43
5	Concatenated metadvice	51
6	Striped data	52
7	Data allocation for thin devices	81
8	Masking view overview	119
9	Storage group SG_1	121
10	Port group PG_1	122
11	Initiator group IG_1	123
12	Masking view MV_1	124
13	Two masking views	125
14	Device masking solution with WWN initiators	143
15	Point-to-point and multi-initiator topologies	146
16	Native iSCSI topology	147
17	Device masking syscalls	148
18	Device masking components	150
19	IPsec network tunnel with Symmetrix-to-Symmetrix endpoints	205
20	IPsec network tunnel with iSCSI host-to-Symmetrix endpoints	206
21	Swap steps using one DRV	226
22	Identify volumes to swap	227
23	Copy volume to DRV	227
24	Copy DRVs to new locations	227
25	DRVs are split	228
26	Initial state of RAID 1 device with SRDF	241
27	Synchronizing the RAID 5 group	241
28	RAID 5 group with SRDF	241
29	Generic SafeWrite list output	264
30	Device view prior to migration session	344
31	Device view after the migration session	350

	Title	Page
1	Configuration change controls	23
2	Configuration Manager operations	31
3	Virtual mirror positions	37
4	Mirror set protection schemes	38
5	Steps to create a device	44
6	Device sizes by Enginuity version	46
7	Metadevice sizes by Enginuity version for auto_meta	51
8	Steps to create an RDF BCV metadevice	55
9	Metadevice data preservation	59
10	Thin device conversions	63
11	Supported thin metadevice features	83
12	SCSI protocol port flags	113
13	Fibre protocol port flags	115
14	LUN base/offset scenarios for multiple hosts with skip holes	174
15	Host platforms and interface configuration flags	176
16	Optimizer strategies for determining swaps	224
17	Symmetrix Optimizer command	231
18	Parameters in the command file	238
19	Action descriptions for symmigrate command	243
20	Audit log session information	250
21	EMC Double Checksum for Oracle functionality by Enginuity level	259
22	symchksum syntax and descriptions	265
23	SYMAPI Quality of Service command	268
24	SIU command summary	363

As part of an effort to improve and enhance the performance and capabilities of its product lines, EMC periodically releases revisions of its hardware and software. Therefore, some functions described in this document may not be supported by all versions of the software or hardware currently in use. For the most up-to-date information on product features, refer to your product release notes.

If a product does not function properly or does not function as described in this document, please contact your EMC representative.

Audience

This document is part of the Solutions Enabler documentation set, and is intended for use by advanced command-line users and script programmers to manage various types of control operations on Symmetrix arrays and devices using the SYMCLI commands of the EMC Solutions Enabler software.

Related documentation

Related documents include:

- ◆ *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*
- ◆ *EMC Solutions Enabler Symmetrix CLI Array Management Product Guide*
- ◆ *EMC Solutions Enabler Installation Guide*
- ◆ *EMC Solutions Enabler SRDF Family CLI Product Guide*
- ◆ *EMC Solutions Enabler Symmetrix SRM CLI Product Guide*
- ◆ *EMC Solutions Enabler TimeFinder Family CLI Product Guide*
- ◆ *EMC host connectivity guides for [your operating system]*

Note: Detailed man page descriptions of all SYMCLI commands can now be found in the companion *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

Conventions used in this document

EMC uses the following conventions for notes, cautions, warnings, and danger notices.

In this document, every use of the word SYMCLI means Solutions Enabler.

Every occurrence of the word MVS in Solutions Enabler software symbols, means OS/390 and z/OS.

Every occurrence of *Symmetrix 4*, refers to Symmetrix 3xxx and 5xxx model numbers.

Every occurrence of *Symmetrix 5*, refers to Symmetrix 8xxx model numbers.

Note: A note presents information that is important, but not hazard-related.



CAUTION

A caution contains information essential to avoid data loss or damage to the system or equipment. The caution may apply to hardware or software.



IMPORTANT

An important notice contains information essential to operation of the software. The important notice applies only to software.

Typographical conventions

EMC uses the following type style conventions in this document:

Normal	Used in running (nonprocedural) text for: - Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus) - Names of resources, attributes, pools, Boolean expressions, buttons, DQL statements, keywords, clauses, environment variables, functions, utilities - URLs, pathnames, filenames, directory names, computer names, filenames, links, groups, service keys, file systems, notifications
Bold	Used in running (nonprocedural) text for: - Names of commands, daemons, options, programs, processes, services, applications, utilities, kernels, notifications, system calls, man pages Used in procedures for: - Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus) - What user specifically selects, clicks, presses, or types
<i>Italic</i>	Used in all text (including procedures) for: - Full titles of publications referenced in text - Emphasis (for example a new term) - Variables
Courier	Used for: - System output, such as an error message or script - URLs, complete paths, filenames, prompts, and syntax when shown outside of running text
Courier bold	Used for: Specific user input (such as commands)
<i>Courier italic</i>	Used in procedures for: - Variables on command line - User input variables
< >	Angle brackets enclose parameter or variable values supplied by the user
[]	Square brackets enclose optional values
	Vertical bar indicates alternate selections - the bar means "or"
{ }	Braces indicate content that you must specify (that is, x or y or z)
...	Ellipses indicate nonessential information omitted from the example

Where to get help

EMC support, product, and licensing information can be obtained as follows.

Product information — For documentation, release notes, software updates, or for information about EMC products, licensing, and service, go to the EMC Powerlink website (registration required) at:

<http://Powerlink.EMC.com>

Technical support — For technical support, go to EMC Customer Service on Powerlink. To open a service request through Powerlink, you must have a valid support agreement. Please contact your EMC sales representative for details about obtaining a valid support agreement or to answer any questions about your account.

Your comments

Your suggestions will help us continue to improve the accuracy, organization, and overall quality of the user publications. Please send your opinion of this document to:

techpubcomments@EMC.com

The Concepts and Procedures part of this product guide provides conceptual information and describes how to perform various types of control operations on Symmetrix devices using the SYMCLI commands of the EMC Solutions Enabler software. These concepts and procedures are described in the subsequent chapters, as follows:

[Chapter 1, “Managing Configuration Changes,”](#) describes configuration concepts and how to manage various types of configurations using the configuration commands of the SYMCLI.

[Chapter 2, “Masking Devices with Autoprovisioning Groups,”](#) describes storage provisioning concepts and how to confine host access to Symmetrix devices using the `symaccess` command of the SYMCLI.

[Chapter 3, “Device Masking,”](#) describes the device masking concepts and how to confine host access to Symmetrix devices using the device masking commands of the SYMCLI.

[Chapter 4, “Device Masking: iSCSI Setup,”](#) describes how to configure your iSCSI driver software and authentication information.

[Chapter 5, “Managing Network IPsec,”](#) describes the IPsec standard and how to manage IPsec network policies, using the IPsec component of the SYMCLI.

[Chapter 6, “Optimizing Array Performance,”](#) describes the Symmetrix Optimizer concepts and how to improve array performance using the Symmetrix Optimizer commands of the SYMCLI.

[Chapter 7, “Enhanced Virtual LUN Technology,”](#) explains how to perform virtual LUN migrations within Symmetrix arrays to change device protection or improve array performance using the SYMCLI.

[Chapter 8, “Performing Double Checksum Operations,”](#) describes the Double Checksum concepts and how to minimize the impact of I/O errors on database consistency during I/O transfers between hosts and Symmetrix storage devices using the Double Checksum commands of the SYMCLI.

[Chapter 9, “Managing Quality of Service,”](#) describes the QOS concepts and how to manage the Quality of Service (QoS) on devices in your storage environment using the QOS commands of the SYMCLI.

Note: For practical examples illustrating how to perform specific array control tasks with SYMCLI command sequences, refer to [Part 2](#) of this guide.

This chapter describes configuration change concepts and how to perform change operations using the Configuration Change component of the SYMCLI. The chapter covers the following topics:

◆ Introduction	22
◆ Configuration change session overview	24
◆ Making configuration changes safely	28
◆ Configuration control operations	30
◆ Setting Symmetrix metrics	35
◆ Planning system protection configurations.....	37
◆ Managing devices	46
◆ Managing metadevices.....	51
◆ Managing spare disks.....	60
◆ Converting devices	61
◆ Setting device attributes	67
◆ Setting device geometry	68
◆ Setting device identifiers.....	71
◆ Deleting devices	74
◆ Managing device reservations.....	76
◆ Configuring Virtual Provisioning	80
◆ Managing device pools.....	90
◆ Mapping devices	96
◆ Mapping CKD devices	98
◆ Unmapping devices	101
◆ Configuring dynamic RDF-capable devices	103
◆ Setting RDF group attributes.....	105
◆ Adding/removing RDF mirrors	107
◆ Swapping RA groups.....	110
◆ Swapping RDF devices.....	111
◆ Setting port characteristics	112

Introduction

The SYMCLI configuration change command, `symconfigure`, is used to perform control operations on Symmetrix® arrays and the array devices and ports. Some of the Symmetrix array controls include setting array-wide metrics, and determining what type of devices the array will support, such as RAID 6 devices. Device controls include creating devices, mapping devices, and configuring device pools. The `symconfigure` command is also used for reserving devices and releasing device reservations.

You can invoke `symconfigure` from the local host to make configuration changes to a locally-connected Symmetrix array, or to an RDF-linked Symmetrix array.

There are three ways to process `symconfigure` commands, as follows:

- ◆ The most common way to use `symconfigure` is placing commands, terminated with a semicolon (;), in a command file. An example follows:

```
symconfigure commit -sid 3160 -file unmap_dev.cmd
```

Where `unmap_dev.cmd` contains:

```
unmap dev 020:024 from dir ALL:ALL;
```

- ◆ Optionally, for UNIX platforms, you can redirect a number of screen entries to `stdin` to save keystroke entries and not deal with the command file.

For example, to prepare a chain of `symconfigure` commands on the screen to be redirected to `stdin`, use the following syntax:

```
symconfigure -sid SymmID prepare <<DELIM
create dev count=3 size=3200 cyl,
emulation=FBA, config=2-Way-Mir,
create dev count=1, size = 3200 cyl,
emulation=FBA, config=unprotected;
DELIM
```

- ◆ A similar screen method can be used with the `-cmd` option. With this option, the commands that would normally be put in a command file are enclosed in quotes. A command can run over many lines, but you cannot press **Enter**. An example follows:

```
symconfigure -sid 256 -cmd "create dev count=1, size = 3200 cyl,
emulation=FBA, config=2-Way-Mir;create dev count=1, size = 3200 cyl,
emulation=FBA, config=unprotected;" -v -nop preview
```

This section describes the `symconfigure` arguments, and the remaining chapter explains the command operations that can be written in a command file.

Configuration changes that are submitted to the array are processed in a session. The `symconfigure` command file format contains various command entries terminated with a semicolon (;). Multiple changes can be made in one session, but all changes must fall into one complete operation. For example, creating a device, adding the device to a device pool, and enabling the device state. Additional command file examples are provided with the actions described later in this chapter.

Table 1 defines the `symconfigure` command arguments:

Table 1 Configuration change controls

symconfigure option	Definition
abort	Attempts to gain control of an existing session to abort it and free any held locks.
commit	Attempts to apply the changes defined in the command file into the specified Symmetrix array.
list	Lists the relevant details, depending on the option: When used with <code>-freespace</code> , shows the free physical disk space within the Symmetrix array as it can be used to create new Symmetrix devices for different emulation modes. Free disk space on unformatted disks is shown as available for all emulation modes. If a physical disk has been partially used to create a device, that device is considered formatted and the rest of the available space can only be used for devices of the same emulation mode. When used with <code>-v</code> , displays configuration information that is not stored in the SYMAPI database and that needs to be retrieved directly from the configuration server. When used with <code>-reserve</code> , shows a summary of all reservations.
prepare	Validates the syntax and correctness of the operations. Verifies the validity of the command file changes and their appropriateness for the specified Symmetrix array. The prepare action has no function for pool sessions.
preview	Makes sure that the command file syntax is correct. Verifies the validity of the command file changes.
query	Returns information about the status of a configuration change session.
release	Releases the specified device reservation.
reserve	Processes the command file to reserve the indicated devices and displays the resulting reserve ID.
show	Shows the details of the specified device reservation.
verify	Verifies that the configuration currently running in the specified Symmetrix array complies with the requirements for host-based configuration changes.

Refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help* for the complete man page description of the `symconfigure` command.

Mixed array environments

Before issuing a `symconfigure` command to a Symmetrix array containing both open system and mainframe devices, you must verify that the mainframe Missing Interrupt Handler (MIH) period for *each* mainframe server attached to the Symmetrix array is set to at least 2 minutes.

To view the current MIH period, use the z/OS command `D IOS,MIH`, and note the value for the DASD device class (for example, `DASD=1:30`).

To change the MIH period, use the z/OS command `SETIOS MIH,DASD=mm:ss`, where `mm:ss` is a period of time in minutes and seconds. Then use the z/OS command `D IOS,MIH` to verify your changes.

Once you have completed your configuration change session, use the `SETIOS MIH,DASD=mm:ss` command to set the MIH period back to its original value.

For more information, refer to your mainframe system administrator.

Configuration change session overview

Solutions Enabler 7.0 running on the Symmetrix V-Max™ Series with Enginuity™ handles configuration changes differently from previous versions of Solutions Enabler. The following is a list of changes in the configuration change feature of Solutions Enabler 7.0 with the Enginuity operating environment for Symmetrix:

- ◆ **Concurrent Provisioning:** Up to four concurrent configuration change sessions are allowed to run at the same time, when they are non-conflicting. This means that multiple parallel configuration change sessions can run at the same time as long as the changes do not include any conflicts on the following:
 - Device back-end port
 - Device front-end port
 - Device
- ◆ Devices and metadevices can be created and mapped in one command. This eliminates the need to perform multiple configuration changes when provisioning new storage.
- ◆ The Symmetrix array manages its own less restrictive device locking.
- ◆ A session ID identifies each running session on the array.

Note: Features that are new in Enginuity 5874 are not supported on arrays running Enginuity 5773 and earlier unless noted.

A session acting on a specified `symconfigure` command file can be verified and checked using the `preview` and `prepare` arguments. The `commit` argument performs these same checks, but then attempts to execute the specified action.

Not all stages are always executed. Use discretion when controlling which stages are to be completed to allow checking and debugging of the command file before the changes are implemented.

Preview The `preview` argument verifies the syntax and correctness of each individual change defined, their correctness as a set, and then terminates the session without change execution (correct within the realm of the host and valid as a possible Symmetrix configuration).

Prepare The `prepare` argument performs the preview checks, validates the change operation (devices are in correct state, etc.), and also verifies the appropriateness of the resulting configuration definition against the current state of the Symmetrix arrays; the argument then terminates the session without attempting to make the configuration change. For Symmetrix V-Max arrays running Enginuity 5874, the `prepare` argument performs the same actions as the `preview` argument.

Note: The `prepare` argument has no function for SAVE device pool sessions.

An example of system output for `symconfigure prepare` command follows:

```
symconfigure -sid 256 -file script.mkdevice -v -nop prepare
```

```
A Configuration Change operation is in progress. Please wait...
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000187940256
{
  create dev count=1, size=3200, emulation=FBA,
```



```

    config=2-Way Mir, mvs_ssid=0000;
}

Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Submitting configuration changes.....Submitted
Validating configuration changes.....Validated.

New symdev: 038B
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 013 of 017 steps.....Executing.
Step 014 of 017 steps.....Executing.
Local: PREPARE.....Done.
Closing configuration change request.....Closed.
Terminating the configuration change session.....Done.

```

The configuration change session has completed successfully.

Commit The `commit` argument completes all checks and verifications, and then attempts to make the requested configuration changes in the specified Symmetrix array.

Query After you commit a command file for execution, it will take a few minutes to complete execution. You can check the status of any configuration change session using the `query` argument.

To check the status of the change session for the next 2 minutes, every 10 seconds on Symmetrix 12345, enter:

```
symconfigure -sid 12345 query -i 10 -c 12
```

This command is useful in RDF environments, where a change to a local Symmetrix array on one host results in a corresponding change to a remote Symmetrix array. The System Manager of a host connected to the remote Symmetrix array can monitor the progress of the change. A `query` is also helpful at sites where the Symmetrix Optimizer is also modifying a configuration by rearranging the placement of a Symmetrix device's mirrors.

Solutions Enabler 7.0 running on a Symmetrix V-Max Series array with Enginuity 5874 adds a `-session_id` option to the `symconfigure query` command, to differentiate between multiple running sessions. The `query` command is used to show if there is an EMC Internal session running.

Below is an example of the `symconfigure query` command on a Symmetrix array running Enginuity 5874:

```
symconfigure -sid 207 query
```

A Configuration Change query is in progress. Please wait...

A Configuration Change operation is in progress. Please wait...

```

Establishing a monitoring session.....Established.

Session ID : 158209 (0x00026a01)
{
  The session changes are in the class of: Modifying symmetrix constraints
  {
    set symmetrix auto_meta_config=Concatenated;
  }

  The Application that initiated the configuration change : SYMCONFIGURE
  The Host that initiated the configuration change       : api1051
  The Process ID that initiated the configuration change  : 31831
  The session length                                     : 12 secs

```

```

    The session status                                     : Running

    The last action requested was: COMMIT
    The state of the last action is: Running
    Step 46 of 65 steps.....Executing.
}

Session ID  : NA
{
    The session changes are in the class of: NA
    The Application that initiated the configuration change : EMC Internal
    The Host that initiated the configuration change       : NA
    The Process ID that initiated the configuration change  : NA
    The session length                                   : 56 secs
    The session status                                     : Running
}

Session ID : 201 (0x000000c9)
{
    The session changes are in the class of: mapping devices
    {
        map dev 100 to dir 15D:0, lun = F0;
    }

    The Application that initiated the configuration change : SYMCONFIGURE
    The Host that initiated the configuration change       : xxxxxx
    The Process ID that initiated the configuration change  : 244
    The session length                                   : 30 secs
    The session status                                     : Running

    The last action requested was: COMMIT
    The state of the last action is: Running
    Step 57 of 105 steps.....Executing.
}

Terminating the monitoring session.....Done.

```

Abort You can abort a change session. The abort action also frees the configuration session lock.

To abort a change session on Symmetrix 12345, enter:

```
symconfigure -sid 12345 abort
```

Because changes made in the RDF operations class will initiate actions on the local Symmetrix and remote Symmetrix arrays, it might become necessary to abort processing on a remote Symmetrix array.

At some point during commit processing, a *point of no return* is reached. Any attempt to abort will be denied once processing has reached this point.

Add the `-session_id` option to the `symconfigure abort` command, to differentiate between multiple running sessions. To abort a change session on a Symmetrix array 343, enter:

```
symconfigure -sid 343 abort -session_id 100
```

If the `session_id` is not known, running the `symconfigure -sid SymmID abort` command will display each running session and prompt for the session ID, as shown in the following example:

```
symconfigure -sid 343 abort
```

```
A Configuration Change operation is in progress. Please wait...
```

Looking for an existing configuration sessions.....Established.

Session ID : 100 (0x00000064)

```
{
  {
    set symmetrix auto_meta = Disabled;
  }
}
```

```
The Application that initiated the configuration change : SYMCONFIGURE
The Host that initiated the configuration change       : api1182
The Process ID that initiated the configuration change  : 15023
The session length                                   : 28 secs
The session status                                    : Running
```

}

Session ID : 105 (0x00000069)

```
{
  {
    map dev 100 to dir 15D:0, lun = F0;
  }
}
```

```
The Application that initiated the configuration change : SYMCONFIGURE
The Host that initiated the configuration change       : api1051
The Process ID that initiated the configuration change  : 244
The session length                                   : 30 secs
The session status                                    : Running
```

}

Session ID : 2100

```
{
  {
    map dev 104 to dir 2C:0, lun = 0F;
  }
}
```

```
The Application that initiated the configuration change : SYMCONFIGURE
The Host that initiated the configuration change       : api171
The Process ID that initiated the configuration change  : 248
The session length                                   : 22 secs
The session status                                    : Running
```

}

Please enter the ID of the session you wish return to cancel abort: **100**

Session ID : 100 (0x00000064)

```
{
  {
    set symmetrix auto_meta = Disabled;
  }
}
```

```
The Application that initiated the configuration change : SYMCONFIGURE
The Host that initiated the configuration change       : api1182
The Process ID that initiated the configuration change  : 15023
The session length                                   : 28 secs
The session status                                    : Running
```

}

Abort the configuration change session 1 for Symmetrix unit 000190300343 (y/[n])? **y**

Aborting configuration changes.....Aborted.

Terminating the configuration change session.....Done.

Making configuration changes safely

Before making configuration changes, you must thoroughly understand your Symmetrix configuration. The following guidelines help to establish safe disciplines as you begin any change that can easily impact stored data:

- ◆ Verify that the current Symmetrix configuration is a viable configuration for host-initiated configuration changes.

To verify your current Symmetrix configuration is ready for changes, enter:

```
symconfigure verify -sid SymmID
```

- ◆ Before creating new Symmetrix devices, check for free physical disk space using the following command:

```
symconfigure list -freespace [-units CYLINDERS|MB] -sid SymmID
```

Free disk space on unformatted disks is shown as available for all emulation modes. New Symmetrix devices will be created first on physical disks that have no prior allocations, causing these disks to be committed to that emulation type.

Examine the distribution of freespace across formatted disks to see if the desired mirroring can be provided using one of the following commands:

```
symdev list -sid SymmID -da all -space  
or  
symdisk list -sid SymmID
```

- ◆ No configuration change is activated in the Symmetrix system until you *commit* the action.

Some classes of change operations may or may not impact current I/O. When possible, before you *commit* any action, stop I/O activity on the Symmetrix devices to be altered during a configuration change session. Consider the *Impact on I/O* statement for each class of operation in this chapter before working on any online devices.

If required, set the impacted devices for change to be Not Ready using the following command form:

```
symld -g DgName not_ready [LdevName [LdevName... ]]
```

Note: If I/O activity on an affected Symmetrix device occurs before or during a *commit* action, the *commit* action might fail. At the very least, heavy I/O activity on unaffected devices will impact how long it takes to commit changes.

- ◆ Ensure that all your critical data is preserved and safe when creating new or changing device configurations. Do not store data on any device that is not mirrored, or RAID-protected.

All final configurations and device attribute adjustments must meet certain open systems guidelines detailed in the E-Lab™ Interoperability Navigator, which can be viewed at <http://elabnavigator.EMC.com>.

Contact the EMC® Customer Support Center for assistance in reverting to your previous configuration should there be unforeseen problems with the new configuration.

- ◆ After committing a `symconfigure` mapping operation, you must update the device mapping information within the host system environment. Attempting host activity with a device after it has been removed or altered, but before you have updated the host's device information, can cause host errors.

To update your hosts, run the utilities specified for your platform as described in [Appendix C, "Updating the Host."](#) The `symcfg discover` command should then be issued to update the SYMAPI database with the new device mapping information. After the host environment is updated, I/O activity can resume with the Symmetrix device.

Impact on I/O

Some classes of change operations might impact current I/O. Consider the *Impact on I/O* statement for each class operation in this chapter before working on any online devices. When possible, before you `commit` any action, stop I/O activity on the Symmetrix devices to be altered for a configuration change session.

For example, set the device(s) for change to be Not Ready using the following command type:

```
symdev -sid SymmID not_ready SymDevName
```

Enable SYMAPI environment option

Before executing any of the class changes, the environment option `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` in the `options` file might need to be enabled or removed. This option or constant is normally enabled, which is required for many of the configuration changes available.

Note: A nonvisible device is a device that is not mapped to your host and/or does not have a physical host device name.

The editable `options` file is located in the SYMAPI configuration directory:

```
/var/symapi/config (for UNIX)
C:\Program Files\EMC\Symapi\config (for Windows)
```

To enable `SYMAPI_CTRL_OF_NONVISIBLE_DEVS`, set:

```
SYMAPI_CTRL_OF_NONVISIBLE_DEVS=ENABLE
```

Configuration control operations

The following are classes of configuration control operations available when applying the `symconfigure` command:

Note: For information on the Engenuity requirements for the configuration control operations, refer to [Table 2 on page 31](#).

Array-wide operations

- ◆ Set the metrics of Symmetrix configuration

Device operations

- ◆ Create/delete Symmetrix devices
- ◆ Automatically create metadevices
- ◆ Configure new Symmetrix devices by copying existing devices
- ◆ Convert a device configuration by adding or removing BCV or DRV attributes
- ◆ Convert an RDF device from static RDF to dynamic RDF
- ◆ Increase or decrease device protection by adding or removing a mirror(s)
- ◆ Reserve devices and release device reservations
- ◆ Set device attributes, emulation type, geometry, identifiers
- ◆ Enable the SCSI3 persistent reservation option for clustered devices
- ◆ Add/remove a spare device
- ◆ Add/remove RDF mirrors to devices
- ◆ Bind/unbind a thin device to a pool
- ◆ Allocate/free space for thin devices

Metadevice operations

- ◆ Form/dissolve metadevices
- ◆ Add/remove metamembers
- ◆ Convert a metadevice's configuration

Device mapping

- ◆ Map/unmap a Symmetrix device to a front-end port
- ◆ Map/unmap a range of Symmetrix devices that form a mainframe CU image to a front-end port
- ◆ Copy device mappings from one EA/EF port to another
- ◆ Add or remove PAV aliases from mainframe devices

Device pools

- ◆ Add/delete pools
- ◆ Add/remove pool members
- ◆ Enable/disable pool members
- ◆ Set thin pool attributes

Configuring RDF

- ◆ Add/remove RDF attributes
- ◆ Swap RDF source/target attributes for an RA group
- ◆ Enable/disable SRDF®/A on an RA group
- ◆ Set RA group attributes

RDF (RA) group parameters

- ◆ Set the session priority
- ◆ Set the minimum cycle time
- ◆ Enable/disable RDF/A transmit idle function
- ◆ Enable/disable RDF/A DSE autostart
- ◆ Set RDF/A DSE pool emulation type

- ◆ Set RDF/A DSE threshold

Front-end port attributes

- ◆ Set/reset SCSI or fibre port flags
- ◆ Set fibre FA loop ID
- ◆ Set host names of port connections
- ◆ Set Gig-E front-end port IP addresses, IPV6 addresses, iSNS name server address

Note: The overall processing time for the changes can vary from 5 minutes to over an hour, depending on the class of changes. RDF changes will also be applied to the remote Symmetrix array. Even more time is needed if the devices are to be synchronized.

Engenuity support for configuration controls

Table 2 lists supported Solutions Enabler configuration control operations for each Engenuity version.

Table 2 Configuration Manager operations (page 1 of 4)

Configuration Manager operation	5670	5671	5771	5772	5773	5874
Bind device:						
Bind thin device to pool					✓	✓
Create device:						
FBA	✓	✓	✓	✓	✓	✓
Celerra [®] FBA	✓	✓	✓	✓	✓	✓
RAID-S (Parity RAID 3+1)	✓	✓				
RAID-S (Parity RAID 7+1)	✓	✓				
RAID-5	✓	✓	✓	✓	✓	✓
RAID-6				✓	✓	✓
CKD	✓	✓	✓	✓	✓	✓
CKD meta devices	✓	✓	✓	✓	✓	✓
AS/400	✓	✓	✓	✓	✓	✓
VME 512 FBA	✓	✓	✓	✓	✓	✓
VDEV		✓	✓	✓	✓	✓
Save devices (for VDEVs)	✓	✓	✓	✓	✓	✓
Thin devices					✓	✓
DATA devices					✓	✓
Delete device	✓	✓	✓	✓	✓	✓
Convert device:						
Adding or removing BCD/DRV	✓	✓	✓	✓	✓	✓
Adding or removing RDF	✓	✓	✓	✓	✓	✓
Dynamic RDF ^a	✓	✓	✓	✓	✓	✓

Table 2 Configuration Manager operations (page 2 of 4)

Configuration Manager operation	5670	5671	5771	5772	5773	5874
Converting static RDF to dynamic RDF ^a	✓	✓	✓	✓	✓	✓
Adding another mirror	✓	✓	✓	✓	✓	
Removing a mirror	✓	✓	✓	✓	✓	
Convert RAID-S group to unprotected devices	✓	✓				
Change device emulation:						
FBA and Celerra FBA	✓	✓	✓	✓	✓	✓
Free device:						
Free thin device					✓	✓
Set device attributes:						
VCMdb	✓	✓	✓	✓	✓	✓
RDB_Cksum	✓	✓	✓	✓	✓	
dyn_rdf ^a	✓	✓	✓	✓	✓	✓
dyn_rdf1_only ^a	✓	✓	✓	✓	✓	✓
dyn_rdf2_only ^a	✓	✓	✓	✓	✓	✓
SCSI3_persist_reserv	✓	✓	✓	✓	✓	✓
Set device geometry					✓	✓
Set device identifiers					✓	✓
Allocate device:						
Allocate thin device					✓	✓
Map device:	✓	✓	✓	✓	✓	✓
AS/400	✓	✓	✓	✓	✓	✓
CKD		✓	✓	✓	✓	✓
Unmap device:	✓	✓	✓	✓	✓	✓
AS/400	✓	✓	✓	✓	✓	✓
CKD	✓	✓	✓	✓	✓	✓
Manage PAV Alias	✓	✓	✓	✓	✓	✓
Unbind device						
Unbind thin device from pool					✓	✓
Form meta	✓	✓	✓	✓	✓	✓
Add meta member:						
Concatenated	✓	✓	✓	✓	✓	✓
Striped ^a	✓	✓	✓	✓	✓	✓

Table 2 Configuration Manager operations (page 3 of 4)

Configuration Manager operation	5670	5671	5771	5772	5773	5874
Remove meta member:						
Concatenated	✓	✓	✓	✓	✓	✓
Dissolve Meta	✓	✓	✓	✓	✓	✓
Swap RA group	✓	✓	✓	✓	✓	✓
Swap hyper (available only to Optimizer)	✓	✓	✓	✓	✓	
Swap RDF device:	✓	✓	✓	✓	✓	✓
Dynamic RDF devices ^a	✓	✓	✓	✓	✓	✓
Set front-end port attributes	✓	✓	✓	✓	✓	✓
Set Symmetrix-wide parameters:						
max_hypers_per_disk	✓	✓	✓	✓	✓	
fba_multi_access_cache	✓	✓	✓	✓	✓	
dynamic_rdf ^a	✓	✓	✓	✓	✓	
raid_5_support	✓	✓	✓			
raid_s_support	✓	✓				
raid_s_members	✓	✓				
raid_5_members	✓	✓	✓			
VCMDB_restricted_access	✓	✓				
concurrent_rdf ^a	✓	✓	✓	✓	✓	
dynamic_concurrent_rdf ^a	✓	✓	✓	✓	✓	
pav_alias_limit	✓	✓	✓	✓	✓	✓
PAV_mode	✓	✓	✓	✓	✓	✓
rdfa_cache_percent		✓	✓	✓	✓	✓
rdfa_host_throttle_time		✓	✓	✓	✓	✓
hot_swap_policy			✓	✓	✓	✓
Add dynamic spare	✓	✓	✓			
Remove dynamic spare	✓	✓	✓			
SRDF operations:						
Add RDF mirror					✓	✓
Remove RDF mirror					✓	✓
Enable/Disable RA Group for RDFA Operation	✓					
Device pools:						

Table 2 Configuration Manager operations (page 4 of 4)

Configuration Manager operation	5670	5671	5771	5772	5773	5874
Create/Delete pools		✓	✓	✓	✓	✓
Add/Remove/Set Snap pool members		✓	✓	✓	✓	✓
Set pool attributes					✓	✓

a. Symmetrix 8000-series or higher.

Setting Symmetrix metrics

You can set some attributes to gain access to new Symmetrix array features using the following command:

```
set Symmetrix MetricName=MetricValue
    [, MetricName=MetricValue];
```

Where:

MetricName — The Symmetrix metric to be set. Possible name values are:

auto_meta — Enables the auto meta feature. Can be enabled only if the other auto meta parameters **min_auto_meta_size**, **auto_meta_config** and **auto_meta_member_size** are set to valid values. Possible values are **ENABLE** or **DISABLE**.

auto_meta_config — Specifies the default meta config when the **auto_meta** feature is enabled. Possible values are **CONCATENATED**, **STRIPED**, or **NONE**.

auto_meta_member_size — Specifies the default meta member size in cylinders when the **auto_meta** feature is enabled. Possible values are listed in [Table 7 on page 51](#).

concurrent_rdf — Enables access to the concurrent RDF feature - multiple R2 devices for a single static R1. Possible values are **ENABLE** or **DISABLE**.

dynamic_rdf — Enables the creation of a set of devices that are RDF-capable, (can be dynamically assigned as RDF1 or RDF2 devices). Possible values are **ENABLE** or **DISABLE**.

dynamic_concurrent_rdf — Enables access to the concurrent RDF feature (multiple R2 devices for a single R1) for dynamic RDF devices. Possible values are **ENABLE** or **DISABLE**.

Note: Enginuity 5874 does not support **concurrent_RDF**, **dynamic_RDF**, and **dynamic_concurrent_RDF**.

fba_multi_access_cache — Determines whether a read request can share cache slots in some conditions. Possible values are **ENABLE** or **DISABLE**.

hot_swap_policy — Indicates whether data from a failed disk should be relocated temporarily, or permanently moved to a hot spare device. Possible values are **TEMPORARY** or **PERMANENT**.

max_hypers_per_disk — Specifies the maximum number of hypers that can be created on a physical disk. Possible values are 1 to 255, where the upper limit varies with the Enginuity level.

min_auto_meta_size — Specifies the size threshold that triggers auto meta creation. When users try to create a device greater than **min_auto_meta_size** and **auto_meta** is enabled then a meta will be created. Possible values are listed in [Table 7 on page 51](#).

pav_alias_limit — If PAV is enabled, specifies the maximum number of aliases that can be assigned to a device. Possible values are:

- Enginuity 5670: 1-7
- Enginuity 5671: 1-15
- Enginuity 5771/5772: 1-255
- Enginuity 5773+: 1-127

`PAV_mode` — Enables the use of PAV (Parallel Access Volumes). Possible values are:

`STANDARD` — Standard PAV volumes with static aliasing can be configured.

`DYNAMIC_STANDARD` — Standard PAV volumes with dynamic aliasing can be configured.

`rdfa_cache_percent` — Sets the percentage of write pending cache that can be used by SRDF/A. This is a value from 0 to 100 percent.

`rdfa_host_throttle_time` — Sets the number of seconds to throttle host writes to SRDF/A devices when cache is full, before dropping RDFA sessions. Throttling will delay a write from the host until a cache slot becomes free. Values are from 0 to 65535.

`raid_5_support` — Enables the ability to create RAID 5 devices in the Symmetrix. Possible values are `ENABLE` or `DISABLE`. This feature is obsolete in Enginuity version 5772 and later.

`raid_s_members` — If RAID-S support is enabled, this identifies the number of members in the Parity RAID set. Possible values are 3 or 7. This also controls the RAID 5 member count. This feature is obsolete in Enginuity version 5771 and later.

`raid_s_support` — Enables the ability to create RAID-S or parity RAID devices in the Symmetrix array. Possible values are `ENABLE` or `DISABLE`. This feature is obsolete in Enginuity version 5772 and later.

`VCMDB_restricted_access` — When enabled, restricts host access to the VCM database as indicated by the user. Possible values are `ENABLE` or `DISABLE`. This metric is obsolete in Enginuity version 5771 and later.

Viewing the Symmetrix metrics

You can view the current settings for these metrics with the following command:

```
symcfg -sid SymmID -v list
```

Subsystem IDs for mainframe systems

Subsystem identifiers (SSID) are used by z/OS (MVS) mainframe systems to locate physical disk controllers. In a Symmetrix array that is configured to support only open systems hosts, the SSID value is not required. On Symmetrix arrays that service both mainframe (CKD devices) and open system hosts (FBA devices), the SSID parameter is required when creating new devices. The SSID parameter is also required when removing a mirror from a device, which results in the mirror becoming a new independent device.

Depending on the type of CKD emulation that is configured, the maximum number of SSID counts may change. It is recommended that you contact EMC Customer Support and ask which SSID was last used and what would be the next recommended ID to use along with how many devices can be placed behind it before you are required to increment the value to the next major range.

To obtain a list of SSIDs that are currently in use, use the following form:

```
symcfg list -ssid -sid SymmID
```

Planning system protection configurations

The SYMCLI maintains configuration and status information for every device on each of the Symmetrix arrays that are accessible from the host. A Symmetrix device can map to a part of a physical disk or to an entire disk. The part of a physical disk to which a Symmetrix device maps is called a hypervolume.

Device mirrors

There are two Symmetrix device configurations that involve various mirror set protection possibilities. [Table 3](#) shows the mirror positions for the Enginuity 5874 virtualized mirror positions. [Table 4](#) provides the mirror protection scheme for arrays running Enginuity 5773 and earlier.

Table 3 Virtual mirror positions (page 1 of 2)

Device/mirror configuration	M1	M2	M3	M4
2-Way-BCV-Mir	RAID 1			
2-Way-Mir	RAID 1			
BCV	Data			
BCV+R-5 ^a	RAID 5			
BCV+R-6 ^a	RAID 6			
2-Way-DRV-Mir	RAID 1			
Parity RAID 3+1 (in multiples of 3)	RAID 5			
Parity RAID 7+1 (in multiples of 7)	RAID 5			
RAID 6	RAID 6			
RDF1	Data	Remote Data		
RDF1+R-5 ^a	RAID 5	Remote Data		
RDF1+R-6	RAID 6	Remote Data		
RDF1-BCV	Data	Remote Data		
RDF1-BCV+R-5 ^a	RAID 5	Remote Data		
RDF1-BCV+R-6	RAID 6	Remote Data		
RDF1-BCV-Mir	RAID 1	Remote Data		
RDF1-Mir	RAID 1	Remote Data		
RDF2	Remote Data	DATA		
RDF2+R-5 ^a	Remote Data	RAID 5		
RDF2+R-6	Remote Data	RAID 6		
RDF2-BCV	Remote Data	DATA		
RDF2-BCV+R-5 ^a	Remote Data	RAID 5		
RDF2-BCV+R6	Remote Data	RAID 6		
RDF2-BCV-Mir	Remote Data	RAID 1		
RDF2-Mir	Remote Data	RAID 1		

Table 3 Virtual mirror positions (page 2 of 2)

Device/mirror configuration	M1	M2	M3	M4
RDF21	Remote Data	Remote Data	DATA	
RDF21+R-5	Remote Data	Remote Data	RAID 5	
RDF21+R-6	Remote Data	Remote Data	RAID 6	
RDF21-Mir	Remote Data	Remote Data	RAID 1	
TDEV	Thin			
RDF1+TDEV	Thin	Remote Data		
RDF2+TDEV	Remote Data	Thin		
Unprotected	DATA			
VDEV	Virtual			
DLDEV				
RDF1 +DLDEV		Remote Data		
RDF2+DLDEV	Remote Data			
RDF21+DLDEV	Remote Data	Remote Data		

[Table 4](#) lists the mirror set protection schemes for devices on Symmetrix arrays running Enginuity 5773 and earlier.

Table 4 Mirror set protection schemes (page 1 of 2)

Device/mirror configuration	M1	M2	M3	M4
2-Way-BCV-Mir	Data	Data		
2-Way-Mir	Data	Data		
3-Way-Mir	Data	Data	Data	
4-Way-Mir	Data	Data	Data	Data
BCV	Data			
BCV+R-5 ^a	Data and Parity	Data and Parity		
BCV+R-6 ^a	Data and Parity			
DRV	Data			
2-Way-DRV-Mir	Data	Data		
Parity RAID 3+1 (in multiples of 3)	RAID Data	RAID Parity (shared)		
Parity RAID 7+1 (in multiples of 7)	RAID Data	RAID Parity (shared)		
RAID 5 ^a	Data and Parity	Data and Parity		
RAID 6	Data and Parity			
RDF1	Data	Remote Data		
RDF1+R-5 ^a	Data and Parity	Remote Data	Data and Parity	
RDF1+R-6	Data and Parity	Remote Data		

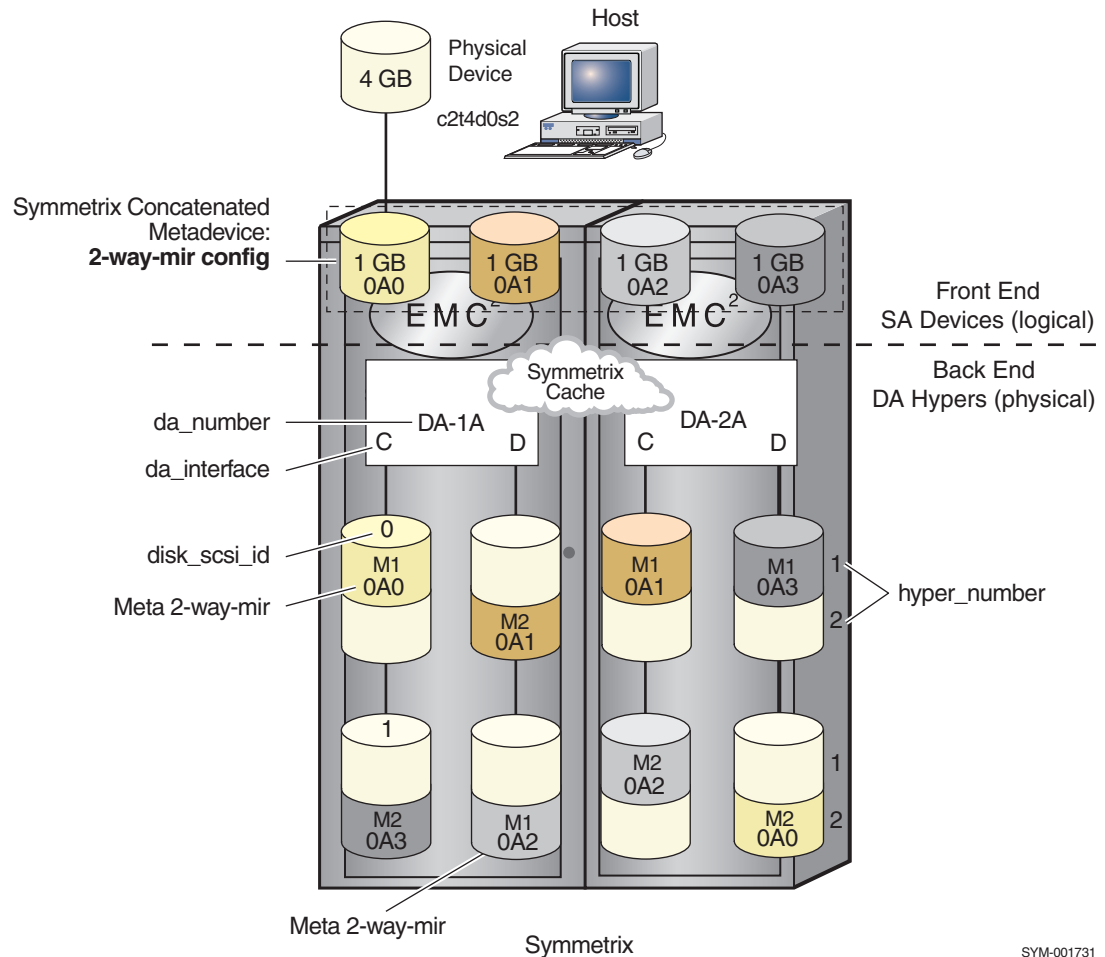
Table 4 Mirror set protection schemes (page 2 of 2)

Device/mirror configuration	M1	M2	M3	M4
RDF1-BCV	Data	Remote Data		
RDF1-BCV+R-5 ^a	Data and Parity	Data and Parity		
RDF1-BCV+R-6	Data and Parity	Remote Data		
RDF1-BCV-Mir	Data	Remote Data	Data	
RDF1-Mir	Data	Remote Data	Data	
RDF1-R-S	RAID Data	Remote Data	RAID Parity	
RDF2	Remote Data	Data		
RDF2+R-5 ^a	Remote Data	Data and Parity	Data and Parity	
RDF2+R-6	Remote Data	Data and Parity		
RDF2-BCV	Remote Data	Data		
RDF2-BCV+R-5 ^a	Remote Data	Data and Parity	Data and Parity	
RDF2-BCV-Mir	Remote Data	Data	Data	
RDF2-Mir	Remote Data	Data	Data	
RDF2-R-S	Remote Data	RAID Data	RAID Parity	
RDF21	Remote Data	Remote Data	Data	
RDF21+R-5	Remote Data	Remote Data	Data and Parity	Data and Parity
RDF21+R-6	Remote Data	Remote Data	Data and Parity	
RDF21-Mir	Remote Data	Remote Data	Data	Data
TDEV				
Unprotected	Data			
VDEV				
DLDEV				

a. This table item is maintained for historical reasons. RAID 5 devices are built on either four or eight hyper members, where data and parity are striped across all members.

2-Way mirror protection

Figure 1 illustrates the devices for a two-way mirror concatenated metadvice configuration (Meta 2-Way-Mir). The metadvice is comprised of four mirrored devices of 1 GB each.



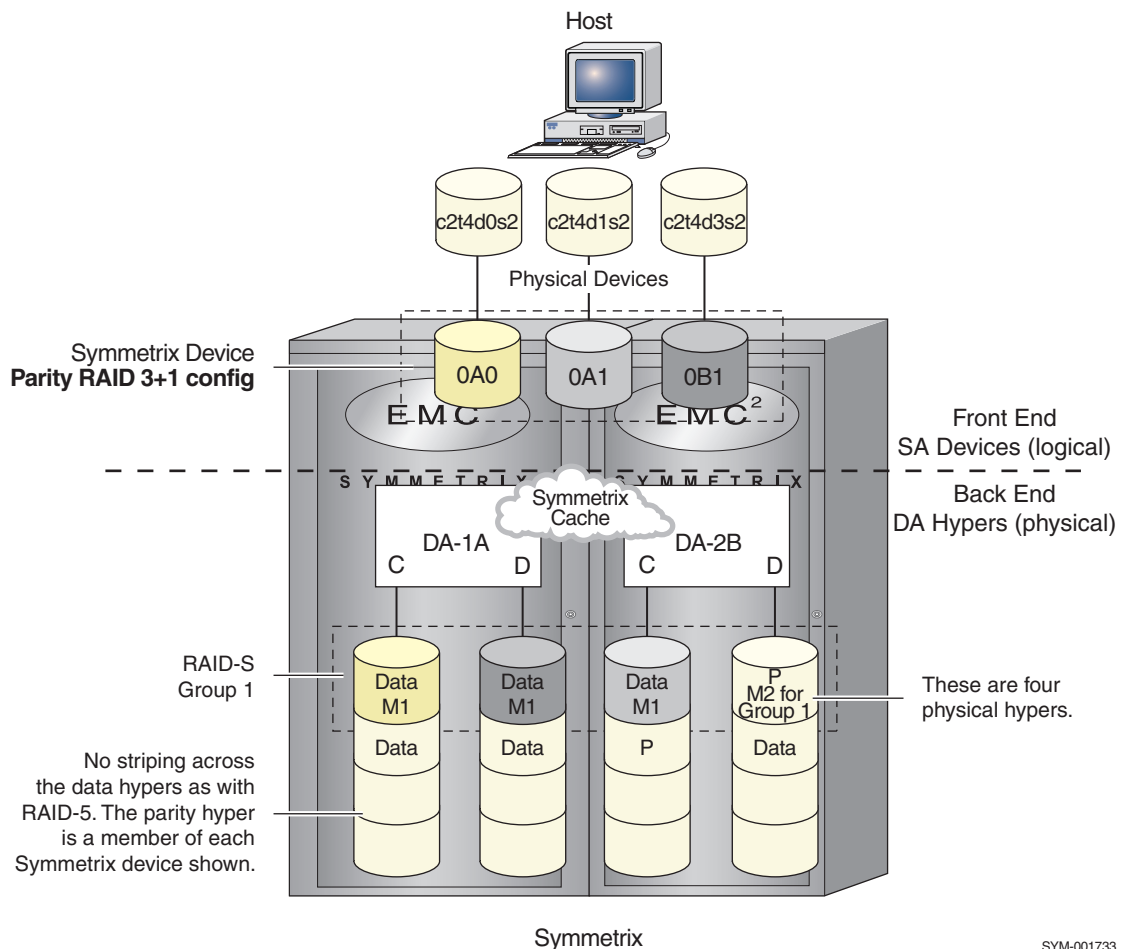
SYM-001731

Figure 1 Symmetrix concatenated metadvice two-way mirror scheme

Parity RAID protection

Parity RAID devices must be created in multiples of three or seven. For example, a Parity RAID 3+1 protection group requires three devices for data and one for shared parity. Any requests to create parity devices outside of the three or seven multiple factor will be denied. The Symmetrix array maps each of the device's data mirrors to the same mirror positions on a separate hyper and creates one parity device to be shared by the DATA devices. The number of Parity RAID (three or seven members) devices must be defined in the Symmetrix attributes.

Figure 2 illustrates the devices of a parity RAID mirror configuration (3+1).



SYM-001733

Figure 2 Symmetrix parity RAID mirror scheme

You can use the `symdev list` command, to get a list of all the available devices. You can then choose a device from this list and get the configuration scheme it is serving by using the `symdev show` command. This information then leads you to back-end information for the device's disk director(s) and corresponding hyper volumes, and their mapping to disk drives. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Array Management Product Guide*.

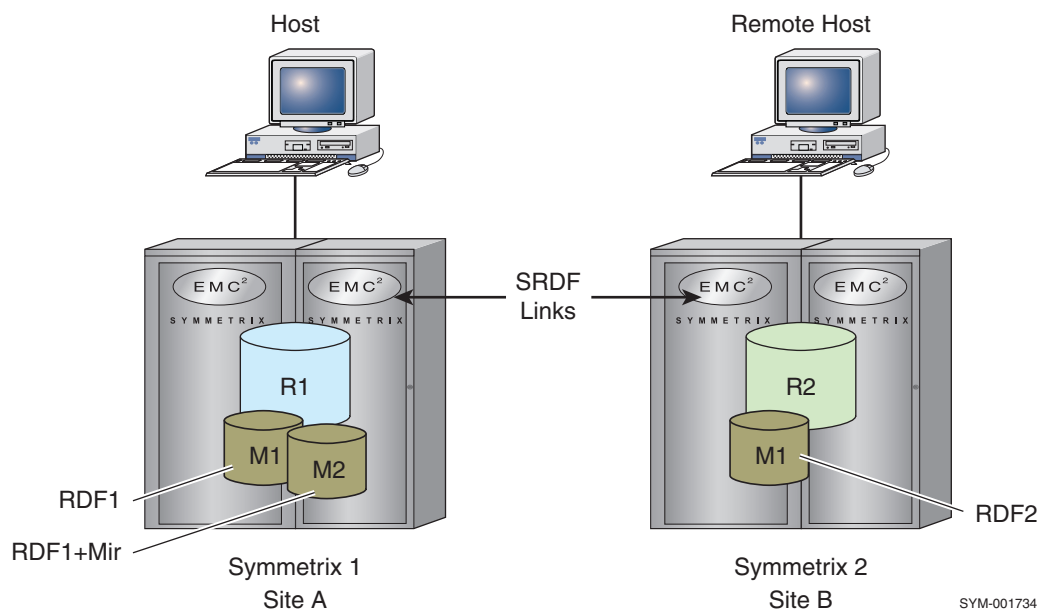
SRDF remote protection

The Symmetrix Remote Data Facility (SRDF) is a business continuance solution that maintains a mirror image of data at the device level in Symmetrix arrays located in physically separate sites.

SRDF devices

SRDF configurations provide for either a unidirectional or a bidirectional data transfer over SRDF links between sites. Devices are designated as either R1 (source) or a R2 (target) mirrors to synchronize and coordinate SRDF transfer activity. Devices can also be concurrent mirrors, such as an R11 device that has two separate R2 pair relationships with R2 devices in different Symmetrix arrays. In addition, with Solutions Enabler version 6.5, a device can have a dual role of both source mirror (R1) to a target mirror (R2) and target mirror (R2) to a source device (R1). This device is called an R21. R21 devices are used in Cascaded SRDF configurations. Refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide* for more details about Cascaded RDF devices.

As shown in [Figure 3](#), a source R1 device is designated an RDF1 type, while a target R2 device is designated an RDF2 type. An R1 or an R2 device can also have associated mirror(s) (M1, M2) for increased protection. These are designated RDF1+Mir or RDF2+Mir as configuration types when using `symconfigure`.



SYM-001734

Figure 3 Simple SRDF device configuration

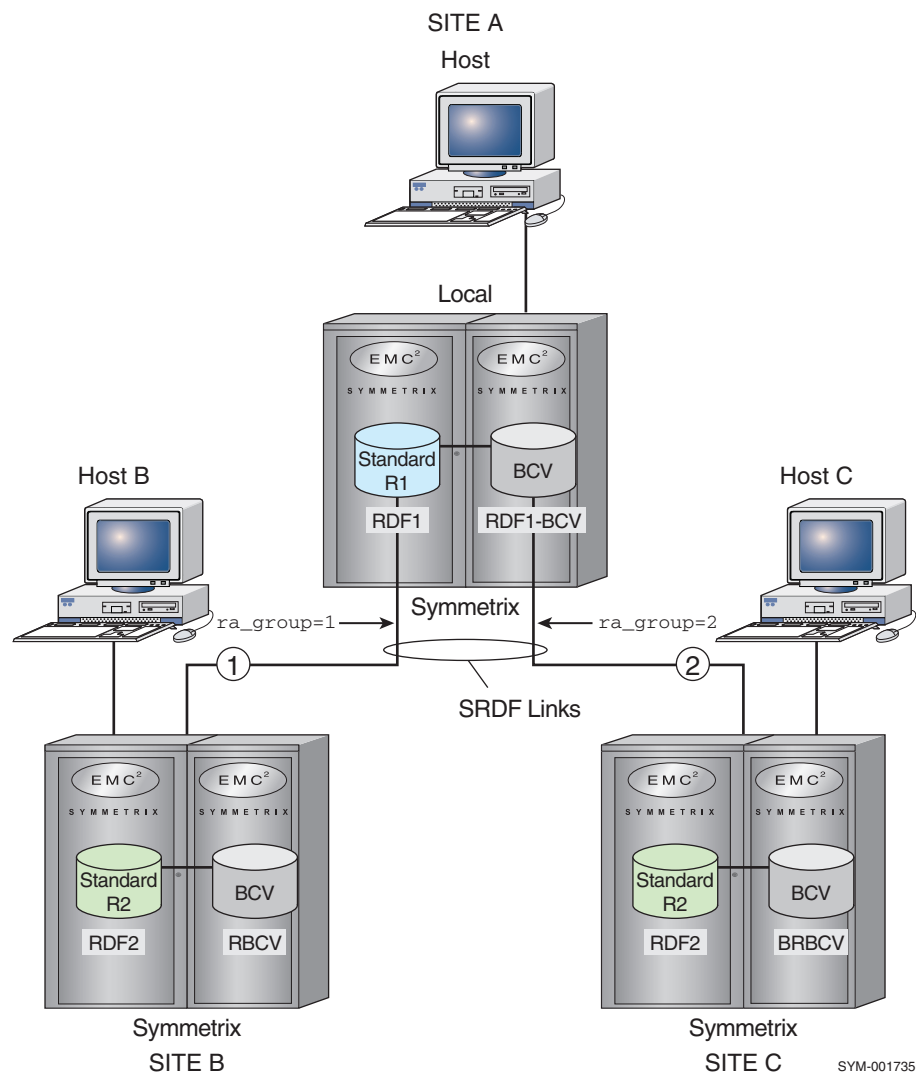
SRDF and BCV devices

You can configure both SRDF and TimeFinder® (BCV) devices that might be in various compounded remote configurations.

As shown in [Figure 4 on page 43](#), you can have multiple sites (for example, remote Sites B and C) on SRDF links to remotely mirror a local Symmetrix array (at Site A). Remote Site B, functioning as a remote mirror to the standard devices at Site A, is most typical. You can also have a third site (remote Site C) to remotely mirror just the BCV devices in the Symmetrix at Site A.

There are two types of SRDF-connected BCV devices:

- ◆ The SRDF-connected BCVs can be paired with R2 mirrors of the local RDF standard devices. In this case, the BCV is designated RBCV, which identifies it as Remote BCV device.
- ◆ The SRDF-connected BCVs can be paired with the R2 mirrors of the local RDF BCV devices, designated RDF1-BCV. The SRDF-connected BCV devices can be remotely associated with one device group of any type. In this case, a BCV device designated BRBCV, which identifies this device as a BCV on a remote host originating from a BCV.



SYM-001735

Figure 4 Various SRDF-connected device configurations with BCVs

Device creation strategy

Creation of certain device (configuration) types may require two sessions. [Table 5](#) describes the sessions required to create a device type and/or convert the various types of devices from one configuration type to another.

Table 5 Steps to create a device (page 1 of 2)

Desired device type	Session 1 create	Session 2
2-Way-BCV-Mir	2-Way-BCV-Mir	
2-Way-Mir	2-Way-Mir	
3-Way-Mir ^a	3-Way-Mir	
4-Way-Mir ^b	4-Way-Mir	
BCV	BCV	
DRV ^c	DRV	
2-Way-DRV-Mir ^d	2-Way-DRV-Mir	
Parity RAID	RAID-S ^d	
RAID 5	RAID-5	
RAID 6	RAID-6	
RAID 5 BCV	BCV+R-5	
RAID 6 BCV	BCV+R-6	
RDF1	RDF1	
RDF1+R-5	RDF1+R-5	
RDF1+R-6	RDF1+R-6	
RDF1-BCV	BCV	Convert to RDF1-BCV
RDF1-BCV+R-5	RDF1+R-5	Convert to RDF1-BCV+R-5
RDF1-BCV+R-6	RDF1+R-6	RDF1-BCV+R-6
RDF1-BCV-Mir	RDF1+Mir	RDF1-BCV-Mir
RDF1-Mir	RDF1-Mir	
RDF2	RDF2	
RDF2+R-5	RDF2+R-5	
RDF2+R-6	RDF2+R-6	
RDF2-BCV	BCV	RDF2-BCV
RDF2-BCV+R-5	RDF2+R-5	RDF2-BCV+R-5
RDF2-BCV+R-6	RDF2+R-6	RDF2-BCV+R-6
RDF2-BCV-Mir	RDF2+Mir	RDF2-BCV-Mir
RDF2-Mir	RDF2-Mir	

Table 5 Steps to create a device (page 2 of 2)

Desired device type	Session 1 create	Session 2
RDF21 ^e +R5	RDF1+R5	Add R2 mirror
RDF21 ^{e,f} +R5	RDF2+R5	Add R1 mirror
RDF21 ^{e,f} +R6	RDF1+R6	Add R2 mirror
RDF21 ^{e,f} +R6	RDF2+R6	Add R1 mirror
RDF21 ^{e,f} -Mir	RDF2-Mir	Add R1 mirror
RDF22 ^{f,g} +R5	RDF2+R5	Add R2 mirror
RDF22 ^{f,g} +R6	RDF2+R6	Add R1 mirror
RDF1+TDEV		
RDF2+TDEV		
DLDEV ^g		
RDF1+DLDEV ^g		
RDF2+DLDEV ^g		
RDF21+DLDEV ^g	RDF2+DLDEV	AddR1 mirror
RDF21+DLDEV ^g	RDF1+DLDEV	AddR2 mirror
Thin device (TDEV)	TDEV	
Unprotected	Unprotected	
Virtual	VDEV	

- a. 3-Way-Mir devices are not supported on Enginuity 5773 and higher.
- b. 4-Way-Mir devices cannot have any BCV devices. 4-Way-Mir devices are not supported on Enginuity 5874 and higher.
- c. Not supported on Enginuity 5874.
- d. Not allowed for Enginuity 5771 and higher.
- e. Only for Enginuity 5773 and higher.
- f. To create an R21 or R22 device, use add rdf mirror with the symconfigure command.
- g. Only for Enginuity 5874 and higher.

For information on creating dynamic-RDF capable devices, refer to [“Configuring dynamic RDF-capable devices” on page 103](#).

When you create an RDF device where a corresponding device must be created on the remote SRDF-linked Symmetrix, a configuration change session is initiated and managed on the remote Symmetrix by the local `symconfigure` utility.

Note: SAVE devices and DATA devices are created following the same procedure as for standard protected devices, with the additional specification of `attribute=savedev` and `attribute=datadev`.

Managing devices

In the command file, you can create (add) one or more Symmetrix devices in a specified Symmetrix array.

To create one or more devices, use the following command:

```
create dev count=n, size=Cylinders,
size=n [MB |GB | CYL]
emulation=EmulationType,
config=DeviceConfig,
[, data_member_count=nn]
[, remote_config=DeviceConfig,
remote_data_member_count=nn,
ra_group=n, [remote_mvs_ssid=nnn],
[dynamic_capability=[dyn_rdf | dyn_rdf1_only | dyn_rdf2_only], ]
[, mvs_ssid=nnn]
[, attribute=ckd_meta | savedev | datadev
[in pool PoolName] [member_state=ENABLE | DISABLE], ]
[, disk_group=nnn, remote_disk_group=nnn]
[, binding to pool=PoolName
[preallocate size=n [MB |GB | CYL]
[remote_pool=PoolName]]
[, [mapping to dir DirNum:PortNum>
[starting] target = <scsi_target>,
lun=scsi_lun, vbus=fibre_vbus
[starting] base_address <cuu_address>]...]
[[device_attr =
[SCSI3_persist_reserv | ACLX]]...];
```

Where:

count — Indicates the number of devices to create.

size — Specifies the size of the device needed in number of cylinders (default), megabytes, or gigabytes.

Table 6 **Device sizes by Enginuity version**

Enginuity version	MBs	CYLs	GBs
Enginuity 5874	245760	262668	240
Enginuity 5773 and earlier	61425	65520	59

To calculate the number of cylinders, use either of the following:

blocks ÷ 960

or

(device size in megabytes) x 10242 ÷ (960 x 512 bytes)

or

(size in megabytes) x 2.1333

Note: Devices to be used as BCV, RDF, or metamembers will need to precisely match corresponding device sizes. Use `symdev/sympd show` or `symdev/sympd list -cyl` commands to see relevant device sizes.

For additional information about sizing devices in cylinders, refer to Chapter 1 of the *EMC Solutions Enabler Symmetrix CLI Array Management Product Guide*.

emulation — Specifies the device emulation type, which currently must be Fixed Block Architecture (FBA), CELERRA FBA, VME512 FBA, or CKD-3390 for z/OS environments.

config — Specifies the desired device configuration type. Possible values are shown in the second column of [Table 5 on page 44](#).

data_member_count — Indicates the number of data members when creating a RAID 5 or RAID 6 devices on a Symmetrix array with Enginuity version 5772 and higher. The user should set the value to 3 or 7 for RAID 5 (3+1) and RAID 5 (7+1) or, 6 or 14 for RAID 6 (6+2) and RAID 6 (14+2).

remote_config — Specifies the desired remote SRDF configuration (if any) from [Table 5 on page 44](#).

remote_data_member_count — Specifies the number of remote data members when creating RDF RAID 5 or RAID 6 devices on a Symmetrix array with Enginuity version 5772 and higher. The user should set the value to 3 or 7 for RAID 5 (3+1) and RAID 5 (7+1) or, 6 or 14 for RAID 6 (6+2) and RAID 6 (14+2).

ra_group — Specifies the RA group number in the SRDF environment.

remote_mvs_ssid — When creating an RDF device in a remote Symmetrix array that also contains CKD devices, a z/OS (MVS) subsystem ID (`remote_mvs_ssid`) value may be provided so the new FBA devices are not seen as part of an existing subsystem ID group. Only one `mvs_ssid` and `remote_mvs_ssid` can be used in a session. They will be applied to all devices created within that session.

dynamic_capability — Specifies the type of dynamic RDF device to create. Possible values are:

`dyn_rdf` — Creates a dynamic RDF device.
`dyn_rdf1_only` — Creates a dynamic R1 RDF device.
`dyn_rdf2_only` — Creates dynamic R2 RDF device.

Note: With Enginuity 5874 and higher, a single device can be set to be dynamic, without automatically creating a device pair.

If creating a dynamic pair, the specified dynamic capability will be applied to the local device, and the corresponding remote device will be assigned a complementary dynamic capability, according to the following:

Local device	Remote device
<code>dyn_rdf</code>	<code>dyn_rdf</code>
<code>dyn_rdf1_only</code>	<code>dyn_rdf2_only</code>
<code>dyn_rdf2_only</code>	<code>dyn_rdf1_only</code>

mvs_ssid — When creating a device in a Symmetrix array that also contains CKD devices, a z/OS (MVS) subsystem ID (`mvs_ssid`) value can be provided so the new FBA devices are not seen as part of an existing subsystem ID group. For more information about SSIDs, refer to [“Subsystem IDs for mainframe systems” on page 36](#).

To obtain a list of SSIDs currently in use, use the following:

```
symcfg -ssid list
```

attribute — Specifies the desired device attribute. Possible values are:

ckd_meta — When creating a device with emulation type of CKD-3390, this indicates that the device should be a striped metadvice. CKD metadvice must be created in sets of four devices.

savedev — When creating a device, this indicates that the device should be a SAVE device. The device will become part of a pool of devices that are used with TimeFinder/Snap for virtual device Snap operations.

datadev — When creating device, this indicates that the device should be a DATA device. The device will become part of a pool of devices that are used with thin devices for virtual operations.

PoolName — Specifies the name of the SAVE or thin pool. It can be from 1 to 12 alphanumeric characters long and include hyphens (-), and underscore (_) characters. The name `DEFAULT_POOL` is reserved to represent the container of all *unpooled* SAVE devices. Thin pools do not use the `DEFAULT_POOL`.

Note: The pool name `DF_DDEV_POOL` is reserved for EMC use only.

member_state — Indicates whether the device(s) being added should be enabled or disabled in the pool.

disk_group — Specifies a disk group when creating a device. A disk group is a set of physical disks set aside to be used to create devices of the protection level. Disk group usage may improve the performance of some configurations.

To see how disk groups are organized in your configuration, use the following commands:

```
symdisk -sid SymmID list -by_diskgroup
and
symdev -sid SymmID list -diskgroup(nn)
```

remote_disk_group — Specifies a remote disk group when creating a device.

binding to pool — Specifies the thin *PoolName*. This option is only available when creating thin devices. When creating RDF thin devices, both devices must be bound to a pool at the same time; just the local RDF or just the remote RDF device cannot be bound individually. Devices do not need to be bound to a pool upon creation, however, they cannot be used until they are bound to a pool.

preallocate size — Indicates the amount of space (pre) allocated to the thin device(s) when it is bound to a pool. If MB or GB is not specified, the size defaults to cylinders.

remote_pool — Specifies the name of the remote thin pool. This option is only for RDF thin devices.

scsi_target — is a hex value for the SCSI target ID.

scsi_lun — Indicates a hex value for the SCSI logical unit number.

fibre_vbus — If mapping to an FA port using volume set addressing, the virtual bus address; else omit.

cuu_address — Indicates a base or alias address for a device being mapped to an EA or EF port. These are mainframe ports which expect devices to be mapped in groups to form CU images. The first digit in the address is the CU image number, which can range from 0 to 0xF. The remaining two digits can range from 00 to 0xFF.

device_attr — Specifies the attributes to be set on the new device.

Example: creating devices

To add four new Symmetrix devices to Symmetrix array 000000012345 as two-way mirrored devices with a size of 1100 cylinders (516 MB), using FBA emulation, and to create the hypervisors to support the new devices in disk group 3, create a command file (called `create.cmd`) containing the following option:

```
create dev count=4, size=1100,emulation=FBA, config=2-Way-Mir
disk_group=3;
```

Then activate the option using the command:

```
symconfigure -sid 345 -file create.cmd -v -noprompt commit
```

Impact on I/O When creating devices, there is no impact on any I/O activity.

Enable SYMAPI environment option

Environment option SYMAPI_CTRL_OF_NONVISIBLE_DEVS in the `options` file must be enabled. If this option is disabled, the created devices, which are nonvisible, will be blocked from having changes performed on them. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Restrictions on creating devices

There are restrictions on the use of unprotected devices created by `symconfigure`, or converted to unprotected devices. Such devices cannot be mapped to hosts for normal data storage use. If an unprotected device is intended for use as a gatekeeper, it should be created with 48 cylinders or less in order to be mapped to a host.

You can create devices with a variety of emulation types: Fixed Block Architecture (FBA), Celerra FBA or VME512 FBA, and CKD for z/OS environments.

You cannot create or convert to SRDF device types when:

- ◆ Domino mode is enabled on any current SRDF pairs.
- ◆ There are any invalid tracks on any of the current SRDF devices.
- ◆ Concurrent RDF is enabled on a device.

When creating devices in an z/OS environment, all new devices must have the same MVS SSID and any remote devices must also have the same remote MVS SSID within a single session. To create devices with different SSID values, you will need to run more than one session.

Copying a device

You can configure available disk space into a set of new devices by copying the attributes of an existing (model) device. To do this you can either specify the quantity of disk space to configure into the new devices, or the number of devices to be created. The quantity of disk space represents the new space that will be available for the host to use, and not the space allocated in the Symmetrix array to manage the request according to the device protection requirements. In addition, you can also override some of the model device's attributes in the new devices. For example, you can copy the attributes of a standard device, but configure the new devices as BCVs.

To configure available disk space, use the following form:

```
configure [n.nn [MB | GB] | nn devices]
copying dev SymDevName
[mapping to dir DirectorNum:PortNum
 [masking hba [awwn=awwn | wwn=wwn |
               iscsi=iscsi | aiscsi=aiscsi]
               host_lun=lun |dynamic_lun]] ]
[overriding
[size=nnn [MB | GB | CYL]]
```

```
[emulation=EmulationType]
[config=DeviceConfig]
data_member_count=n
[mvs_ssid=nn]
[disk_group=n];
```

Where:

***n.nn* [MB | GB]** — Specifies the quantity of disk space to configure.

nn devices — Specifies the number of devices to create.

SymDevName — The Symmetrix device name of the model device.

DirectorNum:PortNum — The mapping attributes of the model device are not copied. If the new devices are to be mapped, you must specify the director/port addresses.

masking hba — The masking attributes of the model device are not copied. If the new devices are to be masked, you must specify the host HBA to which the devices should be masked. This option is not available on arrays running Enginuity 5874.

host_lun — Specifies the LUN addresses to be used for each device that is to be added for the host HBA.

dynamic_lun — Specifies to use the dynamic LUN addressing features but does not require a LUN address for each device. The LUN addresses are assigned based on what may already be in use for that host HBA.

overriding — Indicates that you will be overriding some of the characteristics of the model device.

size — Specifies the size of the new devices.

emulation — Indicates the emulation type of the new devices.

config — Specifies the configuration type of the new devices. Possible values are shown in the second column of [Table 5 on page 44](#).

data_member_count — Specifies the number of data members when creating a RAID 5 or RAID 6 devices on a Symmetrix array with Enginuity version 5772 and higher. The user should set the value to 3 or 7 for RAID 5 (3+1) and RAID 5 (7+1) or, 6 or 14 for RAID 6 (6+2) and RAID 6 (14+2).

mvs_ssid — When creating devices in a Symmetrix array that also contains CKD devices, a z/OS (MVS) subsystem ID (*mvs_ssid*) value can be provided so the new FBA devices are not seen as part of an existing subsystem ID group. For more information about SSIDs, refer to [“Subsystem IDs for mainframe systems” on page 36](#).

disk_group — Specifies the disk group in which to place the new devices. If this option is not specified, the model device’s disk group is *not* used. Instead, SymmWin will place the hypers on any available disk.

Example To configure 6 GB of available free space into BCVs matching an existing standard device, and mapping them to port 14B:0, enter:

```
symconfigure commit -sid 3420 -file cfg_bcv.cmd
```

Where *cfg_bcv.cmd* contains:

```
configure 6 gb copying dev 07A,
mapping to dir 14B:0, overriding
config=bcv;
```

Managing metadevices

Metadevices allow individual devices to be concatenated to create larger devices. The devices assigned in a meta sequence do not need to be adjacent.

The *metahead* is the first device in the metadevice and is responsible for receiving incoming commands. When an incoming command for the metahead is processed, the Enginuity software determines which metadevice member should execute the command.

Several operating systems, such as Windows and some open systems environments, can have larger volumes than are provided by standard Symmetrix physical disk devices. A metadevice is a logical volume set created from individual Symmetrix hypers to define volumes larger than the current Symmetrix maximum hyper volume size of approximately 16 GB. Metadevices are functionally the same as logical volume sets implemented with the host volume manager software. Physically, a metadevice is two or more Symmetrix volumes presented to a host as a single addressable device. It consists of a metahead device, some number of member devices (optional), and a metatail device (see [Figure 5](#)).

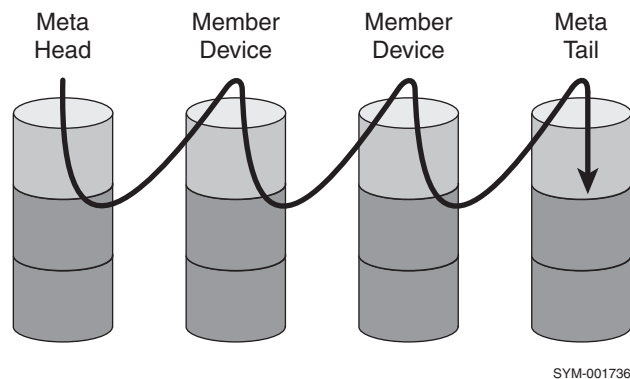


Figure 5 Concatenated metadevices

Metadevice size requirements

Metadevices may contain as many as 255 members. The maximum size metadevice that has been tested by EMC is 60 terabytes, which is the maximum size supported at this time. Metadevices can consist of nonsequential and nonadjacent volumes.

Table 7 Metadevice sizes by Enginuity version for auto_meta

Enginuity version	CYLs	GBs	Min_auto_meta_size	Auto_meta_member_size
Enginuity 5874	255 * 262668	61322.81	262669	262668
Enginuity 5773 and earlier	255 * 65520	15296.26	65521	65520

Meta performance

By allowing individual physical disk devices to be grouped together into a metadevice and the capability of metadevice addressing, Symmetrix enhances disk system functionality. To increase throughput and further improve performance, Symmetrix provides multiple I/O drive queues for metavolumes.

Accessing data in a metadevice

Addressing of data contained in a metadevice can be organized in two different ways:

- ◆ Concatenated devices

- ◆ Striped devices

Concatenated devices

Concatenated devices are volume sets that are organized with the first byte of data at the beginning of the first device (Figure 5). Addressing continues to the end of the first device before any data on the next device is referenced. When writing to a concatenated device, the first metadevice member receives all the data until it is full, and then data is directed to the next member and so on.

Striped devices

Metadevice addressing by striping divides each metamember device into a series of stripes, addressing a stripe from each device before advancing to the next stripe on the first device (as shown in Figure 6). When writing to a striped volume, equal size stripes of data from each participating drive are written alternately to each member of the set.

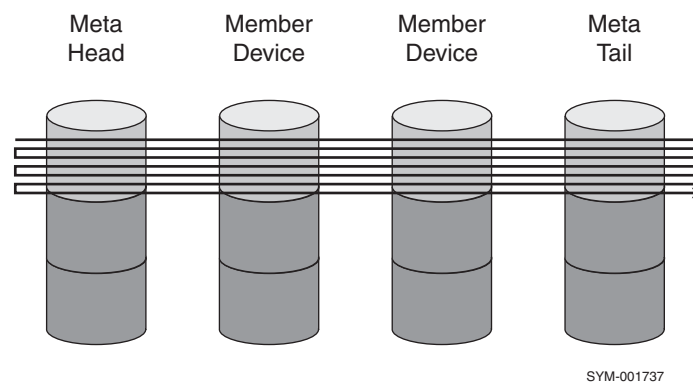


Figure 6 Striped data

Striping data across the multiple drives in definable cylinder stripes benefits *sequential writes* by avoiding stacking multiple writes to a single spindle and disk director. This scheme creates a large volume, but additionally balances the I/O activity between the disk devices and the Symmetrix disk directors.

Striped metadevices are created with a stripe size of 1 cylinder (1920 blocks) for Enginuity 5771 and later.

Note: If comparing the sum of the metamember's individual cylinder size to the metahead cylinder size, they may not match. A `symdev show` of the metahead device will only show the useable cylinder size.

Creating metadevices

Metadevices can be created in either of the following ways:

- ◆ Formed
- ◆ Automatically created (Solutions Enabler 6.5.1 and higher)

If the Symmetrix array is running Solutions Enabler 6.5.1 or higher, it is easier to use the automatic meta configuration feature.

Forming metadevices

When you initially create a metadevice, you must form the head and add the members in the same command file. If using the `count` option, no members need to

be specified. The configuration server will automatically select members from the pool of unmapped devices that match the metahead in size, emulation and configuration type.

To form a metadvice, use the following command:

```
form meta from dev SymDevName,
    config=MetaOption
    [, stripe_size=<MetaStripeSize>[cyl]]
    [, count=<member_count>];
```

where:

config — Specifies the meta configuration type. Possible values are concatenated or striped.

stripe_size — Specifies the size of the striped metadvice. This value can be expressed in blocks or cylinders. Possible sizes in 512 byte blocks are 1920, 3840, 7680, 15360, 30720, and 61440. The stripe size must be 1920 for Engenuity Version 5x69 and later, which is the default for all versions of Engenuity. If you do not specify a stripe size when creating a striped meta, two cylinders will be assigned. For information on metavolume stripe sizes, refer to [“Striped devices” on page 52](#).

cyl — Indicates that the `stripe_size` is expressed in cylinders. The size of a cylinder for FBA emulation is 960 512-byte blocks.

count — Indicates the total number of devices for the configuration server to add to the new metadvice, including the head. This option may be omitted if you want to specify the members using the `add dev` command.

Automatic metadvice creation

Beginning with Solutions Enabler V6.5.1 running Engenuity version 5773, metadvice can be created in a single configuration change session. Four Symmetrix array-wide settings have been added for this feature, as follows:

- ◆ **auto_meta** — This is a Symmetrix-wide setting (ENABLE/DISABLE) that can be used to enable automatic creation of metas. The default value is DISABLE.
- ◆ **auto_meta_member_size** — When **auto_meta** is enabled, this setting indicates the number of cylinders of each meta member: The default value is 0, therefore this value must be changed.
- ◆ **auto_meta_config** (striped or concatenated) — When **auto_meta** is enabled, this setting indicates the configuration can be set to striped or concatenated. The default value is NONE, therefore this value must be changed.
- ◆ **min_auto_meta_size** — This setting is a size threshold that triggers auto meta creation. When the user tries to create a device with size greater than or equal to **min_auto_meta_size**, a meta will be created. Refer to [Table 7 on page 51](#) for metadvice sizes.

Symmetrix-wide meta settings can be set using the `symconfigure` command. The `symconfigure` command is typically executed by specifying a command file containing a series of actions that are performed in a configuration change session. The command file supports the following new metrics in the `set symmetrix` command:

```
set symmetrix [auto_meta = <ENABLE|DISABLE>]
    [min_auto_meta_size = n [MB | GB | CYL]]
    [auto_meta_member_size n [MB | GB | CYL]]
    [auto_meta_config = [striped | concatenated | NONE]];
```

Restrictions

- ◆ The `min_auto_meta_size` cannot be set less than the `auto_meta_member_size`. Refer to [Table 7 on page 51](#) for metadvice sizes.
- ◆ The metric `auto_meta` can be enabled only if `min_auto_meta_size` and `auto_meta_member_size` and `auto_meta_config` are set to valid values.
- ◆ Users can override the `auto_meta_member_size` and the `auto_meta_config` in the create device command line only if `auto_meta` is `ENABLED` and the total device size is greater than `min_auto_meta_size`.
- ◆ Internal device types DATA, SAVE, and DRVs cannot be metadvice, so the parameters `min_auto_meta_size` and `meta_member_size` are not applicable when creating these devices.
- ◆ To reset the current default `meta_member_size` set the value to 0.

Note: The `meta_member_size` on the Symmetrix array can only be in cylinders, Solutions Enabler converts values entered with MB and GB to cylinders. The conversion formula will calculate a number "n" where n is the minimum number of cylinders required to fit the requested capacity.

Metadvice creation

The `create dev` command will create the new metadvice.

- ◆ If `auto_meta` is `ENABLED`, and the input size is greater than or equal to `min_auto_meta_size`, a metadvice will be created automatically. If the array-wide settings are not set, and a request is made to create a device greater than the maximum single device size, the request will be rejected.
- ◆ AS/400 devices: If `auto_meta` is enabled, the input size is an optional field for creating AS400 metadvice. Solutions Enabler will create the correct AS400 meta size using the array-wide meta settings.
- ◆ CKD devices: The procedure for creating CKD metadvice has not changed.

The following shows the updated `create dev` command syntax:

```
create dev count=n,
size = n [MB | GB | CYL],
emulation=EmulationType,
config=DevConfig
[, data_member_count=nn]
[, remote_config=DevConfig,
remote_data_member_count=nn,
ra_group=n, [remote_mvs_ssid=nnn],
[dynamic_capability =
[dyn_rdf | dyn_rdf1_only | dyn_rdf2_only ], ]
[, mvs_ssid=nnn, ]
[, attribute=<ckd_meta | savedev | datadev>
[in_pool=PoolName]
[member_state=<ENABLE | DISABLE> ]
[meta_member_size = <n> [MB | GB | CYL]],
[meta_config = [striped | concatenated]],
[, disk_group=nnn,
[remote_disk_group=nnn] ]
[, binding to pool=PoolName
[preallocate size = n [ MB | GB | CYL]]
[remote_pool=PoolName]];
```

The output of `symcfg list` will show the 4 new settings.

Note: This command also supports creating static and dynamic RDF metadvice, creating BCV metadvice, and binding thin metadvice to pools.

RDF metadevices

The `create dev` command syntax also can create static and dynamic RDF metadevices. To use this feature, both the local and the remote Symmetrix arrays must be running Enginuity V5773 or later and the local Symmetrix array must have `auto_meta` enabled. The `auto_meta` feature may or may not be enabled on the remote Symmetrix array. However, any `auto_meta` settings on the remote Symmetrix array are ignored.

Example For example, assume that the local array is enabled for `auto_meta` and `min_auto_meta_size` is set to 30000. The following command file will create a 2 member meta, on both the local and the remote Symmetrix arrays:

```
create dev count=1, size = 60000, emulation=FBA, config=RDF1,ra_group=20,
remote_config=RDF2,dynamic_capability=dyn_rdf;
```

AS/400 metadevices

The process of creating AS/400 metadevices has been simplified. All AS/400 devices are a fixed size and the emulation type determines the size of the device.

Therefore, with Solutions Enabler V6.5.1, running Enginuity V5773 and earlier, the size field is optional. Solutions Enabler determines the size automatically from the emulation type.

When `auto_meta` is enabled, Solutions Enabler will attempt to create an AS/400 non metadvice and an AS/400 metadvice with or without a size value entered.

When `auto_meta` is disabled, Solutions Enabler will attempt to create an AS/400 non meta device, but will reject a request to create an AS/400 metadvice.

Example The following command creates an AS/400 meta device on a 5773 box with emulation AS/400_M4328_50.

```
create dev count=1, emulation= AS/400_M4328_50, config=2-Way-Mir;
```

This will create an 8-member metadvice where total cylinder size is 143576 (native mode).

Enable SYMAPI environment option

Environment option `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` in the `options` file must be enabled (or not present in the options file), because metadvice must be unmapped for this operation. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Metadvice creation steps

The automatic creation of metadvice can be accomplished with one configuration change session. Metadvice configurations that require two sessions (`form` and `convert`) are shown in [Table 8](#).

Table 8 Steps to create an RDF BCV metadvice

Desired device type	Session 1 form	Session 2 convert
Meta RDF1-BCV	Meta-BCV	→ Meta RDF1-BCV
Meta RDF2-BCV	Meta-BCV	→ Meta RDF2-BCV

- a. In multiples of three or seven.
- b. A set of RAID 5 devices.

Meta configuration restrictions

All metavolumes must be unmapped before configuring them as either a metahead or a member. If not, the `form meta` commands for metadevices will be rejected.

Note that RDF devices cannot be used to form metadevices in one step. When this is necessary, the RDF devices must first be converted to non-RDF devices (as shown in [Table 8 on page 55](#)), the metadevices must be formed separately on both Symmetrix arrays, and then the metahead on one Symmetrix array can be converted back to an RDF device.

Also, once the RDF metadevice has been established, it cannot be modified in one step. If any modifications are required, it will again have to be converted to a non-RDF device first.

The following are restrictions when configuring metadevices:

- ◆ The devices must be unmapped.
- ◆ Metamembers cannot be removed from a striped metadevice.
- ◆ To create a striped metadevice, all members must be the same size and have the same mirror protection.
- ◆ To remove a member from a concatenated metadevice, the member must be the tail. You cannot remove inner members.
- ◆ Only the head of a meta may have its type converted.
- ◆ Only the head of a meta is mapped or assigned to the host.
- ◆ All metas must contain at least one metamember. When a meta is formed, at least one member must be added.
- ◆ All metadevices must be composed of devices of the same BCV type (you must convert from RDF devices in order to change metas).
- ◆ Metadevices must be composed of devices that are all of the same FBA emulation type.
- ◆ You must create CKD metadevices using the `create dev` command, not the `form dev` command.
- ◆ VDEV metadevices must be composed of virtual devices that are of the same capacity, stripe size, count, and emulation type as the standard devices with which they will be used.
- ◆ An RDF device cannot be used to form a metadevice (refer to [Table 8 on page 55](#) for converting a metadevice for RDF operation). The configuration manager cannot form a metadevice from RDF devices that are RDFA-protected (that is, you cannot convert these devices to non-RDF, form the metadevice, and convert them back to RDF).

Expanding an existing metadevice

Once you have successfully formed an initial meta with metamembers, you can add additional metamembers as necessary. The following sections show how to add an additional metamember or members to an existing concatenated or striped metadevice.

Note: RAID 5 BCVs are not supported for the `protect_data` option. You can protect data by using a mirrored BCV device.

Adding members to a concatenated meta

To add additional members to an existing concatenated metadvice, use the following form:

```
add dev SymDevName[:SymDevName] to meta SymDevName;
```

Example To add Symmetrix devices 0013 and 0014 to concatenated meta 0010, enter:

```
add dev 0013:0014 to meta 0010;
```

Adding members to a striped meta

To add additional members to an existing striped metadvice, use the following form:

```
add dev SymDevName[:SymDevName] to meta SymDevName
[, protect_data=[TRUE | FALSE],
bcv_meta_head=SymDevName];
```

Where:

protect_data — Possible values are TRUE or FALSE. The protect_data option is only for striped metas. When set to true, the configuration manager automatically creates a protective copy to the BCV meta of the original device striping. Because this occurs automatically, there is no need to perform a BCV establish. When enabling protection with the protect_data option, you must specify a BCV meta identical to the existing (original) striped meta.

bcv_meta_head — When adding new members to an existing, striped metadvice, if the data on the metadvice is to be protected, you must specify the name of a bcv_meta that matches the original metadvice in capacity, stripe count, and stripe size.

Example To add Symmetrix devices 0013 and 0014 to striped meta 0010, enter:

```
add dev 0013:0014 to meta 0010,
protect_data=TRUE,
bcv_meta_head=00CA;
```

Removing metamembers

To remove a metamember or members from the tail of a concatenated metadvice, use the following form:

```
remove dev SymDevName[:SymDevName] from meta SymDevName;
```

Example To remove Symmetrix device 0014 from meta 0010, enter:

```
remove dev 0014 from meta 0010;
```

Dissolving a metadvice

To dissolve a metadvice, use the following form:

```
dissolve meta dev SymDevName;
```

Example To dissolve metadvice 0010, enter:

```
dissolve meta dev 0010;
```

Note: A dissolve action removes the metahead and all its members.

Configuring thin metadevices

The same commands (`form meta`, `dissolve meta`, `convert meta`) can be used to configure a thin metadvice.

Follow these guidelines when forming a thin metadvice:

1. Only thin devices can be used to create a thin metadvice. Mixing of thin and non thin devices to create a meta is not allowed.
2. Only unbound thin devices can be formed into a metadvice.
3. For a striped metadvice, all members should have the same device size.

Dissolving a meta

Only unbound thin metas can be dissolved.

Expanding a meta

Striped metadevices and concatenated metadevices can be expanded, however, `protect_data` is not supported, and the new member has to be unbound. As a part of the expansion process if the original metadvice was bound to a thin pool, the new metamember will be bound to the same pool.

Converting a meta

Concatenated thin metadevices cannot be converted to striped thin metadevices.

Binding a thin metadvice to a pool

When binding a thin metadvice to a pool, use the `symconfigure` command to bind the metahead and give one preallocated size. Solutions Enabler will distribute the preallocated size among all the metamembers.

The distribution will depend on the type of the metadvice, as follows:

Striped metadvice — The preallocated size will be distributed evenly among the members.

Concatenated metadvice — Starting from the metahead, each member receives a minimum of the total preallocated size remaining, based on the specified `size_of_member`. For example, if 175 GB were preallocated for a metadvice with 3 members, each of 100 GB, Solutions Enabler will allocate 100 GB to the head and 75 GB to the 2nd member and 0 GB to the last member.

Data preservation concerns

When managing metadevices, data is not written directly to the devices involved, and therefore the original data is essentially preserved. However, due to the reorganization upon completion of the device management actions, the host is unable to access all of the original data. Therefore, it is the responsibility of the attached host to handle the change to the physical geometry of the device.

Note: The expansion of striped metadevices is an exception to this rule. In this case, you can provide a BCV meta to preserve the original data, which will be rewritten to the newly configured striped metadevice.

Table 9 lists the possible data management operations and their effects on data integrity.

Table 9 Metadevice data preservation

Metadevice type with data	Operation	Data integrity preserved status
Metadevices concatenated	Adding a member	Preserved
Metadevices concatenated	Removing a member	Preserved ^a
Metadevices concatenated	Dissolving to non meta	Not preserved ^b
Metadevices concatenated	Converting to striped	Configurable ^c
Metadevices striped	Adding a member	Configurable ^c
Metadevice striped	Dissolve	Not preserved ^b

a. Preserved up to where the metadevice is removed.

b. Preserved but there is no host component to piece the data together.

c. Protection is a user-configurable option.



CAUTION

Dissolving striped or concatenated devices will lose access to data.

Adding to concatenated metadevices preserves the metadevice data, but makes the new members's original data inaccessible.

Removing concatenated members will lose access to the data on the removed member. Use these commands with great caution.

Managing spare disks

Note: Engenuity 5772 and higher does not support adding or removing spare disks.

You can reserve physical disks as dynamic (hot) spares. When a physical disk is reserved as a dynamic spare it is not allocated to any device and is held in reserve to support the hypervisors of a Symmetrix disk that has failed.

Adding a spare disk

A spare disk can only be created from a new disk containing no hypervisors. To reserve a disk as a dynamic spare, use the following command:

```
create spare count=n[, format = [512 | 520]];
```

Where:

count — The number (positive integer) of disks to be set aside as spares.

format — The recording format to be used on a spare disk. Values are 512 or 520.

Deleting a spare disk

To remove a disk as a dynamic spare, making it available for normal storage, use the following command:

```
delete spare disk=[director_num, da_interface, scsi_id];
```

Where:

director_num — The director identity number, such as 16A.

da_interface — The DA SCSI path (c, d, e, or f).

scsi_id — The SCSI target ID (hex value).

For example:

```
delete spare disk=[16A, D, 0];
```

Viewing existing spares

To view the set of existing spares, use the following command:

```
symdisk list -hotspare
```

Impact on I/O

The following are restrictions/conditions in lieu of impact on I/O activity:

- ◆ When adding or removing Symmetrix spares—no restriction on I/O.
- ◆ A spare cannot be removed while in use.

Converting devices

In the command file, you can convert the configuration type of an existing device or devices using the following command:

```
convert dev SymDevName[:SymDevName] to DeviceConfig
[emulation=EmulationType,]
[ra_group=n, remote_dev=SymDevName,
invalidate=invalidate_opt,
[remote_mvs_ssid=nnn],
start_copy=YES|NO ],
[mvs_ssid=nnn] [raidset = TRUE | FALSE];
```

To convert an RDF device or devices from static RDF to dynamic RDF, use the following command:

```
convert rdf dev SymDevName[:SymDevName] to dynamic;
```

Where:

SymDevName — Specifies the Symmetrix device name of the device targeted for change. To target more than one device, indicate the first and last devices in a series separated by a colon (:).

DeviceConfig — Specifies the desired device configuration type. Possible values are shown the first column of [Table 5 on page 44](#).

emulation — Indicates the device's emulation type.

ra_group — Specifies the RA group number in the SRDF environment.

remote_dev — Specifies the remote Symmetrix device name of the particular device targeted for change. If you specify a range of *SymDevNames* in the first line of the convert statement, the remote *SymDevName* value will be increased incrementally to arrive at the corresponding device number.

invalidate_opt — Indicates the RDF device to invalidate so that a full copy can be initiated from the remote mirror. Allowed values are R1 (invalidate the source), or R2 (invalidate the target). The value NONE is allowed if the *start_copy* option is set to NO and the devices are not dynamic RDF devices.

remote_mvs_ssid — Specifies the remote z/OS (MVS) subsystem ID that will be assigned to any device created as a result of removing any mirror(s). If not provided, the original MVS SSID will be assigned when available. If the MVS SSID group is full, you must supply a new MVS SSID. Only one *remote_mvs_ssid* can be used in a session. It will be applied to all devices created within that session.

start_copy — Indicates whether an RDF pair should be synchronized after the configuration change is committed.

Note: When creating RDF devices, all conversions within a session must have:

- DeviceConfig settings that reflect the same destination RDF type (RDF1 or RDF2).
- The same *ra_group* number.
- The same *invalidate* option.
- The same *start_copy* option.

mvs_ssid — Specifies the z/OS (MVS) subsystem ID that will be assigned to any device created as a result of removing any mirror(s). If not provided, the original MVS SSID will be assigned when available. If the MVS SSID group is full, you must supply a new MVS SSID. Only one *mvs_ssid* can be used in a session. It will be applied to all devices created within that session.

raidset — When requesting to convert a RAID-S group to unprotected devices, set **raidset** equal to **TRUE** and list the first RAID-S member. It is not necessary to list the other members.

Examples To convert two existing BCV devices (001C and 001D) to an RDF1-BCV configuration and to invalidate the source R1 and synchronize the RDF pair, enter in the command file:

```
convert dev 001C:001D to RDF1-BCV, ra group=1, remote_dev=001c,
invalidate=R1, start_copy=YES;
```

RDF devices

To convert RDF device 0014 from static RDF to dynamic RDF, enter in the command file:

```
convert rdf dev 0014 to dynamic;
```

RAID 5 device to BCV

To convert an existing RAID 5 device to a RAID 5 BCV, use the following form:

```
convert dev DeviceNumber to BCV+R-5;
```

Note: You can only add the BCV attribute to existing RAID 5 devices; converting non RAID 5 devices to RAID 5 BCVs is not allowed.

RAID-S set to unprotected

To convert a RAID-S set whose members are RDF devices (either RDF1+R-S or RDF2+R-S) to unprotected RDF devices, create a command file that requests any member of the RAID-S be converted:

```
convert dev 00D5 to RDF1, raidset=TRUE;
```

Then run the **symconfigure** utility:

```
symconfigure -sid 05605 -file cvt_raids.cmd -v -noprompt commit
```

All members will be converted.

Note: If there is a problem converting any member of a RAID set, the SYMAPI log file will contain the details.

Converting thin devices

To convert a thin device TDEV to an RDF1 thin device:

```
convert dev 0015 to RDF1+TDEV;
```

Thin devices can be converted to other thin device configurations. [Table 10](#) lists the valid conversions.

Table 10 Thin device conversions

Original thin device	Converted to
TDEV	RDF1+TDEV
TDEV	RDF2+TDEV
RDF1+TDEV	TDEV
RDF2+TDEV	TDEV

Impact on I/O

The following are restrictions/conditions to avoid impact on I/O activity:

- ◆ When adding DRV attributes, devices must be unmapped.
- ◆ When adding/removing RDF attributes, there are no restrictions on I/O. The RDF pair must be split or failed over. If failed over, the R1 device must be unmapped.
- ◆ When adding/removing BCV attributes, there are no restrictions on I/O. The standard/BCV pair must be split.

Device group invalid warning

When converting devices that are currently in a device group, some groups will be declared invalid after particular operations.

One example would be if you are performing TimeFinder operations and have placed in a group, one standard (STD) device and one TimeFinder (BCV) device.

`symconfigure` enables you to change the BCV device to a non-BCV device. This conversion would leave the group in an invalid state.

There are two ways to handle this situation:

- ◆ Remove the TimeFinder (BCV) device from the group before you begin the conversion process.
- ◆ Or, after the device has been converted, export (`symdg export`) the device group and remove from the file `symdg.txt` the invalid device and import (`symdg import`) the device group.

Enable SYMAPI environment option

Environment option `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` in the `options` file must be enabled (or not present in the options file) if the device is not currently mapped to the user host or you are working on DRV devices. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Restrictions

The following are restrictions for converting devices:

- ◆ The `convert` command can be used for three different classes of device configuration changes, as long as the class types are performed in separate sessions.

Note: Starting with Engenuity version 5669, changes for multiple operation classes can be executed in the same session, except for dynamic RDF changes and device pool changes.

The three class types that cannot be used in the same session are:

- Add/remove BCV/DRV attributes
- Add/remove RDF attributes
- Increase/decrease mirroring
- ◆ Only the head of a metadevice can have its type changed. The metamember(s) will automatically have the changes applied.
- ◆ Engenuity versions below 5669 cannot create R1/R2 pairs if the R2 is larger than the R1.
Engenuity versions 5669 and above can create R1/R2 pairs with the R2 device being larger than the R1, but swap operations will not be allowed.
Full swap operations require the R1 and R2 devices to be the same size.
- ◆ You cannot convert one member of a raidset to unprotected without converting all the members to unprotected.

Converting a metadevice

To convert a metadevice's configuration from concatenated to striped, use the following command:

```
convert meta SymDevName config=MetaOption
[, stripe_size=MetaStripeSize>[cyl]],
[, protect_data=[TRUE|FALSE],
[bcv_meta_head=SymDevName];
```

Where:

stripe_size — Specifies the stripe size in 512 byte blocks.

protect_data — Specifies whether the data needs to be protected. Possible values are TRUE or FALSE. This option can only be used while converting a metadevice from concatenated to striped.

bcv_meta_head — Specifies the Symmetrix device name.

Metadevices can be converted from concatenated metadevices to striped devices using the `convert meta` command. The command can also be used to change the stripe size of a striped metadevice.

When converting from concatenated metadevices to striped devices, use the `protect_data` option to preserve data on the BCV metadevice during conversion. The BCV metadevice must match the original standard meta in size, configuration, and member count.

The `bcv_meta_head` option specifies which BCV metadevice to use.

Example To convert a concatenated metadevice 0030 to a striped metadevice and preserve data during the conversion using BCV metadevice 01F, enter:

```
convert meta 0030, config=striped, stripe_size=1920,
protect_data=TRUE, bcv_meta_head=01f
```

Note: The specified BCV metadevice must be identical to the original metadevice in capacity, stripe count, and stripe size.

Changes to mirror protection

Note: This section applies to Symmetrix arrays running Enginuity 5773 and earlier.

The mirroring protection of a device can be increased using a separate session from BCV, DRV, and RDF change sessions. Device configuration changes to apply or remove the BCV, DRV, or RDF attribute that results in a change to the device's mirroring, will be denied.

For example, converting a 2-Way-BCV to an unprotected device will be denied. This involves two changes being made to a device, removing a mirror and removing the BCV attribute. For this type of configuration, you can only remove a mirror to form a BCV device or remove the BCV attribute to form a 2-Way-MIR device.

The mirroring protection of a device can also be decreased, and this too must be done using a separate session from the BCV, DRV, and RDF change sessions.

As a result of removing mirrors from a device, a new device is created from the discarded mirrors, resulting in a new Symmetrix device. If the original or new device is unprotected, it cannot be mapped to a host.

VDEV configuration restrictions

Symmetrix virtual devices are configured as full-size devices, but use minimal disk storage on the back-end. On Symmetrix DMX™ models, VDEVs reside in cache only and are depicted as having no back-end hypers. The format must be either CKD or FBA. Virtual devices can also be used to form a VDEV metadvice. Other Symmetrix devices cannot be converted to form virtual devices. To configure a virtual device, set `config=vdev` in the device file.

Note: Creating virtual devices (VDEVs) requires Enginuity version 5671 or higher.

SAVE device configuration restrictions

Symmetrix SAVE devices work in conjunction with virtual devices and can be unprotected¹, mirrored, or RAID 5/RAID 6.² SAVE devices cannot be part of a metavolume or grouped. Other Symmetrix devices cannot be converted to form SAVE devices. Configuring virtual and SAVE devices requires careful planning to anticipate what percentage of the data may change during a virtual copy session. Creating the SAVE device pool too small could result in data changes being lost during a copy session. Make sure that you create enough SAVE devices and set the size of the SAVE devices large enough to handle any expected changes. To configure a SAVE device, set `attribute=savedev` in the request to create a new device.

RDF configuration changes

Making changes to RDF device configurations and swapping RA groups requires that a corresponding change be made on the remote Symmetrix array. The `symconfigure` command performs these changes in parallel. If another user is executing a

1. Configuring SAVE devices as unprotected is not recommended and requires special approval by EMC.
2. For a given device emulation type, all SAVE devices must have the same protection.

configuration change on the remote Symmetrix, the RDF change will not be applied to either side.

Device locking

Note: This section describes device locking on arrays running Enginuity 5773 and earlier. With Enginuity 5874 and later, the Symmetrix V-Max array manages its own less restrictive device locking.

Symmetrix device locks prevent other control operations (that is, SRDF, TimeFinder) from being performed on devices being modified by a configuration change session. Device locks are engaged by the `symconfigure` command during the *preview* stage. For information on the various phases of running a configuration change session, refer to [“Configuration change session overview” on page 24](#).

Device locks are retained until the configuration change session has completed and changes have been activated on the Symmetrix. To view a list of the locked devices on the Symmetrix array, use the `symdev list -lock` command. Device locking is also engaged by the other Symmetrix control operations for blocking devices during control actions.

Held devices

EMC’s TimeFinder/Snap operations allow for devices to be marked as *held*, to be available for snap session copies. Devices are attached and detached as snap target devices using the `symsnap attach` or `detach` command. Held devices are set as Not Ready to the host and are not available for configuration changes. To view a list of devices currently held, use the `symdev list -held` command.

Setting device attributes

You can set the device attributes or emulation of a number of devices in a range using the following command:

```
set device SymDevName[:SymDevName]
    [emulation=EmulationType]

    [attribute=[NO] device_attr];
```

Where:

emulation — Specifies the device emulation type, which can be the following: FBA, CELERRA FBA or VME512 FBA.

attribute — Indicates if a device attribute restricts how a device can be accessed. Possible values include:

- RDB_Cksum (Enginuity 5773 and earlier)
- VCMdb (for device masking on Enginuity 5773 and earlier)
- SCSI3_persist_reserv (persistent group reservation)
- dyn_rdf (This option provides the most flexibility in performing dynamic RDF operations)
- dyn_rdf1_only**
- dyn_rdf2_only**
- ACLX (Enginuity 5874 and higher)

** These options allow a device to become an R1 or R2 device only; therefore, using them prevents the ability to perform RDF swaps.

Examples

To convert five devices (0015 to 0019) to Celerra FBA emulation, enter in the command file:

```
set device 0015:0019 emulation=CELERRA_FBA;
```

To reset the device attribute of five devices (0015 to 0019) removing RDB_Cksum, enter the following in the command file:

```
set device 0015:0019 attribute=NO RDB_Cksum;
```

Enable SYMAPI environment option

The environment option SYMAPI_CTRL_OF_NONVISIBLE_DEVS in the options file must be enabled (or not present in the options file) if the device is not mapped to the user's host. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

You can set emulation and attributes in the same session (command file). For more information on classes, refer to [“Configuration control operations” on page 30](#).

Note: The SCSI_persist_reserv attribute is only used when you have a Sun Cluster 3.0 environment, where two or more channels access the device.

Restrictions

The following are restrictions that exist while setting device characteristics:

- ◆ When setting the device emulation type, the devices must be unmapped. No I/O to the devices involved.
- ◆ When setting the attribute type to a mapped device, it is recommended that you minimize the I/O activity to the affected devices.

Setting device geometry

Solutions Enabler version 6.5 provides the ability to set the geometry for a disk on a Symmetrix arrays managed by a Solaris host.

Prior to DMX-4, the block size and sectors per track were reported differently. During a migration from an earlier version, for example from a DMX-3 to a DMX-4, Solaris looks at the new devices and think that formatting is required (due to the change in block size), and will attempt to wipe the disks by invoking the `format` utility.

By setting the reported device geometry using Solutions Enabler to look like an older Symmetrix array (making a DMX-4 look like a earlier version of DMX, or to emulate a CLARiiON® disk) will satisfy Solaris and allow the migration to proceed. This feature is useful when migrating data from devices residing in earlier Symmetrix arrays to DMX-4 arrays running Enginuity version 5773.

Note: This feature is only for Solaris hosts using the `format` utility.

Symmetrix arrays running Enginuity version 5773 will display (with `list` and `show` commands) the specified (emulated) device geometry, therefore, when a device's geometry is changed to emulate an a different device type, the display will not show the native, or true geometry, but the one that was specified.

Note: Throughout this section the following terms will be used:

`Symm-6` will be used to represent Symmetrix arrays with Enginuity versions 56xx and earlier, and `Symm-7` will represent Symmetrix arrays with Enginuity version 57xx and later.

Some factors to consider when using the new disk geometry feature are:

- ◆ The disk geometry specified will remain as the geometry for the disk, until it is explicitly changed again.
- ◆ If a copy of a target disk is made, the user needs to remember to change the geometry of the new target too. As an example, if a device is replicated using RDF from `Symm-6` (source) to `Symm-7`, the user may decide to change the geometry on the replicated `Symm-7` device. If the `Symm-7` device were to be cloned, the clone device does not automatically receive the emulated geometry—it must be explicitly set.
- ◆ The user needs to know the exact geometry of the source disk and needs to ensure that before the migration or replication occurs, this geometry is set for the target disk.

Note: There is also an array-wide setting called `FBA Geometry Emulation` that can only be turned on by an EMC Customer Engineer. When this is set to `Enabled`, all devices on a `Symm-7` will emulate `Symm-6` geometry. When set to `Enabled` on a Symmetrix array running Enginuity 5874, all devices will emulate `Symm-6` geometry.

However, even if this setting is enabled, the option of setting a geometry on an individual device level still exists, and this setting will take precedence over the array-wide setting.

This feature is set using the `symconfigure` command, which is executed using a command file. The syntax for setting the device geometry follows:

```
set dev SymDevName[:SymDevName]
  geometry [= SYMM-6 | = SYMM-7 | = CLARIION]
  cyls = n;

set dev SymDevName[:SymDevName]
```

```
geometry = <NO GEOMETRY>;
```

Where:

SYMM-6 — Refers to the device geometry in Symmetrix arrays running Enginuity 56xx and earlier (64 sectors/track, 15 tracks/cylinder).

SYMM-7 — Refers to the device geometry in Symmetrix arrays running Enginuity 57xx and later (128 sectors/track, 15 tracks/cylinder).

CLARiION — Refers to the CLARiON disk geometry (10 sectors/track, 16 tracks/cylinder; each sector is 512 bytes)

n — Refers to the device cylinder size.

NO GEOMETRY — Clears the current geometry if the device has an emulated geometry set.

Blocks and sector size for an FBA device is 512 bytes. The device geometry cannot be set on the following devices:

- Metamember
- Vault device
- SFS device
- SAVE device
- DATA device
- DRV
- Mapped device
- Non-FBA device
- VCMDB device

[“Example 8: Updating a disk label with emulated device geometry” on page 319](#) provides an example of the procedure and system output for updating a disk label for a data migration.

Showing device geometry

Disk geometry parameters will display if the `-v` option or the `-geometry` option is specified for the following commands:

- ◆ `symld list -v`
- ◆ `symdev list -v`
- ◆ `sympd list -v`
- ◆ `symld show -geometry`
- ◆ `symdev show -geometry`
- ◆ `sympd show -geometry`

Example The following is an example of the command for displaying the device geometry. The changes to the `symdev show` output for device geometry are shown in **bold**.

```
symdev -sid 016 show 0317 -geometry
```

```
Device Physical Name      : Not Visible

Device Symmetrix Name     : 0317
Device Serial ID         : N/A
Symmetrix ID             : 000192600175

Number of RAID Groups    : 1

Attached BCV Device      : N/A
```

```

Attached VDEV TGT Device : N/A

Vendor ID           : EMC
Product ID          : SYMMETRIX
Product Revision    : 5874
Device WWN          : 60000970000192600175533030333137
Device Emulation Type : FBA
Device Defined Label Type: N/A
Device Defined Label : N/A
Device Sub System Id : 0x1000
Cache Partition Name : DEFAULT_PARTITION

Device Block Size   : 512

Device Capacity
{
  Cylinders      :      958
  Tracks         :      14370
  512-byte Blocks :    1839360
  MegaBytes      :        898
  KiloBytes      :    919680
}

Effective Device Geometry: Native
{
  Sectors/Track      :      128
  Tracks/Cylinder    :        15
  Cylinders          :      958
  512-byte Blocks    :    1839360
  MegaBytes          :        898
  KiloBytes          :    919680
}

```

...(output shortened for this example...)

The field **Effective Device Geometry** can have four possible values:

- ◆ User Defined — Indicates the user has defined a geometry for this device.
- ◆ Native — Indicates the current device geometry is the same as the native geometry for the Symmetrix array.
- ◆ Array wide emulation — Indicates that the array-wide flag for FBA Geometry Emulation is set to Enabled and the effective geometry shown in the output is derived from this setting.

Note: Even if the array-wide setting is turned on, the user can still define the geometry at the individual device level and this setting will take precedence over the array-wide setting.

- ◆ N/A — For non FBA devices.

Setting device identifiers

Solutions Enabler 6.5 and later supports device identifier management on Symmetrix arrays running Enginuity 5773. This support allows names and identifiers to be defined for Symmetrix devices, HP devices, and VMS devices. The definition will be configured in a command file and processed with the `symconfigure` command.

The format of the command file for setting device identifiers follows:

```
set dev SymDevName[:SymDevName]
[[device_name = 'DevName'] | [NO DevName]]
[[hp_identifier = 'hp_id'] | [NO hp_identifier]]
[[vms_identifier = vms_id] | [NO vms_identifier]];
```

This format accommodates setting a device name or identifier on a single device, a range of devices, or multiple ranges of devices. Device identifiers do not have to be unique for all devices.

Restrictions for device_name

- ◆ The `device_name` can be less than or equal to 64 characters in length.
- ◆ Any character may be used except quotes, which are used to mark the start and end of the input.
- ◆ The names are case sensitive.
- ◆ There is no default device name for a device.

Restrictions for hp_id

- ◆ The `hp_id` can be less than or equal to 128 characters in length.
- ◆ Any character may be used except quotes which are used to mark the start and end of the input.
- ◆ The identifiers are case sensitive.
- ◆ There is no default HP identifier for a device.

Restrictions for vms_id

- ◆ The `vms_id` here is an unsigned 4-byte integer. A `vms_id` can be a number between 0 and 32766. This value will be accepted and displayed as a decimal value.

Note: The device identifier can include a `device_name` and the `hp_id` or a device name and the `vms_id`. The device identifier cannot include both the `hp_id` and the `vms_id`.

Identifier exclusions

The user cannot set device identifiers on the following devices:

- ◆ Metamembers. The device identifier of the metahead device will display, if applicable.
- ◆ The following internal devices (device identifiers display as N/A):
 - VAULT
 - SFS
 - DRV
 - SAVE (device names will be allowed for this device type)
 - DATA (device names will be allowed for this device type)

Viewing device identifiers

The user has the ability to see any one of the device identifiers through the `symdev list` command. CLI commands are usually limited to an 80-character output, but if the user requests the `hp_id` of a device to be displayed, the line could be over 80 characters in length. A new form of the `symdev list` command will be created to display the device identifiers.

Note: This CLI command does not work in offline mode.

The following example commands display each of these identifiers and their sample output:

```
symdev -sid 237 list -identifier device_name -range 0370:0372
```

```
Symmetrix ID: 000190300237
```

```

                                Device
-----
Sym  Config          Attr Device Name
-----
0370 2-Way Mir          oracle_database_accounts_device
0371 2-Way Mir          oracle_database_accounts_device
0372 2-Way Mir          oracle_database_accounts_device
0373 2-Way Mir          N/A

```

```
symdev -sid 237 list -identifier device_nice_name -range 0370:0373
```

```
Symmetrix ID: 000190300237
```

```

                                Device
-----
Sym  Config          Attr Nice Name
-----
0370 2-Way Mir          accounts_device
0371 2-Way Mir          accounts_device
0372 2-Way Mir          0001903002370372
0373 2-Way Mir          0001903002370373

```

```
symdev -sid 237 list -identifier hp_id -range 0370:0373
```

```
Symmetrix ID: 000190300237
```

```

                                Device
-----
Sym  Config          Attr HP Identifier
-----
0370 2-Way Mir          hp_oracle_device
0371 2-Way Mir          hp_oracle_device
0372 2-Way Mir          N/A
0373 2-Way Mir          N/A

```

```
symdev -sid 237 list -identifier vms_id -range 0370:0373
```

```
Symmetrix ID: 000190300237
```

```

                                Device
-----
Sym  Config          Attr VMS ID
-----
0370 2-Way Mir          8816
0371 2-Way Mir          8817

```


0372	2-Way Mir	192
0373	2-Way Mir	8819

Note: Device name, device nice name, HP device identifiers, and VMS device identifiers cannot be used in any control command. All these device identifiers are only for display. Both HP device identifiers and VMS device identifiers can be set on any host and any devices.

Deleting devices

In the command file, you can delete one or more Symmetrix devices from the specified Symmetrix array. Deleting a device frees the space that it once occupied for future use.

To delete one or more devices, use the following command:

```
delete dev SymDevName[:SymDevName][,raidset = TRUE | FALSE];
```

Where:

raidset — When requesting to delete a RAID-S group, set **raidset** to **TRUE** and list the first RAID-S member. It is not necessary to list the other members; all will be deleted.

Examples To delete Symmetrix device 0015 from Symmetrix array 000000012345, create a command file containing the following:

```
delete dev 0015;
```

Then activate the option using the command:

```
symconfigure -sid 0015 -file delete_dev.cmd -v -noprompt commit
```

To delete all devices in a RAID-S set, create a command file with a statement that only identifies the first member of the RAID-S set:

```
delete dev 015C, raidset = TRUE;
```

Then run the **symconfigure** utility:

```
symconfigure -sid 05605 -file delete_raids.cmd -v -noprompt commit
```

All members will be deleted.

Restrictions on deleting devices

The following are restrictions for deleting devices:

- ◆ The Symmetrix array must be running Enginuity version 5670 or higher.
- ◆ The device emulation must be one of the following types:
 - SYMAPI_EMULATION_FBA
 - SYMAPI_EMULATION_CELERRA_FBA
 - SYMAPI_EMULATION_VME_512_FBA
 - SYMAPI_EMULATION_CKD_3380
 - SYMAPI_EMULATION_CKD_3390
 - SYMAPI_EMULATION_AS400_n
- ◆ The device must not have an attached BCV or DRV.
- ◆ The device must not have any snap sessions.
- ◆ If the device is a DATA device it must be disabled and have no used tracks.
- ◆ The device must not be:
 - Mapped to a front-end port
 - Part of an RDF consistency group
 - A source or target of a clone session
 - Held
 - A virtual device that is in use
 - An RDF device
 - A metamember

- An SFS device
 - A SAVE device¹
 - The VCM database device
 - Masked by VCM
 - Bound if a thin device
- ◆ You cannot delete one member of a raidset without deleting all the members.

1. Starting with Solutions Enabler version 6.0, you can delete a SAVE device as long as it is disabled with no active tracks.

Managing device reservations

With Enginuity version 5670 and higher, you can use the configuration change functionality to reserve devices and front-end mapping addresses for future configuration and masking operations. When using this feature, you reserve the devices/addresses you plan on using, verify that no one else has reserved the resources, and release the reservations when the task is complete.

All reservations are assigned a reserve ID, indicating that the specified devices/addresses are reserved. Any attempt to use the reserved devices/addresses will return a message indicating that the devices/addresses are reserved.

There are two types of device reservations:

- ◆ **Enforced** — Reservations are enforced by the SYMAPI library, and require that you specify the reserve ID to use the devices. This is the default behavior when reserving devices. Applications developed before Solutions Enabler V6.4 may not be able to process reservation IDs.
- ◆ **Advisory** — Reservations are enforced by co-operating applications. Some applications can ignore advisory reservations, allowing knowledgeable users to make configuration changes on reserved devices, provided that their changes are compatible with the reserving task's goal.

Both types of reservations can have expiration dates associated with them, which will automatically release a reservation if the user fails to explicitly do so.

Device reservations are enabled (TRUE) by default in the `options` file. To disable device reservations, set the `SYMAPI_ENABLE_DEVICE_RESERVATIONS` parameter in the `options` file to FALSE.

Device reservations are enforced (TRUE) by default in the `options` file. To disable the enforcing of device reservations, set the `SYMAPI_ENFORCE_DEVICE_RESERVATIONS` parameter in the `options` file to FALSE.

Note: For information on changing the `options` file parameters, refer to the *EMC Solutions Enabler Installation Guide* or the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

Reserving devices

To reserve devices, use the following form:

```
symconfigure -sid SymmID [-expire ExpirationDate]
               -f[file] CmdFile [-noprompt] [-v]
               -owner Owner -comment UserComment
               [-enforce | -advise]
reserve
```

Where:

ExpirationDate — Specifies the date and time for a device reservation to expire. This is an optional parameter, and if not specified, defaults to no expiration. The format for this parameter is:

```
[mm/dd[ /yy] :] [hh:mm[:ss]]
```

If you only provide the `hh:mm`, the current day will be assumed. If you only provide the `mm/dd`, the current year will be assumed. You can also specify a four-digit year.

CmdFile — Specifies the name of any ASCII text file containing a set of commands to process at a later time. This file can be used in the following ways:

- To reserve devices for specific configuration change operations, in which case the file will list configuration change commands.
- To reserve devices for non-specific operations, in which case the contents of the file will use the following form:

```
reserve dev SymDevName[:SymDevName]
```

Using this method allows you to reserve devices for other applications.

Owner — Specifies the name of the owner of the reservation (up to 31 characters long).

Usercomment — Indicates a user-specified comment detailing the device reservation (up to 255 characters long).

-enforce — Specifies an enforced reservation (default).

-advise — Specifies an advisory reservation.

When reserving devices, be sure to note the returned reserve ID for subsequent processing.

Example To reserve a set of devices to be deleted at a later time, enter:

```
symconfigure -sid 3241 -file delete.cmd reserve
-owner "LabMgr" -comment "Deleting RAID-S devices"
-expire 04/15/2005 -enforce
```

where delete.cmd contains:

```
delete dev 0015, raidset=true;
delete dev 0017, raidset=true;
```

Committing changes on reserved devices

When committing changes on devices reserved with the Enforced flag, you must supply the appropriate reserve ID. If you do not have the reserve ID and someone else has reserved the devices, the commit will fail. If you have reserved the devices, or no one else has reserved the devices, then the commit will succeed.

When committing changes on devices reserved with the Advisory flag, some applications may not require a reserve ID. However, the `symconfigure` and `symmask` commands do require a reserve ID.

To commit changes on devices reserved with the Enforced flag, use the following form:

```
symconfigure -sid SymmID -f[file] CmdFile commit
[-reserve_id=ResvID[,ResvID[,ResvID]]]
[-remote_reserve_id=ResvID[,ResvID[,ResvID]]]
```

Example To commit the changes in the command file delete.cmd, enter:

```
symconfigure -sid 3241 -file delete.cmd commit -reserve_id 5
```

Viewing existing reserved devices

To view existing reservations, use the following commands:

```
symconfigure -reserved list
```

```
symconfigure -reserve_id ResvID show
```

Where:

ResvID — The device reservation ID.

Example To view the set of devices reserved on Symmetrix 54, enter:

```
symconfigure -sid 54 list -reserved
```

The following displays:

Symmetrix ID: 000387940054

S Y M M E T R I X D E V I C E R E S E R V A T I O N S

Reserve ID	Date Reserved	Flags	Owner	Devices	Port Addresses
-----	-----	-----	-----	-----	-----
000001	09/27 11:51	E-	Kevin	01A5	-
Comment: Testing metadvice routines					
000005	10/10 13:45	EL	JaneH	01B6	01C:0 [C9]
				01B6	01C:1 [C9]
Comment: Same device, multiple mapping addresses					
000006	10/11 11:06	E-	SteveM	002E:0030	-
				0049:004B	-
				002D	15C:0 [2,0,3]
				0107	01C:0 [4]
				014D	01C:1 [25]
				0004	15C:0 [1,0,4]
				0047:0048	15C:0 [1,0,5:1,0,6]
Comment: Set up more devices for RCopy testing					
000008	10/17 15:34	EV	Shiran	01B7:01BA	15C:1 [0,1,0:0,1,3]
				01BC:01BF	15C:1 [0,1,4:0,1,7]
Comment: Reserve some devices for VSA port					
000002	10/19 11:27	E-	MikeS	0115:0145	-
Comment: Deleting RAID-S devices					

Legend for Flags:

Type of Reservation (T) :	E = Enforced	A = Advisory
Mapping Address Mode (M) :	L = Lun addressing	V = Volume Set addressing
	T = Target/lun	C = Channel addressing

To request details on reservation 000008, enter:

```
symconfigure -sid 54 show -reserve_id 8
```

Symmetrix ID: 000387940054

```

Reserve ID           : 000008

Usage                : Mapping
Flags                : Enforced

Reserved Date        : 10/17/2005 15:34:04
Expiration Date      : None

Devices              : 01B7:01BA
Front End Paths
{
  Director Number    : 15C
  Director Port Number : 1
  VSA [VBUS,TID,LUN] : [0,1,0] - [0,1,3]
}

Devices              : 01BC:01BF
Front End Paths
{
  Director Number    : 15C
  Director Port Number : 1
  VSA [VBUS,TID,LUN] : [0,1,4] - [0,1,7]
}

Owner                : Shiran
Hostname              : api196
Application           : SYMCONFIGURE
User Data             : 0
User Comment          : Reserve some devices for VSA port

```

Releasing reserved devices

When releasing device reservations, you must supply the appropriate reserve ID. Performing a configuration change on reserved devices will not release them. You must do this as an independent step.

To release reserved devices, use the following form:

```

symconfigure -sid SymmID [-noprompt] -reserve_id
ResvID[, ResvID[, ResvID]]
release

```

Where:

ResvID — Specifies the device reservation ID.

Example To release the set of devices reserved in the example on [page 77](#), enter:

```
symconfigure -sid 3241 release -reserve_id 5,7
```

Configuring Virtual Provisioning

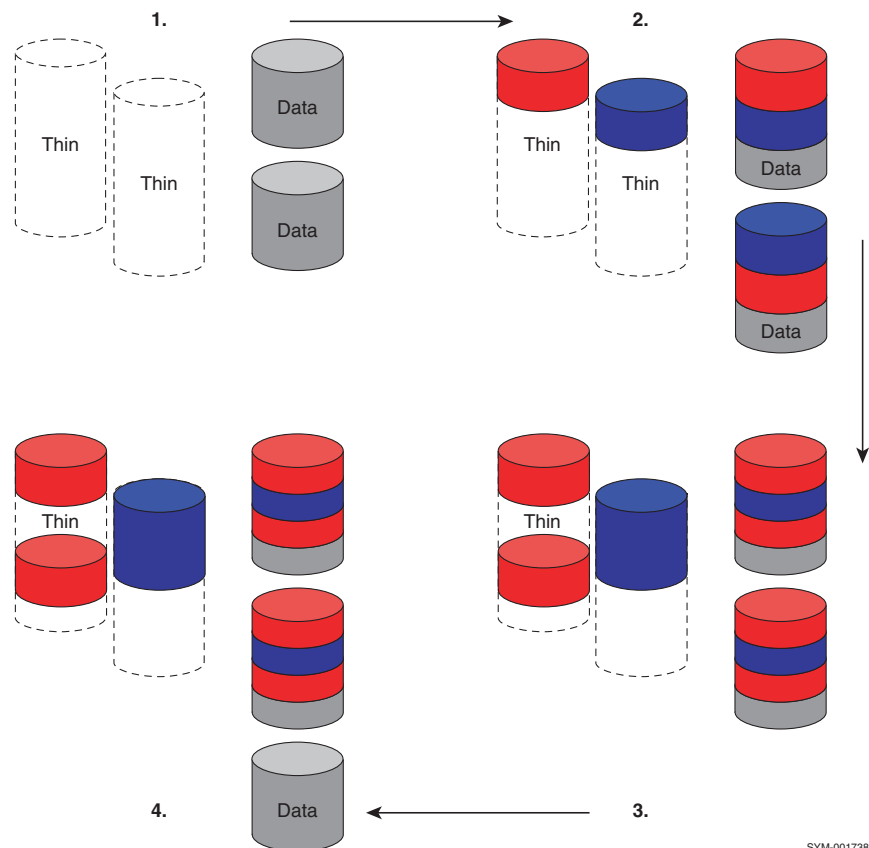
Virtual Provisioning™ provides for the separation of the physical storage devices from the storage devices as perceived by the host systems. The actual physical storage is not perceived by the host. This allows alterations in the physical storage to occur without causing changes which affect the hosts. The result is nondisruptive provisioning and more efficient tiering. Virtual Provisioning has the ability to define to the host servers an apparent disk configuration that is far larger than current needs. It may be possible to provision an amount of storage for applications which will last the life of the applications themselves. This means applications need to be disrupted much less frequently to make changes or add storage.

Virtual Provisioning also means simplification of choices as to which volumes to use for which files. Whether these choices are made by people or software, the choices will be less often restricted by limited space. The choices made will more often reflect the needs of simplicity of management, or performance requirements. Wide striping allows the data to be spread evenly across a broad pool. It allows all data in an application group, known in this case as a thin pool, to be spread fairly evenly across all the drives supporting that group. This balancing of the workload across physical devices can translate into improved performance. It can reduce queuing and smooth out response times. It also reduces performance management effort.

The Virtual Provisioning feature introduces three new concepts: *thin devices* (`TDEV`), *DATA devices* (`datadev`), and *thin pools* (`thin`). Thin devices can be created with an inflated capacity, because the actual storage space for the data written to the thin devices is on the DATA devices. In this way, when additional storage is needed, more DATA devices can be created in the thin pool. [Figure 7 on page 81](#) provides an overview of the data flow of the write operations:

1. The user allocates thin devices that are larger in size than available storage. No storage is initially allocated to these thin devices.
2. As the user writes to thin devices, space gets allocated on the DATA devices in the thin pool.
3. At some point, the physical space available will start to get exhausted.

4. The user adds more DATA devices to the thin pool to accommodate the growing thin devices.



SYM-001738

Figure 7 Data allocation for thin devices

DATA devices are not visible to the host.

Thin devices bound to pools containing DATA devices are the components used for the Virtual Provisioning feature provided with Solutions Enabler V6.5 and Enginuity 5773 and later.

Creating DATA devices

DATA devices are similar to SAVE devices, in that they are not visible to the host and must be contained in a pool before they can be used. Thin pools can only contain devices of the same emulation and protection type, however, the DATA devices can be different sizes.

Use a command file (`symconfigure`) to create DATA devices. The syntax related to creating DATA devices is shown below:

```
create dev count=n,
size = n [MB | GB | CYL],
emulation=EmulationType,
config=DeviceConfig
[, attribute=<ckd_meta | savedev | datadev>
[in pool=PoolName]
[member_state=<ENABLE | DISABLE] ]
```

Where:

- create dev count** — Specifies the number of DATA devices you are creating.
- size** — Indicates the size of the DATA device. If MB or GB is not specified, the size defaults to cylinders.
- emulation** — Specifies FBA or CKD. AS/400 devices cannot be DATA devices.
- config** — Specifies the device configuration. The following configuration types are supported for DATA devices: 2-Way-Mir, RAID5 (3+1 and 7+2), and RAID6 (6+2 and 14+2).
- attribute** — Specifies the device type datadev.
- in pool** — Specifies the name of the thin pool.
- member_state** — Indicates the state of the DATA device; enable or disable.

DATA device configuration rules

The following configuration rules apply to DATA devices:

- ◆ The only supported emulation types are FBA and CKD.
- ◆ There is no default thin pool.
- ◆ A DATA device cannot be replicated (no BCV, RDF, and so on)
- ◆ A DATA device cannot be a metadvice.
- ◆ DATA devices are not visible to the host; they cannot be mapped or masked.

Creating thin devices (TDEVs)

Thin devices are devices that do not have storage allocated to them when they are created. To a host operating system, they look like regular devices with their configured capacity. The host treats them as regular devices and writes and reads from these devices like regular devices.

Use a command file (`symconfigure`) to create thin devices. The syntax related to creating thin devices is shown below:

```
create dev count=n,
size = n [MB | GB | CYL],
emulation=EmulationType,
config=DeviceConfig
[, remote_config=DeviceConfig,
[, binding to pool=PoolName
[preallocate size = n [ MB | GB | CYL]]
[remote_pool=PoolName]];
```

Where:

- create dev count** — Specifies the number of thin devices you are creating.
- size= n [MB |GB |CYL]** — Specifies the size of each thin device. If MB or GB is not specified, the size defaults to cylinders.
- emulation** — Indicates FBA or CKD. AS/400 devices cannot be thin devices.
- config** — Specifies TDEV (local thin device) or RDF1+TDEV (local RDF thin device).
- remote_config** — Specifies RDF2. The `remote_config` option is only used when creating both local and remote RDF thin devices.
- binding to pool** — Specifies the name of the local thin pool.

preallocate size — Specifies the amount of space preallocated to the devices. The amount of space must be less than or equal to the available space in the pool and less than or equal to the allocated size of the thin device.

remote_pool — Specifies the name of the remote thin pool.

Example of create/bind

To create ten 2 GB thin devices, bind them to thin pool `tp_pool` and preallocate 1 GB to each thin device, enter:

```
symconfigure commit -sid 12345 -file add_new_tdevs.cmd
```

Where `add_new_tdevs.cmd` contains:

```
create dev count=10, size=2GB
    emulation=FBA, config=TDEV,
    binding to pool tp_pool, preallocate size=1 GB;
```

Thin devices that were not bound to a data pool during device creation, can be bound using the following syntax:

```
bind tdev SymDevName[:SymDevName]
    to pool PoolName
    preallocate size = n [MB | GB | CYL];
```

The same metadvice commands (`form meta`, `dissolve meta`, `convert meta`) can be used to configure a thin meta device. [Table 11](#) provides the supported thin metadvice features.

Table 11 Supported thin metadvice features

Metadvice feature	Thin device support
Form a thin meta	- Only thin devices can be used to create a thin metadvice. Mixing of thin and non thin metadvice to create a metadvice is not allowed. - Only unbound thin devices can be formed into a metadvice. - For a striped metadvice, all members should have the same device size.
Dissolve a thin meta	Only unbound thin metadvice can be dissolved.
Expand a thin meta	A bound striped metadvice cannot be expanded. An unbound striped or concatenated metadvice can be expanded; however, the new meta member must be unbound. As part of the expansion process, the new metamember will be bound to the same thin pool. The <code>protect_data</code> option is not supported.
Convert a thin meta	Concatenated thin metadvice cannot be converted to striped metadvice. See Table 10 on page 63 for supported thin device conversion types.

Thin device configuration rules

The following rules apply to thin devices:

- ◆ The only supported emulation types are FBA and CKD.
- ◆ Thin devices can only be bound to one pool.
- ◆ The thin pool to which the thin devices are bound must have free tracks.
- ◆ A thin device inherits its protection type from the pool to which it is bound. Whatever configuration type the DATA devices have, the thin devices have.
- ◆ Many thin devices can use the same pool.

- ◆ PowerPath® consistency groups are not supported for thin devices.
- ◆ RDF21 devices cannot be thin devices.
- ◆ The only supported RDF configuration is when both the R1 and the R2 are thin devices.
- ◆ Only non meta CKD thin devices can be bound to a CKD data pool.

Converting thin devices

Thin devices can be converted to other thin device configurations. [Table 10 on page 63](#) lists the valid conversions.

Creating thin pools

Thin pools can be created at the same time as creating the DATA devices. The same naming conventions that apply to SAVE pools and RDF/DSE pools apply. However, there is no default thin pool. Refer to “[Managing device pools](#)” on [page 90](#) for how to create a device pool.

Thin pool restrictions

The following restrictions apply to thin pools:

- ◆ A thin pool can only contain DATA devices with one emulation type. The pool itself is not defined to be a pool for an emulation type. Instead, the first DATA device added to the thin pool defines its emulation type.
- ◆ All DATA devices in a thin pool must have the same protection. For example, if a thin pool has the first DATA device defined as a 2-Way-Mir, all subsequent added DATA devices must be 2-Way-Mir devices.
- ◆ A DATA device of a specific protection and emulation can be put only into an empty thin pool, or into a pool that *only* contains DATA devices that have the same protection and emulation type.
- ◆ A DATA device can only be removed from a thin pool if it is disabled and has no used tracks on it.
- ◆ DATA devices cannot be disabled and moved from a thin pool in the same session.
- ◆ If the total pool capacity needed to store existing thin device data will be insufficient as a result of a `disable dev SymDevStart:SymDevEnd` command, none of the devices in the requested range will be disabled.

Binding and unbinding devices

Use the following command to bind thin devices to a thin pool:

```
bind tdev SymDevName[:SymDevName] to pool PoolName
    preallocate size = nnn [MB | GB | CYL];
```

Where:

preallocate size — The amount of preallocated space. The space must be less than or equal to the available space in the pool and less than or equal to the specified size of the thin device. If MB or GB are not specified, cylinders are used by default.

Use the following command to unbind thin devices from pools:

```
unbind tdev SymDevName[:SymDevName] from pool PoolName;
```

A thin device cannot be unbound from a pool if any of the following occurs:

- ◆ Device is mapped to a front-end port or in the Ready state
- ◆ Device is masked by VCM

- ◆ Device has active snap sessions
- ◆ Device is held
- ◆ Device is a source or target of a clone (src or tgt) session
- ◆ Device is a metamember
- ◆ Device is a part of enabled RDF CG group
- ◆ Device is an RDF device

Setting up Virtual Provisioning

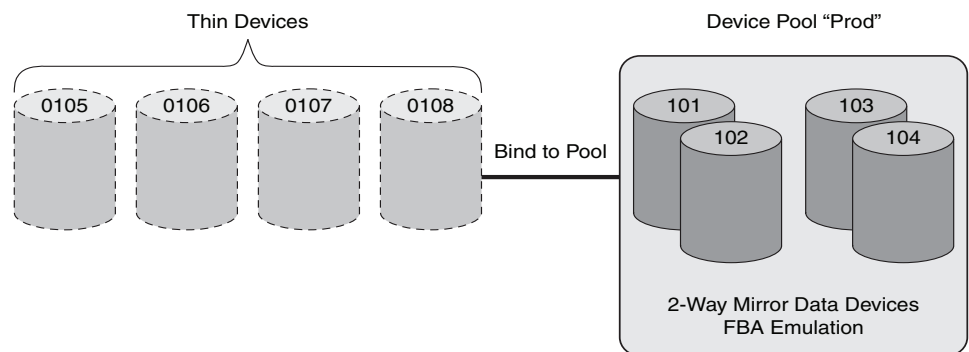
Setting up To set up a Virtual Provisioning environment:

1. Create a pool of DATA devices. An example follows, using the `symconfigure` command file syntax:

```
create pool Prod type =thin;
create dev count =4 config=2-Way_Mir, attribute=datadev,
emulation=FBA, size=1000;
add dev 0101:0104 to pool Prod type=thin member_state =ENABLE;
```

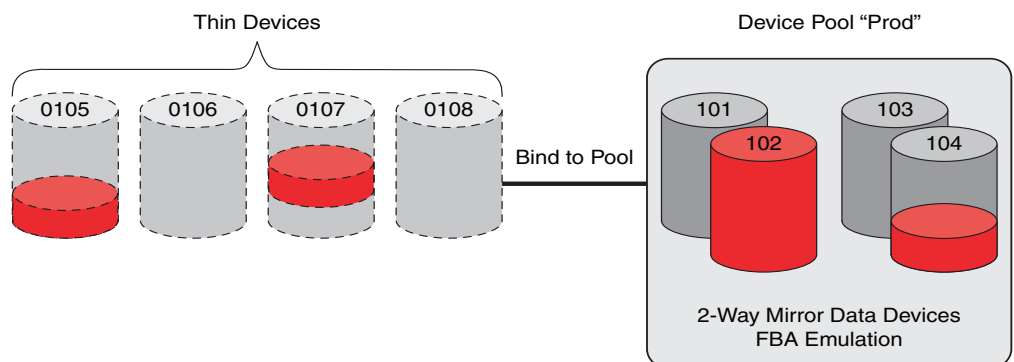
2. Create the thin devices. An example follows:

```
create dev count=4, config=TDEV, emulation=FBA, size=4000;
bind tdev 0105:0108 to pool Prod;
```



SYM-001739

Using When an application first writes to a location on a thin device, space is allocated on the DATA devices for this write. Space is allocated in "chunks." Applications view thin devices as regular devices and have no idea that the storage backing them does not have the same capacity as the total capacity of the thin devices.



SYM-001740

When the thin pool starts filling up, storage administrators always have the capability of adding more DATA devices to the pool to increase the overall pool size.

Allocating space on a thin device

Space can be allocated for the thin device through the `preallocate` action during the `bind` command, or through the `symconfigure allocate` command.

The syntax of the command allows administrators to populate a region of the thin device by giving a starting cylinder on the thin device and the number of cylinders to allocate, as shown:

```
allocate tdev SymDevName start_cyl= nnn
end_cyl=last_cyl | size=nnn MB |GB |CYL;
```

Examples `allocate tdev 254 start_cyl = 0, end_cyl = last_cyl;`
(Allocate the entire device).

`allocate tdev 254 start_cyl = 0, size = 2500;`
(Allocate 2500 cylinders to device 254 starting from 0.)

`allocate tdev 254 start_cyl = 2500, end_cyl = last_cyl;`
(Allocate device 254 starting from 2500 cylinder to the end of device.)

Freeing up unwritten allocated space

If the administrator wants to free up the unwritten allocated space, then they do so using the `symconfigure free` command. The syntax of the command is similar to the syntax of the `allocate` command.

```
free tdev SymDevName start_cyl=nnn
end_cyl= last_cyl | size= nnn MB |GB |CYL;
```

Viewing thin pools Thin pools display some information that is more relevant to just those pools than to Snap and RDFA/DSE pools. To display this information, use the `-detail` option, as shown in the following example:

```
symcfg list -sid 343 -pool -thin -mb -detail
```

Symmetrix ID: 000190300343

S Y M M E T R I X T H I N P O O L S							
Pool Name	Dev Emul	Dev Config	Total MBs	Enabled MBs	Alloc MBs	Alloc (%)	Subs (%)
xt_tp_2way	FBA	2-Way Mir	1920	1920	377	19	3596
xt_tp_raid5	FBA	RAID-5 (3+1)	1796	898	105	11	0
HR_THIN_R5	FBA	RAID-5 (7+1)	3360	2880	246	8	88
tp_pool	FBA	2-Way Mir	17263	17263	0	0	0
Total MBs			24339	22961	729	3	312

The display has the following additional information in it that is particularly relevant for thin devices and thin pools:

Total MBs — Indicates the total capacity of the thin pool. This includes the sum of the enabled and disabled DATA devices in the pool.

Enabled MBs — Shows the total capacity of the enabled DATA devices in the pool. For the pool HR_THIN_R5, the **Enabled** capacity is less than the **Total** capacity. This indicates that there are some DATA devices in this pool, that are in the disabled state.

Alloc MBs — Is the amount of space that has been allocated to thin devices bound to this pool. Hence, in the example shown, the HR_THIN_R5 pool has enabled DATA devices that total 3360 MB. Out of this, 246 MB have been allocated to accommodate the writes happening to the thin devices that are bound to this pool.

Alloc % — Shows the percentage of space that has been allocated in the pool.

Subs % — Indicates the subscribed percentage. This is the ratio of the capacity of the thin devices that are bound to this pool with respect to the total enabled capacity of this pool. The ratio is expressed as a percentage.

Refer to [“Managing device pools” on page 90](#) for additional examples of displaying thin pools.

Viewing thin devices

Using the `-tdev` option in `symcfg list`, a user can review the thin devices in the system. The displayed list can be filtered by emulation type or by the pool name to which the thin devices belong.

```
symcfg list -sid 343 -tdev -mb -RANGE 01c0:0240
```

```
Symmetrix ID: 000190300343
```

```
Enabled Capacity (MBs) :      22961
Bound   Capacity (MBs) :      71583
```

S Y M M E T R I X T H I N D E V I C E S								
Sym	Pool Name	Dev Emul	Total MBs	Pool Subs (%)	Alloc MBs	Pool Alloc (%)	Wrtn MBs	Pool Wrtn (%)
01C0	xt_tp_2way	FBA	17263	899	133	7	0	0
01C1	xt_tp_2way	FBA	17263	899	134	7	0	0
01C2	xt_tp_2way	FBA	17263	899	138	7	28	2
01C3	xt_tp_2way	FBA	17263	899	137	7	0	0
01C4	-	FBA	17263	0	0	0	0	0
01C8	-	FBA	17263	0	0	0	0	0
01F6	-	FBA	937	0	0	0	0	0
01F7	-	FBA	1875	0	0	0	0	0
01FD	-	FBA	93	0	0	0	0	0
021E	-	FBA	375	0	0	0	0	0
0222	-	FBA	375	0	0	0	0	0
0226	-	FBA	375	0	0	0	0	0
0237	HR_THIN_R5	FBA	187	7	0	0	0	0
0238	HR_THIN_R5	FBA	187	7	1	0	0	0
0239	-	FBA	3072	0	0	0	0	0
023A	-	FBA	1171	0	0	0	0	0
023B	HR_THIN_R5	FBA	1078	37	192	7	0	0
0240	HR_THIN_R5	FBA	1078	37	1	0	1	0
Total MBs			169174	737	740	3	29	1

Setting pool limits

Users are able to configure a maximum subscription ratio for a thin pool. (This ratio is the total host-perceived capacity of thin devices, divided by physical thin pool capacity). The value for `max_subs_percent` can be set for a thin pool when it is created with the following command file syntax:

```
create pool PoolName,
    type = <snap | rdfa_dse | thin>
    [, max_subs_percent= <n | NONE>];
```

It can also be set after the thin pool is created using the following form:

```
set pool PoolName,
    type = <thin>,
    max_subs_percent= <n | NONE>;
```

Refer to [“Managing device pools” on page 90](#) for details about configuring device pools.

The following example shows a thin pool set to a maximum subscription percent of 250:

```
set pool LARGE, type=thin, max_subs_percent=250;
```

The thin pool is viewed here using the `symcfg show` command:

```
symcfg -sid 343 -pool LARGE -thin -detail -all -mb show
```

```
Symmetrix ID: 000190300343
```

```
Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : TH
Dev Emulation          : FBA
Dev Configuration      : RAID-5(3+1)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 198
Max. Subscription Percent : 250
```

```
Enabled Devices(198):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)   State
-----
032F      1024         0        1024    0  Enabled
0330      1024         0        1024    0  Enabled
0331      1024         0        1024    0  Enabled
..
..
03F3      1024         0        1024    0  Enabled
03F4      1024         0        1024    0  Enabled
-----
MBs      202851         0        202851    0
}
```

```
Disabled Devices(2):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)   State
-----
03F5      1024         0        1024    0  Disabled
}
```


03F6	1024	0	1024	0	Disabled
-----	-----	-----	-----	-----	
MBs	2049	0	2049	0	
}					

In the previous example, no thin devices were bound to device pool `LARGE`.
Refer to [“Example 9: Configuring virtual provisioning” on page 327](#) for a complete Virtual Provisioning example.

Managing device pools

There are three types of device pools, Snap pools, SRDF/A DSE pools, and thin pools.

TimeFinder/Snap uses SAVE device pools to store pre-update images or change tracks during a virtual copy session. Symmetrix supports the creation of multiple named device pools, allowing `symsnap` commands to select a particular pool. This alleviates contention for SAVE device space among several snap sessions and eliminates the possibility of a single session using all pool space.

SRDF uses SRDF/A DSE pools to extend the cache space available for SRDF/A cycles by off-loading some or all of the cycle data from cache to pre-configured disk storage pool. These pools are configured the same as Snap pools.

Thin devices (used for Virtual Provisioning) are bound to DATA device pools. Each write to a thin device by the host is written in the bound DATA device pool. The DATA device pools do not have a default pool.

The following are attributes of device pools:

All device pools

- ◆ When a pool is created, it will be created as either: `snap`, `rdfa_dse`, or `thin` pool.
- ◆ Device pools are made up of devices that are specifically configured to be used for pools.
- ◆ Devices that do not belong to any pools are available to be added to a pool. Devices in a pool are available for use once they are enabled.
- ◆ Devices can be removed from a pool, but must be disabled and "drained."

Snap pools

- ◆ A special Snap pool `DEFAULT_POOL` exists by default, which contains SAVE devices that have not been assigned to any named pools but are available for Snap operations.

SRDF/DSE pools

- ◆ SRDF/A DSE pools can be associated (shared) with multiple SRDF/A groups.
- ◆ A single SRDF/A group can have, at most, four RDFA DSE pools associated with it. There can be at most one FBA, one CKD3390, one CKD3380, and one AS400 pool.
- ◆ An SRDF/A DSE pool can contain only one type of device from the list of FBA, CKD3390, CKD3380, and AS400.

Thin pools

- ◆ A DATA device pool can only have devices of one emulation type in them. The pool itself is not defined to be a pool for an emulation type. Instead, the first DATA device added to the pool defines its emulation type.
- ◆ A DATA device pool can be bound with multiple thin devices.
- ◆ A DATA device pool can only contain FBA devices.
- ◆ DATA devices cannot be disabled and moved from a pool in the same session.
- ◆ DATA devices for thin provisioning do not have a default pool.
- ◆ If the total pool capacity needed to store existing thin device data will be insufficient as a result of a `disable dev SymDevStart:SymDevEnd` command, none of the devices in the requested range will be disabled.

- ◆ All DATA devices in a DATA device pool must have the same protection. As an example, if a DATA device pool has the first device as a RAID 5, 3+1 DATA device, all subsequent added devices must have RAID 5, 3+1 protection.
- ◆ A DATA device of a specific protection and emulation can be put only into an empty thin pool, or into a pool that *only* contains DATA devices that have the same protection and emulation type.
- ◆ A DATA device can only be removed from a pool if it is disabled and has no used tracks on it.

This section explains how to create and delete device pools, add and remove devices from pools, and enable and disable their members.

Note: For more information about SAVE devices and TimeFinder/Snap operations, refer to the *EMC Solutions Enabler TimeFinder Family CLI Product Guide*.

For information about using SRDF/A DSE pools, refer to the *EMC Solutions Enabler SRDF Family CLI Product Guide*.

For information about using Virtual Provisioning and DATA device pools, refer to [“Configuring Virtual Provisioning” on page 80](#).

Creating a device pool

To create a device pool, use the following form:

```
create pool PoolName, type = snap | rdfa_dse | thin
[, max_subs_percent= <n | NONE>];
```

Where:

type — Specifies the type of device pool. Snap pools are for TimeFinder/Snap operations, SRDF/A DSE pools are for SRDF/A operations, thin pools are for Virtual Provisioning operations with thin devices.

PoolName — Specifies the name of the device pool. It can be from 1 to 12 alphanumeric characters long and include hyphens (-), and underscore (_) characters.

max_subs_percent — Specifies the maximum total capacity that can be bound to the pool. This can have a value between 0 and 65534. A value of 100 indicates that the total capacity of the thin devices bound to a pool cannot exceed 100 percent of the actual capacity of the pool. A value of 200 indicates that the total capacity of the thin devices bound to a pool cannot exceed twice the amount of the actual capacity of the pool.

A value of 1 will allow only a maximum thin capacity that totals 1/100th the actual physical capacity of the pool. The largest number 65534 indicates that the total capacity of the thin devices bound to a pool can be 65534 percent of the total physical capacity of the pool.

New device pools are disabled at creation. To enable the pool, you must add and enable one or more devices in the pool.

The maximum number of device pools per Symmetrix array is 510.

Example

To create a new pool named `Finance`, enter:

```
create pool Finance, type=snap;
```

Deleting a device pool

You can only delete a device pool if it does not have any members.

To delete a device pool, use the following form:

```
delete pool PoolName, type= snap | rdfa_dse | thin;
```

Example To delete a pool named Finance, enter:

```
delete pool Finance, type=snap;
```

Adding devices to a pool

You can only add disabled and inactive devices to a pool. While adding the devices, you can specify whether they should be enabled or disabled in the pool.

To add devices to a pool, use the following form:

```
add dev SymDevName[:SymDevName] to pool PoolName
type= snap | rdfa_dse | thin
[, member_state=ENABLE | DISABLE];
```

Where:

member_state — Indicates whether the device(s) being added should be enabled or disabled in the pool.

Example To add and enable devices 001B:001D to a thin provisioning data pool named ACT20, enter:

```
add dev 001B:001D to pool ACT20, type=thin, member_state=ENABLE;
```

Removing devices from a pool

You can only remove disabled and inactive devices from a pool. Once you remove a device from a pool, the device is available to be added to any other pool.

To remove devices from a pool, use the following form:

```
remove dev SymDevName[:SymDevName] from pool PoolName,
type=snap | rdfa_dse | thin;
```

Example To remove device 001E from a pool named HR30, enter:

```
remove dev 001E from pool HR30, type=rdfa_dse;
```

Enabling/disabling devices

You can enable or disable devices for use in a pool. The devices in a pool do not all have to be in the same state (enabled or disabled). If all the devices in a pool are disabled, then the pool is disabled. If at least one device in a pool is enabled, then the pool is enabled. To enable or disable a range of devices, all the devices must be in the same pool. To disable a device, all sessions must be terminated, and have no used tracks.

If a DATA or SAVE device has used tracks and is disabled, the data will be drained to other devices in the pool if there is enough space. When complete, the device state changes to Disabled.

To enable or disable a device in a pool, use the following form:

```
enable | disable dev SymDevName[:SymDevName] in pool PoolName,
type=snap | rdfa_dse | thin;
```

Where:

PoolName — Specifies the name of the device pool. It can be from 1 to 12 alphanumeric characters long and include hyphen (-), and underscore (_) characters.

Example To enable the use of three new DATA devices in a pool named *Finance*, enter:

```
enable dev 01B:01D in pool Finance, type=thin;
```

Monitoring a device pool

To monitor a device pool's usage, the `symcfg monitor` command has been added, as follows:

```
symcfg [-sid SymmID] [-i Interval] [-c Count] [-offline]
      [-percent <1-100> -action ScriptFile [-norepeat]]
      [-snap | -rdfa_dse | -thin] [-pool PoolName]
monitor
```

Where:

-action — Selects a script that should be run when the specified percent value is encountered. The full pathname to the action script must be specified. The first argument passed to the script is automatically set to the percent value. This option requires the `-percent` option.

-c — Indicates the number (count) of times to display or to acquire an exclusive lock on the Symmetrix host database, the local Symmetrix, and the remote Symmetrix arrays.

-i — Specifies the repeat interval in seconds to display or to acquire an exclusive lock on the Symmetrix host database, the local Symmetrix, and the remote Symmetrix arrays. The default interval is 10 seconds. The minimum interval is 5 seconds.

monitor — Checks the total percent full of the devices currently configured in the pool and can optionally execute a script file if a specified percentage is encountered.

-norepeat — Specifies that the action script should only be run once if the threshold has been met. Used with the action script option on the `monitor` command.

-percent — Causes the action script to be executed when the percent full argument is encountered.

-pool — Specifies an SRDF/A DSE, snap, or thin pool.

-snap — Specifies that the type of the pool to be monitored is a snap pool.

-rdfa_dse — Specifies that the type of the pool to be monitored is an RDF/A DSE pool.

-thin — Specifies that the type of the pool to be monitored is a thin pool.

Viewing a device pool

To view a device pool, use the `symcfg list` command. An example that lists all pools in a Symmetrix array is shown below:

```
symcfg -sid 343 list -pool
```

```
Symmetrix ID: 000190300343
```

S Y M M E T R I X P O O L S									
Pool Name	Type	Device Emul	Device Config	Total Tracks	Enabled Tracks	Used Tracks	Free Tracks	Full (%)	Status
DEFAULT_POOL	SN	FBA	Mixed	2324370	2324370	0	2324370	0	Enabled
DEFAULT_POOL	SN	3390	Unknown	0	0	0	0	0	Disabled
DEFAULT_POOL	SN	3380	Unknown	0	0	0	0	0	Disabled
DEFAULT_POOL	SN	AS400	Unknown	0	0	0	0	0	Disabled
xt_snap_2way	SN	FBA	2-Way Mir	198000	132000	0	132000	0	Enabled
range_t_pool	SN	N/A	Unknown	0	0	0	0	0	Disabled
range_pool2	SN	FBA	2-Way Mir	198000	198000	0	198000	0	Enabled
xt_dse_fba	RS	FBA	2-Way Mir	132000	66000	0	66000	0	Enabled
xt_tp_2way	TH	FBA	2-Way Mir	92160	61440	12080	49360	19	Enabled
xt_tp_raid5	TH	FBA	RAID-5 (3+1)	57480	28740	3380	25360	11	Enabled
HR_THIN_R5	TH	FBA	RAID-5 (7+1)	107520	92160	7880	84280	8	Enabled
tp_pool	TH	FBA	2-Way Mir	552420	552420	0	552420	0	Enabled
Total Tracks				3661950	3455130	23340	3431790	0	

Legend for Pool Types:

SN = Snap, RS = Rdfa DSE TH = Thin

Some notes on the example output follow:

Note that the pool state is Disabled, if all devices in the pool are disabled. The pool state is enabled, if any device is enabled.

Free Tracks is (Enabled - Used); the disabled tracks are not counted when counting free tracks.

Full % is (Used Tracks/Enabled Tracks * 100); disabled tracks are not counted.

Viewing a thin provisioning pool

To list only the Thin Provisioning pools in this output format, use the `-thin` option. The example below shows the display in MBs rather than the default cylinders:

```
symcfg -sid 343 list -pool -thin -mb
```

Symmetrix ID: 000190300343

```

                                S Y M M E T R I X   P O O L S
-----
Pool      Ty
Name      p Dev  Dev      Total  Enabled  Used    Free  Full  Sta
          e Emul Config    MBs      MBs      MBs    MBs  (%)   e
-----
xt_tp_2way TH FBA  2-Way Mir    1920    1920    377    1542   19  Ena
xt_tp_raid5 TH FBA  RAID-5(3+1)    898     898    105     792   11  Ena
HR_THIN_R5 TH FBA  RAID-5(7+1)   2880    2880    246    2633    8  Ena
tp_pool    TH FBA  2-Way Mir   17263   17263     0   17263    0  Ena

Total
MBs                22961    22961    729    22231    3

```

Legend for Pool Types:

SN = Snap, RS = Rdfa DSE TH = Thin

The disabled DATA devices in thin provisioning pool HR_THIN_R5 can be seen by adding the option `-all` to the example, as follows:

```
symcfg -sid 343 show -pool HR_THIN_R5 -thin -all
```

Symmetrix ID: 000190300343

```

Symmetrix ID      : 000190300343
Pool Name         : HR_THIN_R5
Pool Type         : Thin
Dev Emulation     : FBA
Dev Configuration : RAID-5(7+1)
Pool State        : Enabled
# of Devices in Pool : 7
# of Enabled Devices in Pool : 6

```

Enabled Devices(6):

```

{
-----
Sym      Total      Used      Free  Full  Device  Session
Dev      GBs       GBs       GBs   (%)   State   Status
-----
022F      0.5       0.0       0.5    0  Enabled  N/A
0230      0.5       0.0       0.4    9  Enabled  N/A
0231      0.5       0.1       0.4   10  Enabled  N/A
0232      0.5       0.0       0.4    9  Enabled  N/A
0233      0.5       0.0       0.4    9  Enabled  N/A
0234      0.5       0.1       0.4   11  Enabled  N/A
-----
GBs       2.8       0.2       2.6    8
}

```

Disabled Devices(1):

```

{
-----
Sym      Total      Used      Free  Full  Device  Session
Dev      GBs       GBs       GBs   (%)   State   Status
-----
022E      0.5       0.0       0.5    0  Disabled  N/A
-----
GBs       0.5       0.0       0.5    0
}

```

<...>

Mapping devices

Note: With Enginuity 5874, when a device is masked it is automatically mapped. Refer to [Chapter 2, page 2-118](#) for more details.

You can map devices to front-end director ports, or a range of devices to consecutive addresses specifying a starting address. This is part of the Symmetrix Device Reallocation (SDR) feature.

To map a Symmetrix device to a director port, use the following command:

```
map dev SymDevName [:SymDevName]to dir DirectorNum:PortNum
[starting][target=ScsiTarget,] lun=ScsiLun
[, vbus=FibreVbus]
[, emulation=CELERRA_FBA]
```

Note: The following options are not supported with Enginuity 5874.

```
[, awwn=awwn | wwn=wwn | iscsi=iscsi]
[, masking host_lun=lun | dynamic_lun];
```

Where:

awwn — Specifies the user-given name or alias WWN of a host HBA port, if updating a VCM database. This value is for Enginuity 5773 and earlier.

dynamic_lun — Specifies to use the dynamic LUN addressing features but does not require a LUN address for each device. The LUN addresses are assigned based on what may already be in use for that host HBA. This value is for Enginuity 5773 and earlier.

emulation — Indicates that the device's emulation type is CELERRA FBA. This option is required when performing operations on a Celerra device, and indicates that you are aware that you are changing the Celerra environment.

host_lun — Specifies the LUN addresses to be used for each device that is to be added for the host HBA. This value is for Enginuity 5773 and earlier.

iscsi — Specifies the iSCSI name, if updating a VCM database. This value is for Enginuity 5773 and earlier.

lun — Specifies the SCSI logical unit number (hex value).

starting — Specifies the starting address for the range of devices.

target — Specifies the SCSI target ID (hex value).

vbus — Specifies the virtual bus (vbus) address for mapping to an FA port if using volume set addressing.

wwn — Specifies the unique 64-bit World Wide Name (WWN) identifier for an HBA port, if updating a VCM database. This value is for Enginuity 5773 and earlier.

Example: Mapping class

To map metahead (device) 0030 to director 16A, port 0, and SCSI target/lun 0, 7, enter in the command file:

```
map dev 0030 to dir 16A:0 target=0, lun=7;
```


To obtain a list of used addresses, including the next available address, use the following command:

```
symcfg list -SA all -address -available
```

To map a device 0032 to director 16A, port 0, and SCSI target/lun 0,2 and update the device masking database by specifying the WWN 20000000c920b484 of the host bus adapter (HBA) port through which a host accesses the device, enter the following command:

```
map dev 0032 to dir 16A:0 target=0, lun=2, wwn=20000000c920b484;
```

Impact on I/O

The following are restrictions/conditions to avoid impact on I/O activity:

- ◆ When mapping, there are no restrictions on I/O if adding a second path.
- ◆ After committing a `symconfigure` mapping operation, you must update the device mapping information within the host system environment. Attempting host activity with a device after it has been removed or altered, but before you have updated the host's device information, can cause host errors. For more information, refer to the *EMC Solutions Enabler SRM CLI Product Guide*.
- ◆ To update your hosts, run the utilities specified for your platform as described in [Appendix C, "Updating the Host."](#) After the host environment is updated, I/O activity can resume with the Symmetrix device.

Enable SYMAPI environment option

Environment option SYMAPI_CTRL_OF_NONVISIBLE_DEVS in the options file must be enabled (or not present in the options file) if the device is not mapped to the user host. For information about enabling this option, refer to ["Enable SYMAPI environment option" on page 29](#).

Mapping CKD devices

You can map or unmap a range of CKD devices from an EA/EF port. In addition, you can also assign PAV alias addresses for devices already mapped.

Mapping a range of devices

To map a range of devices to an EA/EF port, use the following form:

```
map dev SymDevName[:SymDevName] to dir
DirectorNum:PortNum, starting
base_address=cuu_address [mvs_ssid=nnn];
```

To map a range of devices to an EA or EF port, using the same addressing from a different port, use the following form:

```
map dev SymDevName[:SymDevName] to dir
DirectorNum:PortNum, copying dir
DirectorNum:PortNum;
```

Where:

base_address — Specifies the base address to be assigned to the first device in the mapping request. It will be incremented by one for each device in the range of devices being mapped.

Mainframe ports expect devices to be mapped in groups to form CU images. The first digit in the address is the CU image number, which can range from 0 to 0xF. The remaining two digits can range from 00 to 0xFF.

You can divide the set of 256 `cu_addresses` available in a CU image into base addresses and aliases, as is required.

mvs_ssid — When mapping a range of devices to an EA/EF port, you may need to change the current `mvs_ssid` assigned to the devices. If the devices are becoming part of an existing CU image, they will be assigned the `mvs_ssid` of the devices already mapped. If a new CU image is being formed and mapped, a new `mvs_ssid` can be assigned during the map request. It is not valid to have some devices with a particular SSID mapped and some unmapped.

copying dir — Specifies a directory from which to copy; used with Enginuity version 5671 for each EA/EF port.

Examples To map devices 01D through 023 to CU image A, where the next free base address is 20 and the SSID is 140, enter:

```
map dev 01D:023 to dir 01C:1 starting base_address=A20 mvs_ssid=140;
```

To map devices 01D through 023 to a second director port, copying the address information, enter:

```
map dev 01D:023 to dir 01C:1 copying dir 01C:0;
```

Copying the mapping of a range of devices

To copy a CU image mapping, all devices in the range must be mapped to the same CU image, or not mapped at all. Devices within the specified range that are not mapped will be ignored as long as they are not mappable (SAVE devices, DRVs, and so on). If a device in the range is mappable, the request will be rejected.

To copy a CU image mapping from one port to another, use the following form:

```
map dev SymDevName[:SymDevName] to dir DirectorNum:PortNum
copying dir DirectorNum:PortNum
```

Example To copy addresses of devices 010 through 050 from port 03C:0 to port 03D:0, enter:

```
map dev 010:050 to dir 03D:0 copying dir 03C:0
```

Assigning PAV alias addresses to mapped devices

When assigning PAV alias addresses to mapped devices, the aliases will be propagated to all director ports to which the devices are mapped. Devices within the range that are not mapped will be skipped. If any devices in the range are mapped to a different CU image than the first device, an error will be returned. If the device range has base addresses with gaps, the aliases will also have gaps.

To assign PAV alias addresses to devices mapped to EA or EF ports, use one of the following forms, according to the version of Engenuity:

- ◆ For Engenuity version 5671, use the following form:

```
add pav alias to dev SymDevName[:SymDevName], starting
alias=cuu_address;
```

- ◆ For Engenuity version 5771, use the following form:

```
add pav alias to dev SymDevName[:SymDevName], alias count=nnn;
```

Mainframe ports expect devices to be mapped in groups to form CU images. The first digit in the address is the CU image number, which can range from 0 to 0xF. The remaining two digits can range from 00 to 0xFF.

To assign a PAV alias address range to a CU image, use the following form:

```
add pav alias_range nnn:nnn to mvs_ssid = nnn;
```

Examples To add the alias A60 to device 01D, which is already mapped, enter:

```
add pav alias to dev 01D starting alias=A60;
```

To add total PAV aliases to device 37a, enter:

```
add pav alias to dev 37a , alias count=32;
```

To add a PAV alias range to a CU image using SSID 140, enter:

```
add pav alias_range addr 0080:009f to mvs_ssid=140;
```

Removing PAV alias addresses from mapped devices

To remove PAV alias addresses from devices mapped to EA or EF ports, use the following forms, according to the version of Enginuity:

- ◆ For Enginuity version 5671, use the following form:

```
remove pav alias from dev SymDevName[:SymDevName],
starting alias=cuu_address;
```

- ◆ For Enginuity version 5771, use the following form:

```
remove pav alias from dev SymDevName[:SymDevName], alias count=nnn;
```

To remove a PAV alias address range from a CU image, use the following form:

```
remove pav alias_range nnn:nnn from mvs_ssid =nnn
```

Example To remove aliases from CU image 7 so that additional devices can be mapped to it, enter:

```
remove pav alias from dev 012:052, starting alias=750;
remove pav alias from dev 012:052, starting alias=790;
```

To remove PAV alias range from device 37a, enter

```
remove pav alias from dev 37a, alias count=32;
```

To remove PAV alias range from a CU image, enter:

```
remove pav alias_range addr 0080:009f from mvs_ssid=140;
```

Unmapping devices

You can unmap devices from front-end directors ports. This is part of the Symmetrix Device Reallocation (SDR) feature.

To unmap devices from a director port, use the following form:

```
unmap dev SymDevName[:SymDevName] from dir
<ALL:ALL | ALL:PortNum |
DirectorNum:ALL | DirectorNum:PortNum>
[, emulation=CELERRA_FBA]
[, devmask_access=remove | retain];
```

Where:

emulation — Indicates that the device's emulation type is CELERRA FBA. This option is required when performing operations on a Celerra device, and indicates that you are aware that you are changing the Celerra environment.

devmask_access — Indicates whether the device masking database should be updated. The `remove` option indicates that device masking access entries for the device should be removed from the VCMDB. The `retain` option specifies that entries remain in the database. This option is for Engenuity 5773 and earlier.

Unmapping a range of devices

Unmapping can be from one or all ports. Since all devices with the same SSID must either be mapped or unmapped, you must provide an SSID when unmapping only some devices in a CU image.

To unmap a range of devices from EA or EF ports, use the following form:

```
unmap dev SymDevName:[SymDevName] from dir
< ALL:ALL | ALL:PortNum | DirectorNum:ALL | DirectorNum:PortNum
[new_ssid=nnn];
```

Examples

To unmap five devices from a CU image so that meta CKDs can replace them, and assign the devices an SSID different from the CU image, enter:

```
unmap dev 13B:13F from dir all:all, new_ssid=0160;
```

To unmap devices 0030 and 0031 from director 16A, port 0, and remove the entries from the device masking database, enter in the command file:

```
unmap dev 0030:0031 from dir 16A:0 devmask_access=remove;
```

Impact on I/O

The following are restrictions/conditions to avoid impact on I/O activity:

- ◆ When unmapping, no I/O activity is allowed on any devices in the specified mapped path. Devices must be made Not Ready or Write Disabled.

For example, to make the device Not Ready:

```
symdg create -type [REGULAR | RDF1 | RDF2] DgName
symld -g DgName -sid SymmID add dev SymDevName
symld -g DgName not_ready
```

If you are unmapping only one path to a multipathed device, you may prefer to write disable that path only:

```
symld -g DgName -SA 16A -p 0 write_disable
```

It is possible to make a device Not Ready without creating a device group:

```
symdev -sid SymmID not_ready SymDevName
```

Note: Do not use the `write_disable` argument with the `symrdf` command, as this write disables the source (R1) device(s) or the target (R2) device(s) to its/their local hosts.

- ◆ After committing a `symconfigure` mapping operation, update the device mapping information within the host system environment. Attempting host activity with a device after it has been removed or altered, but before you have updated the host's device information, can cause host errors. For more information, refer to the *EMC Solutions Enabler SRM CLI Product Guide*.

To update your hosts, run the utilities specified for your platform as described in [Appendix C, "Updating the Host."](#) After the host environment is updated, I/O activity can resume with the Symmetrix device.

Configuring dynamic RDF-capable devices

Note: Dynamic RDF is enabled by default on Symmetrix arrays running Enginuity 5874 and higher. This section applies to Symmetrix arrays running Enginuity 5773 and earlier.

Devices can be configured with `symconfigure` to be dynamic RDF-capable devices. Dynamic RDF functionality enables you to create, delete, and swap SRDF pairs while the Symmetrix array is in operation. Using dynamic RDF technology, you can establish SRDF device pairs from non-SRDF devices, and then synchronize and manage them in the same way as configured SRDF pairs.

In addition, when enabled, you can dynamically swap the RDF device designations of a specified device group using the `symrdf` command. Source R1 device(s) become target R2 device(s) and target R2 device(s) become source R1 device(s). Swaps using dynamic RDF perform faster.

Note: Dynamic swap is not supported if Enginuity versions are mixed between your local and remote Symmetrix arrays.

For information on using the devices with RDF commands, refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide*.

The following steps must be performed before dynamic RDF operations can be performed:

1. The Symmetrix must have the dynamic RDF feature enabled.
2. The devices on which you want to perform dynamic RDF operations must have the dynamic RDF attribute set.

These actions must be performed on both the local and remote Symmetrix arrays.

Enabling dynamic RDF on the device

Next, identify the non RDF devices that you want available for use as dynamic RDF devices. Create a command file that enables the dynamic RDF operations for those devices. The following are possible attributes:

```
dyn_rdf      (Using the dyn_rdf option provides the most flexibility)
dyn_rdf1_only *
dyn_rdf2_only *
```

* These options limit a device to either an R1 or R2 device, and therefore, using them prevents the ability to perform RDF swaps.

The following example shows such a file:

```
set dev 0020:002A attribute=dyn_rdf;
set dev 0040:0045 attribute=dyn_rdf;
set dev 0056:005A attribute=dyn_rdf;
```

Then, activate the option using the following command:

```
symconfigure -sid 5605 -file set_device_dyn_rdf.cmd -noprompt -v
commit
```

To view the set of devices currently available for use in dynamic RDF operations:

```
symdev list -dynamic
```

Sample output

Symmetrix ID: 000000005605

Device Name		Directors			Device		
-----		-----			-----		
Cap							
Sym	Physical	SA	:P DA	:IT Config	Attribute	Sts	(MB)
-----		-----			-----		
000A	Not Visible	???:?	02B:D0	Unprotected	Grp'd	RW	516
000B	Not Visible	???:?	15A:D0	Unprotected	Grp'd	RW	516
000C	Not Visible	???:?	01B:D0	Unprotected	Grp'd	RW	516
000D	Not Visible	???:?	16A:D0	Unprotected	N/Grp'd	RW	516
000E	Not Visible	???:?	02A:D0	Unprotected	N/Grp'd	RW	516
0012	Not Visible	???:?	01B:C1	Unprotected	Grp'd	RW	516
0013	Not Visible	???:?	16A:C1	Unprotected	Grp'd	RW	516
0014	Not Visible	???:?	02A:C1	Unprotected	Grp'd	RW	516
0015	Not Visible	???:?	15B:C1	Unprotected	N/Grp'd	RW	516
0016	Not Visible	???:?	01A:D1	Unprotected	N/Grp'd	RW	516

For information on using the devices with RDF commands, refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide*.

Setting RDF group attributes

RDF group attributes allow you to assign priorities to SRDF/A sessions, and to set the minimum amount of time before attempting an SRDF/A cycle switch.

To set RDF group attributes, use the following command:

```
set [ra | rdf] group GroupNum
[session_priority = SessionPriority] |
[minimum_cycle_time = CycleTime]
[rdfa_transmit_idle = <ENABLE | DISABLE>] |
[rdfa_dse_pool = <PoolName | NO POOL>
emulation = [fba | ckd-3390 | ckd-3380 | as400] ] |
[rdfa_dse_threshold = <% of system WP>] |
[rdfa_dse_autostart = <ENABLE | DISABLE>];
```

Where:

GroupNum — The RDF (RA) group number.

session_priority — The priority used to determine which SRDF/A sessions to drop if cache becomes full. Values range from 1 to 64, with 1 being the highest priority (last to be dropped).

minimum_cycle_time — The minimum time to wait before attempting an SRDF/A cycle switch. Values range from 5 to 59 seconds.

rdfa_transmit_idle — Indicates whether this group has transmit idle support enabled.

rdfa_dse_pool — The name of a collection of SAVE devices used for SRDF/A DSE.

emulation — The pool emulation type.

rdfa_dse_threshold — Specifies the percentage of the Symmetrix array's write pending limit. Once the cache usage of all active groups in the Symmetrix array exceeds this limit, data tracks for this group start to spill over to disks. Valid values are from 20 to 100. The default value is 50.

Note: Prior to Solutions Enabler V.70, the DSE threshold had to be set in a separate command file from the other DSE options. With Solutions Enabler V.70, you can now set all DSE options in a single configuration change command file as long as the threshold setting is the first command listed in the file.

rdfa_dse_autostart — Specifies whether SRDF/A DSE is automatically activated when SRDF/A session is activated for the group. Valid values are ENABLE or DISABLE. DISABLE is the default.

Examples To set the SRDF/A session priorities for two different RA groups, enter:

```
symconfigure commit -sid 12345 -file setup_rdfa.cmd
```

where setup_rdfa.cmd contains:

```
set ra group 24, session_priority=1;
set ra group 42, session_priority=8;
```

To set the RDF/A DSE threshold, associate an RDF group with a pool, and activate DSE automatically, enter:

```
symconfigure commit -sid 12 -file setup_dse.cmd
```

where `setup_dse.cmd` contains:

```
set rdf group 7 rdfa_dse_threshold=20
set rdf group 7 rdfa_dse_pool=r1pool, emulation=fba;
set rdf group 7 rdfa_dse_autostart=enable;
```

Note: Place the command for setting the threshold first in the file.

Enabling RDF/A

To enable SRDF/A (asynchronous) for a group, use the following form:

```
enable rdfa on ra_group=n,
  make_group_swappable=[TRUE | FALSE];
```

Disabling RDF/A

To disable SRDF/A (asynchronous) for a group, use the following form:

```
disable rdfa on ra_group=<nnn>,
  delete_support_devices=[TRUE | FALSE];
```

Adding/removing RDF mirrors

Prior to Enginuity version 5773, an SRDF device could be a source device (R1) or a target device (R2), but could not function in both roles simultaneously, unless it was a concurrent device. Solutions Enabler supports setting both Cascade and Concurrent RDF environments. This will be supported with `symconfigure`, using two new commands, `add rdf mirror` and `remove rdf mirror`.

These commands can set up both dynamic and static RDF device configurations.

The command file syntax follows:

```
add rdf mirror to dev SymDevName[:SymDevName]
    ra_group=n, mirror_type = [RDF1 | RDF2],
    remote_dev = SymDevName,
    invalidate = <invalidate_opt>,
    start_copy = [YES | NO]
    [,rdf_mode = [sync|semi|acp_wp|acp_disk|async]];

remove rdf mirror from dev SymDevName[:SymDevName],
    ra_group=n;
```

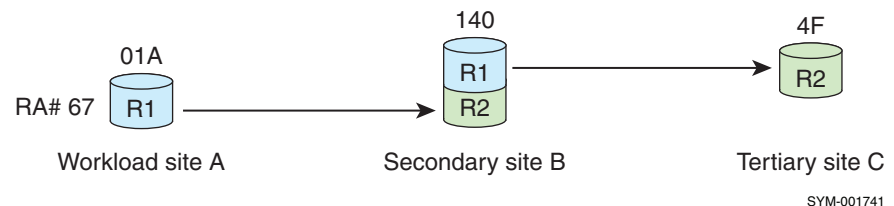
Where:

invalidate_opt — Indicates the RDF device to invalidate so that a full copy can be initiated from the remote mirror. Allowed values are R1 (invalidate the source), or R2 (invalidate the target). The value NONE is allowed if the `start_copy` option is set to NO and the devices are not dynamic RDF devices.

Note: These commands will work with any version of Enginuity.

Example

The following example shows how to add a mirror from each site.

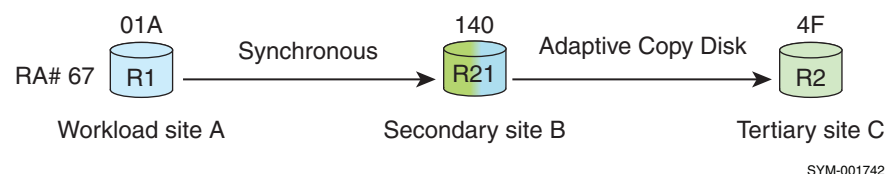


From site A, the `add mirror` command would be as follows:

```
add rdf mirror to dev 01A ra_group=67, mirror_type = RDF1,
remote_dev = 140, invalidate=R1, start_copy = YES,
rdf_mode = sync;
```

From site C, the `add mirror` command would be as follows:

```
add rdf mirror to dev 4F ra_group=67, mirror_type = RDF2,
remote_dev = 140, invalidate=R1, start_copy = YES,
rdf_mode = acp_disk;
```



Possible dual-mirrored device types include the following:

- ◆ R11 — A concurrent R1 device with 2 R1 RDF mirrors
- ◆ R21 — An RDF device with 2 RDF mirrors. One an R2 mirror, and the other an R1 mirror.
- ◆ R22 — A concurrent R2 device with 2 R2 RDF mirrors.

Add/remove mirror restrictions

R21 devices

The following restrictions apply to adding and removing mirrors for R21 devices:

- ◆ To set up a dynamic R21 device, the device must be both `dyn_rdf1` and `dyn_rdf2` capable. Use the following command to set this value:

```
set dev SymDevName[:SymDevName]
    attribute =dyn_rdf;
```

- ◆ When adding the first device to an RDF group with `-rdf_mode async`, all subsequent devices that are added to the RDF group must also be added with `-rdf_mode async`.
- ◆ If adding a dynamic RDF mirror to form an R21 device the second hop (RDF link) can only be in `async` or `ac_disk` mode.
- ◆ If a mode is not specified, the default is synchronous mode.
- ◆ When adding the first device to an RDF group with `-rdf_mode sync|semi|acp_wp|acp_disk`, subsequent devices cannot be added with `-rdf_mode async`.
- ◆ If the RDF device is a part of a consistency group, a mirror cannot be added to form an RDF21 device.
- ◆ Devices cannot be added or removed if there is an active SRDF/A session.
- ◆ RDF mirrors cannot be added to VDEVs, SAVE devices, DATA devices, and thin devices.
- ◆ The command `swap RA group` will not support swapping RDF groups with devices that have a cascaded or a concurrent RDF setup.
- ◆ The following conditions must be met when adding a remote mirror to make a device an R21 device:
 - The R21 device should be able to see its current RDF partner, i.e. the links cannot be partitioned.
 - The remote partner of the R21 cannot be an R21 device or a concurrent RDF device.

R22 devices

The following restrictions apply to adding and removing mirrors for R22 devices:

- ◆ R22 devices are only supported on Gig-E and Fibre directors.
- ◆ Creating an R22 device means to add 2 RDF mirrors. Each RDF mirror of a single device must belong to a different RDF group.
- ◆ R22 devices cannot be BCV devices.
- ◆ The two mirrors of the R22 device cannot be paired with the two R1 mirrors of a single concurrent R1 device.
- ◆ RDF mirrors cannot be added to VDEVs, SAVE devices, and DATA devices.
- ◆ An R22 device cannot be converted from static to dynamic.

- ◆ The command `swap RA group` will not support swapping RDF groups with devices that have a cascaded or a concurrent RDF setup.

Static RDF configuration restrictions

The following restrictions apply to static RDF configurations:

- ◆ Static RDF configurations can only be configured in `sync`, `ac_wp`, or `ac_disk` mode.
- ◆ Static RDF R21 -> R2 leg has to be `AC_DISK` mode.
- ◆ The `start_copy` value should be set to `no` and `invalidate` set to `NONE` if adding a remote mirror to a device leads to a concurrent RDF configuration.

Note: Currently, the `symconfigure` command enforces a restriction that if `invalidate` is set to `R1` or `R2` `start_copy` should be set to `yes`.

For information about control operations for SRDF/Cascaded Replication, refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide*.

Swapping RA groups

You can swap all devices in an RA group from target to source, using the following command:

```
swap ra group nnn,
    refresh=R1|R2, start_copy=YES|NO;
```

Where:

refresh — The RDF device (R1 source or R2 target) to refresh.

start_copy — Whether an RDF pair should be synchronized after the configuration change is committed.

Example: Swapping RA group

To swap RA group 1 from the R1 source to an R2 target group, and then refresh the R2 device, enter:

```
swap ra group 1, refresh=r2, start_copy=yes;
```

Refresh data concerns

The refresh action identifies which device does *not* hold a valid copy of the data before the swap operation begins. If you determine that the R1 holds the valid copy, the action of `refresh R2` will obtain a count of the tracks that are different on the R2 and will copy (refresh) these tracks from the R1 to the R2 device. The result will be the reverse if you use `refresh=R1` as the action.

Restrictions and I/O impact

The following are restrictions on swapping RA groups:

- ◆ All the devices in the RA group must be of the same type (RDF1 or RDF2) if FarPoint™ is enabled.
- ◆ Only one RA group may be swapped per configuration session.
- ◆ When swapping source and target attributes—no restriction on I/O to R2, but no I/O allowed to the R1 device.
- ◆ When swapping the RA group personalities that engage ESCON directors in a FarPoint connection, be aware that FarPoint buffer settings cannot be adjusted using `symconfigure`. If your FarPoint buffers are set to customized parameters other than default values, an EMC representative will need to be called to adjust the buffer settings after the swap has taken place.

Note: To swap Cascaded or Concurrent RDF groups, use the `symrdf -g DgName swap` command.

For information about swapping R1/R2 device designations, refer to [“Swapping RDF devices” on page 111](#).

Enable SYMAPI environment option

Environment option `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` in the `options` file must be enabled (or not present in the options file) if the devices are not mapped to the user host. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Swapping RDF devices

You can swap the RDF personality of the RDF device designations of a specified device group, if you have any of the following:

- ◆ An SRDF license and a Configuration Manager license.
- Or
- ◆ An SRDF license and a Configuration Manager license, and Dynamic RDF is disabled in your Symmetrix configuration.
- Or
- ◆ An SRDF license and Dynamic RDF is enabled in your Symmetrix configuration.

Source R1 device(s) become target R2 device(s) and target R2 device(s) become source R1 device(s).

Use the following form to perform an R1/R2 swap:

```
symrdf -g DgName [-force][-symforce] [-bcv|-all]
          [-bypass] [-v|-noecho] [-noprompt]
          [-i Interval] [-c Count]
swap [-refresh R1|R2]
```

For more information about this command, refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide*.

Note: A swap cannot be done between an R21 -> R2 pair on Symmetrix arrays running Enginuity 5773 because the action attempts to create an R22 device.

Note: R1/R2 swap operations are not supported for Enginuity version 5669 or later configurations where the R2 device is larger than the R1 device.

Impact on I/O

The following are restrictions/conditions in lieu of impact on I/O activity:

- ◆ When swapping source and target attributes, there are no restrictions on I/O to the R2 device, but no I/O is allowed to the R1 device.
- ◆ When swapping the RA group personalities that engage ESCON directors in a FarPoint connection, FarPoint buffer settings cannot be adjusted using `symconfigure`. If your FarPoint buffers are set to customized parameters, other than default values, an EMC Representative will need to be called to adjust the buffer settings after the swap has taken place.

Swap preparation

If your RDF devices are enabled for dynamic swap, ensure that both the R1 and R2 devices are in the same state, either failed over or suspended with both sides write-disabled to the source host, before attempting a swap.

Enable SYMAPI environment option

Environment option `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` in the `options` file must be enabled (or not present in the options file) if the devices are not mapped to the user host. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Setting port characteristics

To set the port characteristics of a specified director, use the following form:

```
set port DirectorNum:PortNum
[FlagName=enable|disable
[,FlagName=enable|disable]]
gige primary_ip_address=IPAddress
    primary_netmask=IPAddress,
    default_gateway=IPAddress,
    isns_ip_address=IPAddress
    primary_ipv6_address=IPAddress,
    primary_ipv6_prefix=<0 -128>,
[fa_loop_id=integer] [hostname=HostName];
```

To reconfigure an existing port by copying the port flag settings of another port, use the following form:

```
set port DirectorNum:PortNum,
copying port DirectorNum:PortNum;
```

When copying the port flag settings from one port to another, both ports must be the same type (Gig-E to Gig-E, FA to FA, and so on). When copying Gig-E ports, IP addresses will not be copied.

Note: Before you can copy the port flags from one port to another, you must first take the port being copied offline with the `symcfg offline` command.

Where:

FlagName — A SCSI or fibre port flag. Possible values for the SCSI protocol flags are in [Table 12 on page 113](#), and the values for the fibre protocol flags are in [Table 13 on page 115](#).



CAUTION

Incorrectly changing the port flags can render your storage system inaccessible. Be sure of your needs before resetting these flags.

gige — Indicates that one or more network address values are going to be specified for a front-end Gig-E director. Addresses should use the Internet standard dot notation.

primary_ip_address — The IP address for a front-end Gig-E port.



CAUTION

When changing the primary IP address, be sure to update all connected hosts with the new address.

primary_netmask — The IP netmask for a front-end Gig-E port.

default_gateway — The gateway or router address for a front-end Gig-E port.

isns_ip_address — The IP address for the Internet Storage Name Service (ISNS) associated with a front-end Gig-E port.

primary_ipv6_address — The IPv6 address for the front-end Gig-E port.

primary_ipv6_prefix — The IPv6 mask prefix for a front-end Gig-E port. The value can be 0 - 128, indicating the number of initial bits in the subnet that are identical.

fa_loop_id — The FA director loop ID (arbitrated loop physical address). (0 - 125) (Hard Addressing must be enabled.) Not applicable for Gig-E ports.

hostname — 12-character hostname.

Table 12 **SCSI protocol port flags (page 1 of 2)**

SCSI protocol flags	Description
Auto_Busy ^a	When enabled specifically for Unisys A-series platforms only, this flag enables the auto-busy mechanism so that the Symmetrix array returns a Busy to all Unisys host requests.
Avoid_Force_Negotiate ^a	When enabled for Sequent V4.2.3 and lower, the Symmetrix array never initiates negotiations. Normal Symmetrix behavior is to initiate negotiations after an offline-to-online transition. This is for hosts that do not handle negotiations.
Avoid_Reset_Broadcast ^a	When enabled, a SCSI bus reset only occurs to the port that received the reset (not broadcast to all channels).
Command_Reordering ^a	When enabled with Tag Command Queueing in use, the incoming SCSI commands become reordered to Simple Queueing. The default is enabled and should only be disabled upon a request from EMC.
Common_Serial_Number	This flag should be enabled for multipath configurations or hosts that need a unique serial number to determine which paths lead to the same device.
Cyl_Count_In_Name ^{ab}	When this flag is enabled, the Symmetrix with the specified port embeds the cylinder count into the product ID returned in the SCSI Inquiry command. Enabled for Pyramid only when it is desirable to embed the Symmetrix support into the Pyramid kernel.
Disable_False_Disconnect ^{ab}	When enabled for debugging, this flag prevents the port from performing a False Disconnect operation. (Default is disabled and currently, you cannot change this flag.)
Disable_Interleaved_Cmds ^a	When enabled (always), metavolume command interleaving is being supported. This allows multiple metamembers to operate at the same time on the same volume.
Disable_Mini_Q ^{ab}	When enabled for debugging, this flag disables the use of the Mini Queue on the port. (Default is disabled and currently, you cannot change this flag.)
Disable_Q_Reset_on_UA	When enabled, a Unit Attention (UA) that is propagated from another director does not flush the queue for this device on this director. Used for hosts that do not expect the queue to be flushed on a 0629 sense (only on a Hard Reset).
Disable_Ultra ^{ab}	When enabled, this flag disables Ultra SCSI on an Ultra capable SA port. (Default is disabled and currently, you cannot change this flag.)
Environ_Set	When enabled, this flag enables the environmental error reporting by the Symmetrix to the host on the specific port.
HP3000_Mode ^c	When enabled for HP MPE 5.0 and Enginuity version 5062 and lower, this flag causes the Symmetrix port to return a SCSI Busy state instead of a 0B44 sense code when an xx3C error occurs.
Linked_Commands ^{ab}	When enabled, this flag enables support of SCSI linked commands. It allows a host to chain SCSI commands in a manner similar to mainframe Channel Command Words (CCWs). (Default is enabled, and currently, you cannot change this flag.)
Negotiate_Reset ^d	When enabled for AS/400 hosts, this flag forces a SCSI negotiation by the Symmetrix array after: • A SCSI reset • An error • A bus device reset This flag is used for AS/400 systems only (default is off).

Table 12 SCSI protocol port flags (page 2 of 2)

SCSI protocol flags	Description
PBAY_Monitor ^a	For the Sequent platforms only to allow emulation of the Sequent PBAY. When enabled, this flag enables low-level polling of the SCSI bus in order to intercept the nonstandard SCSI operations required for a Sequent PBAY disk subsystem. Must be used for the Sequent cluster operation for the Symmetry system for Sequent V4.2.x operating systems only. Must not be used on versions higher than V4.2.x or for any NUMA-Q systems and also not used for Fibre Channel.
SCSI_3	When enabled, the Inquiry data is altered when returned by any device on the port to report that the Symmetrix supports SCSI 3 protocol. When this flag is disabled, the SCSI 2 protocol is supported.
SCSI_Support	When enabled, this flag provides a stricter compliance with SCSI standards for managing device identifiers, multi-port targets, unit attention reports, and the absence of a device at LUN 0.
Sequent ^c	When enabled, the Symmetrix on the specified port does not force wide or synchronous negotiations, and sets the task timeout window to be 15 seconds before aborting a process. In addition, a busy status is returned instead of an 0B44H when aborting a command on a timeout.
Server_On_AS400	When enabled for AS/400 platforms, this flag indicates that the port is to behave as a server, returning server inquiry data, rather than AS/400 data.
Set_Qerr ^a	This flag should be enabled for SGI platforms only to flush the queue on a contingent allegiance condition (CAC). Must be used for V5.3 and V6.2 SGI operating systems and cluster environments. Not used on versions later than V6.2.
Siemens ^c	For Siemens R-Series platforms only. When this flag is enabled for Siemens, the Symmetrix array returns in the sense data error 0B48, instead of 0B44 for normal behavior.
Soft_Reset	When enabled for a Bull/GCOS-7 host, the Symmetrix port supports the SCSI Soft Reset option.
SPC2_Protocol_Version ^e	This flag should be enabled (default) in a Windows 2003 environment running Microsoft HCT test version 12.1. When setting this flag, the port must be offline. Note: Reboot the host after setting this flag.
Sunapee ^c	When enabled for Sun PDB clusters, this flag enables the Sunapee option on the port.
Sync_Transfer ^{ab}	When enabled, this flag enables SCSI synchronous negotiations. (Default is enabled, and currently, you cannot change this flag.)
Tagged_Commands ^{ab}	When enabled, this flag enables support for tagged commands. (Default is enabled, and currently, you cannot change this flag.)
Wide_Transfer ^{ab}	When enabled, this flag enables SCSI Wide operation. (Default is enabled, and currently, you cannot change this flag.)

- a. Obsolete in Enginuity version 68 and higher.
- b. Not available for host-based configuration changes.
- c. Obsolete in Enginuity 5874 and higher.
- d. Can only be set on SA ports with Enginuity 5568 and higher.
- e. Requires Enginuity version 5670 and higher.

Table 13 Fibre protocol port flags

Fibre protocol flags	Description
AC LX	When enabled, allows storage provisioning using Initiator Groups. This flag is applicable for Engenuity 5874 and higher.
AS400	This flag should be enabled for any AS400 (iSeries) hosts connecting to the port.
Auto_Negotiate	When enabled, allows two fibre ports to <i>handshake</i> and settle on an optimal speed for data transfer.
Class_2_Service ^{a,b}	This flag should be enabled for a Class 2 fibre protocol connection that requires an acknowledgement for each frame transmitted. (You cannot change this flag .)
Disk_Array ^a	When enabled (default), the port is represented as a disk array. This port information appears in the Inquiry data.
Generic_VSA ^{ab}	When enabled, the generic volume set addressing mode is selected. GVSA mode allows hexadecimal addressing. (You cannot change this flag .)
Global__3rdParty_Logout ^a	When enabled (default), an extension is provided to the existing third-party logout required by the standard. In addition to logging out the hosts who are logged in to the port receiving the third-party logout, the logout propagates to other Symmetrix fibre ports that share volumes with the port that had received the logout.
Hard_Address ^a	When enabled (default), the FA director attempts to get the loop_id specified when it initializes on the loop (hard-assigned addressing). When disabled, soft addressing is being used.
Init_Point_to_Point ^c	When enabled, specifies a point-to-point (direct or switched) topology in the initialization sequence. When disabled (default), it is initialized as an arbitrated loop.
Non_Participating ^c	When enabled along with the Hard_Address flag, the Fibre Channel director only uses hard-assigned addressing when it initializes on the loop. Otherwise, soft-assigned addressing is used during loop initialization (the default).
OpenVMS	Enabled for an OpenVMS fibre connection.
Unique_WWN ^c	When enabled (default) for all environment configuration changes and new environments to ensure unique World Wide Names (WWN) within the fibre environment (uses Symmetrix serial numbers and port numbers). When disabled, you don't have to change WWNs.
VCM_State ^d	Enabled for device masking or the Volume Logix software, which provides volume configuration management controls to handle access to Symmetrix devices. (Disabled is the default.) This flag is only applicable for Engenuity 5773 and earlier.
Volume_Set_Address ^a	When enabled along with the Disk_Array flag for HP-UX hosts, the volume set addressing mode is selected. VSA mode allows octal addressing.

a. Obsolete in Engenuity version 68 and higher.

b. Not available for host-based configuration changes.

c. Not available for Gig-E ports.

d. Obsolete in Engenuity 5874 and higher.

Example: Setting port characteristics

To turn on hard-assigned addressing and set the FA loop ID to 3 for director 03A, port 0, enter:

```
set port 03A:0 Hard_Address=enable, fa_loop_id=3;
```

Impact on I/O

When setting port attributes, it is recommended that you temporarily suspend I/O activity to the effected ports during this operation.

Enable SYMAPI environment option

Environment option SYMAPI_CTRL_OF_NONVISIBLE_DEVS in the `options` file must be enabled (or not present in the options file) if there is no device from the local host mapped to this port. For information about enabling this option, refer to [“Enable SYMAPI environment option” on page 29](#).

Viewing port flags

You can display the possible port flags and their current status with the following command:

```
symcfg -SA all [-P #] -v list
```

The following is a partial example of output:

```
Director Identification: SA-14B
  Director Type           : SCSI
  Director SCSI Capability : N/A
  Director Status         : Online

  Number of Director Ports : 2
  Director Port Status     : [ON, ON, NA, NA]

  Director Symbolic Number : 14B
  Director Numeric Number  : 30
  Director Slot Number     : 14

Port: 0

  SCSI Flags
  {
    Tagged_Commands      : Enabled
    Linked_Commands      : Enabled
    Sync_Transfer        : Enabled
    Wide_Transfer        : Enabled
    Negotiate_Reset      : Disabled
    Soft_Reset           : Disabled
    .
    .
    .
```

To view the settings of port flags that you cannot modify, use the following command:

```
symcfg -sid SymmID -FA # -p # list -v -all
```

This chapter explains how to confine host access to Symmetrix devices using Autoprovisioning Groups and the `symaccess` command of the SYMCLI. The chapter covers the following topics:

◆ Autoprovisioning Groups.....	118
◆ Discovering host HBAs	120
◆ Creating groups and views.....	121
◆ Managing masking views.....	126
◆ Managing storage groups	128
◆ Managing port groups.....	130
◆ Managing initiator groups	133
◆ Displaying Autoprovisioning information.....	137
◆ Autoprovisioning Groups examples	139

Autoprovisioning Groups

Solutions Enabler V7.0 with Enginuity 5874, introduces an easier, faster way to provision storage. Autoprovisioning Groups allow storage administrators to create groups of host initiators, front-end ports, and logical devices. These groups are then associated to form a masking view, from which all controls are managed.

Note: Autoprovisioning Groups are not supported on Symmetrix DMX arrays running Enginuity 5773 and earlier. Storage administrators should continue to use the `symmask` and `symmaskdb` commands to mask devices in Symmetrix DMX arrays.

What's different?

Most of the applications running on Symmetrix arrays require a fault tolerant environment, with multiple paths to devices, and clusters of servers. An easier way was required to mask multiple devices, ports, and HBAs. Previous versions of Solutions Enabler required one command for each initiator/port combination through which the devices would be accessed.

A new command, `symaccess`, provides all the storage provisioning requirements for Symmetrix V-Max arrays running Enginuity 5874.

With `symaccess`, the following device masking functions are no longer supported:

- ◆ There is no device masking database. There is no initialization, backup, or restore. With the `symaccess` command, all groups and views are backed up to a file, and can be restored from a backup file.
- ◆ Volume visibility and the LUN offset on an initiator are no longer supported.
- ◆ Heterogeneous host types are no longer supported. The HBA flags option is supported.

How it works

Storage provisioning with `symaccess` allows you to create a group of devices, a group of director ports, a group of host initiators, and with one command, associate them in what is called a *masking view*. Once a masking view exists, devices, ports, and initiators can be easily added or removed from their respective groups.

This feature reduces the number of commands needed for masking devices, and allows for easy management of the masking view.

The `symaccess` command is used to create and manage the groups and views. This command also contains some features similar to those found in the `symmask` and `symmaskdb` commands (a login history command and initiator attributes).

Note: The `symaccess` command is not supported on Symmetrix DMX arrays.

The steps for creating a masking view are:

1. Create a storage group (one or more devices).
2. Create a port group (one or more director/port combinations).
3. Create an initiator group (one or more host WWNs or iSCSIs).
4. Create a masking view containing the storage group, port group, and initiator group.

When a masking view is created, the devices are automatically masked and mapped.

[Figure 8](#) shows a masking view.

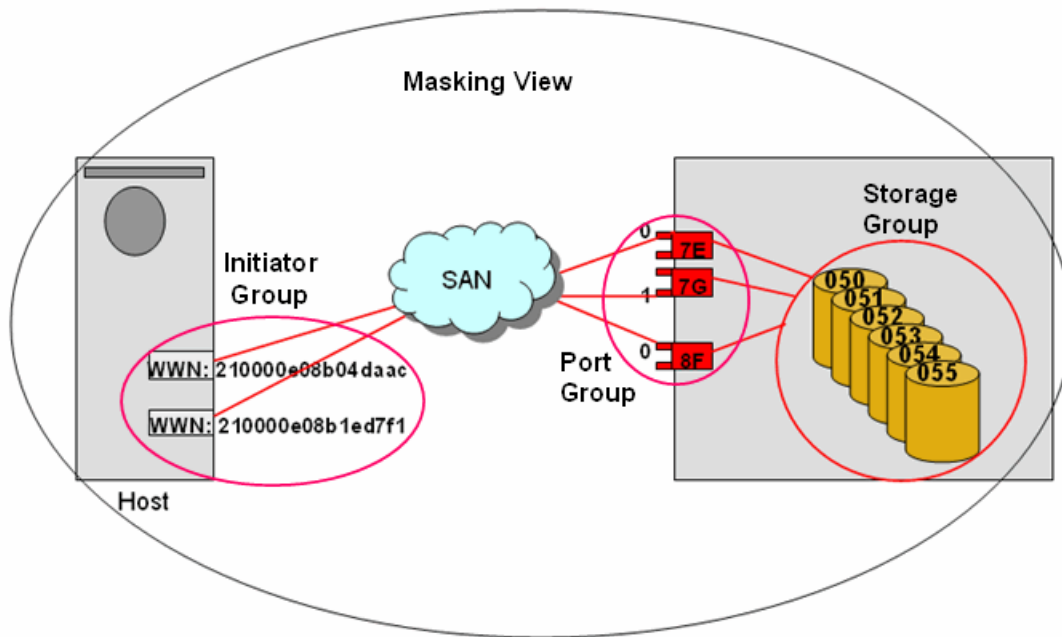


Figure 8 Masking view overview

After the masking view is created, any objects (devices, ports, initiators) added to a group automatically become part of the masking view.

This chapter covers the following topics:

[“Discovering host HBAs” on page 120](#)

[“Creating groups and views” on page 121](#)

[“Managing masking views” on page 126](#)

[“Managing storage groups” on page 128](#)

[“Managing port groups” on page 130](#)

[“Managing initiator groups” on page 133](#)

[“Displaying Autoprovisioning information” on page 137](#)

[“Autoprovisioning Groups examples” on page 139](#)

Discovering host HBAs

During the initial setup, an administrator runs `symaccess discover` on the controlling host to search the environment for Symmetrix devices on each HBA by using the following command:

```
symaccess discover hba
```

Note: It is assumed that the host from which these commands are run has access to the Symmetrix devices.

When the `symmask discover` finds a host HBA, it reads the login history table and performs the following:

1. Creates an ASCII alias and writes it to the login history table.

Note: There is a `-rename` option that can be used with this command to force the discovered hostname/HBA name (or IP address) to be written to the login history table and the initiator group.

2. Prints the initiator identifier (WWN/iSCSI) of the HBAs that are connected to the masked channel and Symmetrix array.

The `symaccess discover` command sends information about this connection back to its host system. The `discover` command is the primary mechanism by which hosts other than the control station can learn about their paths to the Symmetrix array. This information can be displayed using the `symaccess list hba` command.

Note: An iSCSI initiator can log in to the array until the first masking view is created on that array. After the first masking view is created, an iSCSI initiator must belong to a masking view before it is able to log in to the array.

Creating groups and views

Use the `symaccess` command for:

[“Creating storage groups” on page 121](#)

[“Creating port groups” on page 122](#)

[“Creating initiator groups” on page 123](#)

[“Creating a masking view” on page 124](#)

The management commands for storage groups, port groups, initiator groups, and masking views are explained later in this chapter.

Creating storage groups

You can create a storage group using a range of devices, device names, device group devices, or a device file. The `symaccess` syntax for creating a storage group is:

```
symaccess create -name GroupName -type storage
[ < [-reserve_id ResvID[, ResvID[, ResvID...]]]
  devs SymDevStart:SymDevEnd | SymDevName, SymDevName, SymDevName... > |
  <-g DgName [-std] [-bcv] [-vdev] [-tgt] > |
  < -file DeviceFileName [src] [tgt] > ]
```

You can add devices at the storage group creation time, but those devices cannot have LUNs assigned to them. The LUNs are assigned by the Symmetrix array when the masking view is created.

Example

In the following example, a range of devices containing storage for a critical application are put into a newly created storage group named `SG_1`:

```
symaccess create -sid 458 -name SG_1 -type storage devs 050:055
```

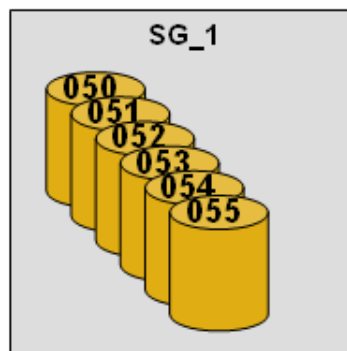


Figure 9 Storage group SG_1

Storage group names can be up to 64 characters, and are not case sensitive. Group names must be unique per group *type*, but different group types can share the same name. For example, a storage group, a port group, and an initiator group can all have the name `Financial_DB`. However, two storage groups cannot be named `Financial_DB`.

Device reservations will be enforced whenever devices are being added to a storage group.

The following sections provide more information about storage groups:

[“Managing storage groups” on page 128](#) explains how to add and remove devices, rename a storage group, and delete a storage group.

[“Displaying Autoprovisioning information” on page 137](#) explains the masking view output.

Creating port groups

Port groups may contain any number of valid front-end ports. A port can belong to more than one port group.

Only Fibre and Gig-E ports on front-end directors will be allowed to be added to a port group. Port groups can have mixed port types.

Ports must have the ACLX flag enabled to be added to a port group. Refer to [“Setting port characteristics” on page 112](#) for information about setting a port flag.

```
create -name GroupName -type port [-dirport Dir:Port [,Dir:Port...]]
```

Example In the following example, a new port group, PG_1 is created containing three front-end ports:

```
symaccess create -sid 458 -name PG_1 -type port -dirport 7E:0, 7G:1, 8F:0
```

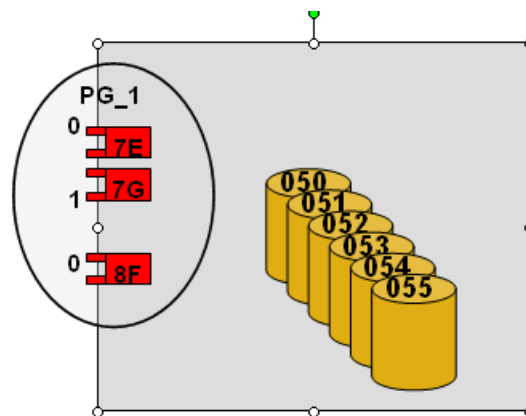


Figure 10 Port group PG_1

The following sections provide more information about port groups:

[“Managing port groups” on page 130](#) explains how to add and remove ports, rename a port group, and delete a port group.

[“Displaying Autoprovisioning information” on page 137](#) explains the masking view output.

Creating initiator groups

An initiator group is a container of one or more host initiators (Fibre or iSCSI). Each initiator group can contain up to 32 entries. An initiator group may also include the name of another initiator group to allow the groups to be cascaded to a depth of one. An HBA may only belong to one group, but may have masking views for both an upper and lower group if cascaded.

You can create an initiator group using the HBA's WWN, iSCSI, a file containing WWNs or iSCSI names, or another initiator group name. The `symaccess` syntax for creating an initiator group is:

```
symaccess -sid SymmID
```

```
create -name GroupName -type initiator [ -wwn wwn | -iscsi iscsi -file  
InitiatorFilename | -ig InitiatorGroupName ]
```

Example In the following example, an initiator group, IG_1 is created containing two WWNs:

```
symaccess create -sid 458 -name IG_1 -type initiator -file IG_1
```

Where the file IG_1 contains:

```
wwn:210000e08b04daac  
wwn:210000e08b1ed7f1
```

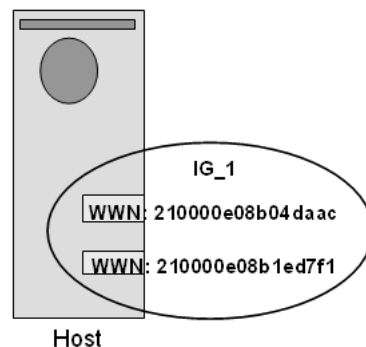


Figure 11 Initiator group IG_1

The following sections provide more information about initiator groups:

[“Managing initiator groups” on page 133](#) explains how to add and remove initiators, rename an initiator group, and delete an initiator group.

[“Displaying Autoprovisioning information” on page 137](#) explains the masking view output.

Creating a masking view

A masking view is a container of a storage group, a port group, and an initiator group. When you create a masking view, the devices in the storage group become visible to the host. The devices are masked and mapped automatically.

Volume dynamic addressing is enabled by default. The Symmetrix array assigns the next available LUN address on the FA port when the masking view is created.

The syntax for creating a masking view is as follows:

```
create view -name ViewName -sg StorageGroupName
           -pg PortGroupName -ig InitiatorGroupName
           [ < [-reserve_id ResvID[, ResvID[, ResvID...]]]
           [-lun Addr]
```

The groups being used must already exist and contain some entries so that a complete view can actually be created.

Example In the following example, a masking view, MV_1, is created containing the previously-created storage group, port group, and initiator group:

```
symaccess create view -name MV_1 -sg SG_1 -pg PG_1 -ig IG_1
```

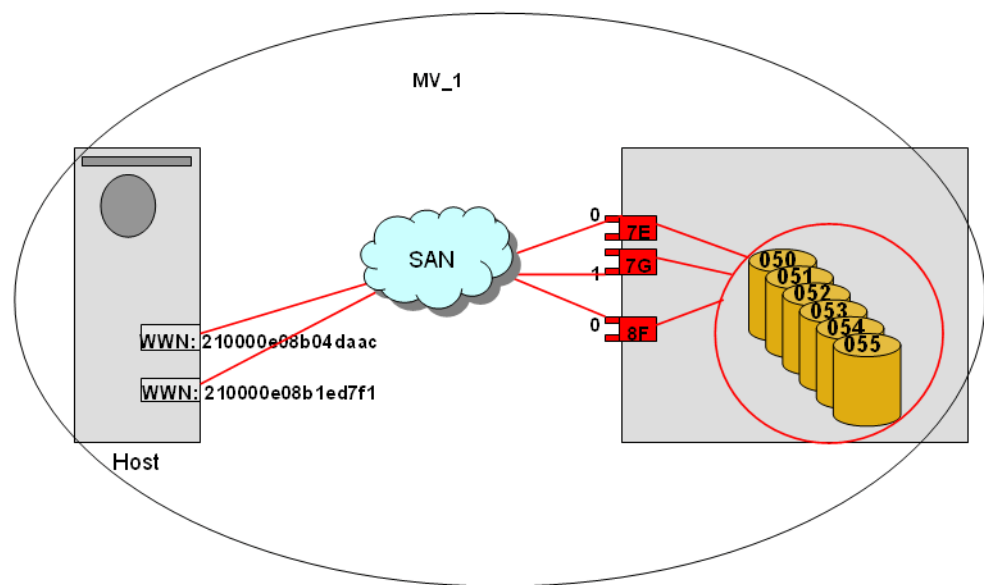


Figure 12 Masking view MV_1

The good news comes after you have your masking view. If you need to add additional storage for your application, you can add devices to the storage group and they will be automatically masked and mapped in the masking view.

The same principle applies for adding additional front-end ports or host initiators.

If you have storage that you want to manage from the same host, you can create a second storage group and then create a masking view with the same port group and initiator group.

Only a single starting LUN may be supplied when the view is created. Supplying this value is optional and the corresponding input flag should be supplied when it's given.

The masking view shown in [Figure 12](#) is a simple example. In a clustered environment, some devices may be seen by the entire cluster, but gatekeeper devices may only need to be seen by individual hosts. To show the flexibility of the storage provisioning functionality, the next section adds additional groups and another masking view to the example.

In this example, two gatekeeper devices are combined into storage group SG_2:

```
create -sid 458 -name SG_2 -type storage devs 087,088
```

One of the ports in the port group PG_1 is placed in a single port group PG_2:

```
create -sid 458 -name PG_2 -type port -dirport 7E:0
```

One of the host initiators is placed in the initiator group IG_2:

```
create -sid 458 -name IG_2 -type initiator -wwn 210000e08b04daac
```

The masking view GKV_2 is created, with the new groups, as follows:

```
create view -name GKV_2 -sg SG_2 -pg PG_2 -ig IG_2
```

[Figure 13](#) shows the groups contained in GKV_2 in the dotted line circles.

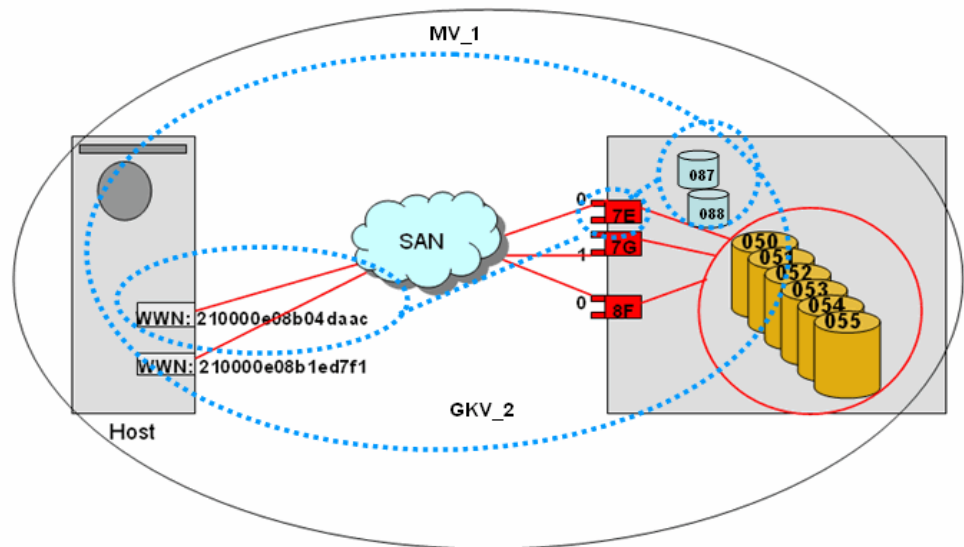


Figure 13 Two masking views

The following sections provide more information about masking views:

[“Managing masking views” on page 126](#) explains how to rename a masking view and delete a masking view.

[“Displaying Autoprovisioning information” on page 137](#) explains the masking view output.

Managing masking views

This section explains how to perform the following actions:

[“Deleting masking views”](#)

[“Naming groups and views”](#)

[“Backing up and restoring views”](#)

Deleting masking views

When a masking view is deleted, all the groups contained in the masking view will remain intact. Any device reservations will continue to be enforced when a masking view is deleted.

To delete a masking view, use the following form:

```
symaccess -sid SymmID delete view -name ViewName
```

For example, to delete masking view `MV_1`, on Symmetrix 458, enter:

```
symaccess -sid 458 delete view -name mv_1
```

Naming groups and views

Names for storage groups, port groups, initiator groups, and masking views can be up to 64 characters in length. Names are not case sensitive. They must be unique per group *type*. In other words, you can have the same name for a storage group, port group, and initiator group, but you can't use the same name for two storage groups within the same array.

Renaming

Use the `rename -new_name` command to rename a storage group, port group, or an initiator group, using the following form:

```
symaccess -sid SymmID rename -name GroupName -type <storage | port | initiator>  
-new_name NewGroupName
```

Use the `rename view -new_name` command to rename a masking view, using the following form:

```
symaccess -sid SymmID rename view -name ViewName -new_name NewViewName
```

If the new name already exists, an error returns.

Cascaded initiator groups

The renaming of a group will be propagated to the higher group if the group is cascaded.

Backing up and restoring views

The masking views, including storage groups, port groups, and initiator groups can be backed up to a file, and restored from the backup file.

Use the following syntax to backup the masking views for a Symmetrix array to a file:

```
symaccess -sid SymmID -f BackupFilename [-noprompt]  
  
backup
```

For example, to back up the masking views on Symmetrix 123 and save the data to a file called `masking_dec5`, enter:

```
symaccess -sid 123-f Masking_dec5 backup
```

Use the following syntax to restore a backup file:

```
symaccess -sid 123-f Masking_dec5 restore
```

Use the `-noprompt` option to eliminate the prompt for confirmation of the operation.

Managing storage groups

After creating a storage group, as explained in [“Creating storage groups” on page 121](#), devices can be added to and removed from the storage group. Other actions on a storage group include renaming the group and deleting the group. This section explains how to perform all these actions using the `symaccess` command.

Adding devices

A storage group can contain up to 4k Symmetrix device numbers, and devices can belong to more than one storage group. When adding devices, specify the device names, a range of devices, a list of devices in a device group, or devices in a device file.

To add devices to an existing storage group, use the following form:

```
symaccess -sid SymmID -name GroupName -type storage
          [-reserve_id ResvID [, ResvID [, ResvID...]]]

add devs SymDevStart:SymDevEnd [-lun Addr] | SymDevName [-lun Addr] |
SymDevName, SymDevName, SymDevName...
[-lun Addr | -lun Addr, Addr, Addr...]

add -g DgName [-std] [-bcv] [-vdev] [-tgt] [-lun addr]

add -file DeviceFileName [src] [tgt] [-lun addr]
```

Device reservations will continue to be enforced when they are added to a storage group.

The next section provides examples of adding devices to a storage group.

Examples To add devices 089, 090, and 091 to storage group SG_Prod on Symmetrix array 458, enter:

```
symaccess -sid 458 -name SG_Prod -type storage add devs 089,090,091
```

To add the BCV devices in device group Prog2 to storage group SG_Prod on Symmetrix array 458, enter:

```
symaccess -sid 458 -name SG_Prod -type storage add -g Prog2 -bcv
```

To add the target devices in a device file ProdFile to storage group SG_Prod on Symmetrix array 458, enter:

```
symaccess -sid 458 -name SG_Prod -type storage add -file ProdFile tgt
```

Where ProdFile contains:

```
011      031
012      033
051      076
094      077
```

Devices 031, 033, 076, and 077 are added to SG_Prod.

Designating the LUN address

When devices are added at the storage group creation time, do not specify a LUN address. The LUN address is determined when the masking view is created.

LUN addresses should only be supplied if the storage group is already contained within a view. If the LUN address is not specified, the Symmetrix array will assign the LUN address.

Removing devices

A storage group should not be deleted until all the devices have been removed. To remove devices from a storage group, use the following form:

```
symaccess -sid SymmID -name GroupName -type storage
    [-reserve_id ResvID[,ResvID[,ResvID...]]] [-unmap]

remove devs SymDevStart:SymDevEnd | SymDevName |
    SymDevName,SymDevName,SymDevName...

remove -g DgName [-std] [-bcv] [-vdev] [-tgt] [-lun addr]

remove -file DeviceFileName [src] [tgt] [-lun addr]
```

Example To remove the BCV devices in device group *Prog2* from storage group *SG_Prod* on Symmetrix array 458, enter:

```
symaccess -sid 458 -name SG_Prod -type storage remove -g Prog2 -bcv
```

Note: A storage group cannot be completely emptied if it is associated with a masking view.

Deleting storage groups

A storage group should be empty before being deleted, unless using the `-force` option.

```
delete <view -name ViewName [-unmap] > |
    < -name GroupName -type
    <storage | port | initiator> [-force] > [-noprompt]
```

For example, to delete storage group *SG_1* from masking view *MV1* on Symmetrix 458, enter:

```
symaccess -sid 458 -view mv1 -name sg_1 -type storage delete
```

Note: A storage group cannot be deleted if it is associated with a masking view.

Use the `-unmap` option to unmap the devices from the ports contained in the port group of the view.

Note: The devices must be in the NOT READY or WRITE DISABLED state to use the `-unmap` option.

Managing port groups

Port groups contain director and port identification and belong to a masking view. Ports can be added and removed. When a port group is no longer associated with a masking view, it can be deleted.

In addition, CHAP authentication can be enabled and disabled on port groups. This section explains the following port-related tasks:

[“Adding ports” on page 130](#)

[“Removing ports” on page 130](#)

[“Deleting port groups” on page 130](#)

[“Using CHAP authentication” on page 131](#)

[“Locking down a Fibre Channel ID” on page 131](#)

Adding ports

Ports can be added to an existing port group by specifying the name of the group and indicating the group type and the director port combinations, using the following syntax:

```
symaccess -sid SymmID -name GroupName -type port
          -dirport Dir:Port[,Dir:Port[,Dir:Port...]]
```

add

To add port 1 of Fibre director 16D to the existing port group PG_1 on Symmetrix 245, enter:

```
symaccess -sid 245 -name PG_1 -type port add -dirport 16D:1
```

Removing ports

```
symaccess -sid SymmID -name GroupName -type port
          -dirport Dir:Port[,Dir:Port[,Dir:Port...]]
```

remove

To remove port 1 of Fibre director 16D from the existing port group PG_1 on Symmetrix 245, enter:

```
symaccess -sid 245 -name PG_1 -type port remove -dirport 16D:1
```

Note: A port group cannot be completely emptied if it is associated with a masking view.

Deleting port groups

To delete a port group, use the following syntax:

```
delete <view -name ViewName> [-force] |
      -name GroupName -type <storage | port | initiator > [-noprompt]
```

Note: A port group cannot be deleted if it is associated with a masking view.

To delete port group PG_1 on Symmetrix 245, enter:

```
symaccess -sid -name PG_1 -type port delete
```

Using CHAP authentication

CHAP (Challenge Handshake Authentication Protocol) allows you to manage a credential name and a CHAP secret, which are similar to a username and a password, though more secure than the standard Password Authentication Procedure (PAP).

To enable CHAP on an iSCSI initiator, use the following form:

```
symaccess -sid SymmID -iscsi iscsi enable chap
```

To set the CHAP credential and secret on a director and port, use the following form:

```
symaccess -sid SymmID -dirport Dir:Port
  set chap -cred Credential -secret Secret
```

To enable CHAP on a specific director and port, use the following form:

```
symaccess -sid SymmID [-dirport Dir:Port]
  enable chap
```

To disable CHAP on a specific director and port, use the following form:

```
symaccess -sid SymmID [-dirport Dir:Port]
  disable chap
```

Locking down a Fibre Channel ID

Fibre Channel ID (FCID) lockdown is a security feature that limits host device access by adding Fibre Channel ID information of a switch within a fabric to device access records in the login history table. This feature handles WWN spoofing and the threat it poses to your networked systems in a shared (same director port) storage port configuration.

For example, to implement the Fibre Channel ID lockdown feature on Fibre Channel 021300 for director 16A, port 0, enter:

```
symaccess -sid 018 set lockdown on 021300
```

This feature lets you set the Fibre Channel ID (FCID) of the WWN of the HBA you want to protect. The FCID is then added to the database record for the WWN of the specified HBA with the specified director and is locked. Once a Fibre Channel ID is locked, no user with a spoofed WWN can log in. If a user with a spoofed WWN is already logged in, that user loses all access through that HBA.



CAUTION

When an HBA logs in to a director port, the Fibre Channel ID accompanies it, telling the director port where to send its response. By specifying Fibre Channel ID information of the switch, the valid physical path through the SAN for a particular HBA is locked down. Only an HBA with a Fibre Channel ID that matches the FCID specified in the device masking record is able to log in to the storage port. It is recommended that at least two HBAs be available on the administrator host. If one HBA becomes locked out, the host will have access through the other HBA and can correct the record in the database.

Lockdown steps

To find the Fibre Channel ID, lock it down, verify that it is locked down, and then force the change to take effect, use the following procedure:

1. Find the WWN. If the device is visible, run `symaccess list hba` to find the device path of the HBA you want to protect.

2. Find the Fibre Channel ID value by referring to [“Finding the FCID of a switch” on page 172](#).
3. Run `symaccess set lockdown` to on with the FCID of the Fibre Channel ID you found in step 2.
4. To finalize the change, either reboot the host or pull the cable from the director and then replace the cable.

Managing initiator groups

This section explains how to manage initiator groups. The following topics are included:

[“Adding initiators” on page 133](#)

[“Removing initiators” on page 134](#)

[“Deleting initiator groups” on page 134](#)

[“Setting HBA flags” on page 134](#)

[“Replacing an HBA” on page 136](#)

Adding initiators

Initiators can be added to an existing initiator group by specifying the initiator type (`-wwn` or `-iscsi`), the initiator group name, or by using an input file. The following is the syntax for adding one or more initiators to an initiator group:

```
symaccess -sid SymmID -name GroupName -type initiator
          -wwn wwn | -iscsi iscsi | -ig InitiatorGroupName |
          -f InitiatorFilename
```

add

Individual initiators

For example, to add initiator `-wwn 10000000c94ef69c` to the initiator group `IG_1` on Symmetrix 245, enter:

```
symaccess -sid 245 -name IG_1 add -type initiator -wwn 10000000c94ef69c
```

File

When using an input file, each initiator must be placed on a new line and start with either `WWN:` or `iSCSI:` or `IG:`, depending on the type of the initiator or initiator group name. The following is an example of the format for an initiator file:

```
WWN:10000000c94ef69c
iSCSI:iscsiname
IG:IgName
#WWN:10000000c94ef69d
```

If the format of the initiator does not match the label at the start of the line, the file return an error. A commented line, which the system ignores, is specified by placing the pound sign (#) at the beginning of a line.

Cascaded initiator groups

An initiator group can be added to another initiator group, only if it does not contain any initiator groups.

The following scenario describes cascaded initiator groups:

- ◆ HOST1 contains WWN1 & WWN2, which are added to IG_1.
- ◆ HOST2 contains WWN3 & WWN4, which are added to IG_2.
- ◆ IG_3 is created and contains IG_1 & IG_2.

In this example, gatekeeper devices for HOST1 can be assigned to IG_1, while different gatekeeper devices for HOST2 can be assigned to IG_2. The application devices needed by both hosts can be assigned to IG_3.

Removing initiators

The `symaccess remove` command removes specified initiators from an initiator group.

An initiator group that is currently associated with a masking view will be allowed to be emptied and the view will remain along with the other two groups. The following is the syntax for removing an initiator from an initiator group:

```
symaccess -sid SymmID -name GroupName -type initiator
          -wwn wwn | -iscsi iscsi | -ig InitiatorGroupName
          | -f InitiatorFilename -login

remove
```

For example, to remove initiator `-wwn 10000000c94ef69c` from the initiator group `IG_1` on Symmetrix 245, enter:

```
symaccess -sid 245 -name IG_1 remove -type initiator -wwn 10000000c94ef69c
```

Add the `-login` option to the command to remove the initiator from the Symmetrix array's login history table.

Deleting initiator groups

To delete an initiator group, use the following syntax:

```
delete <view -name ViewName> [-force] |
      -name GroupName -type <storage | port | initiator > [-noprompt]
```

To delete initiator group `IG_1` on Symmetrix 245, enter:

```
symaccess -sid -name IG_1 -type initiator delete
```

Note: An initiator group cannot be deleted if it is associated with a masking view.

Setting HBA flags

Symmetrix arrays running Solutions Enabler V7.0 and Enginuity 5874, only allow you to set the HBA port flags on a per initiator basis. This feature allows specific host flags to be enabled and disabled on the director port.

Note: Setting HBA port flags replaces setting the heterogeneous host configuration flags. To switch to setting HBA port flags, the heterogeneous host configuration must be disabled for a given director/port/HBA and all flags must be reset.

To set (or reset) the HBA flags, use the following form:

```
symaccess -sid SymmID -wwn wwn | -iscsi iscsi

set hba_flags <on <flag,flag,flag...> <-enable | -disable> |
off [flag,flag,flag...]>
```

Where:

hba_flags — Sets the record in the database to hold information on the HBA port setting that may differ than the current setting on the FA.

on | **off** — Turns HBA flags on or off.

flag — Specifies the overridden HBA port flags from the values [in brackets]:

Common_Serial_Number	[C]	AS400	[AS4]
Disable_Q_Reset_on_UA	[D]	OpenVMS	[OVMS]
Environ_Set	[E]	SCSI_3	[SC3]
Siemens	[S]	Sunapee	[SCL]
Volume_Set_Addressing	[V]	Sequent	[SEQ]
Avoid_Reset_Broadcast	[ARB]	SPC2_Protocol_Version	[SPC2]
		SCSI_Support1	[OS2007]

-enable — Enables the specified HBA port flag(s) on a per initiator basis.

-disable — Disables the specified HBA port flag(s) on a per initiator basis.

For example, to turn on HBA flags and enable the Common_Serial_Number and SCSI_3 flags, and disable the Disable_Q_Reset_on_UA flag, on an HBA with the WWN ID 210000e08b0995b7 for Symmetrix 031, director 16A port 0, enter:

```
symaccess -sid 031 set hba_flags on C,SC3 -enable -wwn 210000e08b0995b7-dir 16A -p 0
symaccess -sid 031 set hba_flags on D -disable -wwn 210000e08b0995b7-dir 16A -p 0
```

The `symaccess show -detail` output has been modified to display the flags that are turned on and off for each HBA initiator that has this feature enabled, as shown in the following example:

```
symaccess -sid 237 -type initiator -detail show Prod1
```

```
Symmetrix ID           : 000190300237
Last updated at       : 08:46:54 AM on Tue Jul 29,2008

Initiator Group Name   : Prod1

  Originator Port wwn   : 10000000c94ef69c
  User-generated Name   : api196/10000000c94ef69c
  FCID Lockdown         : No
  Heterogeneous Host    : No
  Port Flag Overrides   : No
  Type                  : Fibre

  Originator Port wwn   : 5006016839a00c5c
  User-generated Name   : 5006016839a00c5c/5006016839a00c5c
  FCID Lockdown         : No
  Heterogeneous Host    : No
  Port Flag Overrides   : No
  Type                  : Fibre

  iSCSI Name           : Symm_iScsi
  User-generated Name   : iScsi_node_alias/iScsi_port_alias
  FCID Lockdown         : N/A
  Heterogeneous Host    : No
  Port Flag Overrides   : No
  Type                  : iScsi

  Group Name           : IniGrp
  User-generated Name   : N/A
  FCID Lockdown         : N/A
  Heterogeneous Host    : N/A
  Port Flag Overrides   : N/A
  Type                  : Initiator Group
```

...

Replacing an HBA

In the event a host adapter fails, or needs replacement for any reason, you can replace the adapter and assign its set of devices to a new adapter by using the `replace` action in the following form:

```
symaccess replace -wwn wwn -new_wwn NewWWN [-noprompt]
```

```
symaccess replace -iscsi iscsi -new_iscsi NewiSCSI [-noprompt]
```

To swap HBAs, it is suggested to:

1. Run `symaccess list logins` to view the old WWN/iSCSI HBAs.
2. Swap HBA boards.
3. Run `symaccess list hba` or `discover` to view the new initiator (for example WWN).
4. Run `symaccess replace` to substitute a new WWN for all occurrences of the old WWN. For example, to replace old WWN 20000000c920b484 with new WWN 20000000c920b393:

```
symaccess -sid 814 replace -wwn 20000000c920b484 -new_wwn 20000000c920b393
```

5. Run `symaccess discover -rename` to establish the new AWWN and assign an AWWN to the new HBA in the login history table.

Displaying Autoprovisioning information

This section shows the output for the masking views using the following syntax:

```
symaccess -sid SymmID | -f BackupFilename
```

```
list view [-name ViewName] [-v]
```

Listing masking views

The following example shows the command and output for listing all masking views on Symmetrix array 237:

```
symaccess -sid 237 list view
```

```
Symmetrix ID           : 000190300237
```

Masking View Name	Initiator Group	Port Group	Storage Group
View1	IG_1	PG_1	SG_1
View2	WinHost	PG_5	Acct
...			

Listing masking views with details

The following example shows the command and output for the same command, with the `-v` (verbose) option added:

```
symaccess -sid 237 list view -v
```

```
Symmetrix ID           : 000190300237
```

```
Masking View Name      : View1
Initiator Group Name    : IG_1
Port Group Name        : PG_1
Storage Group Name     : SG_1
Last updated at        : 09:23:44 PM on Tue Jun 23,2008
```

```
Masking View Name      : View2
Initiator Group Name    : WinHost
Port Group Name        : PG_5
Storage Group Name     : Acct
Last updated at        : 08:13:17 AM on Tue Jul 12,2008
```

Viewing group details

Storage group, port group, and initiator group details can be displayed using the `symaccess list` and `symaccess show` commands, as follows:

```
symaccess -sid SymmID | -f BackupFilename
```

```
list [-name GroupName] [-v]
```

```
list -type <storage [-dev SymDevName] | port
      [-dirport Dir:Port] | initiator [-wwn wwn |
      -iscsi iscsi]> [-v] [-name GroupName]
```

```
show GroupName -type <storage | port | initiator [-detail]>
show view ViewName
```

An optional group name may be supplied as input and only the details on that group return.

An optional director and port number may be supplied and only the port groups that belong to the director and port return.

To display a list of the initiator groups, in verbose mode, use the following form:

```
list -type initiator [-wwn wwn | -iscsi iscsi] -v
```

To display a specific initiator group, in verbose mode, use the following form:

```
list -name GroupName -v
```

To display the details of a specific initiator group, use the following form:

```
show GroupName -type initiator -detail
```

Autoprovisioning Groups examples

Group Examples To create a initiator group named `initexample`, and to add WWN initiator `210000e08b04daac` to it, enter:

```
symaccess -sid 234 -type initiator -wwn 210000e08b04daac -name initexample create
```

To create a storage group named `storeexample`, and to add device `0026` to it, enter:

```
symaccess -sid 234 -type storage devs 0026 -name storeexample create
```

To create a port group named `portexample`, and to add director `7E` and port `1` to it, enter:

```
symaccess -sid 234 -type port -dirport 7E:1 -name portexample create
```

To add devices to a storage group named `storeexample`, enter:

```
symaccess -sid 234 -type storage -name storeexample add devs 0027
```

To remove WWN initiator `210000e08b04daac` from initiator group named `initexample`, enter:

```
symaccess -sid 234 -type initiator -wwn 210000e08b04daac -name initexample remove
```

To delete an initiator group named `initexample`, enter:

```
symaccess -sid 234 -type initiator -name initexample delete
```

To list all initiator, port and storage groups, enter:

```
symaccess -sid 234 list
```

View Examples To create a view named `viewexample`, containing initiator group `initexample`, port group `portexample`, and storage group `storeexample`, enter:

```
symaccess -sid 234 -name viewexample -sg storeexample -pg portexample -ig initexample create view
```

To delete a view named `viewexample`, enter:

```
symaccess -sid 234 -name viewexample delete view
```

To rename a view named `viewexample` to `mvexample`, enter:

```
symaccess -sid 234 rename view -name viewexample -new_name mvexample
```

To create a view, storage, port and initiator group named `TEST`, and to add devices to storage group `TEST`, initiators to initiator group `TEST`, and director ports to port group `TEST`, enter:

```
symaccess -sid 234 create view -name TEST -wwn 210000e08b04daac -dirport 7E:1 devs 0026
```

Backup and Restore Examples To back up groups and views of Symmetrix ID 234 to a file `backup_from_lab`, enter:

```
symaccess -sid 234 backup -file backup_from_lab
```

To restore group and view information in file `backup_from_lab` to Symmetrix ID 234, enter:

```
symaccess -sid 234 restore -file backup_from_lab
```


This chapter describes the device masking concepts and how to confine host access to Symmetrix devices using the device masking commands of the SYMCLI. The chapter covers the following topics:

◆ Device masking overview	142
◆ Supported topologies.....	146
◆ Device masking VCMDB security enhancement.....	148
◆ Initial device masking setup	149
◆ Device masking configuration	150
◆ Discovering host HBAs	153
◆ How to add and remove masked devices	155
◆ Device masking VCMDB maintenance.....	159
◆ How to view the VCMDB.....	163
◆ HBA initiator management.....	169
◆ Fibre Channel-to-host interface management.....	171

Device masking overview

This chapter provides device masking information for Symmetrix arrays running Enginuity 5773 and earlier.

Note: Symmetrix arrays running Enginuity 5874 cannot use the `symmask` and `symmaskdb` commands described in this chapter. To mask devices on those arrays, refer to [Chapter 2, “Masking Devices with Autoprovisioning Groups.”](#)

SYMCLI device masking provides the ability to assign and mask access privileges of host bus adapters (HBAs) to Symmetrix directors and devices by associating one or more devices with an HBA-to-FA connection (known as a *masked channel*) that you define in the Symmetrix-based device masking database, known as the VCMDB. The VCMDB maintains all access records for an array and the Symmetrix array monitors host access to resolve any conflicts that might arise from multiple hosts having visibility to the same devices.

The device masking commands can also be used to configure heterogeneous hosts with shared access to the same FA port, which is useful in an environment with different host types. However, you can also use Fibre Channel ID lockdown security to protect an HBA from predatory WWN spoofing.

Note: For a detailed introduction to Solutions Enabler, SYMCLI, and the Symmetrix array, refer to the *EMC Solutions Enabler Symmetrix Array Management CLI Product Guide*.

How Symmetrix device masking works

If you define masked channels that allow multiple hosts to connect to a single Symmetrix director, potential conflicts may arise as multiple hosts have access to discover and use the same set of devices. Device Masking allows you to control host access to a set of devices by maintaining a set of entries in the VCMDB on the array that defines the relationship between masked connections and devices.

Each entry includes a host's HBA identity (the HBA port WWN), its associated FA port, and a range of devices mapped to the FA port that should be visible only to the corresponding HBA. Once you make this VCMDB entry and activate the configuration, the Symmetrix makes visible to a host those devices that the VCMDB indicates are available to that host's initiator WWN through that FA port.

The VCMDB on each Symmetrix array specifies the devices that a particular host can access through a specific director. Each director can control access to as many as 256 unique WWNs or 512 iSCSIs (beginning with Enginuity version 5771). As many as 128 fiber director ports, and 64 multi-protocol (iSCSI) ports (depending on the Symmetrix model) can be configured within the device masking VCMDB.

You can initialize, back up, and restore this database. In addition, you can list, add, and remove database entries, clear the database, and manage WWN and iSCSI names.

Host access examples

When a host attempts to access a Symmetrix storage device, as shown in [Figure 14](#), the host HBA initiator name (supplied when the host logs in to the fabric or arbitrated loop) is passed to the Symmetrix director port.

The Symmetrix records the connection, stores the initiator name in a *login history table* in its memory, and thereafter grants access to only the devices that are available to that initiator through that director port (as specified in the device masking VCMDB). However, if CHAP authentication is enabled in a native iSCSI topology, the VCMDB will first check the credential and secret before granting access.

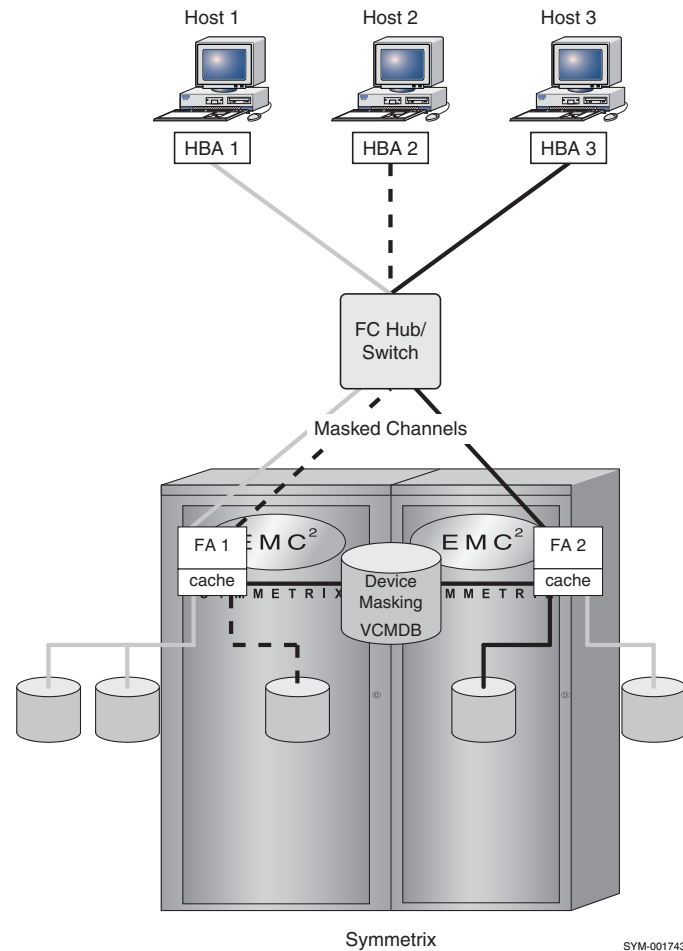


Figure 14 Device masking solution with WWN initiators

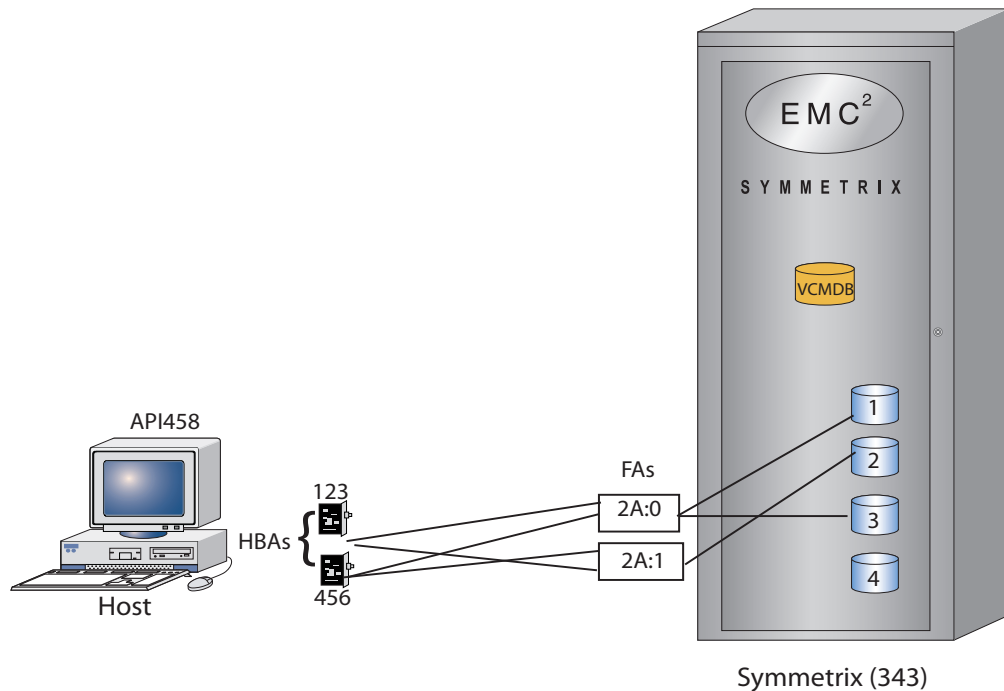
The host HBA port then sends I/O requests directed at particular Symmetrix devices to the director port. Each request includes the identity of the requesting HBA (from which its WWN or iSCSI can be determined) and the identity of the requested device, with its director and logical unit number (LUN).

The software that runs on a Symmetrix system processes each I/O request to verify that the HBA is allowed to access that device. Any request for a device that an HBA does not have access to, returns an error to the host.

In [Figure 14](#), device masking grants Host 1 access to two of the three devices available through FA1, and grants Host 2 access to a third device. Similarly, Host 3 is granted access to only one of the two devices available through FA2, reserving the second device for use as a spare.

Solutions Enabler 7.0 provides the `-host` option to allow the host alias name to be used instead of the initiator types (wwn, alias, or iscsi). In addition, the `ALL` parameter has been added to the `-port` and `-director` options.

These additions can reduce the number of commands required to mask devices, as shown in the following example.



For the above setup there are four records in the VCMDb:

```
WWN 123 on Director port 2A:0 with alias api458/123 + bitmap representing devices 1 and 3
WWN 123 on Director port 2A:1 with alias api458/123 + bitmap representing device 2
WWN 456 on Director port 2A:0 with alias api458/456 + bitmap representing devices 1 and 3
WWN 456 on Director port 2A:1 with alias api458/456 + bitmap representing device 2
```

To mask device 4 with one command:

```
symmask -sid 343 -host api458 -dir ALL -p ALL add dev 4
```

Managing access for unsupported host platforms

Device masking can also manage access for host platforms that are not supported by this release. If a host can log on to the Symmetrix arrays using a Fibre Channel interface, its access can be controlled.

However, device masking cannot automatically discover the WWNs of host HBAs on unsupported platforms. Instead, you must manually set up a record in the database for these hosts.

Note: Because hosts on unsupported platforms cannot run device masking commands used for verifying host initiator name bindings, you must manage and update names without SYMCLI. Contact EMC Customer Support for help with hosts on unsupported platforms.

Device masking controls overview

SYMCLI commands `symmask` and `symmaskdb` specifically support the device masking control and monitor operations. With these commands, you can define and query the Symmetrix devices that each host's HBA ports are permitted to access.

The `symmask discover` command can be run on both the control station and the managed hosts. The `symmask discover` action locates paths to the device masking database (VCMDB) and assigns alias names (AWWN/AISCSI) to the HBAs residing on the host on which the command is run if they are NULL. You can optionally use the `rename` action to generate aliases to be assigned.

For detailed man page descriptions of these commands and their options, refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

Database device locking

During the execution of the `symmask` or `symmaskdb` commands, the SYMCLI sets a Symmetrix External Lock (SEL) on the Symmetrix where the device masking database (VCMDB) resides. This lock ensures that only one host can make changes to the database at any one point in time.

If during the processing of a `symmask` or `symmaskdb` command, the host fails, or a Ctrl/C is performed in the middle of the command, the lock might not release and could lock out further needed changes or control actions. If a device masking command is interrupted and the lock is not released, future invocations of a device masking command will display the following error message:

```
The operation failed because another process has an exclusive lock
on the local Symmetrix.
```

To further examine the presence of this lock, use the following form:

```
symcfg -sid SymmID list -lock -lockn ALL
```



CAUTION

Use the release action only if you believe the lock was forgotten and there are no other operations in progress to the database.

The command will list Symmetrix external locks being held. For this case, it will show a number 14 device masking lock and the length of time it has been on.

To release this lock, use the following form:

```
symcfg -sid SymmID -lockn 14 release
```

Supported topologies

Device masking supports Fibre Channel (point-to-point, arbitrated loop, and switch fabric) and native iSCSI topologies.

Fibre Channel topology

Figure 15 illustrates two different Fibre Channel network configurations that device masking supports:

- ◆ Point-to-point — Network A (left) is configured with direct connections from one HBA on each host to one FA on the array, providing each host with access to a different set of devices.
- ◆ Multi-initiator — Network B (right) is configured with multiple hosts accessing the same Symmetrix devices through a common fabric.

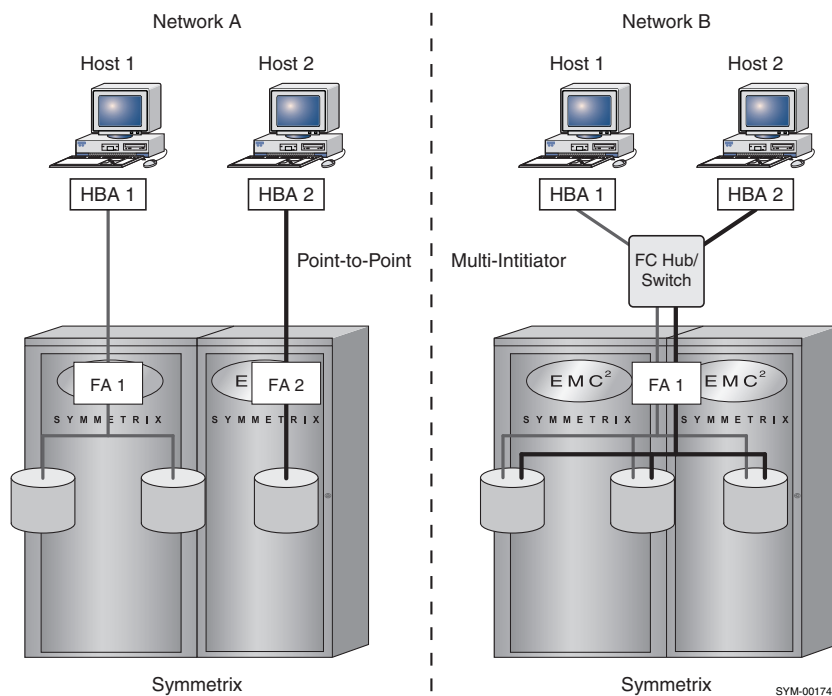


Figure 15 Point-to-point and multi-initiator topologies

Native iSCSI topology

In contrast to the Fibre Channel topology, in a native iSCSI environment, hosts are connected to a Symmetrix DMX array through an Ethernet switch, as shown in Figure 16.

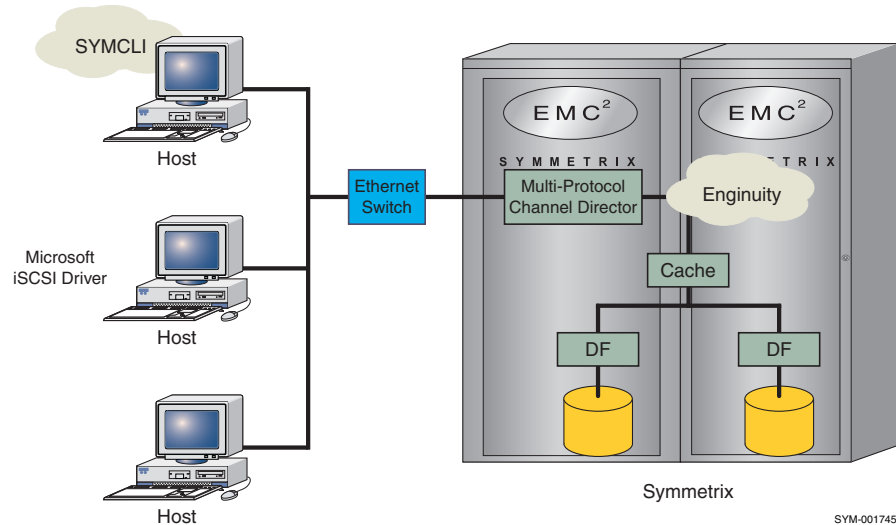


Figure 16 Native iSCSI topology

CHAP

Native iSCSI support standards require that a security protocol be available. Enginuity version 5670 provides the Challenge Handshake Authentication Protocol (CHAP), which can be enabled or disabled by the user. Refer to [“iSCSI software driver configuration” on page 187](#) for more information about CHAP authentication.

HBA initiator support

Both HBA and Symmetrix director ports in the topology are uniquely identified by a name (WWN or iSCSI). For ease of use, you can associate an ASCII nickname (AWWN and AISCsi).

SYMCLI device masking supports the following HBA initiators:

- ◆ World Wide Name (*wwn*) and alias for World Wide Name (*awwn*)
- ◆ Native iSCSI over TCP/IP (*iscsi*) and alias for iSCSI over TCP/IP (*aiscsi*)

Refer to [“Discovering host HBAs” on page 153](#) for more information about WWN and iSCSI initiators.

Note: An iSCSI initiator can log in to the array (through a Gig-E port) before the VCMDb is initialized. After the database is initialized, only iSCSI initiators with a masking record are allowed to log in. If the database is initialized by a host connected to the array through a Gig-E port, a record for the iSCSI initiator of that host will be added to the database so that access to the array is not lost as a result of this command.

Device masking VCMDB security enhancement

By default, the device masking VCMDB is accessible to all HBAs that log in to the director port where the database is configured. Thus, any host with access privileges can effectively modify the contents of the database if it has device masking installed.

However, to prevent an unauthorized host from changing the database, an enhancement to the Symmetrix operating system (Enginuity versions 5265 and higher) allows you to control a host's access to the database device through the contents of the database records. This enhancement allows only HBAs with valid records in the database to access the database. This enhancement works with all versions of device masking.

Beginning with Enginuity version 5670, the VCMDB can be unmapped from any director that is not being used for masking control.

Note: If you have PowerPath installed, you should keep the VCMDB mapped.

VCMDB management updates

The Solutions Enabler SYMCLI version 5.3 includes some changes to the management of the VCMDB. In prior versions of SYMCLI, the host system wrote directly to the VCMDB in the Symmetrix array, which was then read by the Enginuity software (see [Figure 17](#)). Any host with access to the VCMDB could update the database.

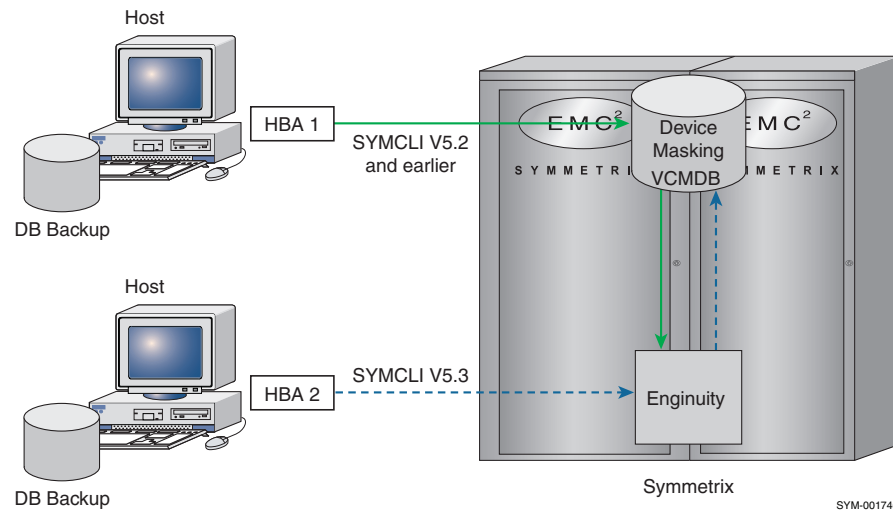


Figure 17 Device masking syscalls

Since Solutions Enabler version 5.3, and Enginuity version 5670, the host communicates with the Enginuity software, which then communicates to the VCMDB.

Initial device masking setup

Before you begin to using the Solutions Enabler Device Masking, it is important to understand your device masking and test your scripts in a controlled environment.

When you set up your device masking environment, you should initialize the device masking VCMDDB (see [“Initializing the database” on page 159](#)). This should only be done on an initial setup, since it clears the device of any and all data.

Once you understand the Solutions Enabler Device Masking functionality described herein, you can begin to develop device masking scripts customized for your environment.

Access control environment setup

If Symmetrix Access Control is being used to protect Symmetrix devices, the host from which you run the device masking commands must be configured in an access control group with an ACL (Access Control List) granting VLOGIX rights. Otherwise, attempted changes to the configuration records in the VCMDDB would fail.

Some controls require DIRCTRL or CFGSYM rights. Initialize, restore, and convert controls require CFGSYM and VLOGIX rights. Authentication and `symconnect` controls require DIRCTRL rights.

For more information about the Symmetrix Access Control `symacl` command, see the *EMC Solutions Enabler Symmetrix Array Management CLI Product Guide*.

Device masking configuration

Configuring device masking involves four steps:

1. “Identify configuration components” on page 150
2. “Initialize and update the database” on page 151
3. “Enable authentication” on page 151
4. “Recommendations for activating the configuration” on page 151

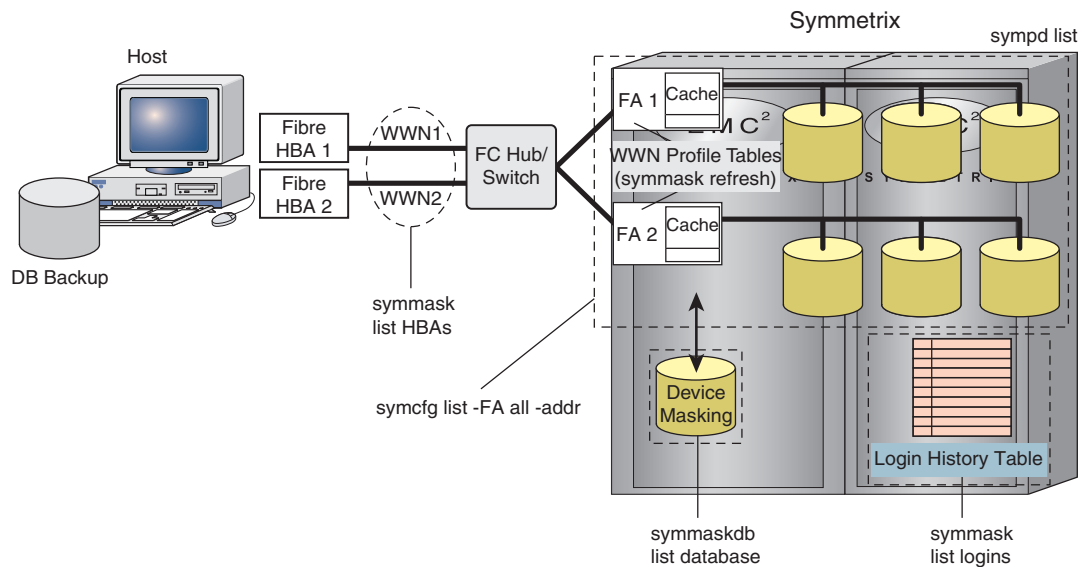
You should be logged on to the control station as Administrator in an MS-DOS command prompt window (on a Windows system) or as root in an xterm window (on a UNIX system).

Identify configuration components

Discover local HBAs on a host that have a channel to a Symmetrix array and generate an AWWN for any HBA that does not have an AWWN assigned yet, updating the VCMDB with the new information.

symmask discover hba

Figure 18 shows device masking components to identify.



SYM-001747

Figure 18 Device masking components

The following commands can be used to obtain the identifiers of each discovered element:

- ◆ List the Symmetrix physical device names of all the device masking devices.
sympd list -vcm
- ◆ List the HBAs on a host and the Symmetrix FA director port to which each HBA is connected. Note the WWN of the HBA that you will be configuring and which FA port connects to that HBA.

symmask list hba

- ◆ Symmetrix director port to which each HBA on the host connects.

```
symmask list logins
```

- ◆ List devices mapped to the FA director that you will be configuring (for example, director 16A).

```
symcfg list -FA 16A -addr
```

Note: SYMCLI device masking supports both World Wide Name (WWN) and native iSCSI (iSCSI) HBA connections.

Initialize and update the database

After you identify each element, you can initialize the database and create records. To initialize the database, use the following command:

```
symmaskdb -sid SymmID init -file BackupFilename [-vcmdb_type <3 | 4| 5>]
```

For more information on initializing the VCMDB, refer to [“Initializing the database” on page 159](#); and for more information on supported database types, refer to [“VCMDB database types” on page 160](#).

Make an entry for the HBA-to-FA connection in the VCMDB, specifying devices that the HBA can access. For example, add a range of devices (0030 through 0034) to the VCMDB on the Symmetrix array (**-sid 814**), specifying the HBA’s WWN and the FA director/port that the HBA connects to.

```
symmask -sid 814 -wwn 20000000c920b484 add devs 0030:0034 -dir 16A -p 0
```

Enable authentication

For detailed information regarding initializing, setting, enabling, and disabling authentication, refer to [Chapter 4, “Device Masking: iSCSI Setup.”](#)

Recommendations for activating the configuration

To ensure that updates to the VCMDB become active and visible to your host, it is recommended that you:

1. Back up the device masking VCMDB to a file by calling `symmaskdb backup`.

Note: During the initial setup you cannot create a backup.

2. Update the Symmetrix array with the configuration changes by performing `symmask refresh`. This calls the Symmetrix director to refresh its WWN/iSCSI-related profile tables in cache with the contents of the device masking VCMDB.



CAUTION

Before running the `symmask refresh` command, or the `symmask -verify` command, make sure there are no HBAs accessing devices in the masked channel (applications running or user activity).

3. Reboot all hosts that have had devices added or removed for the changes to take effect.

Note: When you reboot a host, you must run `symcfg discover` to scan the Symmetrix devices and refresh the SYMAPI configuration database.

4. When configuration of the database is complete, use `sympd list` to view the Symmetrix devices that can be seen by the host.

Discovering host HBAs

During the initial setup, an administrator runs `symmask discover` on the controlling host to search the environment for Symmetrix devices on each HBA by using the following command:

```
symmask discover hba
```

Note: It is assumed that the host from which these commands are run has access to the Symmetrix devices.

When the `symmask discover` finds a host HBA, it reads the login history table and performs the following:

1. Checks whether an alias exists in the device masking VCMDB. If one does, this command writes it to the login history table.
2. If there is no alias in the device masking VCMDB record, or the login history table, it creates an ASCII alias and writes it to the login history table.

Note: There is a `-rename` option that can be used with this command to force the discovered hostname/HBA name (or IP address) to be written to the login history table and the device masking VCMDB. This will overwrite any existing AWWN/iSCSI record you have previously established.

3. Prints the initiator identifier (WWN/iSCSI) of the HBAs that are connected to the masked channel and Symmetrix array.
4. If the `-rename` option was used with the `symmask discover` command, the initiator identifier and its ASCII alias are written to the device masking VCMDB.

The `symmask discover` command sends information about this connection back to its host system. The `discover` command is the primary mechanism by which hosts other than the control station can learn about their VCMDB paths to the Symmetrix array. This is displayed using the `symmask list hba` command.

Using alias names

Whether you have defined alias names (AWWN/iSCSI) for the various HBAs, or choose to use those assigned by the SYMAPI server during discovery, alias names can be used in the command line, replacing the cumbersome numeric identifiers.

These names, which are stored in the Symmetrix array's login history table, identify the HBAs connected to the network interface. Alias names can be shorter in length and much more recognizable than the cryptic WWNs/iSCSIs.

ASCII format

All alias names (auto-generated or user-defined) have two parts separated by a slash (/), such as *ALIAS/ALIAS*. An ASCII alias names generated by the `discover` action consists of two parts: the name of the host and the name of the HBA.

- ◆ For Fibre configurations, the adapter number takes the form of the WWN or iSCSI to guarantee uniqueness. For example, the AWWN for a host whose TCP/IP hostname is `john4554b`, on adapter `10000000c920cf87`, would be `john4554b/10000000c920cf87`.
- ◆ For Native iSCSI configurations, the values are hostname/IP address.

For more information on renaming Identifiers, refer to [“HBA initiator management” on page 169](#).

Renaming identifiers

When using various `symmask` actions (such as adding or removing devices in the device mask) you can target an HBA path by specifying an AWWN or AISCsi in the command line. Once you have established the aliases with the identifiers in the history table and database, you can rename existing aliases with `symmask rename` action using the following form:

```
symmask -sid SymmID -iscsi iscsi rename aiscsiNew
```

For example, you are working with Symmetrix 0128 and you want to change your HBA of 20000000c920b484 to Solar2b, enter:

```
symmask -sid 0128 -wwn 20000000c920b484 rename Solar2b/b4
```

You can run `symmask list logins` to display the contents of the login history table to examine the existing alias names on a specified Symmetrix array.

How to add and remove masked devices

The Symmetrix devices you want to isolate can be assigned to a specified masked channel (HBA to director port). You can add or remove devices from these masked channels using their Symmetrix device name, or device group name.

Adding devices

To add a device or devices to a specified HBA/director-port channel, use the following syntax:

```
symmask -sid SymmID -wwn wwn | -awwn awwn |
        -iscsi iscsi | -aiscsi aiscsi | -host HostName
        -dir <#|all> -p <#|all> [-celerra]
        [-reserve_id ResvID[, ResvID[, ResvID]]]

add devs SymDevStart:SymDevEnd |
        <SymDevName | SymDevName, SymDevName, SymDevName...>
        [-dynamic_lun | -lun Addr | <Addr, Addr, Addr...>]
        [-noprompt] [-remove_unmapped]
```

For example, to add devices 0014 and 0015 on Symmetrix 0128 for access to all directors and ports on Host3b, enter:

```
symmask -sid 0128 -host Host3b -dir all -p all add devs 0014,0015
```

If the devices are not addressed to the specified FA, a warning message displays.

If the devices are already assigned in the database to any WWN, an informational prompt displays. To turn off this functionality, use the `-noprompt` option.

Adding metadevices

To add metadevices, add only the *SymDevname* of the device that is designated as the metahead.

Using reserved devices

If you are using devices that have been reserved, you must supply the device reservation ID.

For example, to add reserved device 0014 on Symmetrix 0128 for access to Host3b using director 16a, port 0, enter:

```
symmask -sid 0128 -host Host3b -dir 16a -p 0 add dev 0014 -reserve_id 5
```

[“Managing device reservations” on page 76](#) provides more information about device reservations.

Adding device group devices

To add device group devices to a specified HBA/director-port channel, use the following syntax:

```
add -g DgName [-std] [-bcv] [-vdev] [-tgt] [-noprompt]
        [-dynamic_lun | -lun Addr] [-remove_unmapped]
```

Use the `-std` option to mask only the standard devices; `-bcv` to mask only the BCV devices; `-vdev` to mask only the virtual devices; and `-tgt` to mask only the target devices. For example, to add the standard devices of device group `prodB` on Symmetrix 0128 for access to Host3b using director 16a, port 0, enter:

```
symmask -sid 0128 -awwn Host3b/4a add -g prodB -std -dir 16a -p 0
```

If no option is specified, only standard devices (`-std`) will be added.

Note: Device group names are not stored, only the device information is stored.

Adding devices from a device file

You can perform add masking operations on devices in a device file. When creating the device file, enter the source device first, a space, then the target device. Device files can contain comment lines that begin with the pound sign (#). For example,

```
#Device list Mask2 for masking
0014 009A
0151 014C
....
```

A Symmetrix ID is required for this option. In addition you can use the keywords `src` and `tgt` to specify that the action should only use the source devices, or only use the target devices.

To mask only the source devices in the `Mask2` file on Symmetrix 0845, to director FA-2A and port 0, enter:

```
symmask -sid 0845 -wwn 50060482d5f003c1 -dir FA-2A -p 0 add -f mask2 src
```

If the `src` or `tgt` keywords are not specified, only the source devices (`src`) are acted upon.

Adding devices with dynamic LUN addressing

When adding a list of devices, a device group, or a device range, you can specify the starting LUN address for each device using the `-lun` option, or have SYMAPI assign the LUN address using the `-dynamic_lun` option.

To add a device or devices to a specified HBA/director-port channel, specifying the LUN address use the following syntax:

```
symmask -sid SymmID -wwn wwn [-awwn awwn] [-iscsi iscsi] [-aiscsi aiscsi]
add devs SymDevStart:SymDevEnd | SymDevName | SymDevName,,...\
-lun Addr | Addr, Addr, Addr...
-dir # -p # [-noprompt] [-dynamic_lun] [-remove_unmapped]
```

Where:

-lun = Specifies starting LUN addresses. You can specify a single starting LUN or multiple LUNs to match the given ranges. For example:

```
symmask add devs 15,18,20 -lun 0 -wwn 20000000c920b484 -dir 14C -p 1 -sid 12
```

-dynamic_lun = Specifies the use of dynamic LUN addressing, however the application assigns the addresses based on what is already in use for the host HBA.

-removed_unmapped = Records that have unmapped devices cannot be converted until those devices are removed. This option removes any unmapped devices when adding dynamically addressed devices in a single step.

Examples For example, to add devices 100:105 on Symmetrix 104 for access to wwn 20000000c920b484 starting at LUN addresses 0, and using the next available LUN numbers, for director 2d, port 0, enter:

```
symmask add devs 100:105 -lun 0 -wwn 20000000c920b484 -dir 2d -p 0 -sid 104
```

When adding devices from a device file or device group, add the `-dynamic_lun` option to the existing syntax for the system to assign the LUN addresses, for example:

```
symmask add -f C:\my_device_file -dynamic_lun -wwn 20000000c920b484 -dir 3c -p 0 -sid 90
```

```
symmask add devs 2C,2E,30 -dynamic_lun -wwn 20000000c920b484 -dir 2a -p 1 -sid 215
```

Removing devices

You can remove some devices associated with an HBA entry, or you can remove the entire set of devices associated with an HBA entry. Removing some devices (but not all) requires syntax similar to that for adding devices and the same refresh, backup, and discover steps after completing the remove operation. You can remove devices from a masked channel at any time. To remove a device or devices from a masked channel, use the following syntax:

```
symmask -sid SymmID -wwn wwn|-awwn awwn|-iscsi iscsi|-aiscsi aiscsi  
remove devs SymDevStart:SymDevEnd|SymDevName|SymDevName,, ,|  
-dir # -p # [-force]
```

The force (`-force`) option may be useful when you want to quickly remove a range of specified device names that might span device names not part of the existing noncontiguous masked channel. It may also be needed when working with metadevices.

After each set of changes, activate the configuration by performing a refresh (`symmask refresh`), back up the database (`symmaskdb backup`), and reboot the affected hosts.

Note: When ever you reboot a host, you must run `symcfg discover` on all the Symmetrix devices and refresh the SYMAPI database.

Examples To remove devices 0031 and 0033 from the 0030-to-0034 range of devices that was added previously:

```
symmask -sid 814 -wwn 20000000c920b484 remove devs 0031,0033 -dir 16A -p 0
```

To remove the remaining devices in the 0030-to-0034 device range, you can specify individual devices or the range with an option (`-force`) that allows you to remove a noncontiguous range. For example:

```
symmask -sid 814 -wwn 20000000c920b484 remove devs 0030:0034 -dir 16A -p 0 -force
```

To remove the entire set of devices that an HBA can access, use `symmask delete` and specify the WWN of the HBA. The delete action removes the HBA entry completely, including any attributes set previously. For example:

```
symmask -sid 814 delete -wwn 20000000c920b484
```

Removing metamembers

To remove metamembers from the device masking VCMDDB, but keep the metaheads in place, use the following form:

```
symmaskdb -sid SymmID -meta_member remove
```

The `-force` option may be necessary when working with metamembers.

Removing device group devices

To remove device group devices from a specified HBA/director-port channel, use the following syntax:

```
symmask -sid SymmID -wwn wwn|-awwn awwn|-iscsi iscsi |-aiscsi aiscsi  
remove -g DgName [-std] [-bcv] [-vdev] [-force] -dir # -p # [-noprompt]
```

Use the `-std` option to remove only the standard devices; `-bcv` to remove only the BCV devices; and `-vdev` to remove only the virtual devices. For example, to remove the standard devices of device group `prodB` on Symmetrix 0128 from access to Host3b using director 16a, port 0, enter:

```
symmask -sid 0128 -awwn Host3b/4a remove -g prodB -std -dir 16a -p 0
```

If no option is specified, only standard devices (`-std`) will be removed.

Note: Device group names are not stored, only the device information is stored.

Removing devices using a device file

To remove the masking entries for only the source devices in the `Mask2` file on Symmetrix 0845, to director FA-2A and port 0, enter:

```
symmask -sid 0845 -wwn 50060482d5f003c1 -dir FA-2A -p 0 remove -f mask2 src
```

If the `src` or `tgt` keywords are not specified, only the source devices are acted upon.

Device masking VCMDB maintenance

After the initial setup, records are added, updated, and deleted from the device masking VCMDB each time a SYMCLI command is issued to add or remove devices in a masked channel, a WWN /iSCSI alias is renamed, one HBA is swapped for another, or an HBA is cleared, removing its device mask allocation.

Initializing the database

For the initial setup of any device masking environment, a device reserved (VCM state enabled) in the Symmetrix array must be initialized and formatted for use as the device masking database, better known as the VCMDB. The initialization clears the disk device of any current data in the process of formatting the database. In most cases, you do not want to clear the data of an existing VCMDB. If you are unsure whether a VCMDB currently exists, issue the command that displays an existing VCMDB. For example, to view the VCMDB if it exists on Symmetrix 0128:

```
symmaskdb -sid 0128 list database
```



CAUTION

The following command is rarely used. Be sure you want to zero out the device masking VCMDB before proceeding.

To initialize and clear the database device, use the following syntax:

```
symmaskdb -sid SymmID init -file BackupFilename [-vcmdb_type <3 | 4 | 5>]
```

For a safeguard, you must specify a backup filename, since this command will try to write the data from this device to a backup file on your host before it clears the current data.

For example, to initialize the database and create backup file *BackupDevMask1* on Symmetrix 0128, enter:

```
symmaskdb -sid 0128 init -file BackupDevMask1
```

If you do not include the `-vcmdb_type` option, the default database type depends on the size of the VCMDB device: type 3 (available beginning with Enginuity 5670) if the device is equal to or greater than 24 cylinders, but less than 48; a type 4 (available beginning with Enginuity 5670) if equal to or greater than 48 cylinders, but less than 96; a type 5 (beginning with Enginuity version 5671) if equal to or greater than 96 cylinders. If the Symmetrix array is running a 57xx Enginuity level greater than version 5770, the default is a type 6 database.

Preventing unauthorized modification of the VCMDB

By default, the VCMDB grants access to all HBAs that log in to the FA director port where the database is configured. Without preventive measures, any host with access privileges can modify the VCMDB if it has the EMC Solutions Enabler Device Masking component or ESN Manager installed.

One way to prevent unauthorized host access, on arrays running Enginuity versions prior to 5771, is by using the Configuration Manager to set the `VCMDB_restricted_access` Symmetrix parameter. This metric is obsolete in Enginuity version 5771 and later. By setting this parameter value to `ENABLE`, you restrict access to the VCMDB to hosts having a VCMDB entry that includes the VCMDB device (the device on which the device masking database resides). You deny

VCMDB access to all hosts except those whose HBAs have added the VCMDB device through the `symmask add devs` command. (You can display the VCMDB device on a Symmetrix array using the `sympd list -vcm` command.)

Prior to enabling this parameter, you should ensure that at least one host HBA has a valid VCMDB entry that includes the VCMDB device. (It is recommended that you have two HBA entries that include this device, in case of an HBA failure.) Without this VCMDB entry, no hosts can access the VCMDB database. To gain access to the VCMDB database again, you would need to reset this parameter to `DISABLE`.

Managing a backup VCMDB file

You can create a backup file containing the current contents of the device masking VCMDB. This is useful when you want to temporarily change the access rights or device masking assignments to various HBAs. Then at some point in time, you can return the device masking environment back to the original masked environment. Often, just backing up the database on a regular basis ensures you can recover your established masked environment in the event of some improper changes or failure.

Note: You cannot reuse any existing backup filename. Always create a new file.

To create a backup database file, use the following syntax:

```
symmaskdb -sid SymmID backup -file BackupFilename
```

For example, to create backup file `BackupDevMask` on Symmetrix 0128, enter:

```
symmaskdb -sid 0128 backup -file BackupDevMask
```

The VCMDB backup files vary in length, depending on how much information is in them.

VCMDB database types

Device masking database types define the number of devices per database record and the number of records you can mask.

Solutions Enabler version 5.3 and Engenuity version 5670 introduced support for a Type 4 device masking VCMDB, which expanded the existing Type 3 VCMDB option.

Beginning with Solutions Enabler version 6.0 and Engenuity version 5671, a Type 5 device masking VCMDB is supported, and Engenuity version 5771 will only support a Type 6 VCMDB. The various VCMDB types are defined as:

- ◆ **Type 3** — Supports up to 8 K devices per record and 32 fibre/32 iSCSI initiator records per port (the VCMDB device must be 24 cylinders or larger)
- ◆ **Type 4** — Supports up to 8 K devices per record and 64 fibre/128 iSCSI initiator records per port (the VCMDB device must be 48 cylinders or larger)
- ◆ **Type 5** — Supports up to 16 K devices per record and 64 fibre/128 iSCSI initiator records per port (the VCMDB device must be 96 cylinders or larger)
- ◆ **Type 6** — Supports up to 64 K devices per record and 256 fibre/512 iSCSI initiator records per port (cannot be specified with SYMCLI)

When initializing the VCMDB database where no database currently exists, SYMCLI defaults to creating a database according the size of the VCMDB device being initialized. For example, a Type 4 VCMDB would be created for a 48-cylinder device. This default behavior is also true for the VCMDB device that currently holds a

database (for example, a Type 4 database on a 96-cylinder device will be initialized as a Type 5 database). A Type 4 or 5 database initializes with direct I/O writes blocked to protect against outside sources corrupting the database. You can also block direct I/O writes to a Type 3 database using `set no_direct_io` option.

You can explicitly create a VCMDB of a specific type in the initialization procedure. This is especially important when you are considering backwards compatibility issues. If you had a larger VCMDB device size (96 cylinders, for example), but you have a need to be compatible with Solutions Enabler version 5.3, you would probably want to specify the smaller size (Type 3 or 4, for example) to maintain backward compatibility. The following command provides an example syntax specifying a VCMDB type:

```
symmaskdb -sid 814 init -file MyInitBackup -vcmdb_type 4
```

Converting a VCMDB type

You can convert a Type 3 database to Type 4 or Type 5, and a Type 4 to a Type 5, if the size of the VCMDB device on the specified Symmetrix array is large enough for the database type. The following is an example converts the VCMDB on Symmetrix 814 to a Type 5 database:

```
symmaskdb -sid 814 convert -vcmdb_type 5 -file MyCvrtBU
```

If you convert from a lower type database to a higher type, any hosts running a Solutions Enabler version that does not support the higher VCMDB type will not be able to access the database. For example, if one host running Solutions Enabler version 6.0 converts an existing VCMDB to Type 5, another connected host running Solutions Enabler version 5.4 will no longer have access to the database until this host is upgraded to version 6.0 or higher.

Restoring a backup VCMDB

You can restore the database from the backup file stored on the host by using the following form:

```
symmaskdb -sid 0128 restore -file MyCvrtBU
```

The database is restored as is; Type 3 restores to Type 3, and Type 4 restores to Type 4. You can use the `convert` or `set` options to alter the resulting environments.

To restore the database from a backup file, but not the authentication information, enter:

```
symmaskdb -sid 0128 restore -file MyCvrtBU -skip_authentication
```

Note: The `-skip_authentication` option is no longer available with Enginuity 5771 and higher.

Restoring a backup and converting its type

To restore from a backup file and convert its database type in a single command specify the `-vcmb_type` option and specify the type: 4 or 5. For example , enter:

```
symmaskdb -sid 0128 restore -file MyCvrtBU -vcmb_type 5
```

Blocking direct writes to the VCMDB

If you have a Type 3 database and would like to block direct writes to the database, use the following form:

```
symmaskdb -sid SymmID set -no_direct_io
```

By setting this, you are indicating that hosts running SYMCLI versions prior to version 5.3 will not be able to write to the database.

If your environment is running Solutions Enabler SYMCLI version 5.3 (or higher) and Engenuity version 5670, enabling this attribute will provide additional security to your database.

Note: Type 4 and 5 databases initialize with direct I/O writes blocked by default.

How to view the VCMDB

Once you have initialized your database and created records, you can view various information about the database. In addition, you can view the login history details.

Viewing the database

You can examine the entire contents of the device masking VCMDB in its entirety, by director and port, by WWN name (or alias), or by iSCSI name (or alias) using the following syntax:

```
symmaskdb -sid SymmID | -file Filename [-v]
[-dir all [-p all] | -dir # [-p <#|all>]]
[-wwn wwn | -awwn awwn |
-iscsi iscsi | -aiscsi aiscsi]
list database
```

Example: Entire VCMDB

For example, to view the device masking VCMDB on Symmetrix 6196, enter:

```
symmaskdb -sid 6196 list database
```

The following is sample output from this command:

```
Symmetrix ID           : 000000006196

Database Type          : Type4
Last updated at        : 04:58:00 PM on Tue Mar 25,2004

Director Identification : FA-2A
Director Port           : 1

User-generated
Identifier      Type  Node Name      Port Name      Devices
-----
10000000c9238053 Fibre api145        i@1f,4000,02   0040:0043
10000000c924e04a Fibre HOST.23.65.70 10000000c924e04a 00BC:00BF
                                00C3:00C6

Director Identification : FA-2B
Director Port           : 1

User-generated
Identifier      Type  Node Name      Port Name      Devices
-----
10000000c9238053 Fibre api145        i@1f,4000,02   None
```

Example: iSCSI connected type 4 database

The following is sample output for a Type 4 database connected through iSCSI:

```
Symmetrix ID           : 000000006208

Database Type          : Type4
Last updated at        : 03:29:45 PM on Fri Jul 25,2004

Director Identification : SE-3A
Director Port           : 0

User-generated
Identifier      Type  Node Name      Port Name      Devices
-----
iqn.2002-06.com* iSCSI iSCSI          microsoft:api210 0001
                                0005:0007
```

0047
004F

Director Identification : FA-14A
Director Port : 0

Identifier	Type	User-generated		Devices
		Node Name	Port Name	
2234567812345678	Fibre	2234567812345678	2234567812345678	0060
1234567812345678	Fibre	1234567812345678	1234567812345678	0060

Example: VCMDB records for a director

To examine the database for records concerning director 2b, port 1 on Symmetrix 6196, enter:

```
symmaskdb -sid 6196 list database -dir 2b -p 1
```

The following is sample output from this command:

```
Symmetrix ID           : 000000006196
Last updated at       : 04:58:00 PM on Tue Mar 25,2004
Director Identification : FA-2B
Director Port         : 1
```

Identifier	Type	User-generated		Devices
		Node Name	Port Name	
10000000c9238053	Fibre	api145	i@1f,4000,@2	None

You can examine the masked assignment of devices to a specific HBA using the following syntax:

```
symmaskdb -sid SymmID list devs -wwn wwn | -awwn awwn|-iscsi iscsi |
-aiscsi aiscsi
```

For example, to examine the devices on Symmetrix 6196 to which host 10000000c9238053 has access, enter:

```
symmaskdb -sid 6196 list devs -wwn 10000000c9238053
```

The following is sample output from this command:

```
Symmetrix ID           : 000000006196
Originator Port wwn    : 10000000c9238053
User-generated Name    : api145/i@1f,4000,@2
```

Sym Dev	Dir:P	Physical Device Name	VBUS	TID	LUN		Attr	Cap(MB)
Name					SYMM	HOST		
0040	2A:1	/dev/rdisk/c1t0d1s2	0	0	1	1		187
0041	2A:1	/dev/rdisk/c1t0d2s2	0	0	2	2		187
0042	2A:1	/dev/rdisk/c1t0d3s2	0	0	3	3		187
0043	2A:1	/dev/rdisk/c1t0d4s2	0	0	4	4		187

Viewing device capacity

You can view the capacity of devices assigned to a particular host with the following command:

```
symmaskdb -sid SymmID list capacity -host HostName
```

When a hostname is specified, such as `api145` for example, SYMAPI looks for records with an the AWWN value of `api145/<wwn>` and pull that record for the device information. If the default AWWN alias has been modified to something other than the host name, this command won't include information for that HBA, or may appear incomplete if only 2 out of 3 HBAs for a given host have the hostname in the alias.

For example, to view the capacity of host `api145` on Symmetrix 6196, enter:

```
symmaskdb -sid 6196 list capacity -host api145
```

The following is sample output from this command:

```
Symmetrix ID      : 000000006196

Host Name         : api145
Identifiers Found : 10000000c9238053

Device  Cap(MB)  Attr  Dir:P
-----  -
0040      187      2A:1
0041      187      2A:1
0042      187      2A:1
0043      187      2A:1

-----

MB Total:      748
GB Total:      0.7
```

This command requires that the first part of the HBA alias be the host name.

You can view which HBAs have been assigned to specific devices with the following command:

```
symmaskdb -sid 6196 list assignment -dev 0040:0043
```

The following is sample output from this command:

```
Symmetrix ID : 000000006196

Device  Identifier          Type  Dir:P
-----  -
0040    10000000c9238053  FIBRE  2A:1
0041    10000000c9238053  FIBRE  2A:1
0042    10000000c9238053  FIBRE  2A:1
0043    10000000c9238053  FIBRE  2A:1
```

Note: The `list database` and `list devs` commands can be targeted to a backup database file on your host by replacing the `-sid` option with a `-file` option that specifies your backup filename.

Viewing the login history table

The `symmask list logins` command is used to view the login history table. This table in the Symmetrix array lists which hosts and HBAs are logged on to a Symmetrix array for all directors and their director ports. You can also use options to restrict this data to a specific director and/or port number.

For example to return login information for Symmetrix 6196, enter:

```
symmask -sid 6196 list logins
```

The following is sample output from this command:

```
Symmetrix ID           : 000000006196
```

```
Director Identification : FA-2A
```

```
Director Port          : 1
```

Identifier	Type	User-generated Node Name	Port Name	FCID	Logged In	On Fabric
10000000c9238053	Fibre	api145	i01f,4000,02	260e13	Yes	Yes
5006048000060d21	Fibre	NULL	NULL	261e13	No	Yes

The identifier field indicates which HBA is communicating with the Symmetrix array. User-generated node and port names are identified as the AWWN or AISCsi alias associated with it. Columns labelled On Fabric and Logged In indicate whether the HBA is connected to a fabric and whether it is logged in to the Symmetrix system.

You can use the verbose (`-v`) option to view the last active login information.

Refreshing the database

The device masking records are located in two places on the Symmetrix array: the database and the local directors. When a record is written to the array, it is stored in the database and then pushed out to the directors at a later time. A `symmask refresh` command will trigger all records within the database to be pushed out to the local directors. A login by an HBA only triggers the record for that specific HBA to be updated.

To view what is currently stored on the local directors to see what (if any) affect a `symmask refresh` command will have, use the `-verify` option, as shown in the following example:

```
symmask refresh -verify -sid 0237
```

The example output follows:

```
Symmetrix ID           : 000190300237
```

```
Database Type          : Type6
```

```
Last updated at        : 12:27:58 PM on Thu Sep 13,2007
```

```
Director Identification : FA-16C
```

```
Director Port          : 0
```

```
VCM Enabled            : No
```

```
Originator Port wwn    : 5006048ad5f004ce
```

```
Type                  : Fibre
```

```
- Record Location      : Director only
```

```
- Devices               : 0011:001E,0037:0040,0048:004B,0C1B:0C20,  
                        0C28:0C38,0C96
```

Director Identification : SE-2D
 Director Port : 0
 VCM Enabled : No

iSCSI Name: iqn.1991-05.com.microsoft:user.eng.emc.com
 Type : iSCSI
 Record Location : Both, Database and Director
 Devices : Do Not Match
 - Director Devices : 0033:0036

iSCSI Name: iqn.1991-05.com.microsoft:user.eng.emc.com
 Type : iSCSI
 - Record Location : Director only
 - Devices : 0033:0036

iSCSI Name: iqn.1991-05.com.microsoft:user.eng.emc.com
 Type : iSCSI
 - Record Location : Director only
 - Devices : 0033:0036

iSCSI Name: iqn.1991-05.com.microsoft:user.corp.emc.com
 Type : iSCSI
 - Record Location : Director only
 - Devices : 0033:0036

iSCSI Name: iqn.1991-05.com.microsoft:user.eng.emc.com
 Type : iSCSI
 - Record Location : Director only
 - Devices : 0033:0036

iSCSI Name: iqn.1991-05.com.microsoft:api188
 Type : iSCSI
 - Record Location : Director only
 - Devices : 0033:0036

Director Identification : FA-16D
 Director Port : 0
 VCM Enabled : No

Originator Port wwn : 10000000c9239664
 Type : Fibre
 Record Location : Both, Database and Director
 FCID Lockdown : On
 FCID Values : Do Not Match
 + Database FCID : 123456
 - Director FCID : 624e13
 Port Flag Overrides : On
 + Database Enabled : Disable_Q_Reset_on-UA(D)
 Environ_Set(E)
 Disabled Flags : Match
 Devices : Do Not Match
 Matching Devices : 001A:001C
 - Director Devices : 0017:0019,0037:0041,0C1B:0C2E,0C43,0C95,
 0C99:0CB6,0CC1:0CE8

Originator Port wwn : 210000e08b04daac
 Type : Fibre
 Record Location : Both, Database and Director
 FCID Lockdown : On
 FCID Values : Match
 Lun Offset : On
 Offset Value : Do Not Match
 + Database Offset : 0100
 - Director Offset : Not Set
 Base Value : Do Not Match
 + Database Base : 0020

```
-      Director Base      : Not Set
      Devices             : Do Not Match
      Matching Devices    : 0017:0019
-      Director Devices    : 001A:001C, 0037:0041, 0C1B:0C2E, 0C43, 0C95,
                           0C99:0CB6, 0CC1:0CE8
```

Legend: + = Added from database, - = Removed from director

This output is similar to the `list database -v` output, with the following differences:

- ◆ Includes the state of the VCM flag on the port.
- ◆ The first column contains either a minus (-) or a plus (+) when information within the records do not match, or when the record is located in only one location and will be completely added or removed.
- ◆ Attributes, such as LUN offset, which are not set in the record(s) are not displayed.

HBA initiator management

An HBA can be specified in `symmask` commands by its unique WWN, by an AWWN alias associated with the WWN, or by an iSCSI name (or its alias). These names appear in the Symmetrix login history table and in the VCMDDB.

Note: Fiber HBAs have a host WWN and a port WWN for each port on the HBA. Most HBAs have only one port, but some have two. In rare cases, an HBA has four ports. HBA identification in device masking always refers to a port WWN.

When you issue the `symmask discover hba` command to update the login history table, the Symmetrix API (SYMAPI) checks the VCMDDB to determine whether an AWWN exists for each WWN record. If not, SYMAPI creates an AWWN that consists of two parts (the name of the host and the name of the HBA) and writes it to the login history table. However, you can rename the AWWN to a shorter name, if you prefer.

The following command allows you to examine AWWNs in the login history table of Symmetrix 814:

```
symmask -sid 814 list logins
```

Creating and renaming aliases

To assign an AWWN to fit your naming requirements, you can use the `symmask rename` command at any time—even before SYMAPI generates an AWWN. For example, the following command assigns the two-part name `Solaris3A/b4` as the AWWN for WWN `20000000c920b484`:

```
symmask -sid 814 -wwn 20000000c920b484 rename Solaris3A/b4
```

If later you decide you want to replace all user-defined AWWNs (like `Solaris3A/b4`) with system-generated AWWNs, the following command overwrites all existing AWWN entries in the VCMDDB:

```
symmask discover hba -rename
```

You can identify the HBA in `symmask` commands through an iSCSI name. The iSCSI is a SCSI-over-IP protocol that tries to take advantage of the IP network for SCSI traffic rather than requiring a new Fibre network. An iSCSI name is used like an IP address or a WWN and is displayed using the `symmask list logins` or the `symmask list hba` commands. A unique iSCSI name is determined by the hardware that logs into a Symmetrix array and, like the AWWN, a two-part alias name is generated from the `symmask discover hba` command.

Swapping a new HBA for a failed HBA

In the event a host adapter fails, or needs replacement for any reason, you can replace the adapter and assign its set of devices to a new adapter by using the `replace` action in the following form:

```
symmask -sid SymmID -wwn wwn|-awwn awwn|-iscsi iscsi|-aiscsi aiscsi  
replace wwnNew | iscsiNew
```

To swap HBAs, it is suggested to:

1. Run `symmask list logins` to view the old WWN/iSCSI HBAs.
2. Swap HBA boards.

3. Run `symmask list hba` or `discover` to view the new initiator (for example WWN).
4. Run `symmask replace` to substitute a new WWN for all occurrences in the database of the old WWN. For example, to replace old WWN 20000000c920b484 with new WWN 20000000c920b393:

```
symmask -sid 814 -wwn 20000000c920b484 replace 20000000c920b393
```

5. Run `symmask discover -rename` or `symmask rename` to establish the new AWWN and assign an AWWN to the new HBA in both the VCMDB and the login history table.
6. Run `symmask refresh` to update the director profile tables (in cache) from the database.

Deleting HBA associations

You can also delete (in the database) the set of devices associated to a host adapter by using the `symmask delete` action with the following syntax:

```
symmask -sid SymID delete -wwn wwn|-awwn awwn|-iscsi iscsi|-aiscsi aiscsi
```

For this database record deletion, you can restrict the action to just devices on a specific Symmetrix director and port with the following option:

```
[-dir #|all -p #|all]
```

In addition, on arrays running Enginuity 5671 or higher, you can use the `-login` option to delete the entry from the login history table as well.

To delete a set of devices associated to a host adapter, it is suggested to:

1. Run `symmask delete` action.
2. Update the Symmetrix array with the configuration change by performing a `symmask refresh`. This calls the Symmetrix director to refresh its WWN/iSCSI-related profile tables in cache with the contents of the device masking VCMDB.



CAUTION

Before running the `symmask refresh` or the `symmask refresh -verify` commands, make sure there are no HBAs accessing devices in the masked channel (applications running or user activity).

Fibre Channel-to-host interface management

Using the device masking commands, you can adjust the protocol characteristics of the Fibre Channel-to-host interface to be compatible with your host platform-specific requirements.

For your specific host communication protocol, the `symmask set` command allows an *advanced* user to adjust the following attributes on a host adapter port basis:

- ◆ Fibre Channel ID (FCID) lock down
- ◆ Device LUN visibility
- ◆ LUN base/offset skip
- ◆ Heterogeneous host configuration

A record for the host adapter port assignment must already exist in the VCMDDB for these channel attributes to be set.



CAUTION

Do not proceed with any of these adjustments unless you are comfortable with your understanding of the details of your HBA interfaces. Improper settings can disable the use of your host with the Symmetrix array.

Locking down a Fibre Channel ID

Fibre Channel ID (FCID) lockdown is a security feature that limits host device access by adding Fibre Channel ID information of a switch within a fabric to device access records in the device masking VCMDDB. This feature handles WWN spoofing and the threat it poses to your networked systems in a shared (same director port) storage port configuration.

For example, to implement the Fibre Channel ID lockdown feature on Fibre Channel 021300 for director 16A, port 0, enter:

```
symmask -sid 018 set lockdown on 021300 -awwn SolarB/1f,0,fca@1,0 -dir 16A -p 0
```

This feature lets you set the Fibre Channel ID (FCID) of the WWN of the HBA you want to protect. The FCID is then added to the database record for the WWN of the specified HBA with the specified director and is locked. Once a Fibre Channel ID is locked, no user with a spoofed WWN can log in. If a user with a spoofed WWN is already logged in, that user loses all access through that HBA.



CAUTION

When an HBA logs into a director port, the Fibre Channel ID accompanies it, telling the director port where to send its response. By specifying Fibre Channel ID information of the switch (in addition to the WWN of the HBA in the device masking record), the valid physical path through the SAN for a particular HBA is locked down. Only an HBA with a Fibre Channel ID that matches the FCID specified in the device masking record is able to log in to the storage port. If the incorrect Fibre Channel ID is added to the device masking VCMDDB, that HBA will lose access and the host utilities may hang on the server with the locked out WWN. It is recommended that at least two HBAs be available on the administrator host. If one HBA becomes locked out, the host will have access through the other HBA and can correct the record in the database.

Lockdown steps

To find the Fibre Channel ID, lock it down, verify that it is locked down, and then force the change to take effect, use the following procedure:

1. Find the WWN. If the device for the device masking VCMDDB is visible, run `symmask list hba` to find the device path of the HBA you want to protect.

Note: If the VCMDDB is unmapped, no PDEVs will be visible when the `symmask list hba` command is issued. The `sympd` command must be called to set an alternate path.

2. Find the Fibre Channel ID value by using one of the following methods:
 - Run `symmask list logins -pdev`, specifying the device path you found in step 1, to find the Fibre Channel ID of the WWN of the HBA you want to protect.
 - Find the Fibre Channel ID value on the switch, refer to [“Finding the FCID of a switch.”](#)
3. Run `symmask set lockdown set to on` with the FCID of the Fibre Channel ID you found in step 2.
4. Run `symmaskdb list database` in verbose mode (`-v`) to verify that the Fibre Channel ID is locked down.
5. Either reboot the host or pull the cable from the director and then replace the cable. This causes the change to take effect. If you reboot, you must run `symcfg discover` to refresh the SYMAPI database.

Effects on other commands

This section describes how locking down a Fibre Channel ID affects other commands:

- ◆ `symmask delete` — Locking down a Fibre Channel ID has no effect on the `delete` action. The specified record is completely cleared from the database.
- ◆ `symmask replace` — Locking down a Fibre Channel ID has no effect on the `replace` action when the cable is simply moved from one HBA to another and not moved at the switch. In this case, the Fibre Channel ID value that is already in place in the database remains the same for the new HBA.

However, if the cable is moved from one port on the switch to another, the FCID value changes. Do not unlock the Fibre Channel ID during this swap. Instead, leave at least one path open to the database device, and reset the FCID value after the swap by recalling the `set` action.

Since you do not have a path from the HBA whose Fibre Channel ID you want to lock down, you cannot use `symmask list logins` to find the FCID value. Instead you must obtain the FCID value from the switch.

Finding the FCID of a switch

This section describes how to find the Fibre Channel ID on Connectrix® and Brocade switches:

- ◆ **Connectrix switch** — Through the hardware view, click the board and then the port of the switch whose Fibre Channel ID you want to find. Right-click to display the port properties window that includes the FCID value.
- ◆ **Brocade switch** — Telnet to the switch and run `nsShow`. Look for the PID value of the WWN of the HBA you want to protect, which is the Fibre Channel ID value.

Format of a FCID

The Fibre Channel ID basically incorporates the port and the domain ID of the switch in the fabric into which the HBA is plugged.

Connectrix ED-1032 and Brocade 1000 series:

2 2 0 4 1 3

Underlined text is the domain. ***Bold Italic text*** is the port.

In this example, the domain is 2 and the port is 04.

Note: For Connectrix, the port is offset by 4.

Brocade 2000 series DS-16B:

0 2 1 3 0 0

Underlined text is the domain. ***Bold Italic text*** is the port.

In this example, the domain is 02 and the port is 3.

Setting device LUN visibility

The device LUN visibility feature allows the host driver to discover devices with noncontiguous LUN addresses. During the process of discovery, the host operating system scans for LUNs starting at 000 and continuing to a point where it does not find a LUN in the sequence. If there is no LUN 000 on the target director, or there is a break in the sequence of LUNs on that target, some operating systems (notably HP-UX and Linux) do not detect the remaining LUNs and fail to discover noncontiguous devices.

To allow your host to detect all these devices, turn on the device LUN visibility feature so that all devices attached to a specified FA director and port are made available to the HBA. For example, to turn on visibility for FA director 16A, port 0, when working with a host HBA whose WWN is 20000000c920b484:

```
symmask -sid 814 set visibility on -dir 16A -p 0 -wwn 20000000c920b484
```

Allowing the host to scan the other devices past a break in the LUN sequence does not change how the host accesses them. For example, if an HP-UX host has LUNs 0000, 0002, 0003, 0004 assigned to it, that is exactly what the host will see. If you need the host to access this same sequence without the break between 0000 and 0002, you can adjust host visibility by offsetting LUN addresses as described in the [“Setting the LUN base/offset skip adjustment.”](#)

Setting the LUN base/offset skip adjustment

Certain host platforms require LUN 000 to be present when it scans the interface for devices. For this feature, LUN refers to the SCSI LUN number. Also, these host types and others cannot see devices beyond the initial contiguous LUN sequence (they cannot skip over masked holes in an array of intended devices). In a device masking environment, this can be a problem when you need to mask out certain devices from the visibility range of certain host platforms.

For these host platforms, the device masking LUN base/offset skip adjustment feature (with 5x68 minimum) provides the ability to specify a LUN *base* and an *offset* hexadecimal value for the skip hole (recorded in the database). When the host asks for a LUN that is equal to, or greater than the skip hole base value, the offset is added to the LUN value requested by the host to render the actual LUN (device) in the

Symmetrix array. The *base* value is essentially the host's first missing LUN in the skip hole. The offset is the hole size (number of addresses needed to skip over the hole).

To set LUN base and offset values for a skip hole within an HBA to director channel, use the following syntax:

```
symmask -sid SymmID set lunoffset on offset base -awwn awwn -dir # -p #
```

For example, (with director 16A/port 0) to make LUNs (devices) 005 through 008 available to host HPB03/1, you need a LUN base address of 000 and an offset of 5 (to skip over 000-004):

```
symmask -sid 018 set lunoffset on 5 0 -awwn HPB03/1 -dir 16A -p 0
```

Multiple hosts and broken sequences

On the other hand, if your masked devices for an HBA-to-FA connection had LUN addresses 0000–0003 and 0007–0009, you would need to specify a LUN base address of 0004 and an offset of 0003 (to renumber LUNs 0007–0009 as LUNs 0004–0006). You can record only one gap per HBA-to-FA connection.

If you have multiple hosts that cannot discover devices with noncontiguous LUN addresses, you need to issue the `symmask set lunoffset` command for each host.

As shown in [Table 14](#) for Scenario 1, you could have LUN devices 000 through 006 assigned to Host A and 007 through 00A assigned to Host B. Because in this case, Host B needs to see LUN 000 first, you would have to set `lunoffset on` with a base value of 000 and an offset of 7. For this case, Host A does not have a problem since there is no hole in its assigned device sequence and it starts with 000.

Table 14 LUN base/offset scenarios for multiple hosts with skip holes

	Host A LUNs	Host B LUNs	<i>base</i>	<i>offset</i>
Scenario 1	000-006		-	-
		007-00A	000	7
Scenario 2	000-002		-	-
	007-008		003	4
		003-006	000	3

For Scenario 2, you could have LUNs 000 through 002 and 007 through 008 assigned to Host A. Host B could have LUNs 003 through 006. Host A's 000-002 is not a problem, but LUNs 007 -008 require a skip hole base value of 003 (because the first visible sequence stopped at 002) and an offset of 4 (hole size). Also, Host B's LUNs 003-006 requires a skip base value of 000 and an offset of 3. This scenario would require two commands: one targeting Host A and one targeting Host B. Only one skip hole per HBA channel can be recorded in the database.

Setting the HBA port flags

Beginning with Solutions Enabler V6.4, setting the HBA port flags on a per initiator basis has been simplified and improved. This feature allows specific host flags to be enabled and disabled on the director port.

Note: Setting HBA port flags replaces setting the heterogeneous host configuration flags in the next section. The heterogeneous host configuration types listed in [Table 15](#) continue to be valid, but will not be expanded. To switch to setting HBA port flags, the heterogeneous host configuration must be disabled for a given director/port/HBA and all flags must be reset.

To set (or reset) the HBA flags, use the following form:

```
symmask -sid SymmID set hba_flags
<on <flag, flag, flag...>
<-enable | -disable> | off [flag, flag, flag...]>
-wwn wwn | -iscsi iscsi -dir <#> -p <#>
```

Where:

hba_flags = Sets the record in the database to hold information on the HBA port setting that may differ than the current setting on the FA.

on | **off** = Turns HBA flags on or off.

flag = Specifies the overridden HBA port flags from the values [in brackets]:

Common_Serial_Number	[C]	AS400	[AS4]
Disable_Q_Reset_on_UA	[D]	OpenVMS	[OVMS]
Environ_Set	[E]	SCSI_3	[SC3]
Siemens	[S]	Sunapee	[SCL]
Volume_Set_Addresssing	[V]	Sequent	[SEQ]
Avoid_Reset_Broadcast	[ARB]	SPC2_Protocol_Version	[SPC2]
		SCSI_Support1	[OS2007]

-enable = Enables the specified HBA port flag(s) on a per initiator basis.

-disable = Disables the specified HBA port flag(s) on a per initiator basis.

For example, to turn on HBA flags and enable the Common_Serial_Number and SCSI_3 flags, and disable the Disable_Q_Reset_on_UA flag, on an HBA with the WWN ID 210000e08b0995b7 for Symmetrix 031, director 16A port 0, enter:

```
symmask -sid 031 set hba_flags on C,SC3 -enable -wwn 210000e08b0995b7-dir 16A -p 0
symmask -sid 031 set hba_flags on D -disable -wwn 210000e08b0995b7-dir 16A -p 0
```

The symmaskdb list database -v output has been modified to display the flags that are turned on and off for each HBA initiator that has this feature enabled, as shown in the following example:

```
Originator Port wwn : 210000e08b0995b7
Type : Fibre
User-generated Name : 210000e08b0995b7/210000e08b0995b7
Visibility : No
FCID Lockdown : No
Lun Offset : No
Heterogeneous Host : No
Port Flag Overrides : Yes
Enabled : Common_Serial_Number (C)
          SCSI_3 (SC3)
Disabled : Disable_Q_Reset_on_UA (D)
Dynamic Addressing : No
Authentication State : N/A
```

Devices

: 0006 0010 0029:002A

Setting the heterogeneous host configuration

Note: Setting the heterogeneous host configuration has been superseded by Setting the HBA port flags. The heterogeneous host configuration types listed in [Table 15](#) continue to be valid, but will not be expanded. To switch to setting HBA port flags, the heterogeneous host configuration must be disabled on the array, and all flags must be reset.

Heterogeneous host configuration is a feature (with Enginuity 5x68 minimum) that allows different host types to share a single director FA port even though they may require different port settings for their distinctive interface protocol.

Turning on heterogeneous host, turns on the bits listed in [Table 15](#), (and turns off the bits listed in footnote a) on the given director/port (for the given WWN or iSCSI). If this feature is enabled for one host type for a WWN, it must be disabled for that WWN before a new host type can be assigned.

This feature can be used in conjunction with the LUN offset skip feature to allow the different hosts their own LUN addressing scheme. With that scheme, the devices they see are different from those seen by any other host on the director.

The following syntax is used to set certain heterogeneous host configuration flags to optimize the host-to-director interface:

```
symmask -sid SymmID -wwn wwn -dir # -p # set heterogeneous on HostConfigFlag
```

Possible *HostConfigFlag* values are listed in the last column of [Table 15](#).

Table 15 Host platforms and interface configuration flags (page 1 of 3)

Host platform	Requirements	Bit ^a	Host configuration flag (<i>HostConfigFlag</i>)
AS/400		AS4	AS400
AS/400	Load source extender	AS4, V	AS400_LSE
Bull Escala/AIX			BULL_AIX
Bull Escala/AIX	PowerPath V1.5.x or earlier	D	BULL_AIX_PP15
Celerra		ARB	CELERRA
HP/DEC AlphaServers Tru64 UNIX 5.x FC-SW		OVMS	DEC_UNIX
HP/DEC OpenVMS		SC3, OVMS	DEC_OVMS
Data General AViiON NUMA 25000 Server		D	DG_AViiON
FSC BS2000/OSD Servers		C, D	FSC_2000
FSC PRIMEPOWER GP7000F Series host			PRIMEPOWER
FSC PRIMEPOWER GP7000F Series host	PowerPath V1.5.x or earlier	D	PRIMEPOWER_PP15
FSC PRIMEPOWER GP7000F Series host	VERITAS DMP	C, D	PRIMEPOWER_DMP
Fujitsu Services ICL Open VME		C	ICL_OPEN
Hewlett-Packard HP-UX		C, V	HP_UX

Table 15 Host platforms and interface configuration flags (page 2 of 3)

Host platform	Requirements	Bit ^a	Host configuration flag (HostConfigFlag)
IBM AIX with FC 6227, 6228, 6239		SC3	IBM_AIX
IBM AIX with FC 6227, 6228, 6239	PowerPath V1.5.x or earlier	SC3, D	IBM_AIX_PP15
IBM AIX with FC 6227, 6228, 6239		C, SC3	IBM_AIX_DMP
IBM AIX with FC 6227, 6228, 6239		C, D, SC3	IBM_AIX_DMP_PP15
IBM AIX with EMC Fibre Channel			IBM_EMCC
IBM AIX with EMC Fibre Channel	PowerPath V1.5.x or earlier	D	IBM_EMCC_PP15
Linux			LINUX
Linux		C	LINUX_DMP
Linux	Veritas Cluster Server	D, C, SC3	LINUX_DMP_VCS
NCR MP-RAS/Windows NT			NCR
NCR MP-RAS/Windows NT	Multiple vendor platforms	D	NCR_MP
NCR MP-RAS/Windows NT	If Windows NT is used with TNT, set FBA Env. Sense key to 4; otherwise, set it to 6	E	NCR_NT
NCR MP-RAS/Windows NT	Multiple vendor platforms:If Windows NT is used with TNT, set FBA Env. Sense key to 4; otherwise, set it to 6	D, E	NCR_NT_MP
Novell NetWare			NOVELL
Novell NetWare	Cluster	D	NOVELL_CLUSTER
Windows NT/Windows 2000			WINDOWS
Windows NT/Windows 2000	PowerPath V1.5.x or earlier	D	WINDOWS_PP15
Windows NT/Windows 2000	HP/Agilent controllers	V	WINDOWS_HP
Windows NT/Windows 2000	PowerPath V1.5.x or earlier, HP/Agilent controllers	D, V	WINDOWS_HP_PP15
Windows NT/Windows 2000	VERITAS VxVM DMP	C	WINDOWS_DMP
Windows NT/Windows 2000	HP/Agilent controllers VERITAS VxVM DMP	C, V	WINDOWS_HP_DMP
Windows NT/Windows 2000	PowerPath V1.5.x or earlier, VERITAS VxVM DMP	C, D	WINDOWS_DMP_PP15
Windows NT/Windows 2000	HP/Agilent and PowerPath V1.5.x, VERITAS VxVM DMP	C, D, V	WINDOWS_HP_DMP_PP15
Sequent NUMA-Q		E, C, SEQ	SEQUENT
Sequent NUMA-Q	FC-SW configurations only	E, C, SEQ, V	SEQUENT_FCSW
FSC Reliant UNIX RM series		E, D, S	RELIANT
Sun			SOLARIS

Table 15 Host platforms and interface configuration flags (page 3 of 3)

Host platform	Requirements	Bit ^a	Host configuration flag (HostConfigFlag)
Sun	PowerPath V1.5.x or earlier	D	SOLARIS_PP15
Sun	VERITAS DMP	C, D	SOLARIS_DMP
Sun	Sun Cluster (ealier than 3.0)	C, D, SCL	SUN_CLUSTER
Sun	Sun Cluster (3.0 or later)	C	SUN_CLUSTER30
VERITAS Cluster (VCS), EMC GeoSpan for VCS		D	VERITAS
VERITAS Cluster (VCS), EMC GeoSpan for VCS	VERITAS DMP	C, D	VERITAS_DMP
VERITAS Cluster (VCS), EMC GeoSpan for VCS	VSC 2.0 or later		VERITAS20
VMware		C, SC3*	VMWARE

a. The following defines the host characteristic for each of the bits used in the table:

AS4	AS/400 secondary port
C	Common serial number for multipaths
D	Disable Queue Reset on Unit Attention (UA)
E	Environmental reports to host from Symmetrix
S	Enable Siemens host RM/400 - RM/600
SCL	Enable Sunapee (for Sun PDB clusters)
SC3	SCSI 3 interface
SEQ	Sequent Host (DYNIX/ptx)
OVMS	OpenVMS Fibre connection
V	Enable volume set addressing
ARB	Avoid Reset Broadcast
SPC2	SPC2 Protocol
SC3*	Bit will only be set if Enginuity Version is lower than 5670
0S2007	Enable SCSI_Support1

Prior to using the device masking SYMCLI commands with iSCSI, you must configure your iSCSI driver software and authentication information. This chapter covers the following topics:

- ◆ [iSCSI authentication overview](#) 180
- ◆ [iSCSI CHAP authentication: Enginuity version 56xx](#) 181
- ◆ [iSCSI CHAP authentication: Enginuity version 57xx](#) 183
- ◆ [iSCSI software driver configuration](#) 187
- ◆ [RADIUS server configuration](#) 201

iSCSI authentication overview

The iSCSI authentication is negotiated during the HBA login phase and can be implemented using CHAP. The implementation and configuration process for iSCSI CHAP authentication varies depending upon which Enginuity version is running on your host. Solutions Enabler supports the following Enginuity versions for iSCSI authentication:

- ◆ **Enginuity version 56xx:** For details on configuring iSCSI authentication, refer to [“iSCSI CHAP authentication: Enginuity version 56xx” on page 181](#).
- ◆ **Enginuity version 57xx:** For details on configuring iSCSI authentication, refer to [“iSCSI CHAP authentication: Enginuity version 57xx” on page 183](#).

In addition, Solutions Enabler provides support for a RADIUS (Remote Authentication Dial-In User Service) server for storing authentication information. For details on configuring your RADIUS server, refer to [“RADIUS server configuration” on page 201](#).

iSCSI CHAP authentication: Enginuity version 56xx

CHAP (Challenge Handshake Authentication Protocol) allows you to manage a credential name and a CHAP secret, which are similar to a username and a password, though more secure than the standard Password Authentication Procedure (PAP).

Requirements

Setting iSCSI authentication requires:

- ◆ Symmetrix DMX running Enginuity version 5670 (or later)

Note: If you are running Enginuity version 57xx (5771 and higher), refer to [“iSCSI CHAP authentication: Enginuity version 57xx” on page 183](#).

- ◆ Fig-E board that manages the front-end connections in the Symmetrix array
- ◆ Gig-E-configured port
- ◆ Host systems that provide driver support for iSCSI
- ◆ An initialized device masking database (VCMDB)

Before an iSCSI host can log in and see any devices, the iSCSI name of the host must have a valid VCMDB database entry specifying the director and port from which it is connecting. For details on configuring your iSCSI drivers, refer to [“iSCSI software driver configuration” on page 187](#).

Setting iSCSI authentication with the SYMCLI

Once you have configured your iSCSI driver software, and masked your devices, you can set the credential name and CHAP secret required for iSCSI authentication. The following command sets the authentication for the iSCSI initiator, `iqn.2002-06.com.microsoft.host210`, using the authentication type of CHAP. The `-credential` option and `-secret` option specify the required authentication information:

```
symmask -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
      set authentication -type CHAP -credential MyCredentials -secret
      MySecret
```

Note: To set iSCSI authentication for a Symmetrix array using Enginuity version 57xx (5771 and higher), requires the use of the `symconnect` command instead of the `symmask set authentication` command, which is valid only for Enginuity version 56xx (5670 and higher). For information about using `symconnect` commands, refer to [“iSCSI CHAP authentication: Enginuity version 57xx” on page 183](#).

Showing authentication information

The `symmask show` command returns the authentication information for a specified iSCSI initiator on a specific Symmetrix array. Note that the CHAP secret is never displayed. For example:

```
symmask -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
      show authentication
```

You can also display authentication data using the `symmaskdb list database -v` command.

When you set authentication using the `symmask` command, authentication is automatically enabled. If you decide to disable authentication, the authentication values are retained in the VCMDB, which allows you to enable them at a later time using the `enable` action. The following command disables authentication:

```
symmask -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210  
disable authentication
```

Restoring authentication data and updating the iSCSI driver

When you restore a database, authentication data is restored to the Symmetrix array from the backup file. Keep in mind that the same authentication data also needs to be set in the host's iSCSI driver software if authentication had been changed since the backup file was generated.

If you had changed your authentication data after the backup file was created, the restored authentication data will be out of sync with the authentication data stored in the iSCSI driver software. To rectify this discrepancy, you must update the iSCSI driver software with the same information contained in the restored authentication data. For more information, refer to [“iSCSI software driver configuration” on page 187](#).

To avoid restoring obsolete authentication data, use the `-skip_authentication` option. For example:

```
symmaskdb -sid 814 restore -file MyDevMaskBackup  
-skip_authentication
```

iSCSI CHAP authentication: Enginuity version 57xx

Beginning with Solutions Enabler version 6.0.2 running on Symmetrix arrays using Enginuity version 5771, connection security for iSCSI ports is set with the `symconnect` command. The `symconnect` functionality allows you to set one-way or two-way CHAP authentication between a host HBA and a Symmetrix array.

If a RADIUS server is configured to store the authentication information, you can set the RADIUS server information and enable the Symmetrix array to look there for the authentication data. For more information on the RADIUS server, refer to [“RADIUS server configuration” on page 201](#).

Setting one-way CHAP authentication

With CHAP one-way authentication, the array challenges the host during the initial link negotiation process and expects to receive a valid credential and CHAP secret in response. When challenged, the host transmits a CHAP credential and CHAP secret to the Symmetrix array. The Symmetrix array looks for this credential and CHAP secret in its own CHAP authentication database or on a RADIUS server (if one is set and turned on). Once a positive authentication occurs, the array sends an acceptance message to the host. However, if the array fails to find any record of the credential/secret pair, it sends a rejection message, and the link is closed.

Setting iSCSI authentication for a Symmetrix authenticator requires that you supply the Symmetrix ID, the iSCSI name, the CHAP credential name (username), and the CHAP secret (password). The following command inserts an entry for the iSCSI initiator (`iqn.2002-06.com.microsoft.host210`) into the Symmetrix CHAP authentication database. The `-cred` and `-secret` parameters specify the authentication data that the host should send in response to a challenge by the Symmetrix array.

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
set chap -cred MyCredentials -secret MySecret
```

The CHAP protocol secret value (*MySecret* in this case) is a user-defined string up to 32 ASCII characters, or 64 binary characters (binary values should be prefixed with the string `0x`) for UNIX users. Windows users need to specify a secret between 12 and 16 characters and a credential name string between 8 and 256 characters.

The `set chap` action automatically enables CHAP authentication for the iSCSI initiator. You can use `disable chap` to disable CHAP authentication and `enable chap` to turn it back on.

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
disable chap
```

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
enable chap
```

Use the `delete chap` command to disable authentication, as follows:

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
delete chap
```

Setting two-way CHAP authentication

With CHAP two-way authentication, the host challenges and authenticates the Symmetrix too. Thus, both the host and Symmetrix array act as authenticators, and both act as peers. Configuring two-way authentication requires that you configure a one-way authentication for each communication direction.

[“Setting one-way CHAP authentication” on page 183](#) shows how to set up the Symmetrix array as an authenticator, which covers one direction of communication. This section describes how to configure authentication from the opposite direction.

The following command illustrates how to set up the Symmetrix array as a peer. In this case we must define what credential and secret the Symmetrix array will send when challenged by the host HBA authenticator:

```
symconnect -sid 6208 -dir 2D -p 0 set chap -cred MyPeerCredentials
          -secret MyPeerSecret
```

The `-dir` and `-p` parameters specify the director (2D) and port (0) on the Symmetrix array (6208) through which a host HBA is connected. Make a note of the `-secret` value set here. This value will have to be set in the iSCSI Initiator properties window as described in [“Configuring iSCSI for two-way chap authentication” on page 195](#).

Note: The host authenticator must have a user authentication database that contains matching authentication data for the authentication to be successful.

The `set chap` command automatically enables CHAP authentication for the Symmetrix director and port. You can use `disable chap` to disable CHAP authentication and `enable chap` to turn it back on:

```
symconnect -sid 6208 -dir 2D -p 0 disable chap
symconnect -sid 6208 -dir 2D -p 0 enable chap
```

Use the `disable chap` command to disable authentication, as follows:

```
symconnect -sid 6208 -dir 2D -p 0 delete chap
```

Initializing an authentication database

In most cases, you do not want to clear the data of an existing authentication database. If you are unsure whether an authentication database currently exists for a Symmetrix array, issue a command that displays any existing Symmetrix CHAP or RADIUS authentication database for the array. For example, to view any authentication database if it exists on Symmetrix 6208:

```
symconnect -sid 6208 -list
```

To initialize and clear an authentication database, you must specify a backup file name to safeguard against clearing data in the database that should not be lost. For example, the following commands create backup files for Symmetrix CHAP and RADIUS authentication databases and attempt to write any current authentication data to these backup files prior to initializing:

```
symconnect -sid 6208 init chap -file MyChapInitBackup
symconnect -sid 6208 init radius -file MyRadiusInitBackup
```


Displaying authentication information

You can use the basic `symconnect list` command to display a Symmetrix array's authentication database (for example, any Symmetrix CHAP or RADIUS authentication database for Symmetrix 20).

`symconnect -sid 20 list`

To display just the Symmetrix CHAP authentication database, include the `chap` option. For example:

`symconnect -sid 20 list chap`

Symmetrix ID : 000190300020

Director Identification : SE-2D

Director Port : 0

Protocol : CHAP

Identifier	Type	State	Credential
SE-2D:0	N/A	ENABLED	symm20
iqn.2002-06.com.microsoft.host210	iSCSI	DISABLED	api210
iqn.2002-06.com.microsoft.host211	iSCSI	ENABLED	api211

CHAP credentials have been defined in the Symmetrix authentication database for two iSCSI initiators, making it possible for the Symmetrix to authenticate either iSCSI HBA logging into its port. Credentials have also been defined for the Symmetrix director and port (SE-2D:0) in the event that two-way authentication is required and the Symmetrix array needs to submit authentication data to the host.

To display only a RADIUS authentication database, include the `radius` parameter. For example:

`symconnect -sid 20 list radius`

Symmetrix ID : 000190300020

Director Identification : SE-2D

Director Port : 0

Protocol : RADIUS_SERVER

Server Rank	State	Server Name	IP Address	Port
Primary	ENABLED	AP22	108.15.139.220	1812 <- default
1st Backup	ENABLED	AP23	108.15.139.221	1812 <- default
2nd Backup	ENABLED	AP24	108.15.139.222	1812 <- default

Protocol : RADIUS

Identifier	Type	State
iqn.2002-06.com.microsoft.host210	iSCSI	DISABLED
iqn.2002-06.com.microsoft.host211	iSCSI	ENABLED

Backing up and restoring authentication information

Backing up authentication information on a regular basis ensures that you can restore a good version of the authentication database in case incorrect changes or other abnormalities occur in the current database.

The following command creates a backup version of the Symmetrix user authentication database in a file called `MyChapBackup` (each time you back up the database, you must specify a new file name):

```
symconnect -sid 6208 -file MyChapBackup backup chap
```

The following command restores this authentication database using the backup version in the file `MyChapBackup`:

```
symconnect -sid 6208 -file MyChapBackup restore chap
```

Similarly, you can backup and restore the authentication database on a RADIUS server by specifying `backup radius` and `restore radius`.

To display the contents of a backup authentication file, use the `symconnect list` command with the filename:

```
symconnect list -file MyChapBackup
```

iSCSI software driver configuration

This section provides the requirements and configuration steps for preparing a host system with a native iSCSI initiator and the VCMDB (residing in a Symmetrix DMX) to communicate over the Microsoft iSCSI software driver and the EMC Multi-Protocol Channel Director.

The following is an overview of the configuration process:

1. Collect information about the host computer and the Symmetrix DMX Multi-Protocol Channel director:
 - Get the iSCSI name and IP address of the Symmetrix director from the Symmetrix array.
 - Get the iSCSI name of the initiator from the host computer.
2. Configure information in the VCMDB of the Symmetrix array that will allow the host computer to access the Symmetrix devices desired:
 - Add access to the Symmetrix devices from the host initiator by the iSCSI name to the VCMDB.
 - Add the iSCSI authentication information (if any) about the host initiator.
 - Refresh the database.
3. Update the iSCSI initiator with the Symmetrix information:
 - Establish a target connection using iSCSI between the host computer and the designated Symmetrix director port with/without authentication.
 - Have the host computer logon to the Symmetrix over iSCSI and establish target devices on the host computer that will persist through a reboot.
4. Perform the following disk administration on the devices on the Symmetrix array, if needed:
 - Format
 - Write signatures
 - Assign drive letters

Configuration of the iSCSI driver can be completed with CHAP authentication as described in “Configuring iSCSI with CHAP authentication” on page 188, or without authentication as described in “Configuring iSCSI without CHAP authentication” on page 196. Refer to the procedure that meets your environment’s authentication requirements.

Requirements

For detailed interoperability information, please refer to E-Lab Interoperability Navigator which can be reached at <http://elabnavigator.EMC.com>.

Note: Your systems must be installed and connected to the network before beginning the configuration.

Installing the iSCSI software initiator

From the Microsoft download site, install the Microsoft iSCSI Software Initiator. The setup installs an icon on your desktop.

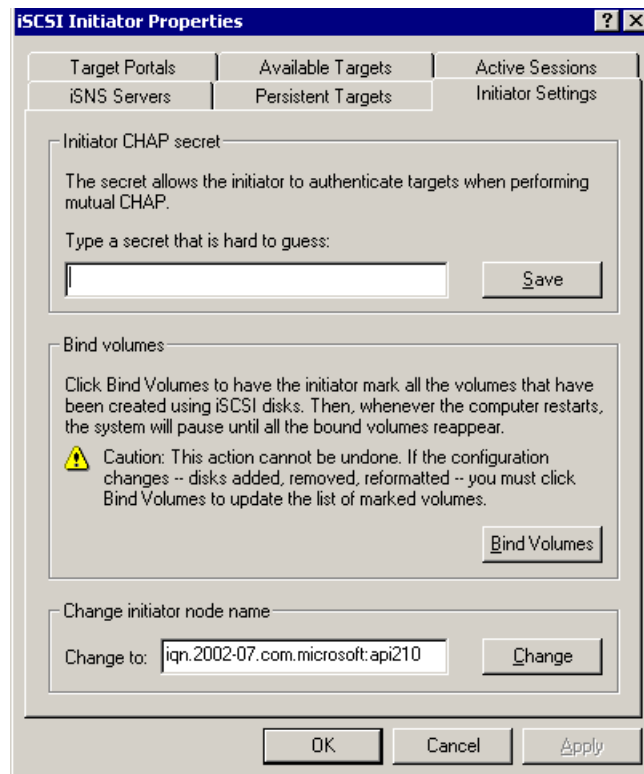
Configuring iSCSI with CHAP authentication

To complete this configuration, you must obtain the iSCSI name of the host, and the iSCSI ID of the multi-protocol director.

Note: This procedure contains steps that require the use of the iSCSI Initiator window and the DOS command window, as follows.

Collect information about the host computer and Symmetrix director

1. Execute the iSCSI Initiator control panel on the host system.
2. Click the **Initiator Settings** tab.



3. Copy the iSCSI ID of your host from the **Change to** field.

Note: Do not enter any data in this window.

4. From the command line, display and copy the IP address of the Symmetrix multi-protocol director, as shown in this example:

```
symcfg -sid 6208 -dir 3a list -v
```

The last two lines of the display contain the iSCSI name and IP address. For example:

```
Symmetrix ID: 000000006208
Product Model           : DMX2000P
Symmetrix ID            : 000000006208
Microcode Version (Number) : 5670 (16260000)
.
iSCSI NAME              : iqn.1992-04.com.emc.5006048000061002
iSCSI IP Address        : 10.10.10.21
```

Configure information in the VCMDB

5. Add a device to create a record in the VCMDB, using the following form:

```
symmask -sid SymmID -iscsi iscsi -dir # -p # add dev #
```

Where:

SymmID — Identifies the Symmetrix ID.
iscsi — Specifies the iSCSI name (from step 3).
 -dir # — Identifies the Symmetrix director number.
 -p # — Identifies the Symmetrix port number.
 add dev # — Specifies the Symmetrix device number(s).

Example: **symmask -sid 6208 -iscsi iqn.2002-07.com.microsoft:api210 -dir 3a -p 0
 add dev 0023**

6. Set the CHAP authentication in the VCMDB using the SYMCLI, which is done differently for each supported Enginuity version:

- For Enginuity version **5771** or higher:

```
symconnect -sid SymmID -iscsi iscsi  

  set chap -cred Credential -secret Secret
```

Where:

SymmID — Identifies the Symmetrix ID.
iscsi — Specifies the host iSCSI name.
Credential — Uses between 8 and 256 alphanumeric characters.
Secret — UNIX: 32 ASCII characters, or 64 binary characters (binary values should be prefixed with the string 0x).
 Windows: 12 and 16 ASCII characters.

- For Enginuity version **5670** or higher:

```
symmask -sid SymmID -iscsi iscsi  

  set authentication -type CHAP  

  -credential Credential -secret Secret
```

Where:

SymmID — Identifies the Symmetrix ID.
iscsi — Specifies the host iSCSI name.
 CHAP — Indicates the authentication type.
CHAPcredential — Uses 8 alphanumeric characters.
CHAPsecret — Uses 12-16 alphanumeric characters.

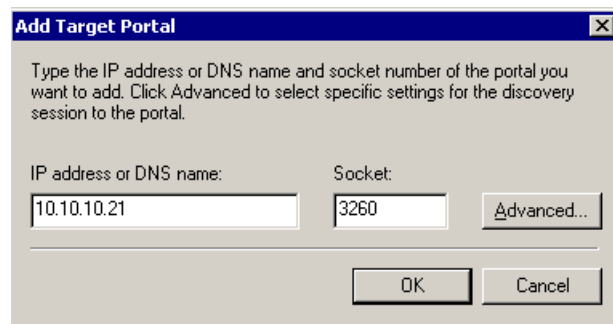
Note: For details about setting iSCSI Authentication, refer to [“iSCSI CHAP authentication: Enginuity version 56xx” on page 181](#) or [“iSCSI CHAP authentication: Enginuity version 57xx” on page 183](#).

7. Refresh the VCMDB, as shown in the following example:

```
symmask -sid 6208 refresh
```

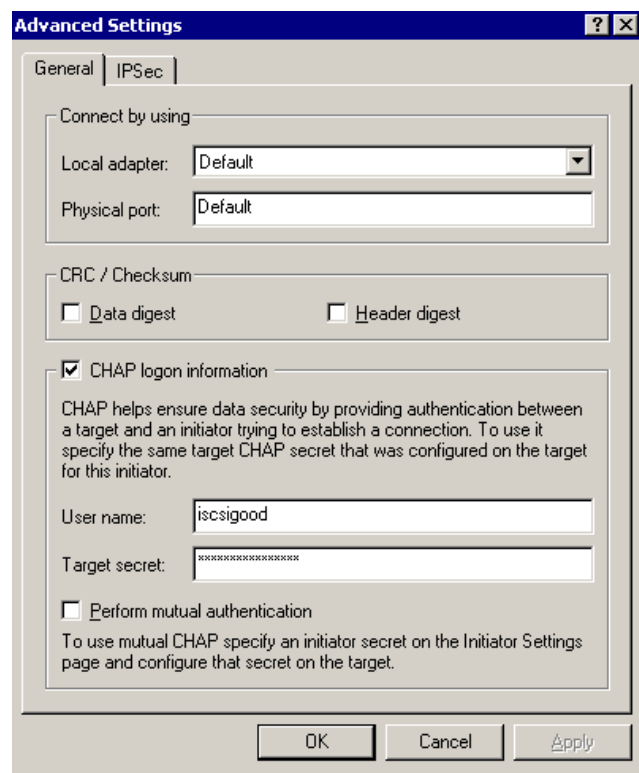
Update iSCSI initiator with Symmetrix information

8. From the iSCSI Initiator Properties window, click the **Target Portals** tab and click **Add**. The **Add Target Portal** dialog box appears.



9. Enter the iSCSI IP address (from step 4) of the Symmetrix multi-protocol director in the **IP address or DNS name** box and click **Advanced**.

The **Advanced Settings** dialog box appears. The iSCSI name automatically displays in the **User name** field. Do not use this for your credential.



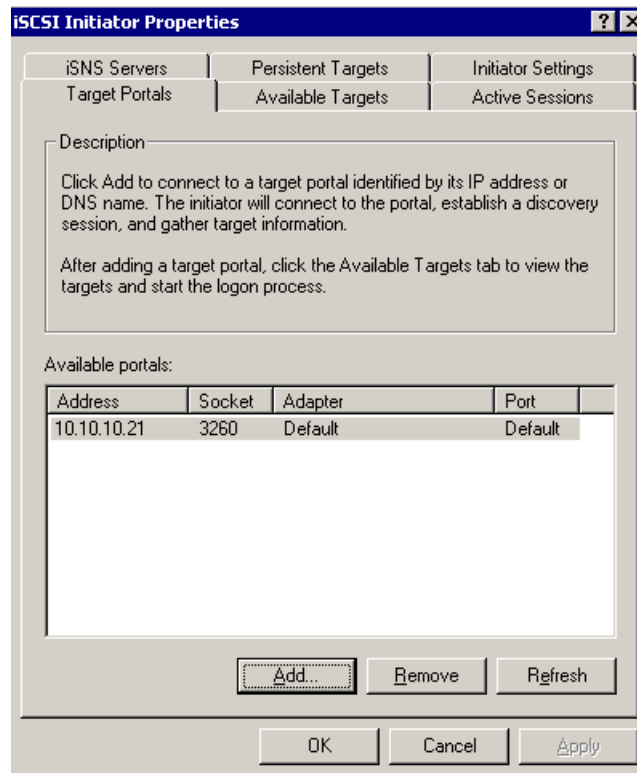
10. Check the box labeled **CHAP logon information**. Change the CHAP credential in the **User name** field, and enter a CHAP secret in the **Target secret** box. Click **OK**.

The credential name string must be between 8 and 256 characters. The CHAP protocol secret value on UNIX can be 32 ASCII characters, or 64 binary characters (binary values should be prefixed with the string 0x). On Windows the secret must be between 12 and 16 ASCII characters.

Note: Do not check or change anything else in this dialog box.

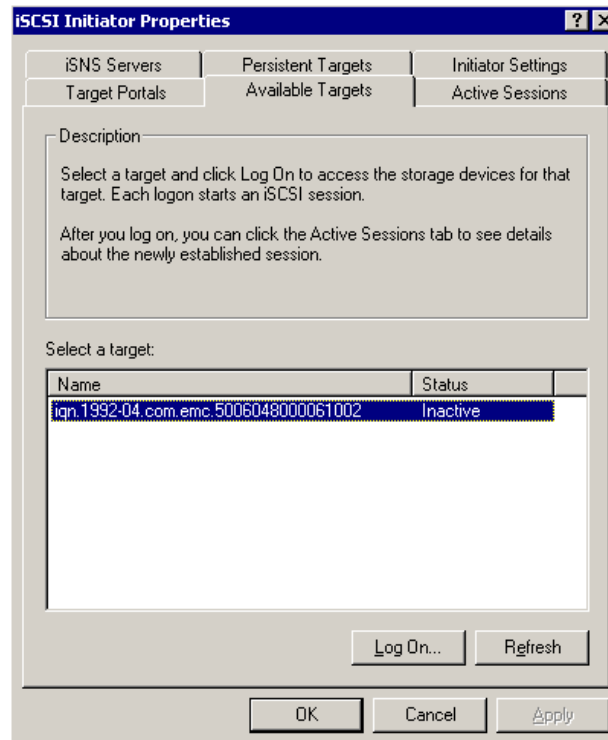
11. Click **OK** to close the **Advanced Settings** dialog box, and click **OK** to close the **Target Portal** dialog box.

The multi-protocol director IP address should appear in the **Available portals** list in the **Target Portals** window.

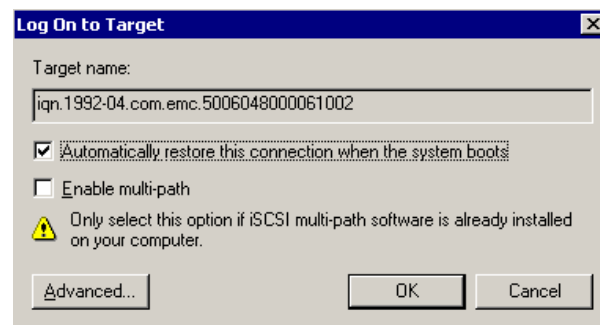


Note: If an error displays, select the IP address from the **Available portals** list and click **Remove**. Begin the configuration procedure again.

12. Click the **Available Targets** tab. The iSCSI name of the Symmetrix multi-protocol director (from step 4) displays in the **Select a target** list.



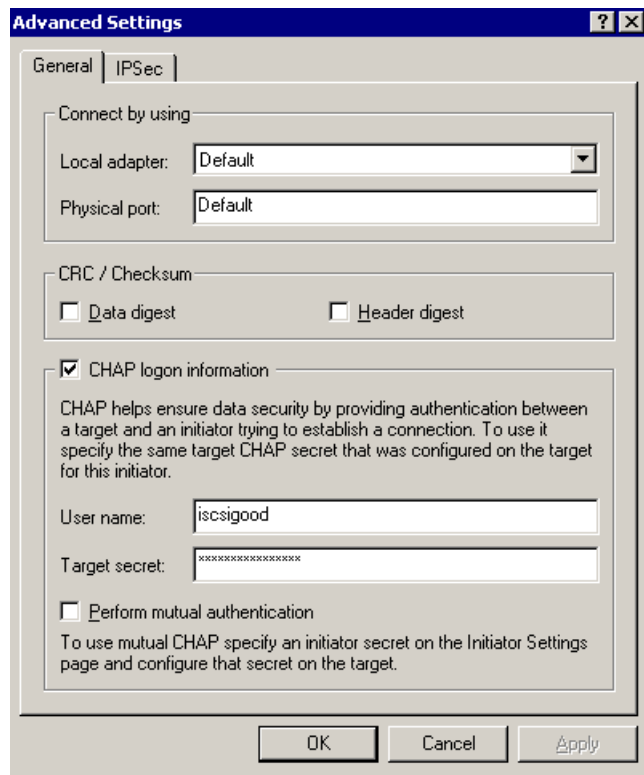
13. Click **Log On**. The **Log On to Target** window dialog box appears.



14. Check **Automatically restore this connection when the system boots**, and click **Advanced**.

The **Advanced Settings** dialog box appears. The iSCSI name automatically displays in the **User name** field.

Do not use this for your credential.

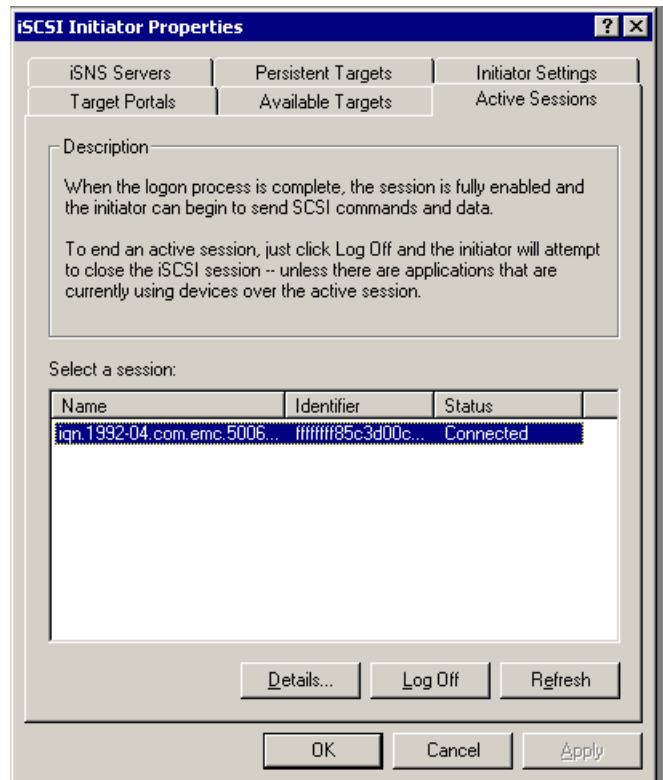


15. Select the box labeled **CHAP logon information**. Change the CHAP credential in the **User name** field, and enter a CHAP secret in the **Target secret** box. Click **OK**.

The credential name string must be between 8 and 256 characters. The CHAP protocol secret value on UNIX can be 32 ASCII characters, or 64 binary characters (binary values should be prefixed with the string 0x). On Windows the secret value must be between 12 and 16 ASCII characters.

Note: Do not select or change anything else in this window.

16. From the iSCSI Initiator Properties window, click the **Active Sessions** tab. An active session should display in the **Select a session** list.



17. Select the session and click **Details** to display the disks found by the iSCSI driver.
 18. Click **OK** to exit from the iSCSI Initiator.

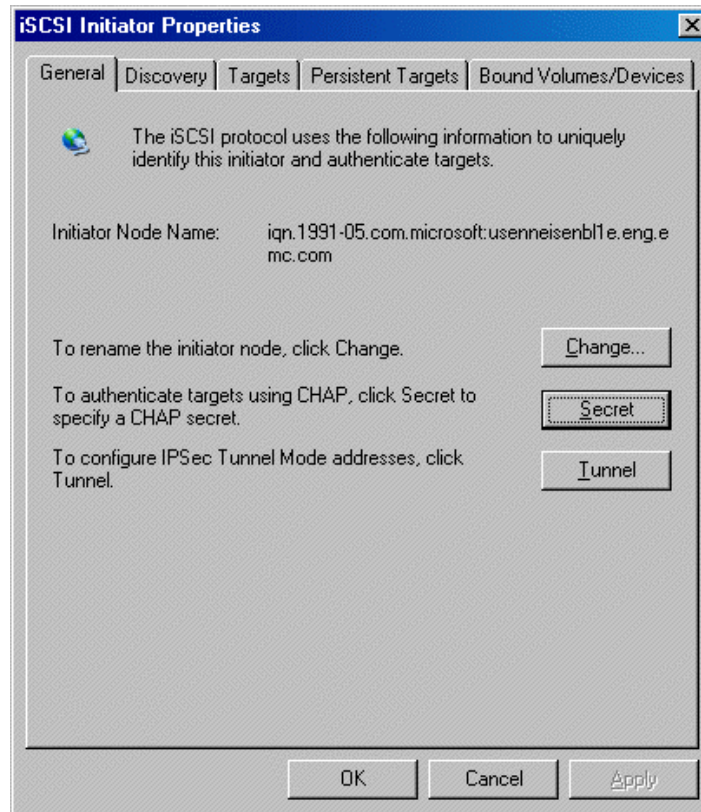
Perform disk administration on the Symmetrix devices

19. Perform any necessary disk administration, such as, formatting, write signatures, and assigning drive letters.
 20. Reboot the host system.

Note: The configuration changes you made will not take effect until you reboot your host.

Configuring iSCSI for two-way chap authentication

For a two-way Chap configuration, after installing your iSCSI Initiator, open your iSCSI Initiator properties, click the **Secret** button on the **General** tab, and set the secret for two-way CHAP authentication to match the value set using `symconnect` as described in [“Setting two-way CHAP authentication” on page 184](#).



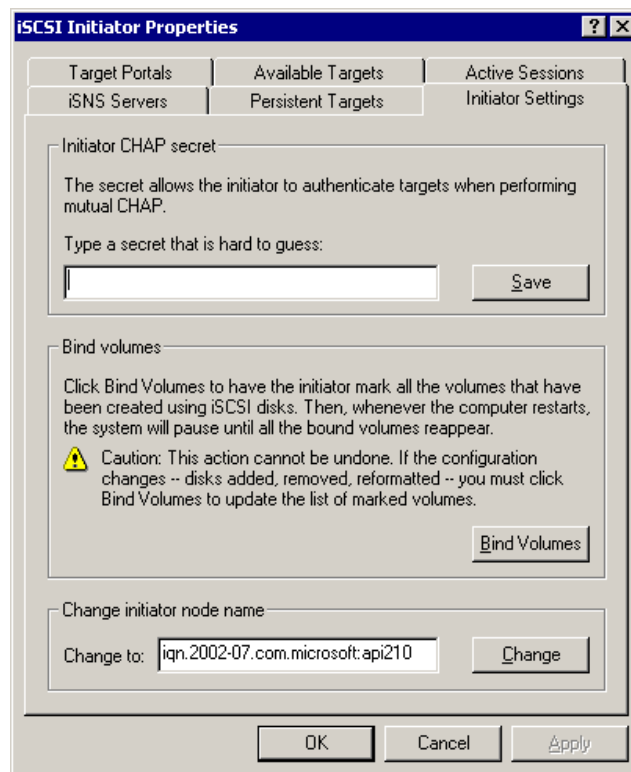
Configuring iSCSI without CHAP authentication

To complete this configuration, you must obtain the iSCSI name of the host, and the iSCSI ID of the multi-protocol director.

Note: This procedure contains steps that require the use of the iSCSI Initiator Properties window and the SYMCLI command window, as follows.

Collect information about the host computer and Symmetrix director

1. Open the iSCSI Initiator on the host system.
2. Click the **Initiator Settings** tab.



3. Copy the iSCSI ID of your host from the **Change to** field at the bottom of the window.

Note: Do not enter any data in this window.

4. List the iSCSI name and IP address of the Symmetrix multi-protocol director, as shown in this example:

```
symcfg -sid 6208 -dir 3a list -v
```

The last two lines of the display contain the iSCSI name and IP address. For example:

```
Symmetrix ID: 000000006208
Product Model : DMX2000P
Symmetrix ID : 000000006208
Microcode Version (Number) : 5670 (16260000)
.
iSCSI NAME : iqn.1992-04.com.emc.5006048000061002
iSCSI IP Address : 10.10.10.21
```

Configure information in the VCMDB

5. From the command line, add a device to create a record in the VCMDB, using the following form:

```
symmask -sid SymmID -iscsi iscsi -dir # -p # add dev #
```

Where:

- *SymmID* = The Symmetrix ID.
- *iscsi* = The iSCSI name copied in step 3.
- *-dir #* = Symmetrix director number.
- *-p #* = Symmetrix port number.
- *add dev #* = Symmetrix device number.

For example:

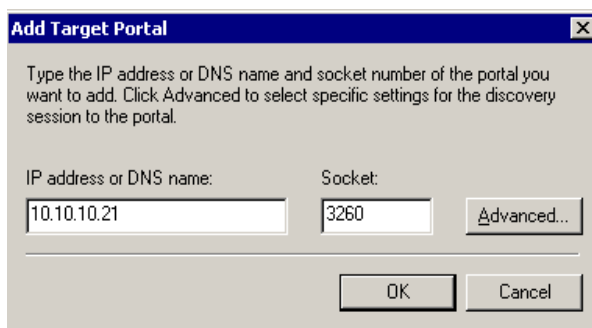
```
symmask -sid 6208 -iscsi iqn.2002-07.com.microsoft:api210 -dir 3a -p 0  
add dev 0023
```

6. Refresh the VCMDB, as shown in the following example:

```
symmask -sid 6208 refresh
```

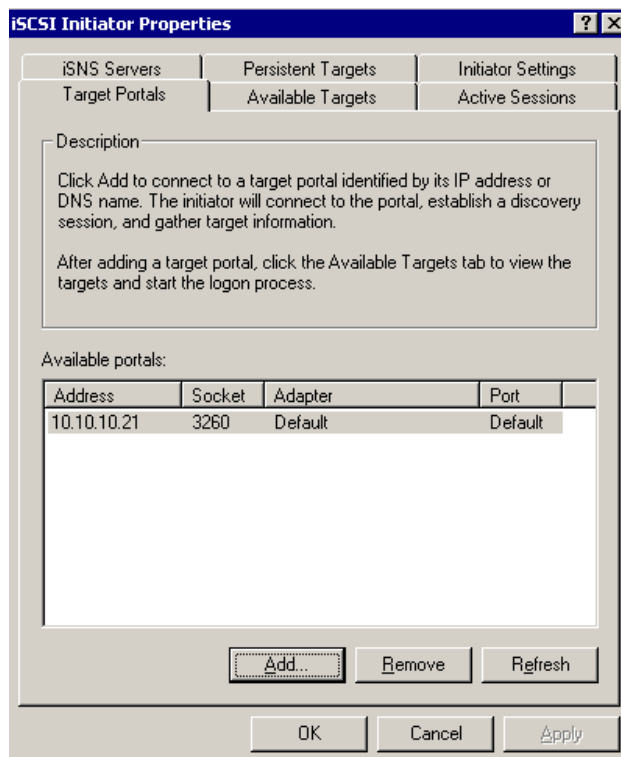
7. From the iSCSI Initiator window, select the **Target Portals** tab and click **Add**.

The **Add Target Portal** dialog box appears.



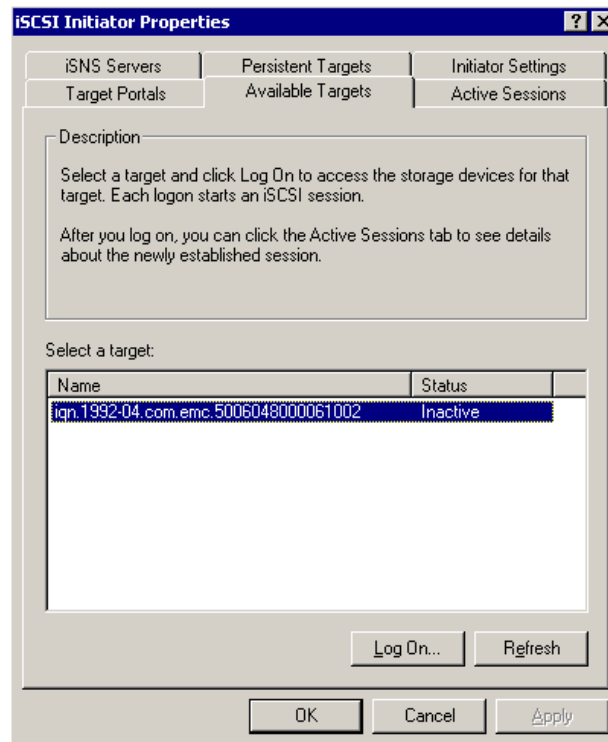
8. Enter the iSCSI IP address (from step 4) of the Symmetrix multi-protocol director in the **IP address or DNS name** box and click **OK**.

The multi-protocol director IP address should appear in the **Available portals** list in the **Target Portals** window.

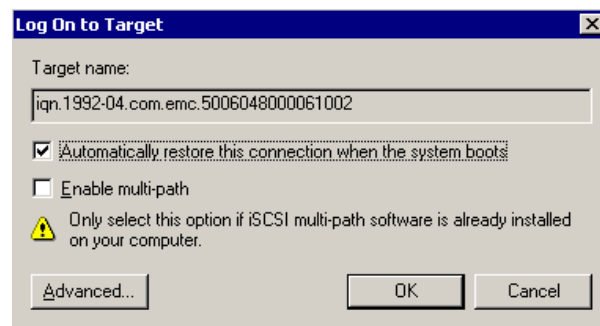


Note: If an error displays, select the IP address from the **Available portals** list and click **Remove**. Begin the configuration procedure again.

9. Click the **Available Targets** tab. The iSCSI name of the Symmetrix multi-protocol director (from step 4) appears in the **Select a target** list.

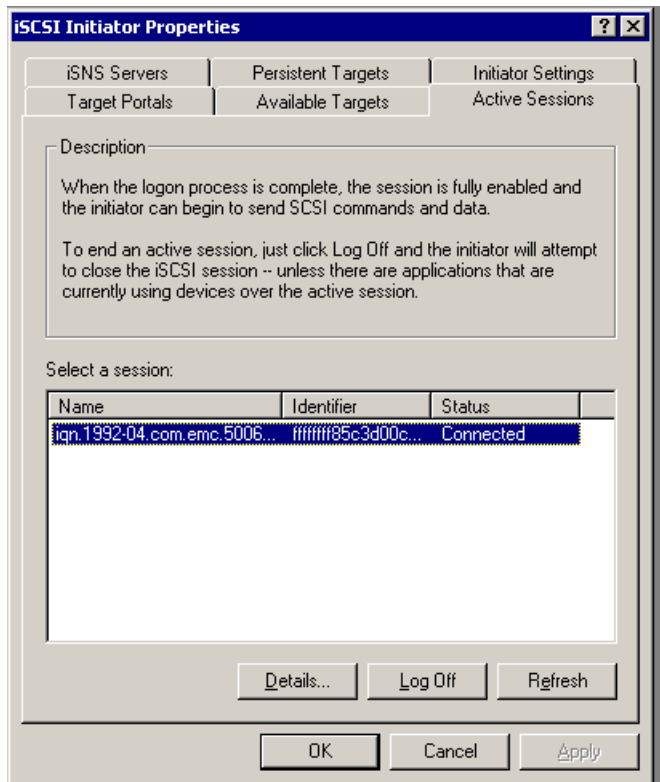


10. Click **Log On**. The **Log On to Target** dialog box appears.



11. Select **Automatically restore this connection when the system boots**, and click **OK**.

12. Click the **Active Sessions** tab. An active session should display in the **Select a session** list.



13. Select the session and click **Details** to display the disks found by the iSCSI driver.
14. Click **OK** to exit from the **iSCSI Initiator Properties** window.

Perform disk administration on the Symmetrix devices

15. Perform any necessary disk administration, such as, formatting, write signatures, and assigning drive letters.
16. Reboot the host system.

Note: The configuration changes you made will not take effect until you reboot your host.

RADIUS server configuration

When using a RADIUS server to store authentication data, you need to use the `symconnect` command to make the SYMAPI server aware of your RADIUS server. Once registered, the Symmetrix SYMAPI server will look to the RADIUS server for matching authentication data when attempting to authenticate a host's credential and secret.

You set the RADIUS server information at the Symmetrix director/port level, letting the Symmetrix know where to go for the authentication information when a host HBA tries to log in. The required information includes the server name, its IP address, its key (a password of up to 256 characters), and its rank (`primary`, `backup1`, or `backup2`). For example:

```
symconnect -sid 6208 -dir 2D -p 0 set radius -server AP22
-key MyServerPassword -ip 108.15.139.220 -rank primary
```

The `-rank` parameter establishes server priority preference if two or three RADIUS servers are specified for a given director-port combination. If a primary and two backup servers are enabled, the system tries them in the obvious order until successful (a server might be unreachable for some reason). If primary is disabled, the system looks to `backup1` and `backup2`.

If the RADIUS server port to be used is different than the server's default port, you can specify it using the `-port` option. You can also include options that specify a server retry interval in seconds (`-i`) and the number of times to retry (`-c`).

The `set radius` command automatically enables RADIUS authentication for the Symmetrix array's director and port. You can use `disable radius` to disable authentication and `enable radius` to turn it back on:

```
symconnect -sid 6208 -dir 2D -p 0 disable radius -rank primary
```

```
symconnect -sid 6208 -dir 2D -p 0 enable radius -rank primary
```

Once you have set and enabled all RADIUS server information and at the Symmetrix director and port level, you can then enable the server authentication feature for a given iSCSI HBA logging into a port. Use either `set radius` or `enable radius` to turn on the RADIUS server authentication for the specified iSCSI initiator. For example:

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
enable radius
```

Similarly, you can use `disable radius` to turn off RADIUS authentication for the iSCSI initiator.

Use `delete radius` to delete the RADIUS authentication entry, as follows:

```
symconnect -sid 6208 -iscsi iqn.2002-06.com.microsoft.host210
delete radius
```


This chapter describes the IPsec standard and how to manage IPsec network policies, using the IPsec component of the SYMCLI. The chapter covers the following topics:

◆ Introduction	204
◆ IPsec and Symmetrix	205
◆ SYMCLI IPsec management component	207
◆ Viewing policy information.....	208
◆ Command file scripting for creating/modifying policies.....	212
◆ Committing a policy-scripted command file	217
◆ Deleting a policy-scripted command file.....	218

Introduction

This section overviews the IPsec standard and how it applies to Symmetrix networks.

IPsec standard

IPsec is a framework of open standards for security of network communication that provides data confidentiality, integrity, and authentication between participating peers at the IP layer of an Ethernet network.

IPsec allows two peers (computers) to trust each other's identity and share a private key for each communication session. Based on the session key, the computer peers establish an encrypted communications channel and can verify that data they receive is what was originally sent. These countermeasures mitigate many security attacks, including spoofing, tampering with data en route, connection hijacking, eavesdropping, and replay of transmission. IPsec is the same security technology used in many corporate VPN solutions, including EMC's.

IPsec technology protects the Internet Protocol (IP) version 4 and 6 (IPv4 and IPv6) standards. IPv4 is the most commonly used, but IPv6, sometimes called IPng (next generation) can provide more addresses, better security, reduced administrative workload, and enhanced support for mobile devices.

More specifically, the IPsec framework is an Internet Protocol standard (RFC 2401) that allows you to create IP network tunnels through an existing IP network. All traffic contained within this tunnel is afforded some configurable measure of protection against hostile exterior entities. This is accomplished by optionally transforming each network packet at the tunnel source in some way, wrapping the result with an IPsec header, and sending this wrapped packet to the destination. The process is reversed at the destination. First, the IPsec wrapper is stripped off, and the packet is transformed again. It then appears as it originally did, and may be processed by conventional means, as if it had never been modified.

IPsec offers three main services to users:

- ◆ Authentication — Verifies that a packet was originated by the sender
- ◆ Integrity — Ensures that packets are not modified in transit
- ◆ Encryption — Hides the packet contents from prying eyes

Not all of these services are necessary or even desirable at the same time, in all situations. Users may selectively employ the ones that fit their current needs.

IPsec policies

An IPsec implementation internally manages database records called policies. They are processed in order, sorting on an attached index number called the policy's priority. In general, policies describe how to select applicable network traffic, and how to apply IPsec protection to it. They contain proposal and transform requirements for the IP tunnel.

More specifically, a policy is a combination of parameters that describe how network traffic will be encrypted. A policy is a single record in the policy database, and is indexed/sorted by Priority. When network traffic appears that might be affected by encryption, the IPsec implementation scans forward, in priority order, through the policy database looking for a policy that matches the traffic. Each policy has proposals and transforms that are also tested, in order. Once a match has been found, searching is terminated, and an entry is created in the Security Associations database that reflects the policy found. Also, each policy has a unique priority.

IPsec and Symmetrix

The Symmetrix subsystem, running Enginuity 5773 in companion with specific crypto processors, provides IPsec traffic services on each of its Gigabit Ethernet (Advanced Multi-Protocol channel) directors. Here, IP network traffic to or from the array can be protected by IPsec applied to it that supports two significant network connection protocols:

- ◆ SRDF — Symmetrix to Symmetrix endpoints, securing SRDF links
- ◆ iSCSI — iSCSI host to Symmetrix endpoints or NAS to Symmetrix endpoints

Figure 19 illustrates an SRDF IPsec link and Figure 20 shows a host endpoint with the iSCSI connection.

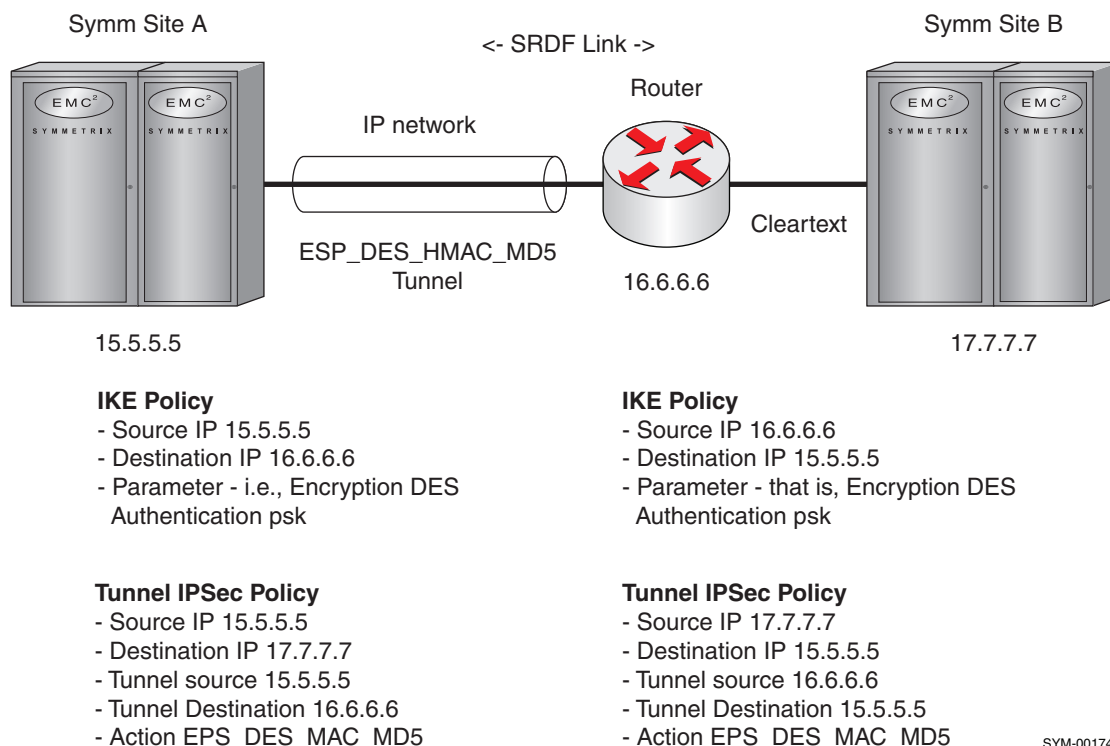


Figure 19 IPsec network tunnel with Symmetrix-to-Symmetrix endpoints

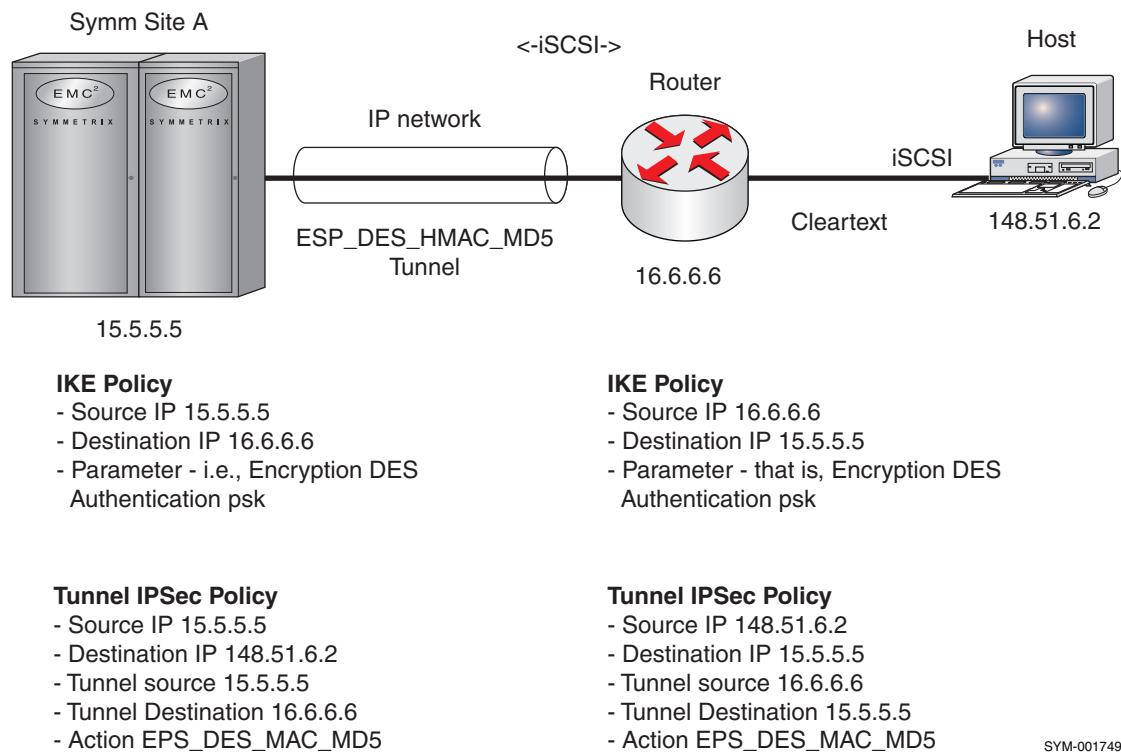


Figure 20 IPsec network tunnel with iSCSI host-to-Symmetrix endpoints

Traffic selection within IPsec is extremely fine-grained, as it is possible to apply IPsec only to iSCSI traffic with a connection to a particular host, or series of hosts, leaving other iSCSI traffic and SRDF untouched. Different destinations can have different types of protection.

Note: With such vast flexibility and versatility, configuration of IPsec is considered somewhat complex. It is recommended that IPsec management, described in this chapter, and the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help* be performed with or by network engineers that understand encryption disciplines and the standard RFC 2401 for IPsec, including details of your specific network and environment. Incorrect intervention in this area, could expose critical information to third parties; especially if the two endpoints are connected with a publicly-visible network.

For more information about the RFC 2401 standard, see the following:

<http://rfc.net/rfc2401.html>

SYMCLI IPsec management component

The SYMCLI IPsec component extends the basic SYMCLI command set to include the Solutions Enabler `symipsec` command. This command allows you to perform all the necessary control operations that manage various types of IPsec security association policy changes in your Symmetrix environment. This IPsec standards compliant command provides policy management to/from the IPsec processor chip on the Symmetrix Gig-E port. It also offers retrieval of policies and associated statistics.

If you employ the Access Control feature with `symacl`, access types BASE and CFGSYM are required.

Information retrieved from the IPsec subsystem is not stored in the SYMAPI database. The IPsec policies are stored in global memory and backed up to the Symmetrix vault, therefore policy database backups and restores are not provided.

The `symipsec` command allows you to display and set values of control parameters for Symmetrix IPsec encryption and authentication support as follows:

- ◆ Lists various active IPsec policies.
- ◆ Shows detail about an active policy.
- ◆ Lists the various statistics and errors in the Engenuity processing of policies.
- ◆ Previews and commits the scripted command file that defines the specific network security policy requirements.
- ◆ Uses command file entries to define the IPsec policy requirements, including details such as proposals, and transforms.

Viewing policy information

With the `symipsec` command, you can return a list of IPsec policies or show detail about a policy being processed on a director and port.

Note: Currently, when specifying port numbers in the `symipsec` command, only port 0 is a valid parameter (and is the default). Specifically targeting Port 1 on a director is not supported.

Listing policies

To list information about policies on a specified Symmetrix array and director, use the following form:

```
symipsec -sid SymmID list -dir #|ALL -port 0
          -priority Level#
          -all
```

Where:

Level# – the policy level index number. The lower the number, the higher the priority level (0-110).

For example, to list the policy priority numbers on director 16D within a specific Symmetrix array, enter:

```
symipsec -sid 343 -dir 16D list port 0 -all
```

The output will list all current policies on the specified director:

```
Symmetrix ID           : 000190300343
Director number        : 16D
Port number            : 0
Policy priorities present :
    0          10      20          30          80          90
    100         110
```

Showing policy detail

To show detail information about policies on a specified Symmetrix array, use the following form:

```
symipsec -sid SymmID show -dir #|ALL -port #|ALL
          -priority Level#
          -all
```

For example, to show the policy details on director 1A for a specific policy (20), enter:

```
symipsec -sid 123456789012 -dir 1A -port 0 show -priority 20
```

For a second example, to show the policy details on director 16D about all current policies, enter:

```
symipsec -sid 343 -dir 16D show -all
```

The output will show detail information about all current policies on the specified director, for example:

```
Symmetrix ID           : 000190300343
Director ID            : 16D
Port number            : 0
Policy priority        : 0
Action                 : Secure
Local Endpoint:
  Endpoint type        : Single
  IP Version            : v4
  IP Address            : 10.0.0.40
  IP Port Number       : ANY
```



```

        IP Protocol Number      : ANY
        Flags                   : N/A
Remote Endpoint:
    Endpoint type               : Single
    IP Version                  : v4
    IP Address                  : 10.0.0.51
    IP Port Number              : ANY
    IP Protocol Number          : ANY
    Flags                       : N/A
Source Port Selectivity        : Packet
Protocol Selectivity           : Policy
Destination Port Selectivity   : Policy
Proposal Set:
    Proposal Set Type           : IKE
    IN SPI                     : 0
    OUT SPI                    : 0
    IN CPI                     : 0
    OUT CPI                    : 0
    IN Nonce                   : 0
    OUT Nonce                  : 0
    IKE Mode                   : Main
    PFS                        : Off
Number of Proposals            : 1
    Proposal #                 : 1
    Proposal type :             : IKE
Number of Transforms           : 1
    Transform #                : 1
    Transform type              : IKE
    Authorization Algorithm     : MD5
    Encryption Algorithm        : 3DES
    Lifetime (bytes)            : 0
    Lifetime (seconds)         : 86400
    ESP Mode                    : Transport
    DH Group                    : DH Group 1
    Authentication Method       : Preshared Key
    Compression Algorithm       : N/A
Symmetrix ID                   : 000190300343
Director ID                    : 16D
Port number                    : 0
Policy priority                 : 10
Action                         : Secure
Local Endpoint:
.
.
.

```

The output pattern repeats for every policy found on the specified director.

Listing policy statistics

To list statistical information about the IPsec processing of a policy or policies, use the following form. You can optionally target a remote endpoint address:

```

symipsec -sid SymmID list -dir #|ALL -port 0
          -stats -type StatsType [-priority Level#] [-remote_addr IPendPt]

```

Where:

StatsType — possible values are:

- ike_errors
- ipsec_details

IPendPt — the IP endpoint address

Level# — the policy level index number. The lower the number, the higher the priority level (0-110).

Examples for retrieving statistical details

For example, to display IPsec SA statistical details for a specific priority (20) on director 1A, enter:

```
sympipsec -sid 123456789012 -dir 1A -port 0 list -stats -type  
ipsec_details -priority 20
```

For a second example, to display IPsec SA statistical details for a specified director 2C, enter:

```
symipsec -sid 067 -dir 2c list -stats -type ipsec_details
```

The output will list statistical detail information about all current policies on the specified director, for example:

```
Statistics from director 02C, port 0
      Local endpoint
Endpoint type      : Single
IP Version         : v6
IP Address         : fd00:abcd:0:0:0:0:1:50
IP Port Number     : 1748
IP Protocol Number : ANY
Flags              : N/A

      Remote endpoint
Endpoint type      : Single
IP Version         : v6
IP Address         : fd00:abcd:0:0:0:0:1:41
IP Port Number     : 48235
IP Protocol Number : ANY
Flags              : N/A

SA Type            : ESP
ESP Mode           : Transport
Encryption Algorithm : AES 128
Authentication Algorithm : XCBC
Compression Algorithm : N/A
Total Rekeys       : 0

      Inbound SA values
SPI                : 4025390203
Byte Limit         : 1099511627776
Seconds Limit      : 5400
Packet Count       : 0
Compressed Packet Count : 0
User Byte Count    : 0
Compressed User Byte Count : 0
Authentication Errors : 0
Replay Errors      : 0
Padding Errors     : 0

      Outbound SA values
SPI                : 4066949324
Byte Limit         : 1099511627776
Seconds Limit      : 5400
Packet Count       : 0
Compressed Packet Count : 0
User Byte Count    : 0
Compressed User Byte Count : 0
```

```
Statistics from director 02C, port 0
      Local endpoint
Endpoint type      : Single
.
.
.
```

The output pattern repeats for every policy found on the specified director.

Examples for retrieving IPsec errors on the director security processor

For example, to display IPsec IKE errors in the security processor on director 2C, enter:

```
symipsec -sid 067 -dir 2c list -stats -type ike_errors
```

The output will list statistical IKE or IPsec errors related to any of the current policies on the specified director, for example:

```
Statistics from director 34, port 0
Endpoint type           : Single
IP Version               : v6
IP Address               : fd00:abcd:0:0:0:0:1:50
IP Port Number          : ANY
IP Protocol Number       : ANY
Flags                   : N/A

      Phase 1 (IKE) errors
Init Failures :          : 0
Response Failures :        : 0
Invalid Cookies :         : 0
No Peer Responses :        : 0
Invalid Peer Responses :    : 0
Logical Failures :         : 0

      Phase 2 (IPsec) errors
Init Failures :          : 0
Response Failures :        : 0
Invalid Cookies :         : 0
No Peer Responses :        : 0
Invalid Peer Responses :    : 0
Logical Failures :         : 0
```

Command file scripting for creating/modifying policies

The command and control of IPsec policy declarations are done through scripting and managing a command file. You can add, modify, or delete policies using various entries in a specified command file.

Command file syntax adding/modifying policies

The following is the command file syntax to add a network policy or modify an existing network policy. Leading pound signs (#) on a command line entry comments out that notation from the command parser:

Note: Currently, you can only define one proposal and one transform per policy declaration.

```
policy add|modify
  -priority Level#
  -action secure|discard|bypass
  [-assoc_ike_policy Level#]
  #only if proposal_type is ipsec
  -local_addr IPAddr
    [-ipproto IPprotocol#|all] [-ipport IPport#|all] [-mask IPAddr]
  -remote_addr IPAddr
    [-ipproto IPprotocol#|all] [-ipport IPport#|all] [-mask IPAddr]
  [-remote_tunnel_addr IPAddr]
  #only if esp_mode is tunnel
  [-selectivity {destip|destport|srcport} SPECIFICITY]
  #for policies with wildcarded ipaddr/port/proto only

  -proposal_set
  -proposal_set_type auto|ike|manual
  [-ike_mode main|aggressive]
  #if proposal_set_type is set to ike mode
  [-pfs on|off]
  #if proposal_set_type is set to ike mode
  [-key_format hex |ascii]
  [-presharedkey Keystring]
  #if proposal_set_type is set to ike mode
  [-inenc_key Keystring -outenc_key Keystring]
  #if transform type is ESP or IKE and proposal is manual mode
  [-inauth_key Keystring -outauth_key Keystring]
  #if proposal is set to manual mode and algorithm is not null
  [-in_spi SPI# -out_spi SPI#]
  #if proposal is set to manual mode and algorithm is AES mode
  [-in_nonce NONCE -out_nonce NONCE]
  #if proposal is set to manual mode

  -proposal
  -proposal_type ike|ipsec

  -transform
  -transform_type ike|esp
  [-auth_alg null|sha1|md5|xcbc]
  [-enc_alg null|des|3des|aes_128|aes_256|aes_cm_128|aes_cm_256]
  #if transform type is esp or ike
  [-dhgroup 1|2|3|4]
  #if transform type is ike
  [-esp_mode tunnel|transport]
  #if transform type is esp
  [-lifetime [LifeParam1][,][LifeParam2]]
  [-auth_method preshared_key|dsa|rsa]
  #if transform type is ike. DSA and RSA not supported.

;
```

Scripting the policy syntax

The beginning section of the policy syntax sets the overall policy requirements. For a policy add or modify task, consider and set the following options:

```
-priority Level#
-action secure|discard|bypass
[-assoc_ike_policy Level#]
-local_addr IPAddr
    [-ipproto IPprotocol#|all][-ipport IPport#|all][-mask IPAddr]
-remote_addr IPAddr
    [-ipproto IPprotocol#|all][-ipport IPport#|all][-mask IPAddr]
[-remote_tunnel_addr IPAddr]
[-selectivity {destip|destport|srcport} SPECIFICITY]
```

Where:

`-priority Level#`

Specifies the index number of the policy to be added or modified. Possible priorities range from 0 to 110. When packets arrive, policies with lower numbered priorities are examined first. Also, any associated IKE policies (`-assoc_ike_policy`) must have a lower priority index number (higher priority) than this object IPsec policy.

`-action secure|discard|bypass`

Setting the policy action to `secure`, requests that IPsec processing will be fully applied to packets that match this policy, requiring packet transformation.

Setting the policy action to `discard`, requests that the matching packets will be dropped completely. For use on unauthorized break-ins.

Setting the policy action to `bypass`, requests that the matching packets will be completely unmodified, and pass straight through the IPsec processor without encryption.

`-assoc_ike_policy Level#`

Specifies the index number of the associated IKE policy to the object IPsec policy. The `-proposal_type` option must be set to `ipsec`. When packets arrive, policies with lower numbered priorities are examined first. Associated IKE policies must have a lower priority index number (higher priority) than the corresponding IPsec policy.

`-local_addr IPAddr`

```
[-ipproto IPprotocol#|all][-ipport IPport#|all][-mask IPAddr]
```

```
-remote_addr IPAddr
```

```
[-ipproto IPprotocol#|all][-ipport IPport#|all][-mask IPAddr]
```

These options primarily define the local and remote endpoint IP addresses. Optionally, you can target an IP protocol number (`IPprotocol#`) or all IP protocols and/or an IP port number (`IPport#`) or all IP port numbers. For example, IP protocol 6 should be used for TCP and 1 for IPv4-ICMP. And for example, IP port 1748 should be used for SRDF and port 3260 for iSCSI host connections. Note the SYMCLI can detect and handle both IPv4 and IPv6 address notations (dots and/or colons) in the command entry.

`-remote_tunnel_addr IPAddr`

Optionally, specifies the IP address of the remote tunnel object. The `-esp_mode` option must be set to `tunnel`.

```
-selectivity {destip|destport|srcport} SPECIFICITY
```

You can specify to use selectivity lists confined to destination or source points and can include *SPECIFICITY* types/protocols for wildcarded proposals only. When an endpoint field's properties have been wildcarded, it determines whether new connections will share an existing security association (selectivity *POLICY*), or if new connections will cause a new security association to be created (selectivity *PACKET*). Selecting *PACKET* results in a more secure configuration, since encryption keys won't be shared between connections, but consumes more resources. Selecting *POLICY* conserves security associations, when this is desired. Properties that may be wildcarded include IP address, IP port number, and IP protocol number.

Scripting proposal set syntax

The proposal set section of the policy syntax sets the proposal settings requirements. For a policy add or modify, consider and set the following options:

```
-proposal_set
-proposal_set_type auto|manual|ike
[-ike_mode main|aggressive]
[-pfs on|off]
[-key_format hex |ascii]
[-presharedkey Keystring]
[-inenc_key Keystring -outenc_key Keystring]
[-inauth_key Keystring -outauth_key Keystring]
[-in_spi SPI# -out_spi SPI#]
[-in_cpi CPI# -out_CPI CPI#]
[-in_nonce NONCE -out_nonce NONCE]
```

Where:

```
-proposal_set
-proposal_set_type auto|manual|ike
```

These options start the proposal set declaration. The `proposal_set_type` can be set to `auto`, `manual`, or `ike` for key management.

Automatic key management or SA management protocol is required to support anti-replay features of AH and ESP modes, and to accommodate on-demand creation of SAs. Multiple keys can be created as a result of this operation.

Manual key management is the simplest form of management where you must manually configure each system with keying data and security association management data relevant to secure communication with other systems. Manual techniques are practical in small, static environments, but it does not scale well.

The Internet Key Exchange (IKE) protocol is a key management protocol standard which is used in conjunction with the IPsec standard. IKE is a hybrid protocol which implements the Oakley key exchange and Skeme key exchange inside the SA and key management protocol (ISAKMP) framework. IKE eliminates the need to manually specify all the IPsec parameters in the crypto maps at both peer endpoints. It allows lifetime limits, changes to occur in keys during IPsec sessions, and allows anti-replay services and dynamic authentication of peers.

IKE options

```
-ike_mode main|aggressive
-pfs on|off
-presharedkey Keystring
```

These options require `proposal_set_type` to be set to `ike`.

IKE phase 1 negotiations are used to establish IKE security associations (SAs). These SAs protect the IKE 2 negotiations. IKE uses one of two modes for phase 1 negotiations: *main* or *aggressive*. Main mode is more intense and secure, but time consuming. Aggressive mode provides faster negotiations, but exposes identities of the peers to eavesdropping.

Perfect Forward Secrecy (PFS) mode for IKE policies can be turned on or off. (Typically, this should be left on, unless you have a special environment.

The preshared key option (`-presharedkey`) specifies the preshared key as the authentication method and defines the preshared key string. It should be the same secret string shared between security points.

Manual options

```
-inenc_key Keystring -outenc_key Keystring
-inauth_key Keystring -outauth_key Keystring
-in_spi SPI# -out_spi SPI#
-in_cpi CPI# -out_CPI CPI#
-in_nonce NONCE -out_nonce NONCE
```

These options require `proposal_set_type` to be set to *manual*.

For non-null `-enc_alg` only, the `-inenc_key` and `-outenc_key` options specify an encryption key string used for encrypting/decrypting inbound and outbound traffic. These must match the corresponding field on the remote endpoint.

For non-null `-auth_alg` only, the `-inauth_key` and `-outauth_key` options specify a hash key string used for authenticating inbound and outbound traffic. These must match the corresponding field on the remote endpoint.

The `-in_spi` and `-out_spi` options specify a Security Parameter Indexing (SPI) number for inbound and outbound traffic in security associations decode.

The `-in_nonce` and `-out_nonce` options specify a random nonce value for inbound and outbound traffic to counter replay attacks.

Proposal options

```
-proposal
-proposal_type ike|ipsec
```

These options start the proposal declaration. Currently, only one proposal per policy can be declared. Compression mode is not supported.

The `proposal_type` specifies the type of proposal to apply to the policy. Possible types supported are:

```
ike -- IP Key Exchange
ipsec -- IPsec
```

Transform options

```
-transform
-transform_type ike|ipsec
```

These options start the proposal declaration. Currently, only one transform per proposal and policy can be declared.

The `transform_type` specifies the type of transform to apply to the policy proposal. Possible types supported are:

```
ike -- IP Key Exchange
esp -- Encapsulation Security Payload
```

Currently, *ah* for authentication header is not supported.

```
-auth_alg null|sha1|md5|xcbc
```

Specifies the authentication algorithm for IKE or ESP policy and transform hash functions. Possible values are:

```
null
```

```
sha1 -- SHA-1 hash authentication algorithm
```

```
md5 -- MD5 hash authentication algorithm
```

```
xcbc -- AES-XCBC-MAC constant key authentication algorithm
```

```
-enc_alg null|des|3des|aes_128|aes_256|aes_cm_128|aes_cm_256
```

Specifies the encryption algorithm for IKE or ESP policy transforms. Possible values are:

```
null
```

```
des -- DES encryption algorithm
```

```
3des-- 3DES or Triple DES (a.k.a. TDES) encryption algorithm
```

```
aes_128 -- Advanced Encryption Standard (AES) encryption (128-bit)
```

```
aes_256 -- Advanced Encryption Standard (AES) encryption (256-bit)
```

```
aes_cm_128 -- Advanced Encryption Standard counter mode (AES-CM)
```

```
aes_cm_256 -- Advanced Encryption Standard counter mode (AES-CM)
```

```
-dhgroup 1|2|3|4
```

Specifies which Diffie-Hellman (dh) group to use for the symmetrical key generation for IKE transforms only. Groups 1 through 4 are supported.

```
-esp_mode tunnel|transport
```

Specifies the ESP transform mode. Possible values are:

```
tunnel - Entire IP packet is encrypted/authenticated (full secure)
```

```
transport - Only the payload (just data) of the IP packet is encrypted/authenticated
```

```
-lifetime [LifeParam1][,][LifeParam2]
```

Specifies the life of the security policy with time and/or data size parameters for IKE and ESP only. Just one, or both values in any order, can be specified:

- A time value in minutes or hours (such as, 90 min or 5 hr)
- A data size value in megabytes or gigabytes (such as, 50 MB or 3 GB)

If both are specified, apply a comma between parameters with not intervening space. The first limit reached, will end the policy life.

```
-auth_method preshared_key
```

Specifies the authentication method for IKE transforms. Possible values are:

```
preshared_key
```

DSA and RSA methods are not supported.

Committing a policy-scripted command file

Once you have completed scripting the policy in the command file. You need to commit the file for execution. However, due to extensive notation work in any policy command file, it is recommended to preview the file for command syntax errors.

Preview the file

To preview the IPsec policy syntax of a command file, use the following command form:

```
symipsec -sid SymmID -dir # -port 0 -file /tmp/commandfile
preview
```

Note: When specifying port numbers in the `symipsec` command, only port 0 is a valid parameter (and is the default). Specifically targeting Port 1 on a director is not supported.

The following example, targets director 1A to preview an IPsec policy addition to the policy database, using file `/tmp/ap`:

```
symipsec -sid 0039 -dir 1A -port 0 -file /tmp/ap preview
```

Commit the file

To commit the IPsec policy syntax of a command file, use the following command form:

```
symipsec -sid SymmID -dir # -port 0 -file /tmp/commandfile
commit
```

The following example, targets director 1A to commit an IPsec iSCSI policy addition to the policy database, using file `/tmp/ap`:

```
symipsec -sid 0039 -dir 1A -port 0 -file /tmp/ap commit
```

Where `/tmp/ap` contains:

```
policy add -priority 50 -assoc_ike_policy 40 -action secure
-local_addr 172.23.195.20 -ipport 3260 -ipproto 6
-remote_addr 50.60.70.80 -ipport 3260 -ipproto 6
-selectivity destip packet -selectivity destport packet
-proposal_set -proposal_set_type auto -proposal
-proposal_type ipsec -transform -transform_type esp
-enc_alg aes_cm_256 -lifetime 90m,5gb
;
```

Deleting a policy-scripted command file

You can delete an existing IPsec policy.

To remove an IPsec policy to the policy database, enter:

```
symipsec -sid 0039 -dir 1A -port 0 -file /tmp/dp commit
```

Where /tmp/dp contains:

```
policy delete -priority 50;
```

This chapter describes the Symmetrix Optimizer concepts and how to improve array performance using of the Symmetrix Optimizer commands of the SYMCLI. This chapter covers the following topics:

◆ What is Symmetrix Optimizer?	220
◆ Configuration requirements	222
◆ How does Symmetrix Optimizer work?	224
◆ Symmetrix Optimizer operations using SYMCLI	231
◆ Setting time windows	234
◆ Migrating devices	236
◆ Setting control parameters	238

What is Symmetrix Optimizer?

Symmetrix Optimizer is one of a family of information management software solutions offered by EMC. Symmetrix Optimizer improves array performance by continuously monitoring access patterns and migrating devices (Symmetrix logical volumes) to achieve balance across the disks in the array. This automated process is based on user-defined parameters and is completely transparent to end users, hosts, and applications in the environment. Migration is performed with constant data availability and consistent protection.

The version of Symmetrix Optimizer that is installed on a Symmetrix depends on the version of the Enginuity operating environment and the version of SYMCLI. The `symoptmz` command sets parameters that control the behavior of the Symmetrix Optimizer algorithm.

This chapter describes how to manage Symmetrix Optimizer using the SYMCLI. Using the Symmetrix Optimizer SYMCLI command, `symoptmz`, users can define the following:

- ◆ Symmetrix devices to be optimized
- ◆ Priority of those devices
- ◆ Window of time that profiles the business workload
- ◆ Window of time in which Optimizer is allowed to swap devices
- ◆ Additional business rules
- ◆ The pace of the Optimizer volume copy mechanism

How Optimizer works

Once initialized with the user-defined parameters, Symmetrix Optimizer operates totally autonomously on the Symmetrix service processor. Symmetrix Optimizer monitors and controls array performance by following these steps:

Step 1: Symmetrix Optimizer builds a database of device activity statistics on the Symmetrix back-end.

Step 2: Using the statistical data collected, configuration information, and the user-defined parameters, the Optimizer algorithm identifies busy and idle devices and their locations on the physical drives. The algorithm tries to minimize average disk service time by balancing I/O activity across physical disks. Optimizer determines which disks require balancing by locating busy devices close to each other on the same disk, and/or by locating busy devices on faster areas of the disks. Optimizer takes into account the speed of the disk, the disk geometry, and the actuator speed to determine faster disks.

Step 3: Once a solution for load balancing has been developed, the next phase is to carry out the Symmetrix device swaps. This is done using established TimeFinder technology (Enginuity 5773 and earlier), which maintains data protection and availability. You can specify whether swaps should occur in a completely automated fashion, or if the device swaps require user approval before the action is taken.

Step 4: Once a swap function completes, Symmetrix Optimizer continues data analysis for the next swap.

Supported environments

Symmetrix Optimizer consists of two major components: the Optimizer processing engine and a client. The Optimizer processing *engine*, which runs continuously on the Symmetrix service processor and collects the statistics, performs the analysis, and executes the migrations. This component is associated with the version of Symmetrix Enginuity software running on the Symmetrix array. Versions of the Optimizer engine (with different functionality) are referred to as different *revs* of Optimizer.

Note: For detailed interoperability information, please refer to E-Lab Interoperability Navigator which can be reached at <http://elabnavigator.EMC.com>.

Since Symmetrix Optimizer operates on Symmetrix devices and physical disks, there are no compatibility requirements at the host or application level.

Symmetrix Optimizer can optimize all open system and mainframe RAID 1 (mirrored), RAID 5, RAID 6, and RDF-protected devices in an array. Optimizer does not work with devices configured for use with the AS/400 operating system. Optimizer does not consider business continuance volumes (BCVs) or dynamic reallocation volumes (DRVs) for swapping (refer to “[Legality of swaps](#)” on page 228 for a complete list of valid swaps).

Configuration requirements

Symmetrix Optimizer must be enabled by EMC from the service processor and DRV's must be configured on the array. In addition, for Symmetrix arrays running Enginuity 5773 and earlier, Optimizer requires an open mirror position to perform its swapping function. Symmetrix arrays running Enginuity 5874 and higher do not require the open mirror position, except when swapping Cascaded RDF devices.

Dynamic reallocation volumes (DRV's)

The DRV is a non-user-addressable logical volume used by Symmetrix Optimizer to temporarily hold customer data while executing reconfiguration (on logical volume granularity). Data is available and protected during the reconfiguration process.

Enginuity 5773 and earlier

As a minimum, two DRV's must be configured for each size and emulation of the volume to be swapped by Optimizer. For example, for Optimizer to swap a 4 GB open system volume, then two 4 GB open system DRV's must be configured. If more than one simultaneous swap is desired, then additional DRV's are needed (two for each swap). If there are volumes of mixed size or emulation within the same array, then DRV's are required for each size and type to be swapped.

Enginuity 5874 and higher

All DRV's are mirrored and only one DRV per swap is required. In addition, the DRV can be equal or larger in size than the logical device that is swapped.

Open mirror position requirement

Symmetrix arrays running Enginuity 5874 and higher do not require the open mirror position.

Optimizer requires an open mirror position to be able to perform a swap, however, Symmetrix arrays (running Enginuity 5773 and earlier) have a four mirror slot limitation per volume. This limitation may make it difficult in certain configurations for Optimizer to obtain the mirror position it requires to perform a swap. For example, a RAID 1 device already has two BCV's established, and Optimizer wants to swap this device, it cannot get a mirror slot for the DRV. If Optimizer does not have a mirror slot available, then it will drop the swap and rerun the analysis. If Optimizer is in Rollback, Manual, or User-Approved mode, then Optimizer will retry a number of times until an error is encountered. Eventually, Optimizer will stop trying if a mirror slot does not become available.

Open configuration lock requirement

Swaps on Symmetrix arrays running Enginuity 5874 and higher do not explicitly prevent other configuration changes from taking place during the swap operation.

For Symmetrix arrays running 5773 and earlier, Optimizer holds a Symmetrix External Lock (SEL) to perform a swap or migration. The configuration lock is an exclusive lock on the Symmetrix array for performing configuration changes. This lock prevents multiple applications from changing the array configuration at the same time. If another application holds the configuration lock when Optimizer wants to create a swap, then Optimizer will behave as previously described (no open mirror position). Optimizer will hold the configuration lock for the duration of the swap or

migration, or until the operation is cancelled, so other applications will not have the ability to perform configuration changes during this time.

Note: Committed swap or migration operations that are scheduled for a later time can be cancelled up until the scheduled time.

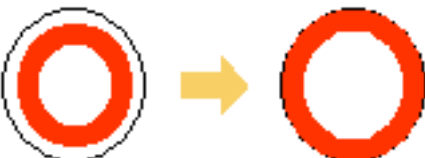
How does Symmetrix Optimizer work?

The analysis methodology and the swap procedure are the core technology and the two main components of the Symmetrix Optimizer solution. The following sections provide a brief description of each.

The analysis

When Symmetrix Optimizer evaluates the performance statistics, it determines potential device swaps based on how well they would improve overall performance. This analysis is based on minimizing disk service time (rotational latency plus seek time plus transfer time). Symmetrix Optimizer uses three strategies when determining which swaps to make.

Table 16 Optimizer strategies for determining swaps

Swap strategy	Description
1. Load Balancing 	Swaps highly active devices on disks with lower activity to even out the load across the physical drives in the Symmetrix system. This decreases the contention on the individual physical disks, improving the performance of both highly active devices and low-activity devices.
2. Minimize Seek 	Performs swaps that relocate highly active devices so that they are closer together on the physical disk and will decrease the seek distance for I/Os to these devices. When Optimizer swaps devices, it will try to achieve this type of configuration since it decreases the overall I/O service time.
3. Use Faster Media 	Optimizer tries to swap highly active devices on the outer zones of the disk. This is because devices located on the outer zones of the disk have faster transfer speeds.

Symmetrix Optimizer's algorithm uses detailed disk performance information that takes into account several drive characteristics, such as those gig-to-gig seek times, zone-bit recording, and bandwidth data.

Performance metrics

Optimizer looks only at back-end activity; it uses the back-end logical device statistics, and the following metrics:

- ◆ DA logical volumes reads
- ◆ DA logical volumes writes
- ◆ Logical volume prefetch
- ◆ DA logical volumes Kb read
- ◆ DA logical volumes Kb written

While modeling the disk service time, different weights are assigned to read, write, and prefetch activity. The assumption is that since writes are done as a background process, they are done in sequence and hence are more efficient. Optimizer does not try to follow DMSP policies; instead, it assumes that reads are equally spread among all the device mirrors, and all mirrors perform all writes.

Modeling service time

Service time is defined as the sum of seeks, latency and transfer time:

- ◆ **Seek time** — Seek time is the time it takes the disk arm to move and position the disk head on the correct track—move from track *X* (serving the previous I/O) to track *Y* (serving the next I/O). Optimizer uses a gig-to-gig database to model seek time for different addresses and different disk drives.
- ◆ **Latency time** — Latency time is the delay for disk rotation. The latency time is a function of the disk rotational speed. Optimizer assumes one-third of a spin for each I/O, assuming the internal disk optimization is on.
- ◆ **Transfer time** — Transfer time is the time that it takes the disk to transfer the data from/to the disk. Transfer time is a function of the data transfer rate, disk bandwidth, and data layout. Optimizer uses the Zone Bit Recording (ZBR) database to model the transfer time of data. The ZBR database includes information about the bandwidth of each zone of the disk.

Accurate seek and latency times are impractical to get because they require a complete trace of I/Os sequence, therefore the Optimizer uses a mathematical model to calculate these metrics.

Finding the best swap

The Optimizer analysis consists of three high-level phases:

- ◆ **Calculate service time** — Model and sum the total service time of each disk and for every time stamp that was marked to be included by the analysis.
- ◆ **Sort disks by activity** — Sort all disks by their modeled total service time.
- ◆ **Find best swap** — Starting at the busiest disk, check all potential swaps. The analysis process models *what-if* scenarios using *virtual swaps* to estimate the impact of a swap on the service time of the affected disks. The philosophy of Optimizer is to check as many swaps as possible in order to guarantee that the best swap is indeed selected.

Currently, 256 hypers are allowed per disk; finding a swap that improves service time by 10 percent or more on these systems is very rare. Since using a percentage of improvement as swap *goodness* criteria is not applicable anymore, Optimizer had to adapt a different method (*M0*) that deals better with the “many hypers contribute smaller chunks” problem. *M0* is defined as “the best you could possibly get” from a swap. Each disk in the system is assigned an *M0* designation, which is defined as the minimum service time a disk can get by replacing one of its hypers by a null hyper (a hyper that performs no I/Os to the disk).

When analyzing a swap, Optimizer checks how close the new-modeled service time is to the disk’s *M0*; the closer to *M0*, the better the swap. Usually, two disks are affected by a single swap; the busier disk’s service time is expected to go down, while the other disk’s service time is expected to go up. In addition to the *M0*, the Optimizer also ensures that the new maximum of service times is less than the old one.

Swap procedure for Engenuity 5874 and higher

Symmetrix V-Max arrays with Engenuity 5874 streamlines the device swap procedure by using only one DRV device during the swap. With Engenuity 5874, only mirrored DRV devices can be created. Refer to [Table 5 on page 44](#) for information about creating mirrored DRV devices.

This section provides the new steps for swapping devices using Symmetrix Optimizer. Refer to [“Swap procedure for Engenuity 5773 and earlier” on page 226](#) if you have an earlier version of Engenuity.

Hypervolumes are swapped using a four-step process. One DRV must be configured per swap. The DRV should be of equal or greater size that the swapped devices. With version 5.1.1 of Symmetrix Optimizer, up to eight simultaneous swaps can happen at one time.

[Figure 21](#) illustrates the steps for swapping device A with device B.

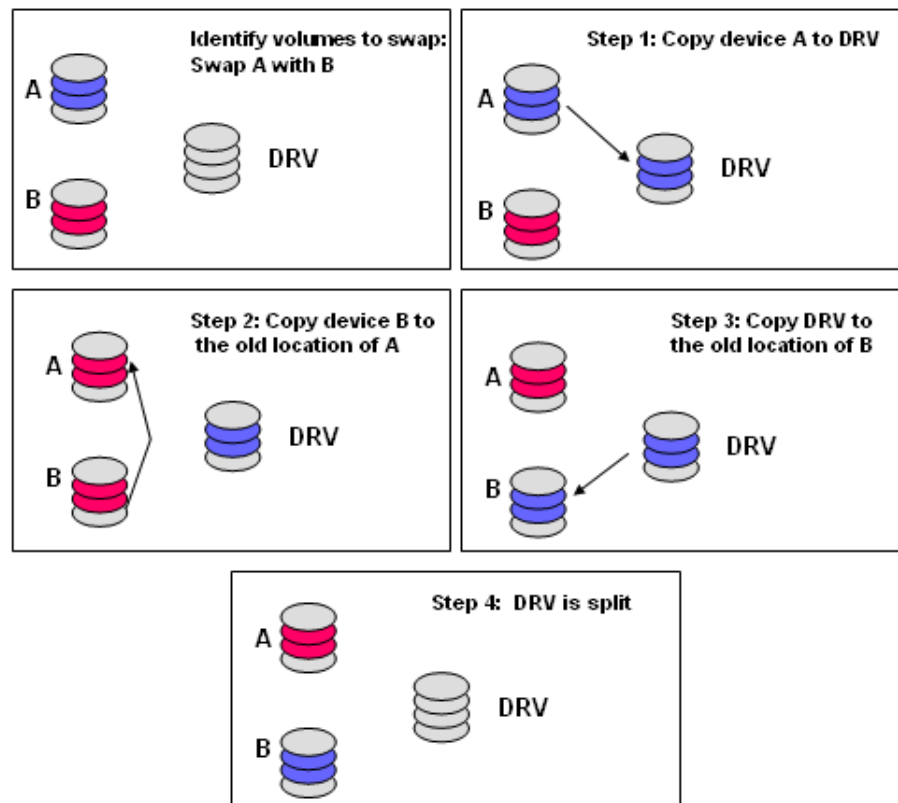


Figure 21 Swap steps using one DRV

Swap procedure for Engenuity 5773 and earlier

The swap procedure relies on the Symmetrix TimeFinder technology and uses DRVs as temporary mirrors. The following section describes the four swap steps when dealing with mirrored devices swaps. In the case of three mirror devices, the swap procedure does not require a DRV.

Step 1: Identify volumes to swap

Symmetrix Optimizer identifies a pair of hypervolumes to swap by recognizable patterns of hypervolume activity and criteria. In [Figure 22](#), assume the red volumes (on A) have high activity and the blue volumes (on B) have low activity.

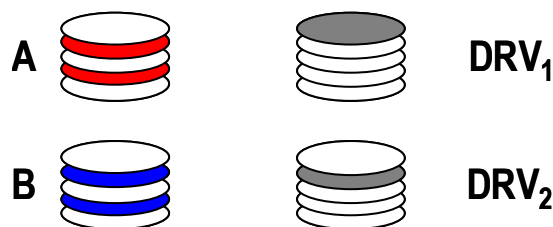


Figure 22

Identify volumes to swap**Step 2: Copy volume to DRV**

Symmetrix Optimizer swap commands are passed to SymmWin, which assigns one DRV (DRV1) as a third mirror for hypervolume A. A second DRV (DRV2) will be assigned as a third mirror for hypervolume B. All tracks on the third mirror are marked invalid. Tracks are copied from the valid mirrors to the two DRVs. After the DRVs are synchronized, the two original swap physical mirrors are marked Not Ready (volume A and volume B) and their attributes are swapped (see [Figure 23](#)). Both hypervolumes still have two (or more) physical mirrors. Host activity to the hypervolumes is now directed to DRVs and the other mirror.

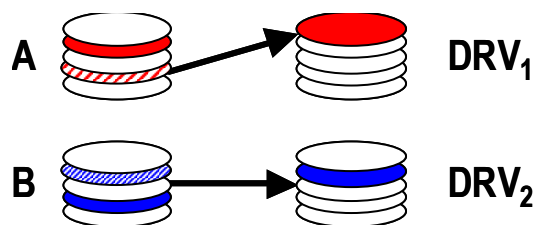


Figure 23

Copy volume to DRV**Step 3: Copy DRV to new location**

The data on the DRVs are now copied to the new location. After the attributes of the original hypervolumes are swapped, SymmWin copies the tracks from the valid mirrors to the two new mirrors and then makes the original hypervolumes Ready (see [Figure 24](#)). This is similar to a BCV restore.

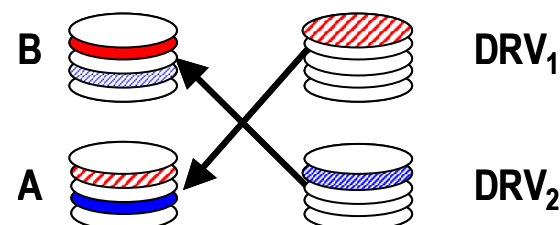


Figure 24

Copy DRVs to new locations**Step 4: DRVs are split**

The final step is to split the DRVs from their standard and mirror hypervolumes after synchronization completes (see [Figure 25](#)). The drive's balance improves and the DRVs are now available for the next swap.

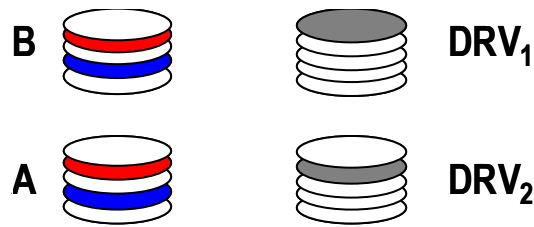


Figure 25 DRVs are split

Note: Swap of three mirror devices does not require a DRV.

Legality of swaps

As a rule, Optimizer will never suggest a swap that conflicts with configuration rules as defined by the Enginuity, SymmWin, or the configuration groups. For Symmetrix 3000, 4000, and 8000 hardware families, all configuration rules are static. However, for the Symmetrix DMX and RMS families, some configuration rules are static and some are more dynamic and defined in relative manner where an Optimizer swap can improve or maintain protection level but will never decrease protection level. (Exceptions for this might be a rollback session, where the user wants to undo a swap.)

Enginuity 67/68 (Symmetrix 3000, 4000, and 8000 families)

For Symmetrix 3000, 4000, and 8000 hardware families, the following rules apply:

- ◆ Swappable devices:
 - RAID 1 (mirrored volumes)
 - RDF devices with two local copies (default, can be controlled from SP)
 - Same size and emulation

Note: FBA and CELLERA_FBA type emulations can be swapped and can coexist on the same drive.

- ◆ Not swappable devices – The following devices are not swappable:
 - Parity RAID (parity and data)
 - CKD Stripe (RAID 1/0)
 - BCV, DRV, SFS
 - AS400
 - Standard, unprotected volumes (default, can be controlled from SP)
- ◆ Other configuration restrictions:
 - **Same disk** — Two mirrors of the same device should never reside on the same disk (for example, 1a:C0).
 - **X/Y bus** — Two mirrors of the same RAID 1 device should never reside on the same memory bus (for example, 1a with 3a).
 - **Dual-Initiator** — Two mirrors of the same RAID 1 device should never reside on the same DA and dual-initiator pair (for example, 1a with 2a).

Enginuity 69/70 (Symmetrix DMX and RMS)

Configuration rules for the Symmetrix DMX and RMS families are relative in manner. Optimizer can improve or maintain protection, but never decrease level of protection. However, some restrictions should always be followed:

- ◆ Swappable devices:
 - Same size and emulation
 - Same protection (Enginuity 5874)
 - Available DRVs for both source and target
 - Currently attached to legal RAID groups
- ◆ Not swappable devices — The following devices are not swappable:
 - Marked by the user as “do not move”
 - Parity RAID (parity and data)
 - CKD Stripe (RAID 1/0)
 - BCV, DRV, SFS, Vault
 - AS400
 - Thin devices
 - SAVE devices and virtual (VDEV) devices
 - Standard, unprotected volumes (default, can be controlled from the Service Processor)
 - Targets of Snap sessions
 - WORM devices or WORM-protected devices
- ◆ Other configuration restrictions:
 - **Same disk** — Two mirrors of the same device should never reside on the same disk (for example, 1a:C0).
 - **Same port** — Two mirrors of the same device should never reside on the same SCSI port (for example, 1a:C).
 - **Same loop** — All mirrors of the same RAID 1 device should never reside on the same disk loop (for example, 1a:C with 16a:C).

Note: Not applicable to RMS.

 - **Power zones** — All mirrors of the same RAID 1 device should never reside on the same power zone.
 - **Same DA** — All mirrors of the same RAID 1 device should not reside on the same DA (CPU) (for example, 1a).

- **Same slot** — All mirrors of the same RAID 1 device should not reside on the same director (slot) (for example, 1a with 1b).
- **Dual-Initiator** — All mirrors of the same RAID 1 device should not reside on the same dual-initiator pair (for example, 1a with 16a).

Note: Not applicable to RMS.

Impact of swaps

To minimize the affect of the swap procedure on the overall Symmetrix performance, the Optimizer will not include the same spindle twice in one swap group. In addition, you can use QoS controls to set a lower BCV priority for the STDs to be swapped, and plan the swap activities to occur on idle business time using the Time Window feature.

Symmetrix Optimizer operations using SYMCLI

The Symmetrix Optimizer is a component of the EMC Symmetrix Management Console. The SYMCLI on the host machine issues commands to display and set the values of the parameters that control the behavior of the Symmetrix Optimizer.

Table 17 describes the main features of the Symmetrix Optimizer command:

Table 17 Symmetrix Optimizer command

symoptmz command argument	Description
clear_stats	Clears disk statistics maintained by Symmetrix Optimizer.
commit	Updates Optimizer with the changes defined in the command file.
disable	Disables the Optimizer algorithm processing.
enable	Enables the Optimizer algorithm processing.
list	Displays Optimizer-specific attributes of Symmetrix devices.
prepare	Performs extra range checks on the changes specified in the command file.
preview	Verifies the syntax of the changes specified in the command file.
query	Queries Symmetrix Optimizer and displays the current state and version information of the Optimizer. If <code>-v</code> is specified, additional information about the version and open sessions displays.
read	Reads from Optimizer's log file.
release	Attempts to gain control of an existing Optimizer API session to abort it and release the Optimizer API lock.
rollback	Rolls back a Symmetrix configuration to that of a previous point-in-time.
set	Sets the values of the parameters that control the Optimizer algorithm behavior.
show	Shows information about the current Optimizer parameters. By default, displays control parameters.
sync	Acquires version information from Optimizer. This argument is supported only in conjunction with <code>-version</code> .

Starting and stopping Optimizer

To start and stop the Optimizer process on the array service processor, use the `symoptmz enable` and `symoptmz disable` commands. Once disabled, the Optimizer process on the Symmetrix service processor will still listen for Optimizer clients (SMC or SYMCLI), but it will not collect statistics or initiate any swaps until it is enabled.

Showing Optimizer information

To show time windows, use the following arguments with the `symoptmz` command:

```
symoptmz -sid SymmID
  show [-parms | -swap_hist | -swap_list | -composite |
        -rollback_list | -activity_list ]
```

Where:

-parms — Displays information about the control parameters of Symmetrix Optimizer. This is the default option for show.

-swap_hist — Displays the history of the swaps known to Optimizer.

-swap_list — Displays all the swap lists and migrations currently known to Optimizer.

-composite — Displays composite time windows. Composite time windows are generated by Optimizer by combining all known user-defined time windows.

-rollback_list — Displays a list of possible rollback points.

-activity_list — Displays all the swap lists and migrations currently known to Optimizer (interchangeable with `-swap_list`).

Reading Optimizer logs

To retrieve the Optimizer activity log, including swaps that have been completed, use the following command:

```
symoptmz read -log_type RUNTIME [-start DateTime] [-stop DateTime]
```

To retrieve the Optimizer error log, use the command:

```
symoptmz read -log_type ERROR [-start DateTime] [-stop DateTime]
```

The log that is returned includes DOS line-feed characters that can be removed using `sed` or another text editor:

```
cat opt_log.txt | sed s/^M$// > cleaned_opt_log.txt
```

In a UNIX shell, the `^M` should be typed using **CTRL-V CTRL-M**.

In addition, the `symoptmz read` command can be used to determine why a swap was not performed.

Committing command file changes

To change any Optimizer parameters, Optimizer must be stopped using the `symoptmz disable` command. The parameters are then set using a command file (for command file parameters, refer to [“Setting control parameters” on page 238](#)); then, validated and committed using the `preview`, `prepare`, and `commit` options. For example, to commit the changes specified in the command file `opt_config.txt`, issue the command:

```
symoptmz -file opt_config.txt commit
```

The `preview` argument checks that the command file has correct syntax. The `prepare` argument also checks syntax and, in addition, performs some range checks. The `commit` argument carries out the same syntax and range checks, and then updates the Optimizer with the modified parameters. It is recommend that a `preview` be run on

your command files, and then any syntax errors corrected. A `prepare` should then be run and any out-of-range figures should be corrected. Finally, a `commit` should be done. The `commit` command will first do a preview and prepare before committing the settings in the command file.

Note: It is not mandatory to execute a `preview` or `prepare` action prior to a `commit`. However, these actions can ensure that the `commit` action will not be rejected or can be used to debug the command file entries.

Approving a swap list

User Approved mode allows you to see what swaps will occur before they take place. To switch Optimizer to User Approved mode, use the `set control_parms swap_mode=USER_OK;` line in the command file. Once this is set, Optimizer recalculates a swap list approximately once every hour as long as samples have been collected for the specified minimum performance period. At any time, the latest swap list can be retrieved using the `symoptmz show -swap_list` command. The swap list can then be approved or declined using the following syntax in the command file:

```
set swap <APPROVE | DECLINE>
    [begin_at=TimeVal,]
    TIMESTAMP=TimeVal;
```

The time stamp specified must be the time stamp returned by the last `symoptmz show swap_list` command. If the command to approve or decline the swap list returns an error, the swap list is probably out of date and a new one is available from the service processor. The latest swap list should then be retrieved again with the `symoptmz show -swap_list` command.

Setting time windows

To set the swap time or performance time windows, use the `symptmz` command with the following syntax in the command file:

```
set time_window id=tw_id,
  type=<SWAP | PERF>,
  flag=<INCLUDE | EXCLUDE>,
  period=<ONCE | WEEKLY | WEEKLY_BY_DAY>,
  starting=DateTime,
  ending=DateTime,
  [days=DayList,
  start_time=hh:mm,
  end_time=hh:mm];
```

Where:

tw_id — Specifies a user-defined name that identifies this time window.

type — Defines the kind of time window being defined. Two types are available:

SWAP: Defines whether a swap is (INCLUDE) or is not (EXCLUDE) performed in the given time window, according the `flag` setting.

PERF: Defines whether a monitoring is (INCLUDE) or is not (EXCLUDE) performed in the given time window, according the `flag` setting.

Using this setting, you could, for example, configure Optimizer to monitor on weekdays from 9 - 5 and only allow swaps on Sundays.

flag — Defines the option for the time window type:

```
INCLUDE
EXCLUDE
```

period — Defines the frequency of occurrence:

ONCE means the action will occur only one time.

WEEKLY_BY_DAY requires the individual days of the week to be specified. The time window applies to each day specified and each day has its own time window.

WEEKLY setting is equivalent to a *range*. The `starting` parameter identifies the first of a series of consecutive days that the recurring time period applies to, and the next day in the list serves as the ending day of the week to which the time period applies.

DateTime — Identifies the date and time in the form of `MMDDYYYY:HHMMSS`.

DayList — is any comma-separated combination of MON, TUE, WED, THU, FRI, SAT, or SUN. For the case of WEEKLY, *DayList* must begin with one of the following: MON_START, TUE_START, WED_START, THU_START, FRI_START, SAT_START, or SUN_START, which identifies the first of a series of consecutive days to which the time window applies. The next entry identifies the day of the week, which concludes the range of days.

Using the `set time_window` command in the command file erases any previously set time windows (both swap and performance), so this command should be used with caution.

Clearing a time window

To clear a time window, use the following syntax:

```
clear time_window type=<SWAP | PERF>;
```

Setting swap priority

Use the following command to set the device priority:

```
symoptmz set swap_priority <none | normal | high>
[-range SymDevStart:SymDevEnd] [-n NumDevs]
```

Use the following command to list the priority settings for all the devices or a range of devices:

```
symoptmz list [-range SymDevStart:SymDevEnd] [-n NumDevs]
```

Performing rollbacks

Rollbacks are used to undo swaps that conflict with your business rules. Rollbacks are performed by first showing the rollback list, as follows:

```
symoptmz show -rollback list
```

Use the following command file syntax to rollback a swap:

```
set swap <APPROVE | DECLINE>
[begin_at=TimeVal,]
TIMESTAMP=TimeVal,
[, ROLLBACK];
```

The rollback feature is an *all or nothing* feature. Optimizer reverses all swaps by going backward from the present to a selected earlier time and undoing each and every swap. Use the User Approved mode as described in [“Approving a swap list” on page 233](#) to undo specific swaps. Rollback swaps can be scheduled for execution according to normal Optimizer policy.

Setting manual swap lists

Enginuity 5773 and earlier

To set a manual swap list, use the following form:

```
set swap_list {Hyper1} with {Hyper2} [, {Hyper3} with {Hyper4},... ]
[begin_at=TimeVal];
```

Where:

Hyper*n* is of the form DDD, I, T, HH:

DDD is the director identifier

I is the director interface

T is the target ID

HH is the hyper number

TimeVal is in the form of MMDDYYYY:HHMMSS.

Enginuity 5874 and later

To set a manual swap list, use the following form:

```
set dev_swap SymDevName1 with SymDevName2
[, SymDevName3 with SymDevName4, ...]
[begin_at=TimeVal];
```

Example

```
Set dev_swap 0030 with 0040;
```

Migrating devices

Note: The `symoptmz migrate` command is not available on Symmetrix V-Max arrays running Enginuity 5874. For those arrays, use the `symmigrate` command.

When devices are moved to a new set of disks, this is called a *migration*. A migration can be used to populate newly added disk drives, or to move devices between high performance and high capacity disks. A migration can only be initiated by a user. When devices are moved to a new specific location, this is called a *relocation*, and it may be used to balance disk usage or to fill in holes left by deleting devices. Relocations are performed by the Optimizer automatically, as needed. A relocation cannot be initiated by a user.

Symmetrix Optimizer `symoptmz` command has been enhanced with the following features for data migrations:

- ◆ The source devices to be migrated can be specified as a device group.
- ◆ The target disks can be a SymmWin disk group number.
- ◆ Users can optionally specify that unmapped and/or unmasked devices can be used as target devices.

When devices or device group devices are migrated to target disks or disk groups, Optimizer looks for free space on the target disks for the migration. If the target disks do not have any free space, users can specify to use unmapped and/or unmasked devices. These options indicate that source devices can be migrated to space occupied by unmapped and/or unmasked devices on the target disks. When this option is specified, Optimizer will NOT look for free space on the target disks but will only look for matching unmapped and/or unmasked devices.

Note: The unmapped and unmasked options are not recommended if there is free space on the target disks.

As with all device migrations, devices on the target disk list must match the size, emulation, and protection type of the source devices for the migration to succeed. To initiate a migration, use the `symoptmz` command with the following syntax in the command file:

```
migrate
  dev[s] SymDevStart1[:SymDevEnd1]
  [, SymDevStart2[:SymDevEnd2], ...]
  TO disk[s] {diskn} [, {diskn}, ...]
  [unmapped=TRUE] [unmasked=TRUE]
  [begin_at=TimeVal];

migrate
  device_group DgName
  TO disk_group_num #
  [unmapped=TRUE] [unmasked=TRUE]
  [begin_at=TimeVal];

migrate
  device_group DgName
  TO disk[s] {diskn} [, {diskn}, ...]
  [unmapped=TRUE] [unmasked=TRUE]
  [begin_at=TimeVal];

migrate
  dev[s] SymDevStart1[:SymDevEnd1]
  [, SymDevStart2[:SymDevEnd2], ...]
```

```

TO disk_group_num #
[unmapped=TRUE] [unmasked=TRUE]
[begin_at=TimeVal];

```

Where:

The *diskn* is of the form DDD, I, T:

DDD is the director Identifier,

I is the Director Interface, and

T is the Target ID

The *TimeVal* is in the form of MMDDYYYY:HHMMSS.

Example An example of a device migration command file follows:

```

migrate devices 240, 245:247 to disks {16B, C, 0}, {4D, C, 0};
migrate device_group Mydgname to disk_group_num 1;
migrate devices 240, 245:247 to disk_group_num 2;
migrate device_group Mydgname to disks {16B, C, 0}, {4D, C, 0};
migrate devices 240, 245:247 to disks {16B, C, 0}, {4D, C, 0} unmapped=TRUE;
migrate device_group Mydgname to disk_group_num 0 unmasked=TRUE;
migrate devices 240, 245:247 to disk_group_num 0 unmapped=TRUE unmasked=TRUE;

```

Setting control parameters

Optimizer control parameters are set using the following syntax in the command file:

```
set control_parms [start_mode=AUTO | MANUAL>,]
[swap_mode=AUTO | USER_OK,]
[min_perf_period=min_perf,]
[workload_period=workload,]
[max_simult_swaps=max_simult,]
[swap_rate=max_swaps];
```

Table 18 describes each parameter.

Table 18 Parameters in the command file

Parameter	Description
start_mode	Determines whether Optimizer is enabled or disabled whenever Optimizer is launched. For example, after rebooting the service processor.
swap_mode	Controls whether Optimizer should automatically swap devices (AUTO) as soon as it finds swaps that would improve performance. If Optimizer is enabled in AUTO swap mode, each day it will make up to the number of swaps specified by the swap_rate parameter, under control of the swap time window settings. If Optimizer is enabled in USER_OK swap mode, it will generate lists of swap suggestions approximately once an hour, and then wait for the user to approve the swap list before proceeding.
min_perf_period	Specifies the amount of samples required initially before a recommendation will be made. You should make sure that the values you specify are long enough (usually a week) for Optimizer to establish a good characterization of your typical workloads. This parameter is expressed in hours. Keep in mind that the Optimizer statistics database holds about 14 days worth of data.
workload_period	Specifies how far back in time Optimizer should consider when the optimization algorithm is run. Be careful not to make this value too large or you may include data that is so old it is no longer representative of your current workloads. This parameter is expressed in hours.
max_simult_swaps	Controls how many swaps Optimizer can perform simultaneously (up to four). The actual value should reflect the number of devices that will be swapped simultaneously. The acceptable values are from two (a single pair swapped) to eight (four pairs swapped) Optimizer swaps.
swap_rate	Sets the maximum number of swaps that Optimizer is allowed to make in a single day. This parameter is only relevant if swap_mode is set to AUTO. Note that reflects the total number of devices to be swapped in a day. A value of 24 would allow 12 pairs of devices to be swapped in a day.

Example The following command file sets up Optimizer to analyze data from the previous seven days and to start figuring out swap suggestions after three days of collecting data. It also sets Optimizer to User Approved mode and sets the maximum number of simultaneous swaps to eight (four pairs of hypervolumes):

```
set control_parms start_mode=AUTO,
min_perf_period=72,
workload_period=168
swap_mode=USER_OK,
max_simult_swaps=8,
swap_rate=50;
```

This chapter explains how to change device protection or improve array performance using Enhanced Virtual LUN Technology. This chapter covers the following topics:

◆ Overview	240
◆ Creating virtual LUN migration sessions.....	243
◆ Migrating to configured space	244
◆ Migrating to unconfigured space.....	246
◆ Managing virtual LUN migration sessions.....	247
◆ Viewing the audit log	250
◆ Virtual LUN migration examples	251

Overview

Enhanced Virtual LUN Technology allows storage administrators to quickly move an application's data to other storage within the same array. A migration can change the protection level and disk type of FBA, CKD, and iSeries Symmetrix devices. A migration can use the storage of existing devices that do not contain live data or unconfigured disk space.

Virtual LUN migration is from source devices to equally-sized target devices, or disks with free space equal to the source devices. A migration will not disrupt hosts or internal applications.

A virtual LUN migration may be initiated in the following circumstances:

- ◆ Traditional data management involves moving data to a lower tier as it ages and becomes less critical to the business. Protection might change as the data ages.
- ◆ Emerging tiered storage configurations include moving data continuously during its lifecycle, either to optimize price/performance or respond to cyclical business changes.

A virtual LUN migration can be performed online or offline.

Solutions Enabler 7.0 includes a new command, `symmigrate`, to perform virtual LUN migrations.

Note: The Symmetrix Optimizer license is required to use enhanced Virtual LUN technology.

Requirements

The following are the requirements for using the `symmigrate` command:

- ◆ Symmetrix V-Max array running Enginuity 5874
- ◆ Solutions Enabler 7.0
- ◆ Any of the following disk types:
 - Enterprise flash
 - Fibre Channel
 - SATA
- ◆ Any of the following device types:
 - Standard Symmetrix devices, unprotected Symmetrix devices
 - Metadevices
 - FBA, CKD, iSeries

Restrictions

- ◆ You cannot migrate protected devices to an unprotected protection type. Migrations of unprotected source devices to unprotected target are supported.
- ◆ You cannot migrate to the same protection type (RAID 5 (3+1) to RAID 5 (3+1) and the same disk group number (disk group 1 to disk group 1). The desired protection type or disk group number must be different from what exists on the source devices.
- ◆ Device types not supported:
 - VDEVs (TimeFinder Snap)
 - Thin devices
 - DATA devices
 - Vault devices/SFS/VCM
 - SAVE devices

What's happening inside the array

Solutions Enabler 7.0 has virtualized the RAID architecture so that RAID 1 and RAID 5 protection mechanisms have been virtualized behind the Symmetrix mirror positions, as RAID 6 was in Enginuity 5772 on Symmetrix DMX arrays.

This allows multiple, independent RAID groups per Symmetrix logical volume. It also reduces the protection positions required.

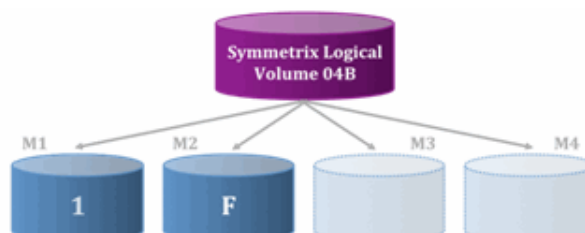


Figure 26 Initial state of RAID 1 device with SRDF

Using virtual LUN technology to migrate a RAID 1 device group device (as shown in [Figure 26](#)) to a RAID 5 device group device, leverages virtual RAID to attach and synchronize the new hypervisors.

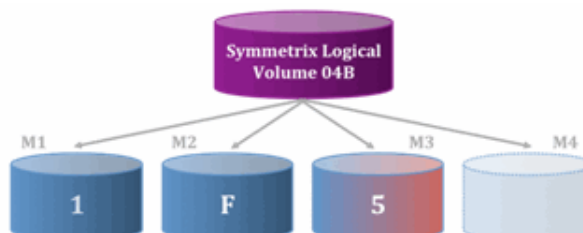


Figure 27 Synchronizing the RAID 5 group

The new target RAID group is associated as an additional mirror to the specified source device. A brief configuration lock is taken for the devices at this time. The system then begins the process of synchronizing the data between the primary and secondary mirrors of the source device, as shown in [Figure 27](#).

Reads and writes are serviced during the migration process.

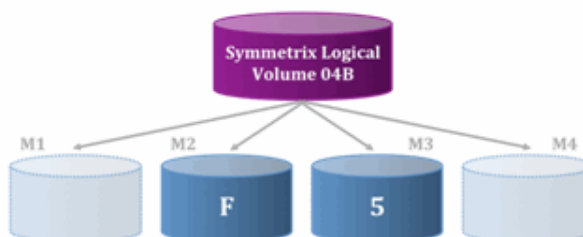


Figure 28 RAID 5 group with SRDF

When the data has been fully synchronized on all of the devices associated with a specific migration session, the system automatically promotes the secondary mirror on the source device to be the new primary mirror, as shown in [Figure 28](#).

The following occurs on the target side:

Configured space — The original primary mirror of the source devices will become the primary mirror of the target devices.

Unconfigured space — The system deletes the original primary mirror and returns the space to the pool of free space on the Symmetrix array.

During the migration the source device will be available for all local and remote replication. The target device will be set by the system as user not ready for the duration of the migration and all operations to set the device to ready will be blocked.

Source devices

Source devices can be specified by using the following:

- ◆ A Device Group
- ◆ A Storage Group
- ◆ A Source file (Containing a list of source devices or device pairs)

Target criteria

Target criteria can be specified as configured space or unconfigured space. A target disk group is specified, as well as the target protection type: Unprotected, RAID1, RAID 5 (3+1 or 7+1), or RAID 6 (6+2 or 14+2), unless using pairs in a file. The target protection type and disk group is assumed when using a device file.

Using files

Device file

A device file (*DevFile*) is a text file that contains pairs of source and target devices and one pair per line in the file though the line continuation character of '\' can be used to wrap lines.

For example:

```
10      20
12      22
14      27
```

Device file for metadvice

Or for a meta device:

```
15 28, 29, 30, 31, \
    32, 33, 34, 39
```

A metadvice to metadvice file is also allowed, if the device sizes are equal, and all members are also equal in size and order.

Source file

A source file (*SrcFile*) is a text file that contains only source devices and one device per line in the file.

For example:

```
9A
9E
9F
```

Creating virtual LUN migration sessions

The CLI will provide the actions listed in [Table 19](#) to create and manage migration sessions.

Table 19 Action descriptions for `symmigrate` command

symmigrate action	Description
<code>validate</code>	Verifies that the information provided is currently allowed. No changes will be made to the device(s) and the command can optionally output a file containing device pairs (using configured space).
<code>establish</code>	Starts a new session and begins the Synchronizing process.
<code>query</code>	Queries for the status of sessions.
<code>verify</code>	Verifies that a session is in a specified state.
<code>list</code>	Lists all the sessions for a given Symmetrix array, or for all Symmetrix arrays.
<code>terminate</code>	Terminates a session and removes it from the Symmetrix array.

Validate

The `symmigrate validate` command is an optional command, that can be used in two different ways. The command can be used as a simple positive/negative test to determine if a migration will succeed. This will be denoted by the return value of the command. The command can also be used when migrating to configured space to output a file that contains the exact pairs of devices that will be used for the migration. This output can be used to verify that the command will produce the exact migration that is desired by the user and to modify the resulting file if it is not.

Note: The file output is only a listing of what the system chose to do at the time the `validate` command was run. To guarantee that the exact pairs that were returned are used, run the `symmigrate establish` command with the file that was output by the `validate` command.

Establish

The `symmigrate establish` command creates the migration session and begins the migration of the source devices to the new target location. The target protection type and disk group number must be provided by the user (unless using a device pair file). This is the point in the migration process that the mirrors are moved to the source device from the target, if using configured space, or are created if using unconfigured space.

Like all other CLI interfaces, the migration control operation will operate on the metahead only. Only the metahead is needed for the control operation; the meta members will be determined by the system. In a configured disk space migration, if the source device is a metadvice, non metadvice can be chosen to match the source device provided that the devices are matched in size and quantity to the meta members. In this case, the output from the `validate` command will be a device-devices list instead of a device-device pair. The source device will be the meta head and the target devices will be a comma separated device list. Alternatively, if there is an available metadvice to use as a target, it can be specified in the standard pair file format.

Note: CKD striped metadvice can only migrate to unconfigured space, and the target protection type is restricted to RAID-1.

Migrating to configured space

A migration to configured space can be performed using a device file or by specifying target criteria.

The following sections explain how to perform each type of migration operation.

Device file Use the following syntax create a migration session using a device file:

```
symmigrate [-v]
            [-i Interval] [-c Count] [-noprompt]
            -name SessionName
            -file DevFile -sid SymmID
validate
establish
```

For example, if the `input.txt` file contains the following pairs:

```
10      20
12      22
14      27
```

The SYMCLI command would be, as follows:

```
symmigrate validate -sid 123 -f input.txt -name DevMigration
symmigrate establish -sid 123 -f input.txt -name DevMigration
```

Target criteria To perform a migration to configured space, and specifying the target criteria, follow this form:

```
symmigrate [-v]
            [-i Interval] [-c Count] [-noprompt]
            -name SessionName
            <-file SrcFile -sid SymmID |
            -sg SgName -sid SymmID |
            -g DgName [-bcv | -tgt]>
            -tgt_config -tgt_dsk_grp DskGrp
            <-tgt_unprotected
            -tgt_raid1
            -tgt_raid5 -tgt_prot <3+1 | 7+1> |
            -tgt_raid6 -tgt_prot <6+2 | 14+2>>

            validate [-outfile OutputFile]
            establish
```

Where:

-name — Indicates a session name when validating or establishing a session.

The source devices can be identified by one of the following:

- ◆ A file, *SrcFile*, which requires the Symmetrix ID
- ◆ A storage group (*-sg*), which requires the Symmetrix ID
- ◆ A device group, with an optional pointer to the BCV or TGT list

The target criteria is specified as follows:

-tgt_config — tells the array to look for configured space.

-tgt_dsk_grp — identifies the disk group location for the migration. The source devices will be in that disk group when the migration is complete.

Protection — the following options specify the protection type that the devices will have when the migration is complete:

-tgt_unprotected — Unprotected. This option is only valid if the source devices are unprotected.

-tgt_raid1 — RAID 1 protection.

-tgt_raid5 — RAID 5 protection with either -tgt_prot 3+1 or 7+1.

-tgt_raid6 — RAID 6 protection, with either -tgt_prot 6+2 or 14+2.

The `symmigrate validate` command can be used to produce a file that will indicate which devices the system identifies as the target *at the time* the `symmigrate validate` command is run.

Migrating to unconfigured space

This section explains how to perform a virtual LUN migration to unconfigured space.

Use the following form when migrating devices to unconfigured space:

```
symmigrate [-v]
            [-i Interval] [-c Count] [-noprompt]
            -name SessionName
            <-file SrcFile -sid SymmID |
            -sg SgName -sid SymmID |
            -g DgName [-bcv | -tgt]>

            <-tgt_unconfig -tgt_dsk_grp DskGrp

            <-tgt_unprotected
            -tgt_raid1
            -tgt_raid5 -tgt_prot <3+1 | 7+1>
            -tgt_raid6 -tgt_prot <6+2 | 14+2>>

validate
establish
```

Where:

-name — Indicates a session name when validating or establishing a session.

The source devices can be identified by one of the following:

- ◆ A file, *SrcFile*, which requires the Symmetrix ID
- ◆ A storage group (*-sg*), which requires the Symmetrix ID
- ◆ A device group, with an optional pointer to the BCV or TGT list

The target criteria is specified as follows:

-tgt_unconfig — tells the array to look for unconfigured space.

-tgt_dsk_grp — identifies the disk group location for the migration. The source devices will be in that disk group when the migration is complete.

Protection — the following options specify the protection type that the devices will have when the migration is complete:

-tgt_unprotected — Unprotected. This option is only valid if the source devices are unprotected.

-tgt_raid1 — RAID 1 protection.

-tgt_raid5 — RAID 5 protection with either **-tgt_prot 3+1** or **7+1**.

-tgt_raid6 — RAID 6 protection, with either **-tgt_prot 6+2** or **14+2**.

An example of a virtual LUN migration of devices in a device group **-tgt** list to unconfigured space, with RAID 5 3+1 protection follows:

```
symmigrate validate
-g mydg -tgt
-tgt_raid5 -tgt_prot 3+1 -tgt_dsk_grp 5
-tgt_unconfig -name DGunconfig

symmigrate establish
-g mydg -tgt
-tgt_raid5 -tgt_prot 3+1 -tgt_dsk_grp 5
-tgt_unconfig -name DGunconfig -nop
```

Managing virtual LUN migration sessions

The `symmigrate` CLI interface will allow the user to list, verify, query, and terminate the migration sessions by using the session name, or by using the session name and specifying the filename, device group, or storage group.

Note: If a session has existed for 2 weeks without being listed or queried, it will be automatically removed at which point it will not be returned by a `query` or `list` command.

The Symmetrix device lock will be held on the source device(s) only when the establish and terminate operations are in progress. The lock will be released from the time the migration enters the Sync In Progress state until the time that the user terminates the session. When the session is terminated, the lock will be taken again.

A limit of 16 concurrent migration and configuration change sessions can be run at one time on a given Symmetrix array.

Listing sessions

Use the `symmigrate list` command to display all the existing migration sessions for a specified Symmetrix array, or all Symmetrix arrays, as follows:

```
symmigrate [-v] [-i Interval] [-c Count]
list [-sid SymmID] [-names] [-detail]
```

To display the migration sessions currently on Symmetrix array 258, enter:

```
symmigrate list -sid 258
```

The display will look similar to the following:

Symmetrix ID: 000192600258

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0010	N/A	0	Migrated	100	Migrate1
0012	0016	0	Migrated	100	Migrate2
0014	N/A	0	Migrated	100	Migrate1
0040	0020	4000	SyncInProgress	80	Application1
0041	0021	6000	SyncInProgress	70	Application1
Total -----					
	Tracks	10000			
	MB(s)	312.5			

The `-detail` option includes the source and target information for each session in the display, as shown in the following sample output:

Symmetrix ID: 000192600258

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	TGT Dsk Grp	TGT Protection	Session Name
0010	N/A	0	Migrated	100	01 RAID-5 (7+1)		Migrate1
0012	0016	0	Synchronized	100	04 RAID-1		Migrate2
0014	N/A	0	Migrated	100	01 RAID-5 (7+1)		Migrate1
0040	0020	4000	SyncInProgress	80	03 RAID-6 (14+2)		Application1
0041	0021	6000	SyncInProgress	70	03 RAID-6 (14+2)		Application1
Total -----							
	Tracks	10000					
	MB(s)	312.5					

Querying sessions

Use the `symmigrate query` command to display the status of a specific migration session, or all migration sessions, as follows:

```
symmigrate [-v] [-i Interval] [-c Count] -name SessionName
            <-file <DevFile | SrcFile> -sid SymmID |
            -sg SgName -sid SymmID |
            -g DgName [-bcv | -tgt]           |
            -sid SymmID>

query [-detail]
```

The `query` command can be used with the session name only, or you can additionally filter the request by adding the specific filename, device group name, or storage group name that was used with that session. Supply the Symmetrix ID, except when using device group.

To display the status of the session named `migrate1` on Symmetrix 258, enter:

```
symmigrate query -name migrate1 -sid 258 -detail
```

Symmetrix ID: 000192600258

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	TGT Dsk Grp	TGT Protection	Session Name
0010	N/A	0	Migrated	100	01	RAID-5 (7+1)	Migrate1
0014	N/A	0	Migrated	100	01	RAID-5 (7+1)	Migrate1

Total		-----					
Tracks		0					
MB(s)		0.0					

Verifying sessions

The `symmigrate verify` command allows you to determine if a migration session is in a specific state. The syntax follows:

```
symmigrate -sid SymmID -name SessionName [-i Interval] [-c Count]
verify [-syncinprog | -migrinprog | createinprog | -synchronized |
-migrated | -failed | -invalid]
```

A virtual LUN migration session is complete when it is in the Migrated state.

To verify whether a migration session named `migrate1` is in the migrated state, enter:

```
symmigrate -sid 258 -name migrate1 verify -migrated -i 30
```

The output is similar to the following:

```
NONE of the devices are in the 'Migrated' state.
NONE of the devices are in the 'Migrated' state.
.....
NONE of the devices are in the 'Migrated' state.
ALL of the devices are in the 'Migrated' state.
```

Terminating sessions

A migration session remains active until it is terminated. When the status of the devices in the session is Migrated, the session is complete and can be terminated. To terminate a session, use the following form:

```
symmigrate -name SessionName
            <-file <DevFile | SrcFile> -sid SymmID |
            -sg SgName -sid SymmID |
            -g DgName [-bcv | -tgt]           |
            -sid SymmID>

terminate
```

Viewing the audit log

For any control command, the API will log the start and end of the operation as well as any errors that may occur. [Table 20](#) provides a description of the information that is saved in the audit log, based on the target selection type.

Table 20 **Audit log session information**

Target selection type	Information in the audit log
Device file	Session Name, Source Devices, Target Disk Group, Target Protection Type, and Target Devices
Configured/unconfigured space with no specified devices	Session Name, Source Devices, Target Disk Group, and Target Protection Type

Virtual LUN migration examples

This section provides several virtual LUN migration examples. [Appendix B, “Virtual LUN Migration Example,”](#) provides the step-by-step input and output for a virtual LUN migration session from start to finish.

1. The migration source and target device pairs are in a file, and the target protection type is RAID-5 (3+1).

Device pairs in `input.txt` are:

```
10      20
11      21
12      22
13      23
```

Usage:

```
symmigrate validate -sid 123 -f input.txt -name Migrate2
```

```
symmigrate establish -sid 123 -f input.txt -name Migrate2
```

2. The user has a storage group `mystoragegrp` that contains devices that he wants to migrate to a new protection type but in the same disk group (group number 1), using configured space. The exact target devices that are chosen are irrelevant.

Usage:

```
symmigrate validate -sid 123  
-sg mystoragegrp  
-tgt_raid6 -tgt_prot 6+2 -tgt_dsk_grp 1  
-tgt_config -name sgconfig
```

```
symmigrate establish -sid 123  
-sg mystoragegrp  
-tgt_raid6 -tgt_prot 6+2 -tgt_dsk_grp 1  
-tgt_config -name sgconfig
```

3. The user has a device group "mydg" that has a TGT list of devices that need to be migrated to unconfigured space in disk group 5. The protection type is to remain the same at RAID-5(3+1). The output file "diskfile" will be created with the chosen physical disks.

Usage:

```
symmigrate validate  
-g mydg -tgt  
-tgt_raid5 -tgt_prot 3+1 -tgt_dsk_grp 5  
-tgt_unconfig -name DGunconfig
```

```
symmigrate establish  
-g mydg -tgt  
-tgt_raid5 -tgt_prot 3+1 -tgt_dsk_grp 5  
-tgt_unconfig -name DGunconfig -nop
```

4. To list all migration sessions established on one Symmetrix array, enter:

```
symmigrate list -sid 123 -detail
```

5. To monitor a migration session named `mysession`, enter:

```
symmigrate query -name mysession -sid 123 -i 5 -c 2
```


Performing Double Checksum Operations

This chapter describes the Double Checksum concepts and how to minimize the impact of I/O errors on database consistency during I/O transfers between hosts and Symmetrix storage devices using the Double Checksum commands of the SYMCLI. This chapter covers the following topics:

- ◆ Overview 254
- ◆ Implementing EMC Double Checksum for Oracle 256
- ◆ Implementing Generic SafeWrite for generic applications 260
- ◆ Syntax and examples 265

Overview

The EMC Double Checksum feature provides a method to help minimize the impact of I/O errors on database consistency during I/O transfers between hosts and Symmetrix storage devices.

For Oracle, **EMC Double Checksum for Oracle** contains a rich set of checks that can be natively performed by the Symmetrix. For each Relational Database Management System (RDBMS) write in the Symmetrix system, checksum values are computed and compared to test the data for any corruption picked up along the way from the host. Although errors of this kind are infrequent, they can have a considerable effect on data availability and recovery. Refer to [“Implementing EMC Double Checksum for Oracle” on page 256](#) for details on this feature.

For generic RDBMS applications, **EMC Double Checksum for Generic Applications** provides the Generic SafeWrite¹ feature to help protect critical applications from incurring an incomplete write, and subsequent torn page, due to a failure with a component connected to the Symmetrix Front End Channel Adapter. Generic SafeWrite is most often used to protect against corruption are from HBA and link failures including server crashes, where essentially, it will help protect against fractured writes that can occur before the data reaches the Symmetrix. Refer to [“Implementing EMC Double Checksum for Oracle” on page 256](#) for details on this feature.

This chapter contains overview and concept information, see the `symchksum` command in the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help* for the complete list of syntax and options.

Traditional methods of preventing data corruption

Data corruption checking is an integral part of most RDBMS products. For instance, Oracle computes a checksum that verifies the data within each page. If corruption occurs, the checksum will be incorrect when the data is read. However, this checking only takes place within the host system—not the storage array.

As a result, if there is corruption after the data has left the host system, it will not be detected until that data is read back into the system, which can be some time — maybe months—later. The RDBMS will issue an alert, and then the data must be rebuilt from backups and database logs. While a corruption remains undetected, the number of database logs required for recovery increases. This causes the data recovery process to be more complex and time-consuming.

Data corruption between host and conventional storage

Although data appears to the host to travel directly to the Symmetrix array, it passes through multiple hardware and software layers. These can lead to problems such as corruption introduced by errors in the operating system or the I/O driver. Hardware can also introduce corruption, such as errors in the host adapter, cable and connector problems, static electricity, and RF noise and interference.

1. Generic SafeWrite has been created with RDBMS in mind. Applications intended for use with Generic SafeWrite include, but are not limited to Microsoft Exchange, Microsoft SQL Server, DB2/UDB and Oracle. Generic SafeWrite is not intended for use with applications where torn pages are not a concern, such as fileshares or FTP servers.

This means that valid data within the RDBMS might arrive corrupted at the storage device. The storage device writes the data as is, because it has no way of validating the data.

Benefits of checking within the Symmetrix array

With this feature, the Symmetrix array can perform error checks on data pages handled within a checksum extent *as they are written to the disk*. The check takes place before the write command is acknowledged. If an error is detected within the blocks of the extent, the I/O can be rejected or reported in a phone home connection or logged in the Symmetrix error log facilities. The error action can be specified by the administrator.

This checking feature minimizes the possibility of data corruption occurring between the host and the Symmetrix array. It improves the recovery time by flagging the error at the time of the "write." When this error condition is raised, and reject I/O is selected, Oracle takes an action, such as taking the tablespace offline.

Implementing EMC Double Checksum for Oracle

The `symchksum` command allows you to perform control operations that manage the checksum I/O extents.

For example, to enable Oracle checksum checking on the extents of all the devices that define the current database instance and then to phone home on error, enter:

```
symchksum enable -type Oracle -phone_home
```

This command requires an Oracle PAK license.

The following are current restrictions for this feature:

- ◆ Oracle databases version 8.0 and above are supported.
- ◆ Data block size is limited to 32 KB.
- ◆ Checksum data objects can only be Oracle data files, control files, and redo logs.
- ◆ Devices cannot be diskless devices (DLDEVs)

The Oracle instance being checked must be started with the following `init.ora` configuration parameter set to true: `db_block_checksum=true`.

Note: Manipulating remote devices over an SRDF link is not supported. The `symchksum` command can only manipulate devices in locally attached Symmetrix arrays.

Other checksum operations

The following additional checks are available:

- ◆ `MagicNumber` — Verifies the 3-bit magic number that appears in Oracle data blocks. (Enabled by default.)
- ◆ `NonzeroDb` — Checks for non-zero data block address. The dba stored in a data block is never zero. (Enabled by default.)
- ◆ `Check_All_Blocks` — Applies checks to each block in a write. Otherwise, only the first block is checked.
- ◆ `Straddle` — Checks that the write does not straddle known Oracle areas.
- ◆ `Check_dba` — Checks that the logical address embedded by Oracle is compatible with the storage address of the writes.

Note: If `-check_dba` is selected, extents from separate datafiles cannot be collapsed. The `-check_dba` option requires data specific to each datafile, which would be lost in an extent collapse.

As a result, each datafile will occupy at least one Symmetrix device extent. When using a logical volume manager and `-check_dba`, this creates a limitation of 200 datafiles per volume group or disk group.

To enable Oracle checksum and add the optional operations of check DBA and check all blocks, enter:

```
symchksum -type Oracle enable -check_dba -check_all_blocks
```


With the addition of these new tests, the output when you list the extents will look similar to the following:

Symmetrix ID: 000187900671

```

D E V I C E S      W I T H      C H E C K S U M      E X T E N T S

                                     Action  Checks
                                     R P C C A N D
                                     e h h M h S l Z F i
                                     j o k a k t l r s
                                     L e n s g D r B D a c
                                     o c e u i B a l B c r
Device Name          Num Blk          g t H m c A d k A t d
-----
/dev/sdj              0048      1 16b Oracle  X X X X X . . . X X X
/dev/sdz              0058      1 16b Oracle  X X X X X . . . X X X

2 Devices with Checksum extents were found.
```

You can use this output to determine which features are enabled on the devices with checksum extents.

Suppressing default features

By default, the `symchksum` command attempts to select the following tests:

- Block checksum value (Checksum)
- Blocksize verification (BlkSize)
- 3-bit magic number (MagicNumber) in Oracle data blocks (not available for redo logs)
- Checks for non-zero data block addresses (NonZeroDbA)

To turn off any of the automatic checksum features, use the `-suppress_feature` option and supply the name of the feature, for example:

```
symchksum -type Oracle enable -suppress_feature MagicNumber
```

The `-suppress_feature` option is only for the operations that run by default. If you want to turn off an option that was manually enabled, such as `-phone_home`, you must disable the checksum operation and begin again with a new `symchksum enable` command.

Enabling checksum options

When using the `symchksum enable` command, the user can decide if they would like to reject the I/O, or have the Symmetrix phone home when a checksum error is detected.

If an I/O is not a multiple of the object block size, the user can choose to reject the I/O. This is called a fractured I/O, and is selected with the `-fractured_reject_io` option. When using this option, the `-reject_io` option must also be used.

When extents are enabled with the `-discard` option, EMC Double Checksum writes blocks to disk until a failed block is detected. The `-discard` option divides a large I/O into smaller units of 32 KB each. When a checksum failure is detected, all blocks

in that unit and subsequent units are discarded. When using this option, the `-reject_io` option must also be used.

The `symchksum enable` command understands the Oracle database structure. The feature can be enabled for tablespaces, control files, redo logs, or the entire database.

Note: For Oracle 9i and above, if the block size for a tablespace is altered, then the user must disable and then re-enable the extents of the tablespace to ensure that the block size of the enabled extents match the block size of the tablespace.

Note: When FF or power down occurs, extents are lost. You must run the `symchksum enable` command again.

Verifying checksum is enabled

The `symchksum` command also allows you to verify that the data file's extents are currently enabled for checksum checking. This provides an easy way to determine if the specified tablespace or instance is fully protected by the Symmetrix checksum feature. The `verify` action will report if all, some, or none of the Oracle data file's extents are enabled for checksum checking. This is useful in environments where the database configuration changes frequently. An example of this is if a new data file was added but checksum was not enabled for the new file.

The `symchksum verify` command understands the Oracle database structure. The feature can be enabled for tablespaces, control files, redo logs, or the entire database.

Validating for checksum operations

The `symchksum` command also allows you to validate your Oracle tablespace or instance for checksum operations without performing any active actions. This is helpful when you want to know if your database environment is configured to support Symmetrix checksum functionality without actually making any changes.

If the validate is successful, you can enable EMC Double Checksum on the specified Oracle database or tablespace. The following items are validated:

- ◆ Oracle version 8 or higher is installed
- ◆ Oracle's checksum initialization parameter is set (`db_block_checksum`)
- ◆ If the Oracle data file is created on a striped LVM, that the LVM stripe width is a multiple of the Oracle block size
- ◆ Oracle data file's block size is less than or equal to 32 KB
- ◆ The Symmetrix Enginuity version supports the checksum functionality
- ◆ Each Symmetrix device (prior to Enginuity V5773) has the Checksum flag set
- ◆ Each Symmetrix device has a supportable number of extents defined

The `symchksum validate` command understands the Oracle database structure. The feature can be enabled for tablespaces, control files, redo logs, or the entire database.

Disabling checksum

The `symchksum disable` command understands the Oracle database structure. The feature can be enabled for tablespaces, control files, redo logs, or the entire database.

The `symchksum disable` command can also be used on a device basis. This capability is not normally used, but is provided in the event the tablespace was dropped before EMC Double Checksum was disabled for that object.

When the `disable` action is specified for a Symmetrix device, the `-force` flag is required. Disabling extents in this way can cause a mapped tablespace or database to be only partially protected, therefore, use this option with caution. All the extents being monitored for checksum errors on the specified Symmetrix device will be disabled.

Functionality by Enginuity level

In support of the Oracle and EMC Hardware Assisted Resilient Data (HARD) initiative, EMC Double Checksum provides end-to-end database integrity by employing sophisticated error detection technology to detect and prevent corruption of data blocks written to Symmetrix arrays. [Table 21](#) shows the **EMC Double Checksum for Oracle** functionality provided in each Enginuity version

Table 21 EMC Double Checksum for Oracle functionality by Enginuity level

Enginuity Level	Block checksum verification	Dial home alert	Discard	Max extents ^a / Symmetrix volume	Magic number detection	Zero dba detection	All blocks option ^b	Straddle ^c
5x68	Yes	Yes	No	32	No	No	No	No
5669	Yes	Yes	No	32	No	No	No	No
5670 rev 49-	Yes	Yes	Forced On	200	No	No	No	No
5670 rev 50+	Yes	Yes	Yes	200	Yes	Yes	No	No
5670 rev 75.71+	Yes	Yes	Yes	200	Yes	Yes	Yes	Yes

- Symmetrix extents are internal to the Symmetrix array and are used to map Oracle tablespace locations; they are not the same as Oracle extents
- For Magic Number Detection and Zero dba Detection, the default mode is to check only the first block in an I/O. If the All Blocks Option is selected in the SYMCLI, then Magic Number Detection and Zero dba Detection checks are performed for each Oracle block. For the checksum check, the only mode is to sum the I/O as a whole; it is not affected by the All Blocks Option.
- Sd an extension to the Oracle and EMC HARD initiative, EMC Double Checksum functionality checks for I/O that overlaps defined database areas. For example, consider a database that has data from blocks 16 to 105. If an I/O is received for blocks 100 to 200, this I/O is said to straddle the database's data. This is unexpected and likely to be a potentially corrupting I/O. If the Straddle option is selected in the SYMCLI, then EMC Double Checksum will consider straddling I/O to be an error.

Implementing Generic SafeWrite for generic applications

Generic SafeWrite (GSW) is used to help protect critical applications from incurring an incomplete write, and subsequent torn page, due to a failure with a component connected to the Symmetrix Front End Channel Adapter.

Torn pages: Using generic safewrite to protect applications

A Relational Database Management System (RDBMS), such as Oracle and Microsoft SQL Server, structure data within database files using pages (also referred to as blocks). Pages within a database are the smallest allocation unit size possible for a database object (such as a table or a row).

For example, the page size for Microsoft SQL Server, and for Oracle, is 8 KB. Although this size is configurable, it is usually set to 8 KB. If an incomplete page is written to a database file, a corruption to the database will occur. The resulting corruption is commonly referred to as a *torn page*.

Torn pages are only detected by most RDBMSs after the corruption has been written, when that area of the database is read, which could be long after when the corruption was introduced. In general, the only recovery from a torn page is to perform a restore from a backup (some RDBMSs allow page-level restores, while others require a complete database restore). Torn pages can occur due to failures in various components that lie between the RDBMS and the storage array. Some of these components include the operating system, file system, logical volume manager, I/O driver, host bus adapter, Fibre or SCSI link and storage adapter.

The EMC Double Checksum Generic SafeWrite feature can help protect critical applications from incurring incomplete writes, and subsequent torn pages, due to a failure with a component connected to the Symmetrix Front End Channel Adapter.

Most often, Generic SafeWrite will be used to protect against corruption that occurs when the HBA and link fails (including server crashes). In this scenario, Generic SafeWrite will protect against fractured writes that occur before the data reaches the Symmetrix array.

Why generic?

Generic SafeWrite is deemed *generic* because the checks performed to ensure complete data are application independent. For instance, Generic SafeWrite will not perform any Oracle- or SQL Server-specific checksums to verify data integrity. It is important to note that for Oracle, EMC Double Checksum for Oracle provides a rich set of checks which can be natively performed by the Symmetrix array. For more information on EMC Double Checksum for Oracle, refer to [“Implementing EMC Double Checksum for Oracle” on page 256](#).

Where to enable Generic SafeWrite

Generic SafeWrite only needs to be enabled for specific devices on the Symmetrix array. For a RDBMS, Generic SafeWrite only needs to be enabled for devices that support data files. The list below gives an example of database files where the supporting devices for these files should have Generic SafeWrite enabled:

Microsoft Exchange:

- .edb files
- .stm files

Microsoft SQL Server:

- .mdf files
- .ndf files

Oracle:

- Data files
- Control files

Note: Manipulating remote devices over an SRDF link is not supported. The `symchksum` command can only manipulate devices in locally attached Symmetrix arrays.

It is recommended to enable Generic SafeWrite for database file devices, though it is not necessary to enable it for database log devices. In general, a RDBMS will write to its respective log file with a 512 byte sector alignment. The RDBMS can therefore determine the last sector that was correctly written and subsequently discard or rollback any transactions that are incomplete.

Note: It is always a best practice to separate the location of database files and log files for a given database onto unique devices. There are cases, however, where the data file and log file may share the same device. In this case, it is still possible to have GSW enabled, however, there will be a performance impact to the log writes which may impact application performance.

There are no restrictions regarding the size of a device or the number of devices where GSW can be enabled. However, for both Generic Safewrite and Oracle Safewrite only SRDF/S and TimeFinder/Mirror are supported. It is also supported to enable GSW on filesystems across all logical volume managers as well as on raw devices, given the OS platforms are supported by the Solutions Enabler Storage Resource Management (SRM) component. When using filesystems on Windows and Linux hosts, for performance reasons, it is strongly recommended to ensure the filesystems are properly aligned with the storage. For more information regarding filesystem alignment, refer to *Using diskpart and diskpart to Align Partitions on Windows Basic and Dynamic Disks* available on EMC Powerlink®.

Configuring Generic SafeWrite

To use Generic SafeWrite, you must:

- ◆ For Enginuity versions prior to 5773, enable the `RDB_cksum` device flag on all devices targeted for Generic SafeWrite use
- ◆ Run a discover operation to update the physical device information in the SYMAPI database
- ◆ Enable Generic SafeWrite on all devices targeted for Generic SafeWrite use with the `symchksum` command

Setting the `RDB_cksum` Symmetrix device flag

Note: Symmetrix arrays running Solutions Enabler V6.5.1 and higher, with Enginuity V5671 and higher, do not require the `RDB_cksum` device flag enabled for Double Checksum and Generic SafeWrite use.

For Enginuity versions prior to 5773 only:

Before you can use Generic SafeWrite, the `RDB_cksum` Symmetrix device flag must be enabled on all devices targeted for Generic SafeWrite use. This change does not turn Generic SafeWrite on, it only allows it to be enabled on the specified devices.

The `RDB_cksum` device flag can be set by using the SYMCLI `symconfigure` command, which will perform a Symmetrix configuration change. For more information on using `symconfigure`, refer to [Chapter 1, “Managing Configuration Changes.”](#)

Note: If `symconfigure` cannot be used, the appropriate device flag can also be set on the array by a EMC Customer Support Engineer.

The following is an example command:

```
symconfigure -sid 54 -f c:\enable_cksum.txt commit
```

Where the `c:\enable_cksum.txt` file contains the following command:

```
set device 0015:0019 attribute=RDB_Cksum;
```

Note: When using `symconfigure`, if metavolumes are used, this flag only the metahead needs to be specified.

Enabling Generic SafeWrite

Once the device flags are set on the Symmetrix array, it is then possible to use the `symchksum` command to enable Generic SafeWrite. Before running the `symchksum` command, confirm the following:

- ◆ The devices that you enabled for Generic SafeWrite are visible to the host from where the `symchksum` command will be run.
- ◆ Run a `symcfg discover` command after presenting devices to a host, in order to update the SYMAPI configuration database with the correct physical drive information.

Using the `symchksum` command, Generic SafeWrite can be enabled by specifying a specific device, a range of devices, or a device group.

Enabling for a device

To enable Generic SafeWrite for a device, use the command syntax shown in the example below:

```
symchksum enable -type generic dev 005 -sid 54
```

Note: If this is a metadvice, only the metahead needs to be specified.

Enabling for a range of devices

To enable Generic SafeWrite for a contiguous range of devices:

```
symchksum enable -type generic -range 005:025 -sid 54
```

Enabling for a device group

To enable Generic SafeWrite for a device group:

```
symchksum enable -type generic -g sql_data -sid 54
```

Note: Enabling Generic SafeWrite on a Composite Group (CG) is currently not supported.

The `symchksum enable -type generic` command automatically sets the Log, Phone Home, and Generic Double Checksum options as described below:

- ◆ Log — Indicates that errors will be sent to the Symmetrix error log. These events should be visible with the `symevent` command.
- ◆ Phone Home — Indicates that an error will initiate a call by the Symmetrix to EMC Customer Service.
- ◆ Generic — Allows two functions to be performed by the Symmetrix array. First, when an incomplete write is detected, it will be rejected and the Symmetrix will force the I/O to be retried from the host. Then, if the host is not available to retry the I/O, the write will be discarded, preventing it from being written to disk.

How to disable Generic SafeWrite

Generic SafeWrite can be disabled using the `symchksum disable -type generic` command as shown in the same examples below.

Disabling for a device

To disable Generic SafeWrite for a device, use the command syntax shown in the example below:

```
symchksum disable -type generic dev 005 -sid 54
```

Note: If this is a metadvice, only the metahead needs to be specified.

Disabling for a range of devices

To disable Generic SafeWrite for a contiguous range of devices:

```
symchksum disable -type generic -range 005:025 -sid 54
```

Disabling for a device group

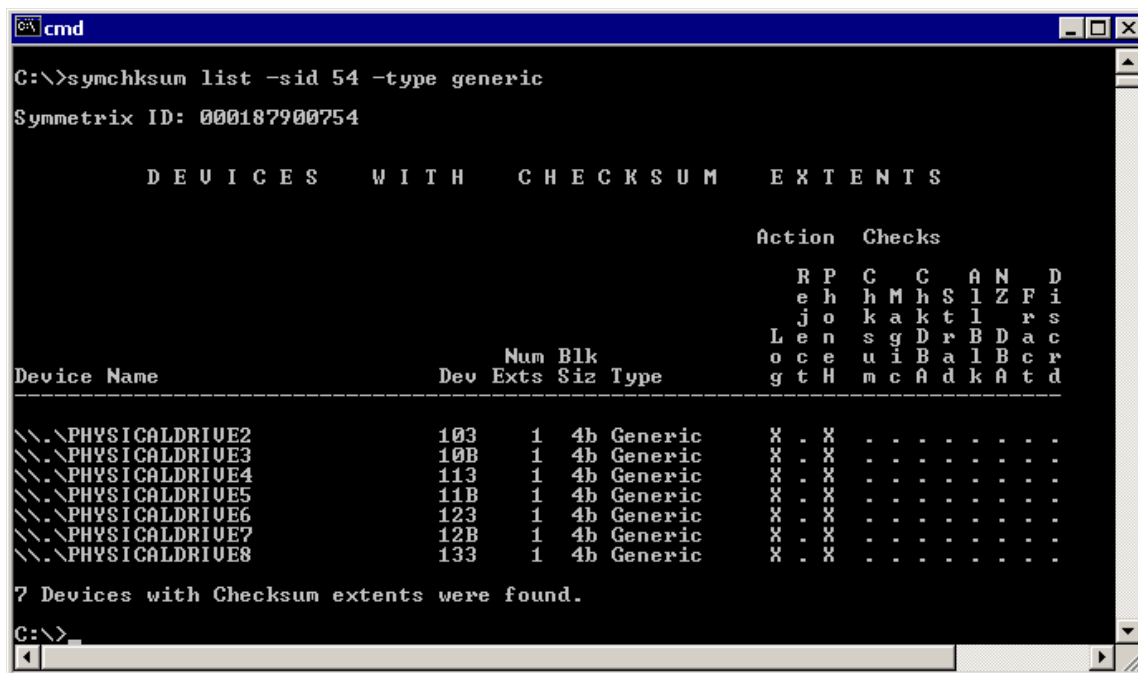
To disable Generic SafeWrite for a device group:

```
symchksum disable -type generic -g sql_data -sid 54
```

Listing Generic SafeWrite devices

To list which devices are Generic SafeWrite enabled, use the `symchksum list` command with the `-type generic` option. Only Generic SafeWrite-enabled devices that are visible to the host running the `symchksum list` command are returned.

The following example shows the expected output from the list command, with Generic for the type and the Log and PhoneH (short for Phone Home) options set.



```

C:\>symchksum list -sid 54 -type generic
Symmetrix ID: 000187900754

          D E V I C E S      W I T H      C H E C K S U M      E X T E N T S

                                     Action   Checks
                                     R P C   C   A N   D
                                     e h h M h S l Z F i
                                     j o k a k t l r s
                                     L e n s g D r B D a c
                                     o c e u i B a l B c r
                                     g t H m c A d k A t d

Device Name                        Num Blk
                                Dev Exts Siz Type
-----
\\.\PHYSICALDRIVE2                103   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE3                10B   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE4                113   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE5                11B   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE6                123   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE7                12B   1 4b Generic  X . X . . . . .
\\.\PHYSICALDRIVE8                133   1 4b Generic  X . X . . . . .

7 Devices with Checksum extents were found.
C:\>

```

Figure 29 Generic SafeWrite list output

The `symchksum show` command can also be used to look at a specific device. For example:

```
symchksum show dev 103 -type generic -mb -sid 54
```

Performance considerations

Application performance should remain unaffected because database log writes, as well as database reads, will be performed normally. However, outside of application performance, there may be a slight increase in the write response time to the database file devices depending on application profile and usage. In general, this response time increase should not impact application performance. Writes to a database file are done asynchronously, therefore write response times to this file are less of a concern than to the log device.

Syntax and examples

This section contains the `symchksum` argument descriptions and several examples of using the SYMCLI `symchksum` command. See the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help* for the complete list of syntax and options.

Table 22 `symchksum` syntax and descriptions

Command	Argument	Description
<code>symchksum</code>	<code>list</code>	Lists all devices that currently have checksum checking enabled. The list can be filtered by using the <code>-type</code> option (to show Oracle or generic).
	<code>show</code>	Displays the extents of a specified device that are having checksum checking performed.
	<code>enable</code>	Enables checksum checking on the extents of the specified devices.
	<code>disable</code>	Disables checksum checking on the extents of the specified devices.
	<code>validate</code>	Validates that a specified database or tablespace is able to have checksum checking enabled.
	<code>verify</code>	Verifies whether the specified database or tablespace has checksum checking enabled on all their devices.

To list the devices on Symmetrix array 3890 that have extents being checked for checksum errors, enter:

```
symchksum list -sid 3890
```

To show all the extents of Symmetrix device 0A1 on Symmetrix array 3890 that are being checked for checksum errors, enter:

```
symchksum show dev 0A1 -sid 3890
```

To enable Checksum on the extents of all the devices that define the current database instance and then to phone home on error, enter:

```
symchksum enable -type Oracle -phone_home
```

To enable Checksum on the extents of all the devices that define the tablespace and then to log on error, enter:

```
symchksum enable -type Oracle -tbs SYSTEM
```

To verify that Oracle tablespace USER01 has Checksum enabled on all the devices that have defined it, enter:

```
symchksum verify -type Oracle -tbs USER01
```

To disable Checksum on the current database instance, enter:

```
symchksum disable -type Oracle
```

Note: Disable by device should only be used under special circumstances. For example, this option can be used to remove extents if a database or a tablespace has been dropped without first doing a normal disable. In this case, disable by device can be used to remove the extents.

To disable (with force) Checksum for all checksum extents on Symmetrix device 0A1 on Symmetrix unit 3890, enter:

```
symchksum disable dev 0A1 -sid 3890 -force
```

This chapter describes the QOS concepts and how to manage the Quality of Service (QoS) on devices in your storage environment using the QOS commands of the SYMCLI. The chapter covers the following topics:

◆ Overview	268
◆ Setting QoS copy parameters and values	269
◆ Setting QoS LRU parameters and values.....	270
◆ Using Symmetrix Priority Controls.....	271
◆ Using dynamic cache partitions.....	273
◆ QoS examples.....	278

Overview

Quality of Service (QoS) allows you more flexibility in managing your Symmetrix system's performance. By increasing the response time for specific copy operations on selected devices, you can increase the overall performance of the other Symmetrix devices.

The QoS (Quality of Service) SYMCLI feature allows you to adjust the data transfer (copy) pace on specified devices, or devices in a device group, for certain operations.

The contention for cache access can be quality of service managed by the least recently used (LRU) ring partitions in the Symmetrix cache.

With Enginuity 5772, you can control the priority service time of devices and control cache partitions for different device groupings.

The `symqos` command provides Quality of Service controls on specified devices. [Table 23](#) lists the features of the QoS command.

Table 23 SYMAPI Quality of Service command

Command	Description
<code>symqos</code>	Invokes quality of service controls on certain devices. Possible operations are: • Displays a list of quality of service priorities assigned to devices and mirrors. • Sets the values and parameters for QoS priorities on the devices and mirrors. • Queries the QoS copy priorities for members of a specified group. • Lists the LRU cache assignments. • Sets the LRU cache assignments to a device or device group. • Resets the LRU assignment back to the default LRU. • Enables/disables priority service time for devices. • Sets priority service time for a range of devices or a device group. • Lists priority service time for a range of devices or a device group. • Enables/disables cache partitioning for a range of devices or a device group. • Creates cache partitions; adds devices to and removes devices from cache partitions. • Lists specific devices in a cache partition.

Setting QoS copy parameters and values

Using SYMCLI, you can set QoS parameters and values for specified devices managing the copy pace (priority) and the LRU assignments.

Copy pace to certain devices

Different copy pace (priority) levels can be set on certain types of devices. Possible copy pace levels on the devices are 0, the highest, to 16, the lowest. Zero is the default.

```
set <BCV|RDF|MIR|CLONE> pace|priority Value [LdevName]
```

To specify which copy operations you are adjusting the pace for, there are five copy operation types to choose from:

- ◆ BCV — TimeFinder copy operations
- ◆ RDF — SRDF copy operations
- ◆ MIR — Mirror copy operations
- ◆ CLONE — Clone (previously called Snap) copy operations

To set a copy pace to 1 during BCV operations for the standard devices in group *ProdDB*, enter:

```
symqos set BCV pace 1 -g ProdDB -nobcv
```

or:

```
symqos set BCV priority 1 -g ProdDB -nobcv
```

Note: The syntax items *pace* and *priority* are interchangeable nouns, which can be used to identify and set the same parameter.

For BCV operations, to set the BCV copy pace value to 1 on logical device *BCV002* of device group *ProdDB*, enter:

```
symqos set BCV pace 1 -g ProdDB BCV002
```

For RDF operations, in device group *ProdDB*, to set the copy pace level to 2 for all STD and BCV devices, enter:

```
symqos set RDF pace 2 -g ProdDB
```

For RDF operations, to set the RDF copy pace value to 1 on just BCV devices of device group *ProdDB*, enter:

```
symqos set RDF pace 1 -g ProdDB -bcv
```

To set a copy pace to 1 during copy operations for mirror device *DEV021* in group *ProdDB*, enter:

```
symqos set MIR pace 1 -g ProdDB DEV021
```

Setting QoS LRU parameters and values

Using SYMCLI, you can set QoS parameters and values for specified devices managing LRU cache I/O operations.

The contention for cache access can be managed by the least recently used (LRU) ring partitions in the Symmetrix cache. You can manage the cache strategy utilizing LRU rings (groups) and assigning certain devices or device groups to them. The syntax follows:

```
set LRU <#,#,# | ALL> [LdevName]
set LRUname <LRUname> [LdevName]
```

To assign all devices of device group `ProdDB` to LRU group 2, enter:

```
symqos -g ProdDB set LRU 2
```

Or, to assign all devices of device group `ProdDB` to an LRU group named `GROUP_2`, enter:

```
symqos -g ProdDB set LRUname GROUP_2
```

Using the `-lru` option of `symcfg list`, you can list the cache-slot allocation and allocation percentage of a specified LRU cache management group number. You can list all the LRUs if you specify `ALL`.

Using the `reset` option of `symqos`, you can reset the LRU assignment back to the default LRU. For example:

```
symqos reset LRU LdevName
```

Using Symmetrix Priority Controls

For Enginuity version 5772, using Symmetrix Priority Controls (SPC), the `symqos` command can also be used to enable and set the priority of service time of host I/O to a device, a range of devices, or devices in a device group. The priority values are from 1 to 16, with 1 being the highest. When this feature is enabled, all devices in the Symmetrix array are set to device priority 1 by default.

The following device types can not be assigned a priority:

- ◆ SFS
- ◆ VAULT
- ◆ SAVE
- ◆ DRV
- ◆ VDEV

For Enginuity 5773, thin devices and DATA devices cannot be assigned a priority.

For Enginuity 5874, thin devices, DATA devices, and diskless devices (DLDEVs) cannot be assigned a priority.

Enabling and disabling priority of service

To enable the priority of service feature for an array, use the following form:

```
symqos -pst -sid SymmID enable | disable
```

For example, to enable host I/O device priority feature for Symmetrix 1234, enter:

```
symqos -pst -sid 1234 enable
```

To disable host I/O device priority feature for Symmetrix 1234, enter:

```
symqos -pst -sid 1234 disable
```

Setting priority of service

When priority of service is enabled, all eligible devices in the array are assigned priority 1 by default. To assign a new priority, use the following form:

```
symqos -pst -sid SymmID
[-range SymDevStart:SymDevEnd | -g DgName] [-std] [-bcv]
set hostio priority Value
```

Where:

-pst — Indicates a priority of service operation.

set hostio priority — Assigns host I/O priority to a range of devices or devices in a device group. When setting the priority for devices in a device group, the priority can optionally be assigned to either STD or BCV devices, or both.

Value — For host I/O priority setting, values are range from 1 (highest priority) to 16 (lowest priority).

For example, to assign all STD devices of device group DeviceGroup to host I/O priority 4, enter:

```
symqos -pst -g DeviceGroup set hostio priority 4 -std
```

If neither STD or BCV is specified, the default is STD.

Listing priority of service

To view the priority of service for a range of devices or devices in a device group, or a given host I/O priority, use the following form:

```
symqos -pst [-hostio_priority Value]

list -sid SymmID [-range SymDevStart:SymDevEnd]
list -g DgName [-std] [-bcv]
```

For example, to list the priority of service time for Symmetrix 131, devices 002 and 003, enter:

```
symqos -pst -sid 131 list -hostio_priority -range 002:003
```

The following is sample output from this command:

```
Symmetrix ID           : 000190300131
Priority Service Time State : Enabled
```

Device			Priority
Sym	Config	Attr	HostIO
0002	2-Way Mir	-	2
0003	2-Way Mir	(M)	2

When listing for device groups, if the device type is not specified, the display will list both STD and BCV devices.

For complete syntax for the `symqos` command, refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

Using dynamic cache partitions

Dynamic Cache Partitioning (DCP) divides the cache memory into multiple partitions with unique names and their device path assignments. Partition areas can be made static or dynamic in size. The dynamic partitioning provides flexibility to the amount of floating memory that can be allocated with a high and low watermark. This allows memory resources to be temporarily donated to other partitions when needed. The `symqos` command allows you to create partitions for different device groupings in addition to the default partition that all devices belong to initially. Each partition will have a target cache percentage as well as a minimum and maximum percentage. In addition, you can donate unused cache to other partitions after a specified donation time.

Note: Thin devices and DATA devices are allowed in cache partitions and should be assigned to the same named cache partition. If this is not possible, any Write Pendings to the thin device will be counted in the cache partition to which thin device belongs, and the cache partition to which the DATA device belongs.

Cache partitioning is enabled and disabled on the Symmetrix array.

Prerequisites

Before creating a cache partition, note the following:

- ◆ There must be an available partition.
- ◆ There must be enough cache left in the default partition that it does not fall below the minimum required cache.

Enabling cache partitions

To enable cache partitions for the Symmetrix array, use the following form:

```
symqos -cp -sid SymmID
enable
disable
```

Examples

For example, to enable each partitions on Symmetrix 301, enter:

```
symqos -cp -sid 301 enable
```

To disable each partitions on Symmetrix 301, enter:

```
symqos -cp -sid 301 disable
```

Using analyze mode

Dynamic Cache Partitioning (DCP) also provides an `analyze` mode, which can be used to determine how much actual cache your applications require. While in `analyze` mode, applications will not fail due to cache partition overflow.

Cache partitions cannot be changed from enabled to `analyze`; the cache partition must be disabled to set the `analyze` mode. The syntax for setting the cache partition mode follows:

```
symqos -cp -sid SymmID
analyze
```

To run in `analyze` mode, set the target percent, and then set all partitions with a minimum of 0%, a maximum of 100%, and a donation time of 0 seconds. After

running in analyze mode, you can list the cache partition settings with the `-usage` option, and determine more accurate cache partition target values.

The `-usage` option is shown in the following example:

```
symqos -cp list -sid 237 -settings -usage
```

```
Symmetrix ID: 000190300237
```

```
Cache Partitioning      : Enabled
Number of Partitions    : 2
Max Num of Partitions   : 8
Min Allowed Target %    : 10
Max Allowed Target %    : 90
XRC Partition State     : Enabled
XRC Partition Name      : XRC_partition
Empty Partition Removal : Remove
Time of Last Modification : 01:12:02 PM on Tue Mar 13,2007
```

Partition Name	Min (%)	Tgt (%)	Max (%)	WP (%)	Time (s)	Device Count	Cache Slots		Used (%)
							WP	Used	
DEFAULT_PARTITION	0	65	100	80	300	1024	2156	395166	95
Lab2	35	35	35	50	300	56	234	6898	65

SRDF/A output

To have the SRDF/A output based on cache partition values, instead of system-wide settings, set `SYMAPI_CP_ANALYSIS_MODE` in the options file to `ENABLE` when running in analyze mode.

The cache partition values display automatically when cache partitions are enabled, but only display in analyze mode when this options file setting is enabled. The default value is `DISABLE`.

The SRDF/A information displayed in CLI commands will show the percent of cache in use, calculated by the target cache percentage and the write pending limit for the partition of the associated RDF group.

RDFA Information:

```
{
  Session Number          : 39
  Cycle Number            : 0
  Number of Devices in the Session : 16
  Session Status          : Inactive
. . .
  Tracks not Committed to the R2 Side: 0
  Time that R2 is behind R1      : 00:00:00
  R1 Side Percent Cache In Use   : 0
  R2 Side Percent Cache In Use   : 0
. . .
```

Empty partition removal

By default, any partition that does not have devices assigned to it will be removed after 4 hours with no activity. This feature appears in the partition settings output (see the DCP settings output).

To change this feature, use the following syntax:

```
symqos -cp -sid SymmID set empty preserve
```

Creating cache partitions

Before you can add devices to a cache partition, it must be created. To create a cache partition, use the following form:

```
symqos -cp -name PartitionName -sid SymmID

        create -target TargetPercent
              -min MinimumPercent
              -max MaximumPercent
              -wp WritePendingLimit
              -time DonationTime
```

Where:

-cp — Indicates a cache partition operation.

PartitionName — Identifies the partition by a unique name. The name can be up to 31 characters.

TargetPercent — Specifies the target cache percentage for a cache partition.

MinimumPercent — Specifies the minimum cache percentage for a cache partition.

MaximumPercent — Specifies the maximum cache percentage for a cache partition.

WritePendingLimit — Specifies the write pending limit percentage for a cache partition. Possible values are 40 - 80.

DonationTime — Specifies the donation time in seconds; the time for when idle cache is made available to other cache partitions.

Adding devices to a cache partition

To a cache partition you can add single devices, a range of devices, or devices in a device group or RA group. The `addall` option must be used with device ranges and RA groups. With device groups, you can specify all devices (`addall`), logical devices (`ld`), standard devices (`-std`), BCV devices (`-bcv`), or virtual devices (`-vdev`).

Device restrictions

When adding devices to a cache partition:

- ◆ Only specify the head device for metadevices and CKD devices.
- ◆ Do not add VAULT devices or DATA devices, as these are blocked from being moved to any new partition.
- ◆ Ensure that all devices in RDF groups that are in Async mode belong to a single partition.

To add devices to a partition, use the following form:

```
symqos -cp -name PartitionName -sid SymmID
        add dev SymDevName

symqos -cp -name PartitionName -sid SymmID
        [-range <startDevRange:endDevRange> | -rdfg GrpNum]
        addall

symqos -cp -name PartitionName -g DgName
        add ld LdevName
        addall [-std] [-bcv] [-vdev]
```

If the device type is not specified, the default is to add only STD devices.

Examples To add device 001 to partition `TestP` on Symmetrix 310, enter:

```
symqos -cp -name TestP -sid 310 add dev 001
```

To add device range 002 through 012 to partition `TestP` on Symmetrix 310, enter:

```
symqos -cp -name TestP -sid 310 -range 002:012 addall
```

To add the BCV devices in device group `Finance` to the partition `TestP` on Symmetrix 310, enter:

```
-symqos -cp -name TestP -sid 310 -g Finance addall -bcv
```

Modifying a cache partition

Use the `modify` argument to change the values of a cache partition, as follows:

```
symqos -cp -name PartitionName -ran-sid SymmID

modify [-target TargetPercent]
        [-min MinimumPercent]
        [-max MaximumPercent]
        [-wp WritePendingLimit]
        [-time DonationTime]
```

Renaming Use the `rename -new_name` option to rename the cache partition, as follows:

```
symqos -cp -name PartitionName -sid SymmID
        rename -new_name NewPartitionName
```

For example, to rename cache partition `TestP` on Symmetrix 310 to `TestA`, enter:

```
symqos -cp TestP -sid 310 rename -new_name TestA
```

Deleting The `delete` option will delete a cache partition. When deleting a cache partition, all devices and cache currently assigned to that partition will move back to the default partition.

Removing devices from a cache partition

The `remove dev`, `rmall`, and `remove ld` options are used to remove devices from a cache partition and return them to the default partition. This can be done for a device range, a device group, or an RDF group.

To remove the BCV devices in device group `Finance` from the partition `TestP` on Symmetrix 310, enter:

```
symqos -cp -name TestP -sid 310 -g Finance remall -bcv
```

Listing cache partitions

When listing cache partitions, the information can be for a range of devices, a device group, an RDF group, a Symmetrix array, or a cache partition name. List cache partition information using the following form:

```
symqos -cp [-name <PartitionName>

list [-range <SymDevStart:SymDevEnd> [-all] | -rdg GrpNum]
list -g DgName [-std] [-bcv] [-vdev]
list [-sid SymmID] -settings [-usage] [-v]
```

The `-settings` option lists the general cache partition configuration for the array, as shown in the following example:

```
symqos -cp -sid 237 list -settings
```

```
Symmetrix ID: 000190300237
```

```
Cache Partitioning      :Analysis
Number of Partitions    :2
Max Num of Partitions   :8
Min Allowed Target %    :10
Max Allowed Target %    :90
XRC Partition State     :Enabled
XRC Partition Name      :XRC_partition
Empty Partition Removal :Enabled
Time of Last Modification :2:11:47 PM on Thu May 24, 2007
```

Partition Name	Min (%)	Tgt (%)	Max (%)	WP (%)	Time (s)
-----	---	---	---	---	-----
DEFAULT_PARTITION	0	100	100	80	300

The `-usage` option includes cache usage information, as shown in the following example:

```
symqos -cp list -sid 237 -settings -usage
```

```
Symmetrix ID: 000190300237
```

```
Cache Partitioning      : Enabled
Number of Partitions    : 2
Max Num of Partitions   : 8
Min Allowed Target %    : 10
Max Allowed Target %    : 90
XRC Partition State     : Enabled
XRC Partition Name      : XRC_partition
Empty Partition Removal :Enabled
Time of Last Modification : 01:12:02 PM on Tue Mar 13,2007
```

Partition Name	Min (%)	Tgt (%)	Max (%)	WP (%)	Time (s)	Device Count	Cache Slots		Used (%)
							WP	Used	
-----	---	---	---	---	-----	-----	-----	-----	-----
DEFAULT_PARTITION	0	65	100	80	300	1024	2156	395166	95
Lab2	35	35	35	50	300	56	234	6898	65

For complete syntax for the `symqos` command, refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

QoS examples

Using SYMCLI, you can get a list of QoS parameters and values for specified devices.

For example, to get a list of copy pace levels for devices 010 through 0B2 in Symmetrix 010000002334, enter:

```
symqos list -sid 2334 -range 10:B2
```

Note: If you omit the `-sid` option, the specified range of devices will apply to all Symmetrix units.

In the preceding example, concerning the range option, the *SymdevStart* value will default to zero if you omit the value as follows:

```
symqos list -sid 2334 -range :B2
```

This will increase the range to include all devices between 000 and 0B2. If you omit the *SymdevEnd* value, the range will list all devices found beyond the *SymdevStart* value on the specified Symmetrix unit as follows:

```
symqos list -sid 2334 -range 10:
```

To list the LRU assignments of devices 00C through 013 to the various LRU cache management groups, enter:

```
symqos list -lru -sid 346 -range 00C:013
```

To get the copy pace levels concerning just the BCV devices of device group *ProdDB*, enter:

```
symqos -g ProdDB query -bcv
```

To get the copy pace levels concerning both the BCV and STD devices of device group *ProdDB*, enter:

```
symqos -g ProdDB query
```

To get the copy pace levels concerning just the STD devices of device group *ProdDB*, enter:

```
symqos -g ProdDB query -nobcv
```

To view the LRU assignment of all devices in a device group, enter:

```
symqos -g ProdDB query -lru
```

To assign all devices of device group *ProdDB* to LRU group 2, enter:

```
symqos -g ProdDB set LRU 2
```

The Operational Examples part of this product guide identifies and focuses on some specific array control tasks that represent the most typical practices in the management of your Symmetrix storage environment. These practical examples illustrate various array control processes by showing the SYMCLI command sequences to accomplish these tasks. These specific management tasks are described in the subsequent chapters as follows:

[Appendix A, “Configuration Change Examples,”](#) provides detailed examples on using the Configuration Change component.

[Appendix B, “Virtual LUN Migration Example,”](#) provides a detailed example of a virtual LUN migration.

[Appendix C, “Updating the Host,”](#) explains how to update a host so that the host recognizes the new Symmetrix configuration.

[Appendix D, “SIU: Overview and Management,”](#) describes the Symmetrix Integration Utilities (SIU) disk management operations for your storage system in a Windows Server environment.

Note: Some of the examples in this section were performed with earlier versions of software. Therefore, your output displays may not look exactly like the ones appearing in these examples.

Configuration Change Examples

This chapter provided detailed examples on using the Configuration Change component. Although these examples illustrate basic configuration concepts, only advanced users or system administrators should change a Symmetrix configuration.

◆ Example 1: Creating devices.....	282
◆ Example 2: Mapping devices.....	288
◆ Example 3: Mapping a single device on multiple EA ports.....	293
◆ Example 4: Mapping a range of devices and adding aliases.....	299
◆ Example 5: Unmapping devices.....	304
◆ Example 6: Setting front-end port attributes.....	308
◆ Example 7: Configuring concurrent RDF devices	313
◆ Example 8: Updating a disk label with emulated device geometry	319
◆ Example 9: Configuring virtual provisioning.....	327
◆ Example 10: Automatically creating metadevices.....	342

Example 1: Creating devices

This hardware setup consists of an HP-UX host (api157) and two Solaris hosts (api145 and api146), each connected to a Symmetrix array (sid 120) running Enginuity version 5568. All commands are issued from Solaris host api145. The example creates four new Symmetrix devices for this Symmetrix array. Each new device is a 2-Way mirrored type device with a size of 220 cylinders (103 MB) and an emulation of FBA.

- ◆ The `symconfigure list -freespace` command displays the Symmetrix array's free physical disk space that you can use to create new devices. The Unformatted column shows free disk space available for any emulation mode. The Formatted column shows free space on disks that have been partially used for devices configured with the same emulation mode (in this case, FBA). The free space on formatted disks can be used only to create new devices of the same emulation mode. The output here shows that there is sufficient free disk space to create four new FBA-emulation devices of 220 cylinders. To display free space in megabytes rather than cylinders (the default), include the `-units MB` option on the command line.

```
symconfigure -sid 120 list -freespace
```

```
Symmetrix ID: 000184500120
```

Emulation	Formatted (Cyl)	Unformatted (Cyl)
-----	-----	-----
FBA	14207954	0

- ◆ For more detailed information about a Symmetrix configuration, use `symconfigure list` with the verbose (`-v`) option. When checking the current number of hypers on disks and their capacity to add additional hypers (new devices), it is particularly important to note the Symmetrix setting for *Max Hypers per Disk*. The output below indicates that each disk in this Symmetrix array may have a maximum of 24 hypers. The ellipsis (.....) represents an area where a portion of the output was omitted for brevity

```
symconfigure -sid 120 list -v
```

```

Symmetrix ID                               : 000184500120

Configuration Server Version                : 5568.31.14
Configuration Server Protocol               : 0x20D
Configuration Server Date                   : 12.28.2001
.....

Max Hypers per Disk                         :          24
FBA Multi Access Cache                     : Enabled
Dynamic RDF Configuration                  : Enabled
RAID-S support                             : Enabled

```

- ◆ Two commands can check free disk space on a Symmetrix array. The `syndisk list` command displays available space (in cylinders) on physical disks and whether those disks can hold additional hypervisors. A disk is identified by its DA director, the DA path or interface (Int), and the disk's SCSI target ID (TID). When you create 2-way mirrored devices, the Symmetrix array maps each mirror of the device to a different disk, accessed through a different memory bus and DA director:

syndisk -sid 120 list

Symmetrix ID: 000184500120

Symmetrix ID : 000184500120
Disks Selected : 60

Ident	Symb	Int	TID	Vendor	Type	Hypers	Capacity (Cyl)	
							Total	Free
DA-1A	01A	C	0	SEAGATE	CHET_73	1	149348	143208
DA-1A	01A	C	1	SEAGATE	CHET_73	2	149348	148909
DA-1A	01A	C	2	SEAGATE	CHET_73	2	149348	148909
DA-1A	01A	C	3	SEAGATE	CHET_73	1	149348	149128
DA-1A	01A	C	4	SEAGATE	CHET_73	1	149348	149128
DA-1A	01A	C	5	SEAGATE	CHET_73	2	149348	148909
DA-1A	01A	D	0	SEAGATE	CHET_73	1	149348	149128
.....								
DA-2A	02A	C	0	SEAGATE	CHET_73	1	149348	149128
.....								

- ◆ The `symdev list` command also displays free disk space. Using the `-da all` and `-space` options displays the distribution of free disk space across those formatted disks in the Symmetrix array that are mapped to DA (disk) directors. You can examine the Hypers column to determine if there are at least two disks (each mirror of a 2-way mirrored device must be on a different spindle) that have fewer hypers than the *Max Hypers per Disk* setting for the Symmetrix array. Both displays show enough available space to allocate the new hypers when the example creates new devices. As time elapses between displays (as is the case with these two displays), the later display is likely to show added numbers of hypers on a disk, depending on the frequency of Symmetrix configuration sessions by this host and other hosts:

symdev list -sid 120 -da all -space

Symmetrix ID: 000184500120

DA			Capacity (MegaBytes)			Details	
Ident	Int	TID	Total	Configured	Unconfigured	Hypers	Format

01A	C	0	70007	778	69229	4	FBA
01A	C	1	70007	675	69332	3	FBA
01A	C	2	70007	726	69281	5	FBA
01A	C	3	70007	600	69407	6	FBA
01A	C	4	70007	675	69332	3	FBA
01A	C	5	70007	572	69435	2	FBA
01A	D	0	70007	103	69904	1	FBA
.....							
02A	C	0	70007	618	69389	6	FBA
.....							

- ◆ The `symdev list` command with the `-cyl` option displays information about all devices in the Symmetrix array, including the capacity (size) of each device. Note that the size of existing 2-Way Mir devices (the type and size that the example will create) is 220 cylinders. Checking device size is useful when you need to size existing RDF, BCV, DRV, and meta devices for matching new devices to their size. Note that device 009A is the last device currently configured in this Symmetrix array. The Symmetrix system will name new devices beginning with 009B:

```
symdev list -sid 120 -cyl
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors		Device			
Sym	Physical	SA :P DA :IT	Config	Attribute	Sts	Cap (CYL)	
0000	Not Visible	????: 01A:C0	Unprotected	N/Grp'd	RW	220	
0001	Not Visible	????: 16A:D4	Unprotected	N/Grp'd	RW	220	
000A	Not Visible	????: 02B:C0	BCV	N/Asst'd	RW	220	
000B	Not Visible	????: 15B:C4	Unprotected	N/Grp'd (M)	RW	660	
000C	Not Visible	????: 15B:C0	Unprotected	N/Grp'd (m)	RW	-	
000D	Not Visible	????: 02B:C4	Unprotected	N/Grp'd (m)	RW	-	
000E	Not Visible	????: 16B:C0	Unprotected	N/Grp'd	RW	220	
000F	Not Visible	????: 01B:C4	Unprotected	N/Grp'd	RW	220	
0010	Not Visible	????: 01A:D0	Unprotected	N/Grp'd (M)	RW	1100	
0011	Not Visible	????: 16A:C4	Unprotected	N/Grp'd (m)	RW	-	
0012	Not Visible	????: 02A:D0	Unprotected	N/Grp'd (m)	RW	-	
0013	Not Visible	????: 15A:C4	Unprotected	N/Grp'd (m)	RW	-	
0014	Not Visible	????: 15A:D0	Unprotected	N/Grp'd (m)	RW	-	
0015	Not Visible	????: 02A:C4	2-Way Mir	N/Grp'd	RW	220	
0016	Not Visible	????: 16A:D0	3-Way Mir	N/Grp'd	RW	220	
0017	Not Visible	????: 01A:C4	4-Way Mir	N/Grp'd	RW	220	
0018	Not Visible	????: 01B:D0	Unprotected	N/Grp'd	RW	220	
0019	Not Visible	????: 16B:D3	Unprotected	N/Grp'd	RW	220	
006F	Not Visible	????: 15B:C2	Unprotected	N/Grp'd	RW	220	
0070	Not Visible	????: 16B:C0	2-Way Mir	N/Grp'd	RW	220	
0071	Not Visible	????: 15A:C0	2-Way BCV Mir	N/Asst'd	RW	220	
0072	Not Visible	????: 02B:C4	3-Way Mir	N/Grp'd	RW	220	
0073	Not Visible	????: 16A:C0	3-Way Mir	N/Grp'd	RW	220	
0074	Not Visible	????: 16B:C3	3-Way Mir	N/Grp'd	RW	220	
0075	Not Visible	04B:0 01A:C0	Unprotected	Grp'd	RW	220	
0076	Not Visible	????: 01A:C2	Unprotected	Grp'd	RW	220	
0077	Not Visible	????: 01A:C3	3-Way Mir	N/Grp'd	RW	220	
0078	Not Visible	????: 01A:C1	RDF1	N/Grp'd	RW	1000	
0079	Not Visible	????: 01A:C4	RDF2	N/Grp'd	WD	1000	
007A	Not Visible	????: 01A:C2	RDF2	N/Grp'd	WD	1000	
007B	Not Visible	????: 01A:C0	RDF1	N/Grp'd	RW	1000	
0098	Not Visible	????: 02A:C3	Unprotected	N/Grp'd	RW	220	
0099	Not Visible	????: 15B:C1	Unprotected	N/Grp'd	RW	220	
009A	Not Visible	????: 01A:C3	Unprotected	N/Grp'd	RW	220	

- ◆ To check if this Symmetrix array is operating in a mixed environment, use the `symcfg list` command with the `-dir all` option to display information about all directors in the Symmetrix array. The following output shows that there are no ESCON directors present, so this is not a mixed environment. A sub-system identifier (SSID) will not be required when creating new devices.:

```
symcfg list -sid 120 -dir all
```

```
Symmetrix ID: 000184500120
```

```

          S Y M M E T R I X      D I R E C T O R S

      Ident  Symbolic  Numeric  Slot  Type           Status
      -----
      DA-1A   01A      1        1    DISK           Online
      DA-2A   02A      2        2    DISK           Online
      RF-3A   03A      3        3    RDF-BI-DIR     Online
      FA-4A   04A      4        4    FibreChannel   Online
      SA-13A  13A      13       13    SCSI           Online
      SA-14A  14A      14       14    SCSI           Online
      DA-15A  15A      15       15    DISK           Online
      DA-16A  16A      16       16    DISK           Online
      DA-1B   01B      17        1    DISK           Online
      DA-2B   02B      18        2    DISK           Online
      RF-3B   03B      19        3    RDF-BI-DIR     Online
      FA-4B   04B      20        4    FibreChannel   Online
      .....

```

- ◆ The `symconfigure verify` command verifies that the current Symmetrix configuration is available for host-initiated configuration changes (that is, a configuration change session can be opened).:

```
symconfigure -sid 120 verify
```

```

A Configuration Change Verification is in progress. Please wait...
  Establishing a configuration change
session.....Established.
  Verifying
configuration.....Verified.
  Terminating the configuration change session.....Done.

The configuration verification session has succeeded.

```

- ◆ The following command uses the `vi` text editor to create a text file named `create_dev.cmd`. As was done here, you can enter into the file the `create dev` entry that defines four new devices as 2-way mirrored type devices, each mirror with a size of 220 cylinders:

```
vi create_dev.cmd
```

```
create dev, count=4, size=220, emulation=fba, config=2-way-mir;
```

- ◆ The `symconfigure` command with the `commit` argument executes the command file and begins the process of creating the devices. Including the verbose (`-v`) option displays the details of the change session as it executes:

```
symconfigure -sid 120 -file create_dev.cmd -v -noprompt commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000184500120
{
    create dev count=4, size=220, emulation=FBA,
    config=2-Way Mir;
}

Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 003 of 011 steps.....Executing.
Step 005 of 011 steps.....Executing.
Step 007 of 011 steps.....Executing.
Step 008 of 011 steps.....Executing.
Step 008 of 011 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 003 of 103 steps.....Executing.
.....
Step 101 of 103 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

- ◆ To monitor the configuration change session while the `symconfigure commit` operation is processing, issue the `symconfigure query` command, as shown in [“Example 5: Unmapping devices” on page 304](#) and [“Example 6: Setting front-end port attributes” on page 308](#).

- ◆ When you add new devices, the SYMAPI host database file on the host issuing the configuration change is updated automatically on successful completion of the `symconfig commit` command. (Automatic update of the local SYMAPI database occurs for all configuration changes except mapping changes.) To refresh the SYMAPI host database files on all *other* hosts attached to that Symmetrix array, you can perform the `symcfg sync` command on those hosts.

To confirm that the new devices have been added correctly, issue another `symdev list` command. The following display shows that four new 2-way mirrored devices (009B, 009C, 009D, and 009E) were added to the Symmetrix array. Recall that the last device name displayed before adding these devices was 009A. Therefore, the Symmetrix system began naming the new devices starting with 009B:

```
symdev list -sid 120
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors		Device		
Sym	Physical	SA :P DA :IT	Config	Attribute	Sts	Cap (MB)
0000	Not Visible	??? :? 01A:C0	Unprotected	N/Grp'd	RW	103
0001	Not Visible	??? :? 16A:D4	Unprotected	N/Grp'd	RW	103
0097	Not Visible	??? :? 16A:C3	Unprotected	N/Grp'd	RW	103
0098	Not Visible	??? :? 02A:C3	Unprotected	N/Grp'd	RW	103
0099	Not Visible	??? :? 15B:C1	Unprotected	N/Grp'd	RW	103
009A	Not Visible	??? :? 01A:C3	Unprotected	N/Grp'd	RW	103
009B	Not Visible	??? :? 01A:C5	2-Way Mir	N/Grp'd	RW	103
009C	Not Visible	??? :? 02B:C1	2-Way Mir	N/Grp'd	RW	103
009D	Not Visible	??? :? 16A:C4	2-Way Mir	N/Grp'd	RW	103
009E	Not Visible	??? :? 15B:C3	2-Way Mir	N/Grp'd	RW	103

Example 2: Mapping devices

The hardware setup remains the same as Example 1: an HP-UX host (api157) and two Solaris hosts (api145 and api146), each connected to a Symmetrix array (sid 120). All commands are issued from Solaris host api145. The example maps the four new Symmetrix devices to front-end director FA-4B so that these devices can be made visible to host api145.

- Using the `-connections` option with `symcfg list` allows you to see the host connections to a Symmetrix array connected to a host running SYMAPI software. The following display shows the front-end directors that each host uses to reach the Symmetrix (sid 120). Although a host can use more than one front-end director to connect to the same Symmetrix array, host api145 here uses only director FA-4B. The “FA” prefix indicates a fibre front-end adapter, while an “SA” prefix indicates a SCSI front-end adapter:

```
symcfg list -connections -sid 120
```

```
Symmetrix ID : 000185500120
```

Symmetrix		Host				
Director	Port	Node Name	IP Address	HW Type	OS Name	OS Revision
FA-5B	0	api146	172.23.65.146	sun4u	SunOS	5.7
FA-4B	0	api145	172.23.65.145	sun4u	SunOS	5.6
SA-14A	1	api157	172.23.65.157	9000/897	HPUX	B.11.00

- The `symdev list` command with the `-noport` option displays devices that are not yet mapped to any front-end (SA) adapter ports, including the newly-created devices (009B through 009E).

```
symdev list -sid 120 -noport
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors		Device			
Sym	Physical	SA :P	DA :IT	Config	Attribute	Sts	Cap (MB)
0000	Not Visible	???:?	01A:C0	Unprotected	N/Grp'd	RW	103
000A	Not Visible	???:?	02B:C0	BCV	N/Asst'd	RW	103
000B	Not Visible	???:?	15B:C4	Unprotected	N/Grp'd (M)	RW	309
000E	Not Visible	???:?	16B:C0	Unprotected	N/Grp'd	RW	103
000F	Not Visible	???:?	01B:C4	Unprotected	N/Grp'd	RW	103
0010	Not Visible	???:?	01A:D0	Unprotected	N/Grp'd (M)	RW	516
0015	Not Visible	???:?	02A:C4	2-Way Mir	N/Grp'd	RW	103
0016	Not Visible	???:?	16A:D0	3-Way Mir	N/Grp'd	RW	103
0017	Not Visible	???:?	01A:C4	4-Way Mir	N/Grp'd	RW	103
0099	Not Visible	???:?	15B:C1	Unprotected	N/Grp'd	RW	103
009A	Not Visible	???:?	01A:C3	Unprotected	N/Grp'd	RW	103
009B	Not Visible	???:?	01A:C5	2-Way Mir	N/Grp'd	RW	103
009C	Not Visible	???:?	02B:C1	2-Way Mir	N/Grp'd	RW	103
009D	Not Visible	???:?	16A:C4	2-Way Mir	N/Grp'd	RW	103
009E	Not Visible	???:?	15B:C3	2-Way Mir	N/Grp'd	RW	103

- ◆ The `sympd list` command displays devices that are visible to this host, meaning that these devices are currently the only ones on the Symmetrix array that have been mapped to a front-end director (04B) and recognized by performing a host update:

```
sympd list -sid 120
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors				Device			
Physical	Sym	SA	:P	DA	:IT	Config	Attribute	Sts	Cap (MB)
/dev/rdsk/clt0d1s2	0029	04B:0	16B:C3	RDF2			N/Grp'd	WD	103
/dev/rdsk/clt0d2s2	0033	04B:0	15A:C3	RDF1			N/Grp'd	RW	103
/dev/rdsk/clt0d3s2	003D	04B:0	02B:D2	Unprotected			N/Grp'd	RW	3

- ◆ This `symcfg list -addresses -available` command displays the VBUS, TID, and LUN addresses associated with front-end director 04B, port 0, and indicates the next available LUN address (generated by adding 1 to the last LUN address used). To decide on a LUN value, consider the LUN conventions for your host platform. The example uses LUN 00B as the address for the first new device:

```
symcfg list -sid 120 -sa 04B -p 0 -addresses -available
```

```
Symmetrix ID: 000184500120
```

Director			Device Name			Attr	Address		
Ident	Symbolic	Port	Sym	Physical			VBUS	TID	LUN
FA-4B	04B	0	-	AVAILABLE			0	0	000 *
			0029	/dev/rdisk/clt0d1s2			0	0	001
			0033	/dev/rdisk/clt0d2s2			0	0	002
			003D	/dev/rdisk/clt0d3s2			0	0	003
			0046	Not Visible			0	0	004
			-	AVAILABLE			0	0	005 *
			0075	Not Visible			0	0	00A
			-	AVAILABLE			0	0	00B *

Legend for Available address:

(*): The VBUS, TID, LUN address values represent a gap in the address assignments or are the next available address in the run

- ◆ The `symconfigure verify` command verifies that the current Symmetrix configuration is available for host-initiated configuration changes (that is, a configuration change session can be opened):

```
# symconfigure -sid 120 verify
```

```
A Configuration Change Verification is in progress. Please wait...
Establishing a configuration change session.....Established.
Verifying configuration.....Verified.
Terminating the configuration change session.....Done.
```

```
The configuration verification session has succeeded.
```

- ◆ The following command uses the vi text editor to create a text file named `map_dev.cmd`. As was done here, you can enter into the file the `map dev` entries that define the director, port, and LUN addresses for the four new devices. If you are mapping to a fibre adapter (FA) port system and are *not* using volume set addressing (as is the case for this example), only the LUN address is required (not the VBUS or TID):

```
vi map_dev.cmd
```

```
map dev 09B to dir 04B:0, lun=00B;
map dev 09C to dir 04B:0, lun=00C;
map dev 09D to dir 04B:0, lun=00D;
map dev 09E to dir 04B:0, lun=00E;
```

- ◆ The `symconfigure` command with the `commit` argument executes the command file and begins the process of mapping the devices:

```
symconfigure -sid 120 -file map_dev.cmd -v -noprompt commit
```

```
Establishing a monitoring session.....Established.
The session changes are in the class of: Mapping/unmapping devices
{
  map dev 009B to dir 4B:0, target=00, lun=0B;
  map dev 009C to dir 4B:0, target=00, lun=0C;
  map dev 009D to dir 4B:0, target=00, lun=0D;
  map dev 009E to dir 4B:0, target=00, lun=0E;
}

The last action requested was: PREPARE
The state of the last action is: Running
PREPARE.....Done.
The last action requested was: COMMIT
The state of the last action is: Running
Step 05 of 44 steps.....Executing.
.....
Step 43 of 44 steps.....Executing.
Monitored session has terminated
Terminating the monitoring session.....Done.
```

- ◆ To monitor the configuration change session while the `symconfigure commit` operation is processing, issue the `symconfigure query` command (as shown in examples 3 and 4).

- ◆ The `symdev list` command shows that devices 009B through 009E are now mapped to a Symmetrix front-end port (04B:0) but that the new devices are not yet visible to the host:

```
symdev list -sid 120
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors		Device			
Sym	Physical	SA	:P DA :IT	Config	Attribute	Sts	Cap (MB)
0000	Not Visible	???:?	01A:C0	Unprotected	N/Grp'd	RW	103
0001	Not Visible	???:?	16A:D4	Unprotected	N/Grp'd	RW	103
0002	Not Visible	13A:0	02A:C0	Unprotected	N/Grp'd	RW	103
0003	Not Visible	13A:1	15A:D4	Unprotected	N/Grp'd	RW	103
0004	Not Visible	13B:0	15A:C0	Unprotected	N/Grp'd	RW	103
0005	Not Visible	13B:1	02A:D4	Unprotected	N/Grp'd	RW	103
0006	Not Visible	14A:0	16A:C0	Unprotected	N/Grp'd	RW	103
0007	Not Visible	14A:1	01A:D4	Unprotected	N/Grp'd	RW	103
0008	Not Visible	14B:0	01B:C0	Unprotected	N/Grp'd	RW	103
0009	Not Visible	14B:1	16B:C4	Unprotected	N/Grp'd	RW	103
000A	Not Visible	???:?	02B:C0	BCV	N/Asst'd	RW	103
000B	Not Visible	???:?	15B:C4	Unprotected	N/Grp'd (M)	RW	309
000C	Not Visible	???:?	15B:C0	Unprotected	N/Grp'd (m)	RW	-
000D	Not Visible	???:?	02B:C4	Unprotected	N/Grp'd (m)	RW	-
000E	Not Visible	???:?	16B:C0	Unprotected	N/Grp'd	RW	103
000F	Not Visible	???:?	01B:C4	Unprotected	N/Grp'd	RW	103
0010	Not Visible	???:?	01A:D0	Unprotected	N/Grp'd (M)	RW	516
0011	Not Visible	???:?	16A:C4	Unprotected	N/Grp'd (m)	RW	-
0012	Not Visible	???:?	02A:D0	Unprotected	N/Grp'd (m)	RW	-
0013	Not Visible	???:?	15A:C4	Unprotected	N/Grp'd (m)	RW	-
0099	Not Visible	???:?	15B:C1	Unprotected	N/Grp'd	RW	103
009A	Not Visible	???:?	01A:C3	Unprotected	N/Grp'd	RW	103
009B	Not Visible	04B:0	01A:C5	2-Way Mir	N/Grp'd	RW	103
009C	Not Visible	04B:0	02B:C1	2-Way Mir	N/Grp'd	RW	103
009D	Not Visible	04B:0	16A:C4	2-Way Mir	N/Grp'd	RW	103
009E	Not Visible	04B:0	15B:C3	2-Way Mir	N/Grp'd	RW	103

- ◆ The `sympd list` command confirms that the new devices are not visible to the host. This command displays only those devices that have a physical device name, which means that host can “see” them:

```
sympd list -sid 120
```

```
Symmetrix ID: 000184500120
```

Device Name		Directors		Device			
Physical	Sym	SA	:P DA :IT	Config	Attribute	Sts	Cap (MB)
/dev/rdisk/c1t0d1s2	0029	04B:0	16B:C3	RDF2	N/Grp'd	WD	103
/dev/rdisk/c1t0d2s2	0033	04B:0	15A:C3	RDF1	N/Grp'd	RW	103
/dev/rdisk/c1t0d3s2	003D	04B:0	02B:D2	Unprotected	N/Grp'd	RW	3

- ◆ Executing the following utilities introduces the new devices to the host in a Sun Solaris environment:

```
drvconfig  
disks  
devlinks
```

- ◆ After performing the proper host procedures to update the host view, you need to complete host addressing by making sure that the host address is recognized in the SYMAPI view. To update the SYMAPI database on your host, issue the `symcfg discover` command:

symcfg discover

This operation may take up to a few minutes. Please be patient...

- ◆ The `sympd list` command shows that the new devices are now visible to the host:

sympd list -sid 120

Symmetrix ID: 000184500120

Device Name	Directors				Device		
Physical	Sym	SA	:P DA	:IT Config	Attribute	Sts	Cap (MB)
/dev/rdisk/clt0d1s2	0029	04B:0	16B:C3	RDF2	N/Grp'd	WD	103
/dev/rdisk/clt0d2s2	0033	04B:0	15A:C3	RDF1	N/Grp'd	RW	103
/dev/rdisk/clt0d3s2	003D	04B:0	02B:D2	Unprotected	N/Grp'd	RW	3
/dev/rdisk/clt0d32s2	009B	04B:0	01A:C0	2-Way Mir	N/Grp'd	RW	103
/dev/rdisk/clt0d33s2	009C	04B:0	16A:D4	2-Way Mir	N/Grp'd	RW	103
/dev/rdisk/clt0d34s2	009D	04B:0	01A:D2	2-Way Mir	N/Grp'd	RW	103
/dev/rdisk/clt0d35s2	009E	04B:0	01A:D2	2-Way Mir	N/Grp'd	RW	103

Example 3: Mapping a single device on multiple EA ports

The following example was performed using Solutions Enabler version 6.0. The hardware setup is a Symmetrix array set to dynamic PAV operation and connected to a mainframe z/OS host. Although you would normally map multiple devices in a CU image (as in [“Example 4: Mapping a range of devices and adding aliases” on page 299](#)), this example describes the simple case of mapping a single device and copying this mapping to all director/ports on an EA board.

The `symcfg list` command with the `-CUimages` option provides information about the CU images configured on the Symmetrix array whose ID is 000000006219. CU images 1 and 2 are currently configured. The "CU status" pertains to the status of the devices in the image: online, offline, or mixed:

```
symcfg -sid 6219 list -CUimages
```

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status             : Offline

Sub System ID          : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 16
Device Ranges          : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48

CU image number: 0x02
CU status             : Mixed

Sub System ID          : 0x0186
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 20
Device Ranges          : 0271:0280
                        : 033B:033E
Number of Base Addresses : 20
Number of Alias Addresses : 0
```

The following `vi` command uses the `vi` text editor to create a text file named `example1MapDev.cfg`. As was done here, you can enter into the file the map dev entries that define the director, port, and base addresses for one or more devices. The first entry maps one device (343) to director 12a, port 0, assigning a base address of 300. The first base address that you assign to a CU image's device set must be a multiple of 0x010. The first digit of the base address represents the CU image number (in this case, CU image 3). The SSID assigned to this device is 192 (only one SSID per command file is allowed). Subsequent entries map device 343 to other directors and

ports on an EA board, copying the mapping from director 12a, port 0. EA boards are used to talk to the mainframe, and the mainframe needs to see the device on all ports on the EA board. The EA board has eight ports, and they are located on slot 12:

vi example1MapDev.cfg

```
map dev 343 to dir 12a:0, starting BASE_ADDRESS=300 MVS_SSID=192;
map dev 343 to dir 12a:1, copying dir 12A:0;
map dev 343 to dir 12b:0, copying dir 12A:0;
map dev 343 to dir 12b:1, copying dir 12A:0;
map dev 343 to dir 12c:0, copying dir 12A:0;
map dev 343 to dir 12c:1, copying dir 12A:0;
map dev 343 to dir 12d:0, copying dir 12A:0;
map dev 343 to dir 12d:1, copying dir 12A:0;
```

The `symconfigure` command with the `commit` argument executes the command file and maps device 343 to the eight directors and ports. The ellipsis (...) indicates output omitted for brevity.

symconfigure -sid 6219 -file example1MapDev.cfg -noprompt -v commit

```
Establishing a configuration change session.....Established.
Processing symmetrix 000000006219
{
  map dev 0343:0343 to dir 12A:0, starting base_address=300, ssid=0192;
  map dev 0343:0343 to dir 12A:1, copying dir 12A:0;
  map dev 0343:0343 to dir 12B:0, copying dir 12A:0;
  map dev 0343:0343 to dir 12B:1, copying dir 12A:0;
  map dev 0343:0343 to dir 12C:0, copying dir 12A:0;
  map dev 0343:0343 to dir 12C:1, copying dir 12A:0;
  map dev 0343:0343 to dir 12D:0, copying dir 12A:0;
  map dev 0343:0343 to dir 12D:1, copying dir 12A:0;
}

Performing Access checks.....Allowed.
Locking devices.....Locked.
Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 011 of 017 steps.....Executing.
Step 013 of 017 steps.....Executing.
Step 015 of 017 steps.....Executing.
Step 016 of 017 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
.....
Step 085 of 108 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

The `symcfg list` command displays the addition of CU image 3 that was just created. Device 343 is mapped to eight ports, and there are no alias assignments yet. The ellipsis (...) indicates omitted output.

`symcfg -sid 6219 list -CUimages`

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status          : Offline

Sub System ID          : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices       : 16
Device Ranges           : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48
```

```
.....

CU image number: 0x03
CU status          : Offline

Sub System ID          : 0x0192
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices       : 1
Device Ranges           : 0343
Number of Base Addresses : 1
Number of Alias Addresses : 0
```

The following `vi` command uses the `vi` text editor to create a text file named `example1AliasDev.cfg`. As was done here, you can enter into the file entries that assign seven alias addresses to device 343:

`vi example1AliasDev.cfg`

```
add pav alias to dev 343 starting alias=303;
add pav alias to dev 343 starting alias=305;
add pav alias to dev 343 starting alias=307;
add pav alias to dev 343 starting alias=309;
add pav alias to dev 343 starting alias=311;
add pav alias to dev 343 starting alias=312;
add pav alias to dev 343 starting alias=313;
```

The `symconfigure` command with the `commit` argument executes the command file and begins the process of adding the alias addresses:

```
symconfigure -sid 6219 -file example1AliasDev.cfg -noprompt -v  
commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000000006219
{
  add pav alias to dev 0343, starting alias=303;
  add pav alias to dev 0343, starting alias=305;
  add pav alias to dev 0343, starting alias=307;
  add pav alias to dev 0343, starting alias=309;
  add pav alias to dev 0343, starting alias=311;
  add pav alias to dev 0343, starting alias=312;
  add pav alias to dev 0343, starting alias=313;
}

Performing Access checks.....Allowed.
Locking devices.....Locked.
Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 011 of 017 steps.....Executing.
Step 013 of 017 steps.....Executing.
Step 015 of 017 steps.....Executing.
Step 016 of 017 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 012 of 064 steps.....Executing.
Step 014 of 064 steps.....Executing.
Step 015 of 064 steps.....Executing.
Step 023 of 064 steps.....Executing.
Step 028 of 064 steps.....Executing.
Step 030 of 064 steps.....Executing.
Step 035 of 064 steps.....Executing.
Step 037 of 104 steps.....Executing.
Step 045 of 104 steps.....Executing.
Step 050 of 104 steps.....Executing.
Step 054 of 104 steps.....Executing.
Step 056 of 104 steps.....Executing.
Step 057 of 104 steps.....Executing.
Step 058 of 104 steps.....Executing.
Step 058 of 104 steps.....Executing.
.....
Step 062 of 104 steps.....Executing.
Step 067 of 104 steps.....Executing.
Step 098 of 108 steps.....Executing.
Step 104 of 108 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```


Another `symcfg list` command shows the addition of 7 aliases for device 343 in CU image 3:

`symcfg -sid 769 list -CUimages`

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status         : Offline

Sub System ID         : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 16
Device Ranges          : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48

CU image number: 0x02
CU status         : Offline

Sub System ID         : 0x0186
Director Port Assignments : EA-12A:0

.....

Number of Devices      : 20
Device Ranges          : 0271:0280
                        : 033B:033E
Number of Base Addresses : 20
Number of Alias Addresses : 0

CU image number: 0x03
CU status         : Offline

Sub System ID         : 0x0192
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 1
Device Ranges          : 0343
Number of Base Addresses : 1
Number of Alias Addresses : 7
```

The `symcfg show` command displays the alias assignment details of CU image 3:

`symcfg -sid 6219 -CUimages show 3`

```
Symmetrix ID      : 000000006219
PAV Aliasing      : DynamicStandardPAV
```

CU image number: 0x03

```
Sub System ID      : 0x0192
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
```

```
Number of Devices   : 1
Number of Base Addresses : 1
Number of Aliases Addresses : 7
```

SymDev	Base Address	Aliases
0343	300	303 305 307 309 311 312 313

Example 4: Mapping a range of devices and adding aliases

This example continues Example 3 by mapping three CKD devices as CU image 4, copying the device mappings to other ports, and adding aliases for each device.

Note: This feature is only supported with Engenuity version 5671.

The `symcfg list` command with the `-CUimages` option displays the three CU images that are currently configured for Symmetrix 6219:

symcfg -sid 6219 list -CUimages

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status          : Offline

Sub System ID         : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 16
Device Ranges          : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48

CU image number: 0x02
CU status          : Offline

Sub System ID         : 0x0186
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 20
Device Ranges          : 0271:0280
                        : 033B:033E
Number of Base Addresses : 20
Number of Alias Addresses : 0

CU image number: 0x03
CU status          : Offline

Sub System ID         : 0x0192
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 1
Device Ranges          : 0343
```

```

Number of Base Addresses      : 1
Number of Alias Addresses    : 7

```

The following `vi` command uses the `vi` text editor to create a text file named `example2MapDevs.cfg`. As was done here, you can enter into the file the `map dev` entry that maps a range of three devices (344, 345, and 346) to director 12a, port 0, assigning base addresses beginning with 400. The first digit of the starting base address represents the CU image number (in this case, CU image 4), and the next two digits (00) specify its position within the image's device list. Subsequent entries map the devices to the other directors and ports on an EA board, copying the mapping from director 12a, port 0. The SSID assigned to these devices is 193:

`vi example2MapDevs.cfg`

```

map dev 344:346 to dir 12a:0, starting BASE_ADDRESS=400 MVS_SSID=193;
map dev 344:346 to dir 12a:1, copying dir 12A:0;
map dev 344:346 to dir 12b:0, copying dir 12A:0;
map dev 344:346 to dir 12b:1, copying dir 12A:0;
map dev 344:346 to dir 12c:0, copying dir 12A:0;
map dev 344:346 to dir 12c:1, copying dir 12A:0;
map dev 344:346 to dir 12d:0, copying dir 12A:0;
map dev 344:346 to dir 12d:1, copying dir 12A:0;

```

The `symconfigure` command with the `commit` argument executes the command file and begins the process of mapping the devices. The ellipsis (...) indicates output omitted for brevity:

`symconfigure -sid 6219 -file example2MapDevs.cfg -noprompt -v commit`

```

Establishing a configuration change session.....Established.
Processing symmetrix 000000006219
{
  map dev 0344:0346 to dir 12A:0, starting base_address=400, ssid=0193;
  map dev 0344:0346 to dir 12A:1, copying dir 12A:0;
  map dev 0344:0346 to dir 12B:0, copying dir 12A:0;
  map dev 0344:0346 to dir 12B:1, copying dir 12A:0;
  map dev 0344:0346 to dir 12C:0, copying dir 12A:0;
  map dev 0344:0346 to dir 12C:1, copying dir 12A:0;
  map dev 0344:0346 to dir 12D:0, copying dir 12A:0;
  map dev 0344:0346 to dir 12D:1, copying dir 12A:0;
}

Performing Access checks.....Allowed.
Locking devices.....Locked.
Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 011 of 017 steps.....Executing.
Step 013 of 017 steps.....Executing.
Step 015 of 017 steps.....Executing.
.....
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.

```

The `symcfg list` command displays the addition of CU image 4 for the Symmetrix array:

symcfg -sid 6219 list -CUimages

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status           : Offline

Sub System ID         : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 16
Device Ranges          : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48
```

```
.....

CU image number: 0x04
CU status           : Offline

Sub System ID         : 0x0193
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 3
Device Ranges          : 0344:0346
Number of Base Addresses : 3
Number of Alias Addresses : 0
```

The `vi` command uses the `vi` text editor to create a command file named `example2AliasDevs.cfg` to add three alias addresses for each of the three devices in CU image 4, beginning at alias addresses 410, 420, and 430:

vi example2AliasDevs.cfg

```
add pav alias to dev 344:346 starting alias=410;
add pav alias to dev 344:346 starting alias=420;
add pav alias to dev 344:346 starting alias=430;
```

The `symconfigure` command with the `commit` argument executes the command file and begins the process of adding the alias addresses:

```
symconfigure -sid 6219 -file example2AliasDevs.cfg -noprompt -v commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000000006219
{
  add pav alias to dev 0344:0346, starting alias=410;
  add pav alias to dev 0344:0346, starting alias=420;
  add pav alias to dev 0344:0346, starting alias=430;
}

Performing Access checks.....Allowed.
Locking devices.....Locked.
Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 013 of 017 steps.....Executing.
Step 014 of 017 steps.....Executing.
Step 016 of 017 steps.....Executing.
Step 017 of 017 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 002 of 064 steps.....Executing.
Step 011 of 064 steps.....Executing.
Step 014 of 064 steps.....Executing.
Step 015 of 064 steps.....Executing.
Step 022 of 064 steps.....Executing.
Step 028 of 064 steps.....Executing.
Step 030 of 064 steps.....Executing.
Step 035 of 064 steps.....Executing.
Step 037 of 104 steps.....Executing.
Step 045 of 104 steps.....Executing.
Step 050 of 104 steps.....Executing.
Step 054 of 104 steps.....Executing.
.....
Step 108 of 108 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

Another `symcfg list` command displays the CU images again, including the addition of aliases for CU image 4.

`symcfg -sid 6219 list -CUimages`

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x01
CU status            : Offline

Sub System ID         : 0x0185
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 16
Device Ranges         : 0231:0240
Number of Base Addresses : 16
Number of Alias Addresses : 48
.....

CU image number: 0x04
CU status            : Offline
Sub System ID         : 0x0193
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 3
Device Ranges         : 0344:0346
Number of Base Addresses : 3
Number of Alias Addresses : 9
```

The `symcfg show` command displays the alias assignment details for CU image 4.

`symcfg -sid 6219 -CUimages show 4`

```
Symmetrix ID           : 000000006219
PAV Aliasing           : DynamicStandardPAV

CU image number: 0x04

Sub System ID         : 0x0193
Director Port Assignments : EA-12A:0
                        : EA-12A:1
                        : EA-12B:0
                        : EA-12B:1
                        : EA-12C:0
                        : EA-12C:1
                        : EA-12D:0
                        : EA-12D:1
Number of Devices      : 3
Number of Base Addresses : 3
Number of Aliases Addresses : 9

SymDev      Base Address      Aliases
0344-0346    400-402          410-412 420-422 430-432
```

Example 5: Unmapping devices

This example unmaps a device (0001) that was previously mapped to multiple paths.

- ◆ The `symdev show` command output provides a section labeled *Front Director Paths* that shows a device's front-end mappings even if there is only one path. The following command shows the mapping of device 0001 before unmapping it. Note that this device is mapped to multiple paths, but only the path that maps to front-end director 04B is visible to this host. If you know that device 0001 has multiple paths, you can also display these paths with `symdev list -sid 120 -range 0001:0001 -multiport`, which provides a very brief display of summary information:

symdev show 0001 -sid 120

```

Device Physical Name      : /dev/rdisk/clt0d36s2

Device Symmetrix Name     : 0001
Device Serial ID         : 20001200
Symmetrix ID             : 000184500120

Attached BCV Device      : N/A
Attached Snap TGT Device : N/A

Vendor ID                : EMC
Product ID               : SYMMETRIX
Product Revision         : 5568

Device Emulation Type    : FBA
Device Defined Label Type: N/A
Device Defined Label     : N/A
Device Sub System Id     : N/A

Device Block Size        : 512

Device Capacity
{
  Cylinders      :      220
  Tracks         :      3300
  512-byte Blocks :    211200
  MegaBytes      :       103
  KiloBytes      :    105600
}

Device Configuration     : Unprotected
Dynamic Spare Invoked    : No
Dynamic RDF Capability    : Not_RDF_Capable
Device Service State     : Normal

Device Status            : Ready              (RW)
Device SA Status         : Ready              (RW)

Front Director Paths (2):
{
-----
                                POWERPATH  DIRECTOR      PORT
                                -----
PdevName                      Type      Num    Type  Num  Sts  VBUS TID LUN
-----
/dev/rdisk/clt0d36s2          N/A      04B   FA    0    RW   000 00 021
Not Visible                   N/A      04A   FA    0    RW   000 00 000
}
-----

```


- ◆ When unmapping a device, there can be no I/O activity in the specified mapped path. If you want to unmap only one path to a device that has paths to multiple front-end directors, you can write-disable just that path. The `symdev` command write-disables the device-0001 path specified by director 04B, port 0:

```
symdev -sid 120 write_disable 0001 -sa 04B -p 0
```

Write Disable operation successfully completed for the device.

- ◆ The `symconfigure verify` command verifies that the current Symmetrix configuration is available for host-initiated configuration changes (that is, a configuration change session can be opened):

```
symconfigure -sid 120 verify
```

```
A Configuration Change Verification is in progress. Please wait...
  Establishing a configuration change session.....Established.
  Verifying configuration.....Verified.
  Terminating the configuration change session.....Done.
```

The configuration verification session has succeeded.

- ◆ The following command uses the `vi` text editor to create a text file named `unmap.cmd`. As was done here, you can enter into the file the `unmap dev` entry that defines the path (director: port) that you want to unmap:

```
vi unmap.cmd
```

```
unmap dev 0001 from dir 04B:0;
```

- ◆ The `symconfigure` command with the `commit` argument executes the command file and begins the process of unmapping the device:

```
symconfigure -sid 120 -file unmap.cmd -v -noprompt commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000184500120
{
  unmap dev 0001 from dir 4B:0;
}

Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 004 of 011 steps.....Executing.
.....
Step 008 of 011 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 003 of 044 steps.....Executing.
.....
Step 043 of 044 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

- ◆ To monitor the configuration change session while the `symconfigure commit` operation is processing, you can issue the `symconfigure query` command or the following UNIX `tail` command from a second window. The following two commands allow you to compare outputs from the `tail` command and from `symconfigure query`:

```
tail -f /var/symapi/log/symapi-20011228.log
```

```
12/28/2001 15:33:47.328 22089 0 EMC:SYMCLI iCfgChgSessionStart Called to start a
local session for SID 000184500120
12/28/2001 15:33:53.452 22089 Establishing session with Local cfg srvr (000184500120)
...Established.
12/28/2001 15:34:00.779 22089 {
12/28/2001 15:34:00.786 22089 unmap dev 0001 from dir 4B:0, lun=021; (generated)
12/28/2001 15:34:00.786 22089 unmap dev 0001 from dir 4B:0;
12/28/2001 15:34:00.794 22089 }
12/28/2001 15:34:05.799 22089 Submitting configuration changes.....Submitted.
12/28/2001 15:34:16.809 22089 Validating configuration changes.....Validated.
12/28/2001 15:34:27.829 22089 Initiating PREPARE of configuration changes..Queued.
12/28/2001 15:34:39.830 22089 PREPARE requesting required resources.....Obtained.
12/28/2001 15:34:40.840 22089 Step 004 of 011 steps.....Executing.
12/28/2001 15:34:47.849 22089 Step 007 of 011 steps.....Executing.
12/28/2001 15:34:53.860 22089 Step 008 of 011 steps.....Executing.
12/28/2001 15:35:16.880 22089 Local: PREPARE.....Done.
12/28/2001 15:35:27.990 22089 Initiating COMMIT of configuration changes...Queued.
12/28/2001 15:35:39.990 22089 COMMIT requesting required resources.....Obtained.
12/28/2001 15:35:40.000 22089 Step 004 of 044 steps.....Executing.
.....
12/28/2001 15:40:56.905 22089 Step 039 of 044 steps.....Executing.
12/28/2001 15:41:18.925 22089 Local: COMMIT.....Done.
12/28/2001 15:41:18.940 22089 0 EMC:SYMCLI process_load_request Switching to FULL
load for 000184500120 because the configuration changed
12/28/2001 15:41:25.573 22089 Terminating session with configuration server..Done.
```

- ◆ The `symconfigure query` command issued from a second window checks the status of the configuration change session 30 times at 10-second intervals:

```
symconfigure query -sid 120 -i 10 -c 30
```

```
Establishing a monitoring session.....Established.
The session changes are in the class of: Mapping/unmapping devices
{
  unmap dev 0001 from dir 4B:0;
}

The last action requested was: PREPARE
The state of the last action is: Running
Step 08 of 11 steps.....Executing.
PREPARE.....Done.
The last action requested was: COMMIT
The state of the last action is: Running
Step 04 of 44 steps.....Executing.
.....
Step 39 of 44 steps.....Executing.
Step 42 of 44 steps.....Executing.
Monitored session has terminated
Terminating the monitoring session.....Done.
```

- ◆ Executing the following utilities updates the host view:

```
drvconfig
disks
devlinks
```

- ◆ The `symcfg discover` command updates the SYMAPI database:

symcfg discover

- ◆ The `symdev show` command output now provides an updated SYMAPI view after unmapping one of the paths of device 0001. The *Front Director Paths* section no longer shows any mapping to this device that the example unmapped. The path using director 04B, port 0, has been deleted:

symdev show 0001 -sid 120

Device Physical Name : Not Visible
 Device Symmetrix Name : 0001
 Device Serial ID : 20001200
 Symmetrix ID : 000184500120

Attached BCV Device : N/A
 Attached Snap TGT Device : N/A

Vendor ID : EMC
 Product ID : SYMMETRIX
 Product Revision : 5568

Device Emulation Type : FBA
 Device Defined Label Type: N/A
 Device Defined Label : N/A
 Device Sub System Id : N/A

Device Block Size : 512

Device Capacity

```
{
  Cylinders      :      220
  Tracks         :      3300
  512-byte Blocks :     211200
  MegaBytes      :       103
  KiloBytes      :     105600
}
```

Device Configuration : Unprotected
 Dynamic Spare Invoked : No
 Dynamic RDF Capability : Not_RDF_Capable
 Device Service State : Normal

Device Status : Write Disabled (WD)
 Device SA Status : Write Disabled (WD)

Front Director Paths (1):

```
{
  -----
                                POWERPATH  DIRECTOR      PORT
                                -----
PdevName                      Type      Num   Type   Num   Sts   VBUS TID LUN
-----
Not Visible                    N/A      04A   FA     0     WD    000 00 000
}
```

Example 6: Setting front-end port attributes

The hardware setup for this example is a Solaris host (api145) connected to a Symmetrix array (sid 120). The host is connected to front-end director FA-4A, which is a Fibre Channel director. The example illustrates how to modify a front-end port attribute (VCM_State).

- ◆ Using the `-connections` option with `symcfg list` allows you to see the host connections to a Symmetrix array. The following display shows the front-end director (FA-4A) that this host uses to reach the Symmetrix (sid 120):

```
symcfg list -sid 120 -connections
```

```
Symmetrix ID : 000184500120
```

Symmetrix		Host				
Director	Port	Node Name	IP Address	HW Type	OS Name	OS Revision
FA-4A	0	api145	172.23.65.145	sun4u	SunOS	5.6

- ◆ The Symmetrix port attribute VCM_State is a fibre protocol flag that can be either enabled or disabled (the default). You need to enable this flag for device masking or Volume Logix software (which provides volume management controls to handle access to Symmetrix devices). The `symcfg list` command provides detailed (`-v`) information about the Symmetrix configuration and the front-end director/port that connects the host to the Symmetrix array. The section *Fibre Specific Flags*, in the example below, shows that VCM_State is currently disabled. The ellipsis (.....) indicates where some output was omitted for brevity:

```
symcfg list -sid 120 -sa 04A -p 0 -v
```

```
Symmetrix ID: 000184500120
```

```

Product Model           : 8430
Symmetrix ID            : 000184500120

Microcode Version (Number) : 5568
Microcode Date          : 11.12.2001

Microcode Patch Date     : 11.12.2001
Microcode Patch Level    : 37

```

```

.....

Number of Configured (Sym) Devices : 159
Number of Visible (Host) Devices  : 6
Number of Configured Actual Disks  : 96
Number of Configured Hot Spares    : 0

Number of Powerpath Devices       : 0
Powerpath Run Time Version        : N/A

SDDF Configuration State          : Enabled
Configuration Change State        : Enabled
WORM Configuration Level          : N/A
WORM Characteristics              : N/A

Symmetrix Configuration Checksum   : 8FB66
Switched RDF Configuration State   : Disabled
Concurrent RDF Configuration State : Disabled
Dynamic RDF Configuration State    : Disabled
RDF Data Mobility Configuration State: Disabled
Access Control Configuration State  : Disabled

```

Device Masking (VCM) Config State : Disabled

Director Identification: FA-4A

```

Director Type           : FibreChannel
Director Status         : Online

Number of Director Ports : 1
Director Ports Status    : [ON,N/A,N/A,N/A]

Director Symbolic Number : 04A
Director Numeric Number  : 4
Director Slot Number     : 4

```

Director Port: 0

```

WWN Node Name           : 50060482bfcfe603
WWN Port Name           : 50060482bfcfe603

```

```

Fibre Channel Loop ID   : 0
Fibre Adapter Type      : N/A

```

SCSI Flags

```

{
  Tagged_Commands       : Enabled
  Linked_Commands       : Disabled
  Sync_Transfer         : Disabled
  Wide_Transfer         : Disabled
  Negotiate_Reset       : Disabled
  Soft_Reset            : Disabled
  Environ_Set           : Disabled
  Cyl_Count_In_Name     : Disabled
}

```

Fibre Specific Flags

```

{
  Disk_Array            : Disabled
  Volume_Set_Addressing : Disabled
  Hard_Addressing       : Enabled
  Non_Participating     : Disabled
  Global_3rdParty_Logout : Disabled
  Init_Point_to_Point   : Disabled
  Unique_WWN            : Enabled
  Generic_VSA           : Disabled
  VCM_State             : Disabled
  Class_2_Service       : Disabled
  OpenVMS               : Disabled
}

```

- ◆ The `symconfigure verify` command verifies that the current Symmetrix configuration is available for host-initiated configuration changes (that is, a configuration change session can be opened):

`symconfigure -sid 120 verify`

```

A Configuration Change Verification is in progress. Please wait...
  Establishing a configuration change
session.....Established.
  Verifying
configuration.....Verified.
  Terminating the configuration change session.....Done.

The configuration verification session has succeeded.

```

- ◆ The following command uses the vi text editor to create a text file named port.cmd. As was done here, you can enter into the file the set port entry that enables the VCM_State flag for director 04A, port 0, of this Symmetrix array:

```
vi port.cmd
```

```
set port 04A:0 vcm_state=enable;
```

- ◆ The symconfigure command with the commit argument executes the command file and begins the process of setting the VCM_State port flag to “enable:”

```
symconfigure -sid 120 -file port.cmd -v -noprompt commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000184500120
{
    set port 4A:0
        VCM_State=enable;
}

Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 004 of 011 steps.....Executing.
Step 006 of 011 steps.....Executing.
Step 007 of 011 steps.....Executing.
Step 008 of 011 steps.....Executing.
Step 008 of 011 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 003 of 039 steps.....Executing.
.....
Step 038 of 039 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

- ◆ To monitor the configuration change session while the `symconfigure commit` operation is processing, you can issue the `symconfigure query` command or the following UNIX `tail` command from a second window. The following two commands allow you to compare outputs from the `tail` command and from `symconfigure query`:

```
tail -f /var/symapi/log/symapi-20011228.log
```

```
12/28/2001 16:10:34.404 22122 0 EMC:SYMCLI iCfgChgSessionStart Called to start a local
session for SID 000184500120
12/28/2001 16:10:40.524 22122 Establishing session with Local cfg srvr
(000184500120)...Established.
12/28/2001 16:10:47.871 22122 {
12/28/2001 16:10:47.878 22122   set port 4A:0 VCM_State=ENABLE;
12/28/2001 16:10:47.881 22122 }
12/28/2001 16:10:52.891 22122 Submitting configuration changes.....Submitted.
12/28/2001 16:11:03.901 22122 Validating configuration changes.....Validated.
12/28/2001 16:11:14.922 22122 Initiating PREPARE of configuration changes...Queued.
12/28/2001 16:11:26.922 22122 PREPARE requesting required resources.....Obtained.
12/28/2001 16:11:28.932 22122 Step 004 of 011 steps.....Executing.
12/28/2001 16:11:34.942 22122 Step 007 of 011 steps.....Executing.
12/28/2001 16:11:41.952 22122 Step 008 of 011 steps.....Executing.
12/28/2001 16:12:03.972 22122 Local: PREPARE.....Done.
12/28/2001 16:12:15.082 22122 Initiating COMMIT of configuration changes...Queued.
12/28/2001 16:12:27.083 22122 COMMIT requesting required resources.....Obtained.
12/28/2001 16:12:28.093 22122 Step 004 of 039 steps.....Executing.
.....
12/28/2001 16:17:13.577 22122 Step 038 of 039 steps.....Executing.
12/28/2001 16:17:19.587 22122 Local: COMMIT.....Done.
12/28/2001 16:17:19.602 22122 0 EMC:SYMCLI process_load_request Switching to FULL load for
000184500120 because the configuration changed
12/28/2001 16:17:26.275 22122 Terminating session with configuration server..Done.
```

- ◆ The `symconfigure query` command issued from a second window checks the status of the configuration change session 30 times at 10-second intervals:

```
symconfigure query -sid 120 -i 10 -c 30
```

```
Establishing a monitoring session.....Established.
The session changes are in the class of: Modifying port configurations
{
    set port 4A:0
        VCM_State=enable;
}

The last action requested was: PREPARE
The state of the last action is: Running
PREPARE.....Done.
The last action requested was: COMMIT
The state of the last action is: Running
Step 06 of 39 steps.....Executing.
.....
Step 38 of 39 steps.....Executing.
COMMIT.....Done.
Monitored session has terminated
Terminating the monitoring session.....Done.
```

- ◆ The `symcfg list` command again provides the detailed (-v) information about the Symmetrix configuration and the front-end director/port that connects the host to the Symmetrix array. The section *Fibre Specific Flags*, in the example below, shows that `VCM_State` is now enabled:

```
symcfg -sa 04A -p 0 list -v -sid 120
```

```
Symmetrix ID: 000184500120
```

```
Product Model           : 8430
Symmetrix ID            : 000184500120

Microcode Version (Number) : 5568 (15BFAA01)
Microcode Date           : 11.12.2001

Microcode Patch Date      : 11.12.2001
Microcode Patch Level     : 37
```

```
.....
```

```
Director Identification: FA-4A
```

```
Director Type           : FibreChannel
Director Status          : Online

Number of Director Ports : 1
Director Ports Status    : [ON,N/A,N/A,N/A]

Director Symbolic Number : 04A
Director Numeric Number  : 4
Director Slot Number     : 4
```

```
Director Port: 0
```

```
WWN Node Name           : 50060482bfcfe603
WWN Port Name            : 50060482bfcfe603

Fibre Channel Loop ID    : 0
Fibre Adapter Type       : N/A
```

```
.....
```

```
Fibre Specific Flags
```

```
{
  Disk_Array             : Disabled
  Volume_Set_Addressing  : Disabled
  Hard_Addressing        : Enabled
  Non_Participating      : Disabled
  Global_3rdParty_Logout : Disabled
  Init_Point_to_Point    : Disabled
  Unique_WWN             : Enabled
  Generic_VSA            : Disabled
  VCM_State              : Enabled
  Class_2_Service        : Disabled
  OpenVMS                : Disabled
}
```


Example 7: Configuring concurrent RDF devices

This example is performed using Solutions Enabler version 6.0. You can use the SYMCLI Configuration Manager to configure concurrent dynamic SRDF pairs while the Symmetrix array is running. In this Fibre Channel example, the controlling host is connected to a local Symmetrix array (sid 000187400011). The local Symmetrix is connected via RDF links to two remote Symmetrix arrays (sid 000187400093 and sid 000000006201). The example uses two different RDF (RA) groups to achieve the connection between each local R1 device and its two remote R2 mirrors.

For brevity, this example shows creating an RDF Group with only one connection (one local director to one remote director). Using recommended practice, you would have at least two sets of RA directors supporting the RDF group.

- ◆ The following command uses the vi text editor to create a text file named `set_sym.cmd`. As shown here, you can enter into this file the `set symmetrix` entries that set the three flags necessary to enable concurrent dynamic RDF in the Symmetrix array attached to your host (these flags may or may not have been set using SymmWin). You will also need to ensure that the `dynamic_rdf` flag is enabled in the two remote Symmetrix arrays (enabling the concurrent flags in the remote arrays is not necessary).

```
vi set_sym.cmd
```

```
set symmetrix dynamic_rdf=enable;
set symmetrix concurrent_dynamic_rdf=enable;
set symmetrix concurrent_rdf=enable;
```

- ◆ The `symconfigure commit` command executes the command file and initiates setting the Symmetrix `dynamic_rdf` parameter and the `concurrent_dynamic_rdf` flag. The ellipsis (.....) indicates output that was omitted for brevity:

```
symconfigure -sid 11 -file set_sym.cmd -v -noprompt commit
```

```
Establishing a configuration change session.....Established.
Processing symmetrix 000187400011
{
  set symmetrix dynamic_rdf_configuration = Enabled;
  set symmetrix concurrent_dynamic_rdf = Enabled;
  set symmetrix concurrent_rdf = Enabled;
}

Submitting configuration changes.....Submitted.
Validating configuration changes.....Validated.
Initiating PREPARE of configuration changes.....Queued.
PREPARE requesting required resources.....Obtained.
Step 008 of 011 steps.....Executing.
Local: PREPARE.....Done.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 012 of 034 steps.....Executing.
.....
Step 032 of 034 steps.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

- ◆ The `symcfg list` command displays current RDF (RA) groups that serve as RDF links to connect local Symmetrix 000187400011 to remote Symmetrix 000187400093 through director 4D:

```
symcfg list -ra 4D -sid 11
```

```
Symmetrix ID: 000187400011
```

S Y M M E T R I X R D F D I R E C T O R S										
Ident	Symb	Num	Slot	Type	Attr	Remote SymmID	Local RA Grp	Remote RA Grp	Status	
RF-4D	04D	52	4	RDF-R1	-	000187400093	27 (1A)	23 (16)	Online	
					-	000187400093	28 (1B)	24 (17)		
					-	000187400093	29 (1C)	25 (18)		
					-	000187400093	30 (1D)	26 (19)		
					-	000187400093	31 (1E)	27 (1A)		

- ◆ The following `symrdf addgrp` command creates a dynamic RDF group that represents another RDF link between Symmetrix 000187400011 and Symmetrix 000187400093. It adds dynamic RDF group 58 on the local Symmetrix, and RDF group 58 on the remote Symmetrix. You must specify a group label (`grp58` in this case) that can be used when modifying or deleting the group. Creation of the dynamic RDF group includes director 4D from the local Symmetrix and 3C from the remote Symmetrix as the director end points of this connection.

It is important to be aware of your network topology when creating dynamic RDF groups between two Symmetrix arrays. To create a dynamic RDF link (a connection) between RA directors, the director end points must be able to see each other through the Fibre Channel fabric. For example, a dynamic RDF link can be created between local and remote directors only if the Fibre Channel zoning is set up so that the two directors can see each other through the fabric:

```
symrdf -v addgrp -label grp58 -rdfg 58 -sid 187400011 -dir 4D
-remote_rdfg 58 -remote_sid 000187400093 -remote_dir 3C -noprompt
```

An RDF Addgrp operation execution is in progress for dynamic group 'grp58'. Please wait...

Successfully Added Dynamic RDF Group 'grp58' for Symm: 000187400011

Note: For brevity, this command creates an RDF group with only one connection (local director 4D to remote director 3C). Using recommended practice, you would have at least two sets of RA directors supporting the RDF group. For example, the previous command could include `-dir 4D,4C` and `-remote_dir 3C,3D`.

- ◆ Another `symcfg list` command verifies the logical connections from the local director (4D) point of view. Dynamic RDF group 58 has been added to both the local and remote Symmetrix arrays:

```
symcfg list -ra 4D -sid 11
```

```
Symmetrix ID: 000187400011
```

S Y M M E T R I X R D F D I R E C T O R S										
Ident	Symb	Num	Slot	Type	Attr	Remote SymmID	Local RA Grp	Remote RA Grp	Status	
RF-4D	04D	52	4	RDF-R1	-	000187400093	27 (1A)	23 (16)	Online	
					-	000187400093	28 (1B)	24 (17)		
					-	000187400093	29 (1C)	25 (18)		
					-	000187400093	30 (1D)	26 (19)		
					-	000187400093	31 (1E)	27 (1A)		
					-	000187400093	58 (39)	58 (39)		

- ◆ Another `symcfg list` displays current RDF (RA) groups that serve as RDF links to connect local Symmetrix 000187400011 to the second remote Symmetrix (000000006201) through director 13A:

```
symcfg list -ra 13A -sid 11
```

```
Symmetrix ID: 000187400011
```

S Y M M E T R I X R D F D I R E C T O R S										
Ident	Symb	Num	Slot	Type	Attr	Remote SymmID	Local RA Grp	Remote RA Grp	Status	
RE-13A	13A	13	13	RDF-R1	-	000000006201	47 (2E)	47 (2E)	Online	
					-	000000006201	50 (31)	50 (31)		

- ◆ The following `symrdf addgrp` command creates a dynamic RDF group that represents another RDF link between Symmetrix 000187400011 and Symmetrix 000000006201. It adds dynamic RDF group 51 on the local Symmetrix, and RDF group 51 on the remote Symmetrix. Creation of the local and remote RDF groups includes director 13A from the local Symmetrix and 13B from the remote Symmetrix array:

```
symrdf -v addgrp -label grp51 -rdfg 51 -sid 187400011 -dir 13A  
-remote_rdfg 51 -remote_sid 000000006201 -remote_dir 13B -noprompt
```

```
An RDF Addgrp operation execution is in  
progress for dynamic group 'grp51'. Please wait...
```

```
Successfully Added Dynamic RDF Group 'grp51' for Symm: 000187400011
```

Note: For brevity, this command creates an RDF Group with only one connection (local director 13A to remote director 13B). Using recommended practice, you would have at least two sets of RA directors supporting the RDF group. For example, the previous command could include `-dir 13A,13B` and `-remote_dir 13B,13C`.

- ◆ Another `symcfg list` command verifies the logical connections from the local director (13A) point of view. Dynamic RDF group 51 has been added to both the local and remote Symmetrix arrays:

```
symcfg list -ra 13A -sid 11
```

```
Symmetrix ID: 000187400011
```

S Y M M E T R I X R D F D I R E C T O R S

Ident	Symb	Num	Slot	Type	Attr	Remote SymmID	Local RA Grp	Remote RA Grp	Status
RE-13A	13A	13	13	RDF-R1	-	000000006201	47 (2E)	47 (2E)	Online
					-	000000006201	50 (31)	50 (31)	
					-	000000006201	51 (32)	51 (32)	

The `symdev list` command with the `-dynamic` option displays those devices on the local Symmetrix that have been configured to be capable of dynamic RDF. These will be the source devices:

```
symdev list -dynamic -sid 11
```

```
Symmetrix ID: 000187400011
```

Device Name		Directors		Device			
Sym	Physical	SA :P	DA :IT	Config	Attribute	Sts	Cap (MB)
0300	/dev/vx/rdmp/c6t0d0s2	14C:0	16A:D8	RAID-5	N/Grp'd	RW	449
0301	/dev/vx/rdmp/c6t0d1s2	14C:0	02A:D8	RAID-5	N/Grp'd	RW	449
0302	/dev/vx/rdmp/c6t0d2s2	14C:0	01A:C4	RAID-5	N/Grp'd	RW	449
0303	/dev/vx/rdmp/c6t0d3s2	14C:0	15A:C0	RAID-5	N/Grp'd	RW	449
0304	/dev/vx/rdmp/c6t0d4s2	14C:0	16A:C3	RAID-5	N/Grp'd	RW	449
0305	/dev/vx/rdmp/c6t0d5s2	14C:0	02A:C3	RAID-5	N/Grp'd	RW	449

- ◆ The following command uses the `vi` text editor to create a text file named `OEA2OEB2.list`. As was done here, you can enter into the file those device names that will constitute one of the sets of dynamic SRDF pairs (those R1/R2 pairs for the local Symmetrix and remote Symmetrix 000187400093). The R1 devices are listed in first column, and the remote R2 devices are listed in the second column on the same line as their respective R1 source. Like the R1 devices, the R2 devices must also be non-RDF devices that have been set with the dynamic RDF attribute:

```
vi OEA2OEB2.list
```

```
300 080
301 081
302 082
303 083
304 084
305 085
```

- ◆ The following command uses the vi text editor to create a text file named OEA206A2.list. As was done here, you can enter into the file those device names that will constitute the second set of dynamic SRDF pairs (those R1/R2 pairs for the local Symmetrix and remote Symmetrix 000000006201):

```
vi OEA206A2.list
```

```
300 150
301 151
302 152
303 153
304 154
305 155
```

- ◆ The symrdf createpair command parses the file called OEA20EB2.list that defines the dynamic SRDF pairs and specifies that the column-1 devices in the file are R1 devices (-type RDF1) on the local Symmetrix (000187400011). Communication is via RDF group 58 (-rdfg 58), which was previously established as the RDF link to remote Symmetrix 000187400093:

```
symrdf createpair -file OEA20EB2.list -sid 11 -rdfg 58 -type rdf1 -establish -noprompt
```

An RDF 'Create Pair' operation execution is in progress for device file 'OEA20EB2.list'. Please wait...

```
Create RDF Pair in (0011,58).....Done.
Mark target device(s) in (0011,58) for full copy from source....Started.
Devices: 0300-0305 ..... Marked.
Mark target device(s) in (0011,58) for full copy from source....Done.
Merge track tables between source and target in (0011,58).....Started.
Devices: 0300-0305 ..... Merged.
Merge track tables between source and target in (0011,58).....Done.
Resume RDF link(s) for device(s) in (0011,58).....Started.
Resume RDF link(s) for device(s) in (0011,58).....Done.
```

The RDF 'Create Pair' operation successfully executed for device file 'OEA20EB2.list'.

- ◆ A second symrdf createpair command parses the file called OEA206A2.list that defines the dynamic SRDF pairs and specifies that the column-1 devices in the file are R1 devices (-type RDF1) on the local Symmetrix (000187400011). Communication is by RDF group 51 (-rdfg 51), which was previously established as the RDF link to remote Symmetrix 000000006201:

```
symrdf createpair -file OEA206A2.list -sid 11 -rdfg 51 -type rdf1 -establish -nop
```

An RDF 'Create Pair' operation execution is in progress for device file 'OEA206A2.list'. Please wait...

```
Create RDF Pair in (0011,51).....Done.
Mark target device(s) in (0011,51) for full copy from source....Started.
Devices: 0300-0305 ..... Marked.
Mark target device(s) in (0011,51) for full copy from source....Done.
Merge track tables between source and target in (0011,51).....Started.
Devices: 0300-0305 ..... Merged.
Merge track tables between source and target in (0011,51).....Done.
Resume RDF link(s) for device(s) in (0011,51).....Started.
Resume RDF link(s) for device(s) in (0011,51).....Done.
```

The RDF 'Create Pair' operation successfully executed for device file 'OEA206A2.list'.

- ◆ The `symdmg` command creates an RDF1 type device group named `dynConc`. The `symld` command adds to the group the R1 devices that were created on the local Symmetrix (sid 11):

```
symdmg create -type rdf1 dynConc
symld -g dynConc -sid 11 -range 300:305 addall dev
```

- ◆ The `symrdf query -rdfg all` command displays the concurrent SRDF pairings for the local R1 devices in the device group `dynConc`. The `-rdfg` option allows you to see the SRDF pairs represented by both RDF (RA) groups. As shown, all concurrent pairs are in the Synchronized state:

```
symrdf -g dynConc query -rdfg all
```

```
Device Group (DG) Name      : dynConc
DG's Type                   : RDF1
DG's Symmetrix ID           : 000187400011
Remote Symmetrix ID         : 000000006201
RDF (RA) Group Number       : 51 (32)
Remote Symmetrix ID         : 000187400093
RDF (RA) Group Number       : 58 (39)
```

Source (R1) View					Target (R2) View				MODES		
-----					-----				-----		
		ST			LI	ST					
Standard		A			N	A					
Logical		T	R1 Inv	R2 Inv	K	T	R1 Inv	R2 Inv		RDF Pair	
Device	Dev	E	Tracks	Tracks	S	Dev	E	Tracks	Tracks	MDA	STATE
-----					-----				-----		
DEV001	0300	RW	0	0	RW	0080	WD	0	0	S..	Synchronized
		RW	0	0	RW	0150	WD	0	0	S..	Synchronized
DEV002	0301	RW	0	0	RW	0081	WD	0	0	S..	Synchronized
		RW	0	0	RW	0151	WD	0	0	S..	Synchronized
DEV003	0302	RW	0	0	RW	0082	WD	0	0	S..	Synchronized
		RW	0	0	RW	0152	WD	0	0	S..	Synchronized
DEV004	0303	RW	0	0	RW	0083	WD	0	0	S..	Synchronized
		RW	0	0	RW	0153	WD	0	0	S..	Synchronized
DEV005	0304	RW	0	0	RW	0084	WD	0	0	S..	Synchronized
		RW	0	0	RW	0154	WD	0	0	S..	Synchronized
DEV006	0305	RW	0	0	RW	0085	WD	0	0	S..	Synchronized
		RW	0	0	RW	0155	WD	0	0	S..	Synchronized

Total		-----			-----			-----			
Track(s)		0			0			0		0	
MB(s)		0.0			0.0			0.0		0.0	

Legend for MODES:

```
M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
D(omino)             : X = Enabled, . = Disabled
A(daptive Copy)      : D = Disk Mode, W = WP Mode, . = ACp off
```

Example 8: Updating a disk label with emulated device geometry

Note: Because the change in geometry is destructive, disk geometry changes should be made to a device prior to it being allocated to a host. If a device needs to be re-purposed and does not contain data that needs to be kept, the device must be unmapped or unmasked and write disabled or not ready before its geometry can be modified.

The following procedure can be run to prepare the host volumes and update the disk label using only native operating system utilities:

1. Have an EMC Customer Engineer enable FBA Geometry Emulation or use Solutions Enabler to set the device geometry to SYMM-7.
2. Synchronize the source and target volumes.
3. Split the target volumes and insure that they are ready and read/write.
4. Print the VTOC of the source volumes to a file, as follows:

```
prtvtoc /dev/rdisk/SourceVolume > FileName
```

5. Transfer the VTOC files to the target host.
6. Update the manufacturer information on the target volumes, as follows:

```
format TargetVolume
type
Auto configure
label
```

7. Update the partition information on the target volumes, as follows:

```
fmthard -s FileName TargetVolume
```

Procedure example

The following example shows the commands and the resulting output of a data migration and update to the disk label.

This example uses Solutions Enabler to modify the target disk geometry and SRDF to perform the migration. The disk group that will be migrated is called `geodg` and contains Symmetrix devices `036` and `037` from the source array which is serial number `0073`, and is a DMX 2000S-M2 running Enginuity 5671:

symcfg list

S Y M M E T R I X						
SymmID	Attachment	Model	Mcode Version	Cache Size (MB)	Num Phys Devices	Num Symm Devices
000387720073	Local	2000S-M2	5671	65536	30	80
000190100414	Remote	DMX3-24	5773	131072	0	144

syminq | grep c2t50060485C708E268d11s2

/dev/rdsk/*d11s2	R1	EMC	SYMMETRIX	5671	7300036000	4417920
/dev/vx/rdmp/*d11s2	R1	EMC	SYMMETRIX	5671	7300036000	4417920

syminq | grep c2t50060485C708E268d12s2

/dev/rdsk/*d12s2	R1	EMC	SYMMETRIX	5671	7300037000	4417920
------------------	----	-----	-----------	------	------------	---------

```
/dev/vx/rdmp/*d12s2 R1      EMC      SYMMETRIX      5671 7300037000      4417920
```

vxprint -htQq

```
dg geodg      default      default  29000      1203016756.36.licoa034

dm geodg01    c2t50060485C708E268d11s2 auto 2048 8831616 -
dm geodg02    c2t50060485C708E268d12s2 auto 2048 8831616 -

v  geodgvol01 -            ENABLED  ACTIVE  17663232 ROUND -      fsgen
pl geodgplex01 geodgvol01  ENABLED  ACTIVE  17663232 CONCAT -      RW
sd geodg01sd01 geodgplex01 geodg01  0      8831616 0      c2t5006* ENA
sd geodg02sd01 geodgplex01 geodg02  0      8831616 8831616 c2t5006* ENA
```

The disk group contains a single Veritas volume that is mounted and contains data:

df -k

Filesystem	kbytes	used	avail	capacity	Mounted on
/dev/dsk/c0t1d0s0	16379106	6176410	10038905	39%	/
/proc	0	0	0	0%	/proc
mnttab	0	0	0	0%	/etc/mnttab
fd	0	0	0	0%	/dev/fd
swap	1430576	112	1430464	1%	/var/run
dmpfs	1430464	0	1430464	0%	/dev/vx/dmp
dmpfs	1430464	0	1430464	0%	/dev/vx/rdmp
swap	1430840	376	1430464	1%	/tmp
/dev/vx/dsk/geodg/geodgvol01	8831616	4213799	4329252	50%	/mp/geodgvol01_mp

ls -l /mp/geodgvol01_mp

```
total 8388678
drwxr-xr-x  2 root    other      3072 Feb 14 16:36 flat_files
drwxr-xr-x  2 root    root        96 Feb 14 14:26 lost+found
-rw-----T  1 root    other    1073741824 Feb 14 16:37 testfile1
-rw-----T  1 root    other    1073741824 Feb 14 16:38 testfile2
-rw-----T  1 root    other    1073741824 Feb 14 16:39 testfile3
-rw-----T  1 root    other    1073741824 Feb 14 16:41 testfile4
```

ls -l /mp/geodgvol01_mp/flat_files

```
total 290
-rw-r--r--  1 root    other      131 Feb 14 16:36 arp.conf
-rw-r--r--  1 root    other     1583 Feb 14 16:36 audiocs.conf
-rw-r--r--  1 root    other     1414 Feb 14 16:36 audioens.conf
-rw-r--r--  1 root    other     1601 Feb 14 16:36 audiotcs.conf
-rw-r--r--  1 root    other     1810 Feb 14 16:36 bofi.conf
-rw-r--r--  1 root    other      135 Feb 14 16:36 clone.conf
-rw-r--r--  1 root    other      129 Feb 14 16:36 cn.conf
-rw-r--r--  1 root    other      139 Feb 14 16:36 conskbd.conf
-rw-r--r--  1 root    other      164 Feb 14 16:36 consms.conf
```

cat /mp/geodgvol01_mp/flat_files/arp.conf

```
#
# Copyright (c) 1992, by Sun Microsystems, Inc.
#
#ident  "@(#)arp.conf  1.4      93/06/03  SMI"
name="arp" parent="pseudo" instance=0;
```


The devices that will be the targets of the migration in Symmetrix 0414 (DMX3-24, Enginuity 5773) are devices 076 and 077. Before beginning the migration, the device geometry is correctly set to SYMM-6 so that they match the source devices on the DMX:

```
symconfigure -sid 14 -cmd "set dev 0076:0077 geometry=SYMM-6 cyls=9204;" commit
```

A Configuration Change operation is in progress. Please wait...

```
Establishing a configuration change session.....Established.
Processing symmetrix 000190100414
Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Locking devices.....Locked.
Validating configuration changes.....Validated.
Preparing configuration changes.....Prepared.
Committing configuration changes.....Started.
Committing configuration changes.....Committed.
Terminating the configuration change session.....Done.
```

The configuration change session has successfully completed.

```
symdev list -geometry_set -sid 14
```

Symmetrix ID: 000190100414

Device Name		Directors		Device			Cap
Sym	Physical	SA :P DA :IT	Config	Attribute	Sts	(MB)	
0076	Not Visible	???:? 16C:C9	2-Way Mir	N/Grp'd	RW	4314	
0077	Not Visible	???:? 16D:DB	2-Way Mir	N/Grp'd	RW	4314	

Note: Setting the device geometry can be performed before or after the devices are configured as SRDF devices.

The devices are configured as SRDF devices using dynamic RDF (RDF group 01 already exists), and is synchronized:

```
cat dynamic
036 076
037 077
```

```
symrdf -sid 73 -file dynamic -rdfg 1 createpair -type R1 -establish -nop
```

An RDF 'Create Pair' operation execution is in progress for device file 'dynamic'. Please wait...

```
Create RDF Pair in (0073,001).....Started.
Create RDF Pair in (0073,001).....Done.
Mark target device(s) in (0073,001) for full copy from source....Started.
Devices: 0036-0037 in (0073,001).....Marked.
Mark target device(s) in (0073,001) for full copy from source....Done.
Merge track tables between source and target in (0073,001).....Started.
Devices: 0036-0037 in (0073,001).....Merged.
Merge track tables between source and target in (0073,001).....Done.
Resume RDF link(s) for device(s) in (0073,001).....Started.
Resume RDF link(s) for device(s) in (0073,001).....Done.
```

The RDF 'Create Pair' operation successfully executed for device file 'dynamic'.

A Symmetrix device group containing the two devices has been created and the devices have been synchronized:

```
symrdf -g vxgeodg query
```

```
Device Group (DG) Name      : vxgeodg
DG's Type                  : RDF1
DG's Symmetrix ID          : 000387720073   (Microcode Version: 5671)
Remote Symmetrix ID        : 000190100414   (Microcode Version: 5773)
RDF (RA) Group Number      : 1 (00)
```

Source (R1) View					Target (R2) View				MODES	
Standard	ST				LI	ST				
Logical	A				N	A				
Device	T	R1 Inv	R2 Inv		K	T	R1 Inv	R2 Inv		RDF Pair
Dev	E	Tracks	Tracks		S Dev	E	Tracks	Tracks	MDA	STATE
DEV001	0036	RW	0	0	RW	0076	WD	0	0	S.. Synchronized
DEV002	0037	RW	0	0	RW	0077	WD	0	0	S.. Synchronized
Total										
Track(s)			0	0			0	0		
MB(s)			0.0	0.0			0.0	0.0		

Legend for MODES:

```
M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
D(omino)           : X = Enabled, . = Disabled
A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

A copy of the VTOC from the source volumes must be saved so that it can be written to the target devices after they are labeled. This is done by saving the output of prtvtoc to a file:

```
prtvtoc /dev/rdsk/c3t50060485C708E247d11s2 > /mig_files/d11_0036.vtoc
```

```
prtvtoc /dev/rdsk/c3t50060485C708E247d12s2 > /mig_files/d12_0037.vtoc
```

```
cat /mig_files/d11_0036.vtoc
```

```
* /dev/rdsk/c3t50060485C708E247d11s2 partition map
```

```
*
```

```
* Dimensions:
```

```
* 512 bytes/sector
* 64 sectors/track
* 15 tracks/cylinder
* 960 sectors/cylinder
* 9204 cylinders
* 9202 accessible cylinders
*
```

```
* Flags:
```

```
* 1: unmountable
* 10: read-only
*
```

* Partition	Tag	Flags	First Sector	Sector Count	Last Sector	Mount Directory
2	5	01	0	8833920	8833919	
7	15	01	0	8833920	8833919	

```
# cat /mig_files/d12_0037.vtoc
```

```
* /dev/rdisk/c3t50060485C708E247d12s2 partition map
*
* Dimensions:
*   512 bytes/sector
*   64 sectors/track
*   15 tracks/cylinder
*   960 sectors/cylinder
*   9204 cylinders
*   9202 accessible cylinders
*
* Flags:
*   1: unmountable
*   10: read-only
*
*
* Partition  Tag  Flags      First      Sector      Last
*           Tag  Flags      Sector      Count      Sector      Mount Directory
*       2      5    01          0    8833920    8833919
*       7     15    01          0    8833920    8833919
```

The VTOC files from the source volumes can then be transferred to the target host.
The SRDF pairs can now be split:

```
symrdf -g vxgeodg split -nop
```

An RDF 'Split' operation execution is
in progress for device group 'vxgeodg'. Please wait...

```
Suspend RDF link(s).....Done.
Read/Write Enable device(s) on RA at target (R2).....Done.
```

The RDF 'Split' operation successfully executed for device group 'vxgeodg'.

The target devices are now available to the target host:

```
./inq -no_dots | grep 0076
```

```
/dev/rdsk/c2t3d11s2 :EMC :SYMMETRIX :5773 :1400076000 : 4417920
/dev/vx/rdmp/c2t3d11s2 :EMC :SYMMETRIX :5773 :1400076000 : 4417920
```

```
./inq -no_dots | grep 0077
```

```
/dev/rdsk/c2t3d12s2 :EMC :SYMMETRIX :5773 :1400077000 : 4417920
/dev/vx/rdmp/c2t3d12s2 :EMC :SYMMETRIX :5773 :1400077000 : 4417920
```

They contain the label from the source DMX. The labels can now be rewritten on the
target devices:

```
...
47. c2t3d11 <EMC-SYMMETRIX-5671 cyl 9202 alt 2 hd 15 sec 64>
    /pci@1f,4000/fibre-channel@5/sd@3,b
48. c2t3d12 <EMC-SYMMETRIX-5671 cyl 9202 alt 2 hd 15 sec 64>
    /pci@1f,4000/fibre-channel@5/sd@3,c
```

```
Specify disk (enter its number): 47
selecting c2t3d11
[disk formatted]
```

```
FORMAT MENU:
```

```
disk      - select a disk
type      - select (define) a disk type
partition - select (define) a partition table
current   - describe the current disk
format    - format and analyze the disk
repair    - repair a defective sector
label     - write label to the disk
analyze   - surface analysis
```

```

defect      - defect list management
backup      - search for backup labels
verify      - read and display labels
save        - save new disk/partition definitions
inquiry     - show vendor, product and revision
volname     - set 8-character volume name
!<cmd>      - execute <cmd>, then return
quit
format> type

```

AVAILABLE DRIVE TYPES:

0. Auto configure
1. Quantum ProDrive 80S
2. Quantum ProDrive 105S
3. CDC Wren IV 94171-344
4. SUN0104
5. SUN0207
6. SUN0327
7. SUN0340
8. SUN0424
9. SUN0535
10. SUN0669
11. SUN1.0G
12. SUN1.05
13. SUN1.3G
14. SUN2.1G
15. SUN2.9G
16. Zip 100
17. Zip 250
18. SUN36G
19. EMC-SYMMETRIX-5773
20. EMC-SYMMETRIX-5773
21. EMC-SYMMETRIX-5773
22. EMC-SYMMETRIX-5671
23. other

```

Specify disk type (enter its number)[22]: 0
c2t3d11: configured with capacity of 4.21GB
<EMC-SYMMETRIX-5773 cyl 9202 alt 2 hd 15 sec 64>
selecting c2t3d11
[disk formatted]
format> label
Ready to label disk, continue? Y

```

```

Specify disk (enter its number)[48]: 48
selecting c2t3d12
[disk formatted]
format> type

```

AVAILABLE DRIVE TYPES:

0. Auto configure
1. Quantum ProDrive 80S
2. Quantum ProDrive 105S
3. CDC Wren IV 94171-344
4. SUN0104
5. SUN0207
6. SUN0327
7. SUN0340
8. SUN0424
9. SUN0535
10. SUN0669
11. SUN1.0G
12. SUN1.05
13. SUN1.3G

```

14. SUN2.1G
15. SUN2.9G
16. Zip 100
17. Zip 250
18. SUN36G
19. EMC-SYMMETRIX-5773
20. EMC-SYMMETRIX-5773
21. EMC-SYMMETRIX-5773
22. EMC-SYMMETRIX-5671
23. EMC-SYMMETRIX-5773
24. other
Specify disk type (enter its number)[22]: 0
c2t3d12: configured with capacity of 4.21GB
<EMC-SYMMETRIX-5773 cyl 9202 alt 2 hd 15 sec 64>
selecting c2t3d12
[disk formatted]
format> label
Ready to label disk, continue? y

format> q
#

```

The target device labels now contain the correct Manufacturer ID and disk geometry. The VTOC that was copied over from the source host can now be written to the target devices:

```
fmthard -s /mig_files/d11_0036.vtoc /dev/rdisk/c2t3d11s2
```

```
fmthard: New volume table of contents now in place.
```

```
fmthard -s /mig_files/d12_0037.vtoc /dev/rdisk/c2t3d12s2
```

```
fmthard: New volume table of contents now in place.
```

The VTOC on the target devices now matches the VTOC on the source devices:

```
prtvto /dev/rdisk/c2t3d11s2
* /dev/rdisk/c2t3d11s2 partition map
*
* Dimensions:
*   512 bytes/sector
*   64 sectors/track
*   15 tracks/cylinder
*   960 sectors/cylinder
*   9204 cylinders
*   9202 accessible cylinders
*
* Flags:
*   1: unmountable
*   10: read-only
*
*
* Partition  Tag  Flags      First      Sector      Last
* Partition  Tag  Flags      Sector      Count      Sector  Mount Directory
*       2       5    01         0    8833920    8833919
*       7      15    01         0    8833920    8833919

prtvto /dev/rdisk/c2t3d12s2
* /dev/rdisk/c2t3d12s2 partition map
*
* Dimensions:
*   512 bytes/sector
*   64 sectors/track
*   15 tracks/cylinder
*   960 sectors/cylinder
*   9204 cylinders
*   9202 accessible cylinders

```

```

*
* Flags:
*   1: unmountable
*  10: read-only
*
*
* Partition  Tag  Flags      First      Sector      Last
*            Tag  Flags      Sector      Count      Sector  Mount Directory
*            2    5    01          0    8833920    8833919
*            7   15    01          0    8833920    8833919

```

The Veritas disk group can be imported, the volume can be started, and the file system can be mounted:

```
vxvg -C import geodg
```

```
vxvol -g geodg start geodgvol01
```

```
vxprint -htQq
```

```

dg geodg          default      default  29000      1203016756.36.1licoa034

dm geodg01        c2t3d11s2    auto     2048      8831616   -
dm geodg02        c2t3d12s2    auto     2048      8831616   -

v  geodgvol01     -              ENABLED  ACTIVE    17663232  ROUND    -      fsgen
pl geodgplex01    geodgvol01    ENABLED  ACTIVE    17663232  CONCAT   -      RW
sd geodg01sd01    geodgplex01    geodg01  0         8831616   0        c2t3d11   ENA
sd geodg02sd01    geodgplex01    geodg02  0         8831616   8831616  c2t3d12   ENA

```

```
fsck -F vxfs /dev/vx/dsk/geodg/geodgvol01
```

```
log replay in progress
```

```
replay complete - marking super-block as CLEAN
```

```
mount -F vxfs /dev/vx/dsk/geodg/geodgvol01 /mp/geodgvol01_R2mp
```

The data is preserved:

```
ls -l /mp/geodgvol01_R2mp
```

```

total 8388678
drwxr-xr-x  2 root    other      3072 Feb 14 16:36 flat_files
drwxr-xr-x  2 root    root         96 Feb 14 14:26 lost+found
-rw-----T  1 root    other    1073741824 Feb 14 16:37 testfile1
-rw-----T  1 root    other    1073741824 Feb 14 16:38 testfile2
-rw-----T  1 root    other    1073741824 Feb 14 16:39 testfile3
-rw-----T  1 root    other    1073741824 Feb 14 16:41 testfile4

```

```
ls -l /mp/geodgvol01_R2mp/flat_files
```

```

total 290
-rw-r--r--  1 root    other      131 Feb 14 16:36 arp.conf
-rw-r--r--  1 root    other     1583 Feb 14 16:36 audiocs.conf
-rw-r--r--  1 root    other     1414 Feb 14 16:36 audioens.conf
-rw-r--r--  1 root    other     1601 Feb 14 16:36 audiotcs.conf
-rw-r--r--  1 root    other     1810 Feb 14 16:36 bofi.conf
-rw-r--r--  1 root    other      135 Feb 14 16:36 clone.conf
-rw-r--r--  1 root    other      129 Feb 14 16:36 cn.conf
-rw-r--r--  1 root    other      139 Feb 14 16:36 conskbd.conf
-rw-r--r--  1 root    other      164 Feb 14 16:36 consms.conf
...

```

```
cat /mp/geodgvol01_R2mp/flat_files/arp.conf
```

```

#
# Copyright (c) 1992, by Sun Microsystems, Inc.
#
#ident  "@(#)arp.conf  1.4      93/06/03  SMI"

name="arp" parent="pseudo" instance=0;

```

Example 9: Configuring virtual provisioning

In the following virtual provisioning example, a command file (`thin_config_file`) is used with the `symconfigure` command to create the environment.

```
symconfigure -sid 343 -f thin_config_file -v -nop commit
```

The file `thin_config_file` contained the commands for `symconfigure`.

Create a pool

The following command file creates an empty thin pool:

```
create pool LARGE type = thin;
```

The pool `LARGE` can be viewed using the following command:

```
symcfg -sid 343 list -pools -mb
```

Symmetrix ID: 000190300343

S Y M M E T R I X P O O L S									
Pool Name	Type	Device Emulation	Device Configuration	Total MBs	Enabled MBs	Used MBs	Free MBs	Full (%)	Status
DEFAULT_POOL	SN	FBA	Mixed	103574	103574	0	103574	0	Ena
DEFAULT_POOL	SN	3390	Unknown	0	0	0	0	0	Dis
DEFAULT_POOL	SN	3380	Unknown	0	0	0	0	0	Dis
DEFAULT_POOL	SN	AS400	Unknown	0	0	0	0	0	Dis
xt_tp_2way	TH	FBA	2-Way Mir	1440	1440	938	501	65	Ena
tp_pool	TH	FBA	2-Way Mir	34525	34525	0	34524	0	Ena
tp_pool2	TH	FBA	2-Way Mir	34525	17262	90	17172	0	Ena
HR_THIN_R5	TH	FBA	RAID-5 (7+1)	3360	3360	309	3050	9	Ena
LARGE	TH	N/A	Unknown	0	0	0	0	0	Dis
Total MBs				177425	160162	1339	158822	0	

Legend for Pool Types:

SN = Snap, RS = Rdfa DSE TH = Thin

Create DATA devices

The following command file creates 200 DATA devices:

```
create dev count = 200
size = 1 GB,
emulation=fba,
config=RAID-6,
data_member_count=3,
attribute=datadev;
```

The file is then committed with the `symconfigure` command.

```
symconfigure -sid 343 -f thin_config_file -v -nop commit
```

A Configuration Change operation is in progress. Please wait...

```
Establishing a configuration change session.....Established.
Processing symmetrix 000190300343
{
    create dev count=200, size=1093, emulation=FBA,
    config=RAID-6, data_member_count=3, mvs_ssid=0000, attribute=datadev;
}
```

```
Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Submitting configuration changes.....Submitted
Validating configuration changes.....Validated.
```

```
New symdevs: 032F:03F6
Initiating PREPARE of configuration changes.....Prepared.
Initiating COMMIT of configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 020 of 078 steps.....Executing.
Step 041 of 151 steps.....Executing.
Step 065 of 151 steps.....Executing.
Step 068 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 078 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 093 of 173 steps.....Executing.
Step 098 of 173 steps.....Executing.
Step 099 of 173 steps.....Executing.
Step 099 of 173 steps.....Executing.
Step 107 of 173 steps.....Executing.
Step 109 of 173 steps.....Executing.
Step 115 of 173 steps.....Executing.
Step 118 of 173 steps.....Executing.
Step 125 of 173 steps.....Executing.
Step 126 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 127, substep 1.....Executing.
Step 127, substep 1.....Executing.
Step 127, substep 1.....Executing.
Step 127, substep 2.....Executing.
Step 127, substep 2.....Executing.
Step 127, substep 2.....Executing.
Local: COMMIT.....Done.
Terminating the configuration change session.....Done.
```

The configuration change session has successfully completed.

The created DATA devices 032F to 03F6 can be viewed using the `symdev list` or the `symcfg list` commands:

symdev -sid 343 list -RANGE 32D:3F6 -all

Symmetrix ID: 000190300343

Device Name		Directors		Device			Cap
Sym	Physical	SA :P DA :IT	Config	Attribute	Sts		(MB)
032D	Not Visible	???:? 15B:CD	Unprotected	N/Grp'd	RW		2063
032E	Not Visible	???:? 02A:C5	Unprotected	N/Grp'd	RW		2063
032F	Not Visible	???:? 01A:CC	RAID-6	N/A (DT)	RW		1025
0330	Not Visible	???:? 01A:D7	RAID-6	N/A (DT)	RW		1025
..							
03F4	Not Visible	???:? 01A:D7	RAID-6	N/A (DT)	RW		1025
03F5	Not Visible	???:? 02A:CB	RAID-6	N/A (DT)	RW		1025
03F6	Not Visible	???:? 02A:D6	RAID-6	N/A (DT)	RW		1025

symcfg -sid 343 list -datadevs -mb

Symmetrix ID: 000190300343

S Y M M E T R I X D A T A D E V I C E S								
Sym	Dev Emul	Dev Config	Pool Type	Pool Name	State	Total MBs	Used MBs	Full (%)
01C9	FBA	2-Way Mir	TH	tp_pool2	Drn	17262	0	0
01CA	FBA	2-Way Mir	TH	tp_pool2	Ena	17262	90	0
01CB	FBA	2-Way Mir	TH	tp_pool	Ena	17262	0	0
01CC	FBA	2-Way Mir	TH	tp_pool	Ena	17262	0	0
01FB	FBA	2-Way Mir	-	-	Dis	468	0	0
..								
03F1	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
03F2	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
03F3	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
03F4	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
03F5	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
03F6	FBA	RAID-6(6+2)	-	-	Dis	1024	0	0
Total MBs						294770	1339	0

Legend for Pool Types:

SN = Snap, RS = Rdfa DSE, TH = Thin, - = N/A

Adding DATA devices to the pool

The following file (thin_config_file) was created to add DATA devices to the pool:

```
add dev 032F:03F6 to pool LARGE type = thin,
member_state = ENABLE;
```

The file was then committed with the `symconfigure` command:

```
symconfigure -sid 343 -f thin_config_file -v -nop commit
```

A Configuration Change operation is in progress. Please wait...

```
Establishing a configuration change session.....Established.
{
  add dev 032F:03F6 to pool LARGE, type=thin, member_state=enable;
}

Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Locking devices.....Locked.
Committing configuration changes.....Reordering.
Adding pool devs .....Done.
Enabling pool devs .....Done.
Committing configuration changes.....Committed.
Terminating the configuration change session.....Done.
```

The configuration change session has successfully completed. The pool can now be viewed, as shown in this example using the `symcfg show` command with the `-detail` and `-all` options:

```
symcfg -sid 343 -pool LARGE -thin -detail -all -mb show
```

Symmetrix ID: 000190300343

```
Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 200
Max. Subscription Percent : None
```

Enabled Devices(200):

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)   State
-----
032F      1024         0        1024    0   Enabled
0330      1024         0        1024    0   Enabled
..
..
03F5      1024         0        1024    0   Enabled
03F6      1024         0        1024    0   Enabled
-----
MBs      204900         0      204900    0
}
```

This output indicates that no thin devices are bound to the device pool `LARGE`.

Setting pool limits

Right now, thin devices totaling any capacity can be bound to the `LARGE` pool. To limit this, the pool subscription percent (`max_subs_percent=`) is set. In the next example, the pool is set to be oversubscribed by 250%, which means that the total capacity of the thin devices that are bound to this pool cannot exceed 250% of the pool capacity.

In the command file, a couple of DATA devices will also be disabled, as follows:

```
disable dev 03F5:03F6 in pool LARGE, type=thin;
set pool LARGE, type=thin, max_subs_percent=250;
```

The following display shows the `LARGE` pool after the configuration change session:

```
symcfg -sid 343 -pool LARGE -thin -detail -all -mb show
```

```
Symmetrix ID: 000190300343
```

```
Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 198
Max. Subscription Percent : 250
```

```
Enabled Devices(198):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)   State
-----
032F      1024         0        1024    0  Enabled
0330      1024         0        1024    0  Enabled
0331      1024         0        1024    0  Enabled
..
..
03F3      1024         0        1024    0  Enabled
03F4      1024         0        1024    0  Enabled
-----
MBs      202851         0      202851    0
}
```

```
Disabled Devices(2):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)   State
-----
03F5      1024         0        1024    0  Disabled
03F6      1024         0        1024    0  Disabled
-----
MBs      2049         0        2049    0
}
```

In this example, there are still no thin devices bound to the device pool `LARGE`.

Creating thin devices and binding them to the pool

The following command file creates the thin devices, and is followed by the `symconfigure` command processing the file:

```
create dev count=100,
      size=500MB,
      config=TDEV,
      emulation=FBA,
      binding to pool=LARGE;
```

```
symconfigure -sid 343 -f thin_config_file -v -nop commit
```

A Configuration Change operation is in progress. Please wait...

```
Establishing a configuration change session.....Established.
Processing symmetrix 000190300343
{
  create dev count=100, size=534, emulation=FBA,
    config=TDEV, mvs_ssid=0000,
    binding to pool=LARGE;
}

Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Submitting configuration changes.....Submitted
Validating configuration changes.....Validated.

New symdevs: 03F7:045A
Initiating PREPARE of configuration changes.....Prepared.
Initiating COMMIT of configuration changes.....Started.
Committing configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 004 of 078 steps.....Executing.
Step 020 of 078 steps.....Executing.
Step 038 of 078 steps.....Executing.
Step 062 of 151 steps.....Executing.
Step 066 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 076 of 173 steps.....Executing.
Step 078 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 097 of 173 steps.....Executing.
Step 099 of 173 steps.....Executing.
Step 099 of 173 steps.....Executing.
Step 107 of 173 steps.....Executing.
Step 109 of 173 steps.....Executing.
Step 115 of 173 steps.....Executing.
Step 124 of 173 steps.....Executing.
Step 127 of 173 steps.....Executing.
Step 173 of 173 steps.....Executing.
Local: COMMIT.....Done.
Binding devices.....Done.
Terminating the configuration change session.....Done.
```

The configuration change session has successfully completed.

The thin devices can be listed using the `symdev list`, or the `symcfg list` commands, as shown:

symdev -sid 343 list -tdev -RANGE 03F7:045A

Symmetrix ID: 000190300343

Device Name		Directors		Device			
Sym	Physical	SA :P DA :IT	Config	Attribute	Sts	Cap (MB)	
03F7	Not Visible	???:?	NA:NA	TDEV	N/Grp'd	RW	501
..							
0459	Not Visible	???:?	NA:NA	TDEV	N/Grp'd	RW	501
045A	Not Visible	???:?	NA:NA	TDEV	N/Grp'd	RW	501

symcfg -sid 343 list -tdev -mb

Symmetrix ID: 000190300343

Enabled Capacity (MBs) : 259439

Bound Capacity (MBs) : 87682

S Y M M E T R I X T H I N D E V I C E S									
Sym	Pool Name	Dev Emul	Total MBs	Pool Subs (%)	Pool Allocated MBs (%)		Pool Written MBs (%)		Status
01BF	-	FBA	17263	0	0	0	0	0	Unbound
01C0	-	FBA	17263	0	0	0	0	0	Unbound
01C1	-	FBA	17263	0	0	0	0	0	Unbound
01C2	tp_pool	FBA	17263	50	0	0	0	0	Bound
01C3	tp_pool2	FBA	17263	100	135	1	0	0	Unbinding
01C4	-	FBA	17263	0	0	0	0	0	Unbound
01C5	-	FBA	17263	0	0	0	0	0	Unbound
..									
0456	LARGE	FBA	500	0	0	0	0	0	Bound
0457	LARGE	FBA	500	0	0	0	0	0	Bound
0458	LARGE	FBA	500	0	0	0	0	0	Bound
0459	LARGE	FBA	500	0	0	0	0	0	Bound
045A	LARGE	FBA	500	0	0	0	0	0	Bound
Total MBs			618532	238	1459	1	4	0	

The pool can be viewed using the `symcfg show` command, as shown:

```
symcfg -sid 343 -pool LARGE -thin -detail -all -mb show
```

```
Symmetrix ID: 000190300343
```

```
Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 198
Max. Subscription Percent : 250
```

```
Enabled Devices(198):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)  State
-----
032F      1024         0        1024    0  Enabled
0330      1024         0        1024    0  Enabled
..
..
03F3      1024         1        1023    0  Enabled
03F4      1024         2        1022    0  Enabled
-----
MBs      202851        75      202776    0
}
```

```
Disabled Devices(2):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)  State
-----
03F5      1024         0        1024    0  Disabled
03F6      1024         0        1024    0  Disabled
-----
MBs      2049         0        2049    0
}
```

```
Thin Devices(100):
```

```
{
-----
Sym      Total      Pool      Pool      Pool
Dev      MBs        Subs      Allocated  Written
              (%)      MBs  (%)      MBs  (%)  Status
-----
03F7      500         0         0    0         0    0  Bound
03F8      500         0         0    0         0    0  Bound
..
..
0459      500         0         0    0         0    0  Bound
045A      500         0         0    0         0    0  Bound
-----
MBs      50062        25        75    0         3    0
}
```

The above display can be altered using the `-gb` option to show the values in gigabytes or it can show the number of Symmetrix tracks if no option is specified.

Binding with preallocate

The following command file creates some thin devices, binds them to the `LARGE` pool, and preallocates space for the thin devices:

```
create dev count=100,
        size=500MB,
        config=TDEV,
        emulation=FBA,
        binding to pool=LARGE,
        preallocate size=400MB;
```

The configuration change session output follows:

```
symconfigure -sid 343 -f thin_config_file -v -nop commit
```

A Configuration Change operation is in progress. Please wait...

```
Establishing a configuration change session.....Established.
Processing symmetrix 000190300343
{
    create dev count=100, size=534, emulation=FBA,
        config=TDEV, mvs_ssid=0000,
        binding to pool=LARGE, preallocate size =427 cyl;
}

Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Submitting configuration changes.....Submitted
Validating configuration changes.....Validated.

New symdevs: 045B:04BE
Initiating PREPARE of configuration changes.....Prepared.
Initiating COMMIT of configuration changes.....Started.
Committing configuration changes.....Queued.
COMMIT requesting required resources.....Obtained.
Step 020 of 078 steps.....Executing.
Step 037 of 078 steps.....Executing.
Step 061 of 151 steps.....Executing.
Step 066 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 071 of 151 steps.....Executing.
Step 076 of 173 steps.....Executing.
Step 078 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 082 of 173 steps.....Executing.
Step 097 of 173 steps.....Executing.
Step 098 of 173 steps.....Executing.
Step 099 of 173 steps.....Executing.
Step 104 of 173 steps.....Executing.
Step 104 of 173 steps.....Executing.
Step 113 of 173 steps.....Executing.
Step 119 of 173 steps.....Executing.
Step 124 of 173 steps.....Executing.
Step 132 of 173 steps.....Executing.
Local: COMMIT.....Done.
Binding devices.....Done.
Allocating devices.....Started.
Terminating the configuration change session.....Done.
```

The configuration change session has successfully completed.

The result of this create, bind, and preallocate action can be seen by looking at the pool, as follows:

symcfg -sid 343 -pool LARGE -thin -detail -all -mb show

Symmetrix ID: 000190300343

```
Symmetrix ID      : 000190300343
Pool Name         : LARGE
Pool Type         : Thin
Dev Emulation     : FBA
Dev Configuration : RAID-6(6+2)
Pool State        : Enabled
# of Devices in Pool : 200
# of Enabled Devices in Pool : 198
Max. Subscription Percent : 250
```

Enabled Devices(198):

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)  State
-----
032F      1024        189        835   18  Enabled
0330      1024        202        822   19  Enabled
0331      1024        168        855   16  Enabled
0332      1024        199        825   19  Enabled
0333      1024        195        828   19  Enabled
..
..
03F3      1024        219        805   21  Enabled
03F4      1024        192        831   18  Enabled
-----
MBs      202851      40200      162651  19
}
```

Disabled Devices(2):

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      MBs        MBs        MBs  (%)  State
-----
03F5      1024          0       1024    0  Disabled
03F6      1024          0       1024    0  Disabled
-----
MBs      2049          0       2049    0
}
```

Thin Devices(200):

```
{
-----
Sym      Total      Pool      Pool      Pool
Dev      MBs        Subs      Allocated  Written
              (%)      MBs    (%)      MBs    (%)  Status
-----
03F7      500         0         0      0         0      0  Bound
03F8      500         0         0      0         0      0  Bound
..
..
0458      500         0         0      0         0      0  Bound
0459      500         0         0      0         0      0  Bound
045A      500         0         0      0         0      0  Bound
045B      500         0        401     0         0      0  Bound
045C      500         0        401     0         0      0  Bound
045D      500         0        401     0         0      0  Bound
}
```



```

..
..
04BD          500      0      401      0      0      0      Bound
04BE          500      0      401      0      0      0      Bound
-----
MBs          100125    49      40200    20      8      0
}

```

Disabling DATA devices

When you disable DATA devices, enough space needs to be available in the pool to accommodate the allocation that has happened on the devices being disabled. The following command file disables two devices:

```
disable dev 032F:0330 in pool LARGE, type=thin;
```

After the configuration change session, the pool LARGE can be viewed, as follows:

```
symcfg -sid 343 -pool LARGE -thin -all -mb show
```

```
Symmetrix ID: 000190300343
```

```

Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 196

```

```
Enabled Devices(196):
```

```

{
-----
Sym      Total      Used      Free  Full  Device  Session
Dev      MBs        MBs        MBs   (%)  State   Status
-----
0331      1024        170        854   16   Enabled  N/A
0332      1024        201        822   19   Enabled  N/A
..
..
03F2      1024        225        799   21   Enabled  N/A
03F3      1024        221        803   21   Enabled  N/A
03F4      1024        195        829   19   Enabled  N/A
-----
MBs      200802      40200      160602  20
}

```

```
Disabled Devices(4):
```

```

{
-----
Sym      Total      Used      Free  Full  Device  Session
Dev      MBs        MBs        MBs   (%)  State   Status
-----
032F      1024        189        835   18   Draining N/A
0330      1024         0        1024    0   Draining N/A
03F5      1024         0        1024    0   Disabled N/A
03F6      1024         0        1024    0   Disabled N/A
-----
MBs       4098        189        3909    4
}

```

Allocating space for thin devices

Space can be allocated for thin devices after they have been bound. The following command file allocates space for thin devices 03FF to 045A:

```
allocate tdev 03FF:045A start_cyl = 0 end_cyl = 500;
```

The following output is shown after the configuration change session:

```
symcfg -sid 343 -pool LARGE -thin -all -mb show
```

```
Symmetrix ID: 000190300343
```

```
Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State             : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 196
Max. Subscription Percent : 250
```

```
Enabled Devices(196):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks (%)  State
-----
0331      16392      6792      9600   41   Enabled
..
..
```

```
Disabled Devices(4):
```

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks (%)  State
-----
032F      16392         0      16392    0   Disabled
0330      16392         0      16392    0   Disabled
03F5      16392         0      16392    0   Disabled
03F6      16392         0      16392    0   Disabled
-----
Tracks      65568         0      65568    0
}
```

```
Thin Devices(200):
```

```
{
-----
Sym      Total      Pool      Pool      Pool
Dev      Tracks     Subs      Allocated  Written
              (%)   Tracks   (%)   Tracks   (%)  Status
-----
03F7      8010         0         12      0         1      0   Bound
03F8      8010         0         12      0         1      0   Bound
03F9      8010         0         12      0         1      0   Bound
03FA      8010         0         12      0         1      0   Bound
03FB      8010         0         12      0         1      0   Bound
03FC      8010         0         12      0         1      0   Bound
03FD      8010         0         12      0         1      0   Bound
03FE      8010         0         12      0         1      0   Bound
03FF      8010         0        7536      0         1      0   Bound
0400      8010         0        7536      0         1      0   Bound
0401      8010         0        7536      0         1      0   Bound
0402      8010         0        7536      0         1      0   Bound
0403      8010         0        7536      0         1      0   Bound
}
```

```

..
..
04BD          8010      0      6480      0          1      0 Bound
04BE          8010      0      6528      0          1      0 Bound
-----
Tracks      1602000    50    1341672    42          179    0
}

```

Freeing space on thin devices

Space that has been allocated but not written to, can be freed on a thin device. If a user allocated or preallocated a large amount of space for a thin device, but is not actually consuming that space, this is the way that the unused space can be freed. The following command file frees space on devices 03FF to 045A:

```
free tdev 03FF:045A start_cyl = 0 end_cyl = last_cyl;
```

The output while the deallocation is in progress looks as follows:

```
symcfg -sid 343 -pool LARGE -thin -all -mb show
```

```
Symmetrix ID: 000190300343
```

```

Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State              : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 196
Max. Subscription Percent : 250

```

```
Enabled Devices(196):
```

```

{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks (%)  State
-----
0331      16392      5580      10812   34  Enabled
..
..
03F4      16392      6348      10044   38  Enabled
-----
Tracks    3212832    1104756    2108076   34
}

```

```
Disabled Devices(4):
```

```

{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks (%)  State
-----
032F      16392          0      16392    0  Disabled
..
..
03F6      16392          0      16392    0  Disabled
-----
Tracks    65568          0      65568    0
}

```

```
Thin Devices(200):
```

```
{
```

```

-----
Sym      Total      Pool      Pool      Pool
Dev      Tracks     Subs     Allocated  Written
              (%)    Tracks    (%)    Tracks   (%)   Status
-----
03F7      8010       0        12        0         1     0   Bound
03F8      8010       0        12        0         1     0   Bound
03F9      8010       0        12        0         1     0   Bound
03FA      8010       0        12        0         1     0   Bound
03FB      8010       0        12        0         1     0   Bound
03FC      8010       0        12        0         1     0   Bound
03FD      8010       0        12        0         1     0   Bound
03FE      8010       0        12        0         1     0   Bound
03FF      8010       0       4092        0         1     0 Dealloc'ing
0400      8010       0       4476        0         1     0 Dealloc'ing
0401      8010       0       4404        0         1     0 Dealloc'ing
..
..
04BE      8010       0       6528        0         1     0   Bound
-----
Tracks    1602000    50    1069296    33        179    0
}

```

After the space is freed, the output looks as follows:

symcfg -sid 343 -pool LARGE -thin -all -mb show

Symmetrix ID: 000190300343

```

Symmetrix ID           : 000190300343
Pool Name              : LARGE
Pool Type              : TH
Dev Emulation          : FBA
Dev Configuration      : RAID-6(6+2)
Pool State              : Enabled
# of Devices in Pool   : 200
# of Enabled Devices in Pool : 196
Max. Subscription Percent : 250

```

Enabled Devices(196):

```

{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks    Tracks    (%)   State
-----
0331      16392       5580      10812    34   Enabled
..
..
03F4      16392       6348      10044    38   Enabled
-----
Tracks    3212832    1104756    2108076    34
}

```

Disabled Devices(4):

```

{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks    Tracks    (%)   State
-----
032F      16392         0      16392     0   Disabled
..
..
03F6      16392         0      16392     0   Disabled
-----
Tracks    65568         0      65568     0
}

```

```

}

Thin Devices(200):
{
-----
Sym          Total    Pool      Pool      Pool
Dev          Tracks   Subs      Allocated  Written
              (%)    Tracks    (%)      Tracks  (%)   Status
-----
03F7          8010      0         12        0         1      0   Bound
03F8          8010      0         12        0         1      0   Bound
03F9          8010      0         12        0         1      0   Bound
03FA          8010      0         12        0         1      0   Bound
03FB          8010      0         12        0         1      0   Bound
03FC          8010      0         12        0         1      0   Bound
03FD          8010      0         12        0         1      0   Bound
03FE          8010      0         12        0         1      0   Bound
03FF          8010      0         12        0         1      0   Bound
0400          8010      0         12        0         1      0   Bound
0401          8010      0         12        0         1      0   Bound
0402          8010      0         12        0         1      0   Bound
..
..
04BD          8010      0        6480      0         1      0   Bound
04BE          8010      0        6528      0         1      0   Bound
-----
Tracks       1602000    50      649488    20         179    0
}

```

Example 10: Automatically creating metadevices

Note: To use this feature, both the local and the remote Symmetrix arrays must be running Enginuity V5773 or later and the local Symmetrix array must have `auto_meta` enabled.

The command below creates an AS400 metadevice on a Symmetrix array running Enginuity 5773. The device emulation is AS/400_M4328_50.

```
create dev count=1, emulation= AS/400_M4328_50, config=2-Way-Mir;
```

This creates an eight-member metadevice where total cylinder size is 143576 (native mode).

The following is example output:

```
symcfg list -sid 343 -v

Symmetrix ID: 000190300343
Time Zone    : EDT

Product Model           : DMX3-6
Symmetrix ID           : 000190300343

Microcode Version (Number) : 5773 (168D0000)
Microcode Date          : 10.26.2007

Microcode Patch Date     : 10.26.2007
Microcode Patch Level    : 30
.....
Cache Size (Mirrored)    : 32768 (MB)
# of Available Cache Slots : 950456
Max # of System Write Pending Slots : 761106
Max # of DA Write Pending Slots : 380552
Max # of Device Write Pending Slots : 38050

Symmetrix Total Operating Time : 17 days, 03:20:51
Symmetrix Power ON Time       : Fri Oct 12 09:35:59 2007
Symmetrix Last IPL Time (Cold) : Tue Oct 23 12:12:05 2007
Symmetrix Last Fast IPL Time (Hot) : Fri Oct 26 16:40:26 2007

Host DB Sync Time          : Mon Oct 29 12:56:50 2007
Symmetrix CLI (SYMCLI) Version : X6.5.0.160 (Edit Level: 862)
Built with SYMAPI Version   : X6.5.0.160 (Edit Level: 862)
SYMAPI Run Time Version     : X6.5.0.160 (Edit Level: 862)
SYMAPI Server Version       : unavailable
.....
SRDF/A Maximum Host Throttle (Secs) : 0
SRDF/A Maximum Cache Usage (Percent) : 94

Auto MetaCreation          : Disabled
Minimum Auto Meta Size     : 65520
Auto Meta Member size      : 1712 cyl
Auto Meta Config           : striped
```

Virtual LUN Migration Example

This appendix provides an example of a Symmetrix Virtual LUN migration.

- ◆ [Virtual LUN migration example](#) 344

Virtual LUN migration example

This section provides a step-by-step `symmigrate` example. It walks through each step that is involved in a single migration session.

In this example, a device group will be used as the source. The source devices are RAID5 (3+1) devices that are specified as a TGT list of devices in a device group named `mydg`.

Device group `mydg` has the following TGT devices: 0020, 0021 with 1439 MBs in size and the protection type is RAID5 (3+1). The disk group number is 4.

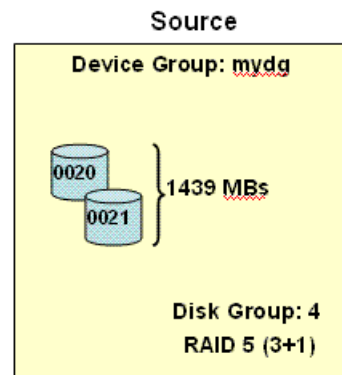


Figure 30 Device view prior to migration session

The following `validate` command will validate the `symmigrate` request and generate the device to device pairs in the output file named `mydgtosgout`:

```
symmigrate validate -nop -v -outfile mydgtosgout
  -g mydg -tgt
  -tgt_raid6 -protection 6+2 -tgt_dsk_grp 1
  -tgt_config
  -name myfullsession
```

'Validate' operation execution is in progress for the device group - `mydg`. Please wait...

STARTING a Migrate 'Validate' operation.

SELECTING the list of Source devices

Devices 0020, 0021 [SELECTED]

The Migrate 'Validate' operation SUCCEEDED.

'Validate' operation successfully executed for the device group - `mydg`.

The following command shows the validated device-to-device pairs in the output file:

```
more mydgtosgout
```

```
0020 0075
0021 0086
```


After the `validate` command, use the `establish` command to establish the migration session by using the validated output file `mydgtosgout`.

```
symmigrate establish -sid 123 -file mydgtosgout -nop -v  
-name myfullsession
```

'Establish' operation execution is in progress for the device list in device file 'mydgtosgout'. Please wait...

STARTING a Migrate 'Establish' operation.

PAIRING of Source and Target Devices:

Devices: 0020-0075, 0021-0086

The Migrate 'Establish' operation SUCCEEDED.

'Establish' operation successfully executed for the device list in device file 'mydgtosgout'.

For the preceding command, the following audit log messages will be generated:

STARTING a Migrate 'ESTABLISH' operation for Device Pairs. Options=(RAID6_6+2, DiskGroup 1, Configured Space)

Symm 000192600123 Source Devices: [0020 0021] Target Devices: [0075 0086]

The Migrate 'ESTABLISH' operation SUCCESSFULLY COMPLETED.

The following `symmigrate list` command will list all the existing migration sessions on this Symmetrix array:

```
symmigrate list -sid 123
```

Symmetrix ID: 000192600123

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0010	N/A	0	Migrated	100	Migrate1
0014	N/A	0	Migrated	100	Migrate1
0020	0075	23024	SyncInProg	50	myfullsession
0021	0086	32209	SyncInProg	30	myfullsession
Total -----					
Tracks		55233			
MB(s)		1726.0			

After the `list` command, use the `symmigrate query` command with the `-detail` option to query the `myfullsession` migration session. This provides more detailed information about this session:

```
symmigrate query -name myfullsession -sid 123 -detail
```

Symmetrix ID: 000192600123

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	TGT Dsk Grp	TGT Protection	Session Name
0020	0075	18420	SyncInProg	60	01	RAID-6(6+2)	myfullsession
0021	0086	27628	SyncInProg	40	01	RAID-6(6+2)	myfullsession
Total -----							
		Tracks	46048				
		MB(s)	1439.0				

The `symmigrate query` command with the `-i` option can also be used to monitor the progress of this migration session:

```
symmigrate query -name myfullsession -sid 123 -i 5 -c 2
```

Symmetrix ID: 000192600123

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0020	0075	13814	SyncInProg	70	myfullsession
0021	0086	23024	SyncInProg	50	myfullsession
Total -----					
		Tracks	36834		
		MB(s)	1151.1		

Copy rate : 4.7 MB/S
Estimated time to completion : 00:02:37

Symmetrix ID: 000192600123

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0020	0075	13010	SyncInProg	72	myfullsession
0021	0086	22310	SyncInProg	52	myfullsession
Total -----					
		Tracks	35320		
		MB(s)	1103.8		

Copy rate : 4.7 MB/S
Estimated time to completion : 00:02:30

As the migration session is in `SyncInProg` state, the following `symdev show` command will display the device RAID group information for both the source and the target devices. Both the primary and the secondary RAID group will be shown for the source device and only the original primary RAID group will be shown for the target device.

The following is the device show output for source device 0020:

symdev show 0020 -sid 123

```

Device Physical Name      : Not Visible
Device Symmetrix Name    : 0020
.
.
.
Device Configuration     : RAID-5              (Non-Exclusive Access)
Device is WORM Enabled   : No
Device is WORM Protected : No

SCSI-3 Persistent Reserve: Disabled

Dynamic Spare Invoked    : No
Dynamic RDF Capability    : RDF1_OR_RDF2_Capable

STAR Mode                : No
STAR Recovery Capability : None
STAR Recovery State      : NA

Device Service State     : Normal

Device Status            : Ready              (RW)
Device SA Status         : N/A                (N/A)

Mirror Set Type          : [RAID-5,RAID-6,N/A,N/A]
Mirror Set DA Status     : [RW,RW,N/A,N/A]
Mirror Set Inv. Tracks   : [0,0,0,0]

Back End Disk Director Information
{
  Hyper Type              : RAID-5
  Hyper Status            : Ready              (RW)
  Disk [Director, Interface, TID] : [N/A,N/A,N/A]
  Disk Director Volume Number : N/A
  Hyper Number            : N/A
  Mirror Number           : 1

  Hyper Type              : RAID-6
  Hyper Status            : Ready              (RW)
  Disk [Director, Interface, TID] : [N/A,N/A,N/A]
  Disk Director Volume Number : N/A
  Hyper Number            : N/A
  Mirror Number           : 2
}

RAID Group Information
{
  Mirror Number           : 1
  RAID Type               : RAID-5
  Protection Level        : 3+1
  Device Position         : Primary
  RAID Group Service State : Normal
  Failing Member Mask     : N/A
  Hyper Devices:
  {
    Device : 0020
    {

```

```

-----
Disk      DA      Hyper      Member      Disk
DA :IT    Vol#    Num Cap(MB)  Num Status  Grp#  Cap(MB)
-----
16B:Dd    841     18   2878     1  RW        4   140014
01C:D9    720     18   2878     2  RW        4   140014
16D:C4    137     18   2878     3  RW        4   140014
01A:Ce    460     18   2878     4  RW        4   140014
}
}

Mirror Number          : 2
RAID Type              : RAID-6
Protection Level       : 6+2
Device Position        : Secondary
RAID Group Service State : Normal
Failing Member Mask    : N/A
Hyper Devices:
{
  Device : 0020
  {
    -----
    Disk      DA      Hyper      Member      Disk
    DA :IT    Vol#    Num Cap(MB)  Num Status  Grp#  Cap(MB)
    -----
    01A:C4    176     55   1439     1  RW        1   140014
    16B:D7    674     55   1439     2  RW        1   140014
    16C:D6    678     55   1439     3  RW        1   140014
    01D:C7    224     55   1439     4  RW        1   140014
    16A:D6    682     55   1439     5  RW        1   140014
    01B:C7    244     55   1439     6  RW        1   140014
    01C:C8    275     54   1439     7  RW        1   140014
    16D:Dd    861     54   1439     8  RW        1   140014
  }
}
}

```

The following is the device show output for target device 0075:

symdev show 0075 -sid 123

```

Device Physical Name      : Not Visible
Device Symmetrix Name     : 0075
.
.
.
Device Configuration      : RAID-6              (Non-Exclusive Access)
Device is WORM Enabled    : No
Device is WORM Protected  : No
SCSI-3 Persistent Reserve: Disabled
Dynamic Spare Invoked     : No
Dynamic RDF Capability     : RDF1_OR_RDF2_Capable
STAR Mode                 : No
STAR Recovery Capability   : None
STAR Recovery State       : NA
Device Service State      : Normal
Device Status             : Not Ready          (NR)

```

```

Device SA Status      : N/A                (N/A)

Mirror Set Type       : [RAID-6,N/A,N/A,N/A]

Mirror Set DA Status  : [NR,N/A,N/A,N/A]

Mirror Set Inv. Tracks : [0,0,0,0]

Back End Disk Director Information
{
  Hyper Type                : RAID-6
  Hyper Status              : Not Ready          (NR)
  Disk [Director, Interface, TID] : [N/A,N/A,N/A]
  Disk Director Volume Number : N/A
  Hyper Number              : N/A
  Mirror Number             : 1
}

RAID Group Information
{
  RAID Type                  : RAID-6
  Device Position          : Primary
  RAID Group Number        : 361
  RAID Group Capacity (MB) : 8631
  RAID Group Ready State   : Not Ready
  RAID Group WriteProtect State : Enabled
  Failing Member Mask     : N/A
  RAID-6 Hyper Devices (6+2):
  {
    Device : 0FA8
    {
      -----
      

| Disk   | DA   | Hyper        | Member     | Disk          |
|--------|------|--------------|------------|---------------|
| DA :IT | Vol# | Num Cap (MB) | Num Status | Grp# Cap (MB) |
| 01A:C4 | 176  | 55 1439      | 1 NR       | 1 140014      |
| 16A:D6 | 682  | 55 1439      | 5 NR       | 1 140014      |
| 01B:C7 | 244  | 55 1439      | 6 NR       | 1 140014      |
| 16B:D7 | 674  | 55 1439      | 2 NR       | 1 140014      |
| 01C:C8 | 275  | 54 1439      | 7 NR       | 1 140014      |
| 16C:D6 | 678  | 55 1439      | 3 NR       | 1 140014      |
| 01D:C7 | 224  | 55 1439      | 4 NR       | 1 140014      |
| 16D:Dd | 861  | 54 1439      | 8 NR       | 1 140014      |


      -----
    }
  }
}

```

The following `symmmigrate query` command shows that the migration operation is finished and the migration session is in the Migrated state:

```
symmmigrate query -name myfullsession -sid 123
```

```
Symmetrix ID: 000192600123
```

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0020	0075	0	Migrated	100	myfullsession
0021	0086	0	Migrated	100	myfullsession
Total -----					
Tracks		0			
MB(s)		0			

When the migration session is in the Migrated state, use the `symmmigrate terminate` command to remove the session, as shown:

```
symmigrate terminate -sid 123 -name myfullsession -nop
```

'Terminate' operation execution is in progress for migration session
- myfullsession. Please wait...

'Terminate' operation successfully executed for migration session
- myfullsession.

For the preceding command, the following audit log messages are generated:

STARTING a Migrate 'TERMINATE' operation for Session Name. Options=(None)

Symm 000192600123 Session Name: [myfullsession]

The Migrate 'TERMINATE' operations SUCCESSFULLY COMPLETED.

When the migration is complete, the devices look as shown in

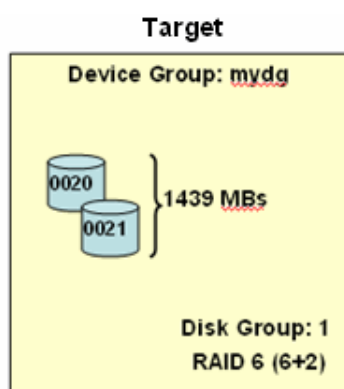


Figure 31 Device view after the migration session

The following is the output for `symmigrate list` command after the `terminate` command (myfullsession no longer displays):

```
symmigrate list -sid 123
```

Symmetrix ID: 000192600123

Src	Tgt	Invalid Tracks	Status SRC => TGT	Done (%)	Session Name
0010	N/A	0	Migrated	100	Migrate1
0014	N/A	0	Migrated	100	Migrate1
Total -----					
Tracks		0			
MB(s)		0			

The following `symdev show` command displays the RAID group information for both the source and the target device after the migration has finished:

```
symdev show 0020 -sid 123
```

Device Physical Name : Not Visible

Device Symmetrix Name : 0020

.

```

.
.
Device Configuration      : RAID-6                (Non-Exclusive Access)

Device is WORM Enabled    : No
Device is WORM Protected : No

SCSI-3 Persistent Reserve: Disabled

Dynamic Spare Invoked     : No

Dynamic RDF Capability     : RDF1_OR_RDF2_Capable

STAR Mode                 : No
STAR Recovery Capability   : None
STAR Recovery State       : NA

Device Service State      : Normal

Device Status             : Ready                    (RW)
Device SA Status          : N/A                      (N/A)

Mirror Set Type           : [RAID-6,N/A,N/A,N/A]

Mirror Set DA Status      : [RW,N/A,N/A,N/A]

Mirror Set Inv. Tracks    : [0,0,0,0]

Back End Disk Director Information
{
  Hyper Type              : RAID-6
  Hyper Status            : Ready                    (RW)
  Disk [Director, Interface, TID] : [N/A,N/A,N/A]
  Disk Director Volume Number : N/A
  Hyper Number            : N/A
  Mirror Number           : 1
}

RAID Group Information
{
  Mirror Number           : 1
  RAID Type              : RAID-6
  Device Position         : Primary
  RAID Group Number       : 361
  RAID Group Capacity (MB) : 8631
  RAID Group Ready State   : Not Ready
  RAID Group WriteProtect State : Enabled
  Failing Member Mask     : N/A
  RAID-6 Hyper Devices (6+2):
  {
    Device : 0FA8
    {
      -----
      Disk    DA      Hyper    Member    Disk
      DA :IT   Vol#   Num Cap(MB) Num Status Grp#   Cap(MB)
      -----
      01A:C4   176    55    1439    1 RW      1    140014
      16A:D6   682    55    1439    5 RW      1    140014
      01B:C7   244    55    1439    6 RW      1    140014
      16B:D7   674    55    1439    2 RW      1    140014
      01C:C8   275    54    1439    7 RW      1    140014
      16C:D6   678    55    1439    3 RW      1    140014
      01D:C7   224    55    1439    4 RW      1    140014
      16D:Dd   861    54    1439    8 RW      1    140014
    }
  }
}

```

}

symdev show 0075 -sid 123

```

Device Physical Name      : Not Visible

Device Symmetrix Name     : 0075
.
.
.
Device Configuration    : RAID-5                (Non-Exclusive Access)

Device is WORM Enabled    : No
Device is WORM Protected  : No

SCSI-3 Persistent Reserve: Disabled

Dynamic Spare Invoked     : No

Dynamic RDF Capability     : RDF1_OR_RDF2_Capable

STAR Mode                 : No
STAR Recovery Capability   : None
STAR Recovery State       : NA

Device Service State      : Normal

Device Status             : Ready                (RW)
Device SA Status          : N/A                  (N/A)

Mirror Set Type           : [RAID-5,N/A,N/A,N/A]

Mirror Set DA Status      : [RW,N/A,N/A,N/A]

Mirror Set Inv. Tracks    : [0,0,0,0]

```

```

Back End Disk Director Information
{
  Hyper Type               : RAID-5
  Hyper Status             : Ready                (RW)
  Disk [Director, Interface, TID] : [N/A,N/A,N/A]
  Disk Director Volume Number : N/A
  Hyper Number             : N/A
  Mirror Number            : 1
}

```

```

RAID Group Information
{
  Mirror Number            : 1
  RAID Type                : RAID-5
  Device Position          : Primary
  RAID Group Number        : 122
  RAID Group Capacity (MB) : 8631
  RAID Group Ready State   : Ready
  RAID Group WriteProtect State : Enabled
  Failing Member Mask      : N/A
  RAID-5 Hyper Devices (3+1):
  {
    Device : 05B1
    {
      -----
      Disk    DA      Hyper      Member      Disk
      DA :IT   Vol#    Num Cap(MB)  Num Status  Grp#   Cap(MB)
      -----
      01A:Ce   460     18      2878      4  RW           4   140014
    }
  }
}

```



```
        16B:Dd      841    18    2878    1 RW        4    140014
        01C:D9      720    18    2878    2 RW        4    140014
        16D:C4      137    18    2878    3 RW        4    140014
    }
}
}
```

After you reconfigure a Symmetrix array by moving, deleting, adding, or modifying one or more devices, you must update the host so that the host recognizes the new Symmetrix configuration.

For some platforms, the `symcfg scan` command is available to perform the host update. These include Sun Solaris, HP-UX, IBM AIX, Tru64/OSF1, and Windows systems.

This appendix describes the procedures for these host systems:

◆ Sun Solaris systems	356
◆ HP-UX systems	357
◆ IBM AIX systems	358
◆ HP Tru64 UNIX systems	359
◆ Windows systems	360

Sun Solaris systems

Whenever devices are newly mapped to a host system, or device channel addresses are changed, you need to perform the following actions to introduce the new devices to the system.

To add new Symmetrix devices while online in the Solaris environment, perform the following steps:

1. Execute the following utilities:

```
drvconfig or symcfg scan  
disks  
devlinks
```

2. Follow the instructions in the Solaris documentation to introduce new devices to the host environment.

HP-UX systems

Whenever devices are newly mapped to a host system or device channel addresses are changed, you need to perform the following actions to introduce the new devices to the system.

To view the results of the mapping changes, use the `ioscan` command in a statement similar to the following:

```
ioscan -fnC disk
```

You can define newly connected physical volumes to the HP-UX host system without rebooting it, using the following form:

```
insf -e
```

For more information, refer to the HP 9000 documentation.

IBM AIX systems

Whenever devices are newly mapped to a host system or device channel addresses are changed, you need to perform the following actions to introduce the new devices to the AIX system:

1. From the **SMIT** menu, select **Devices > Fixed Disk > Add a Disk**.
2. Select the **EMC SYMMETRIX** definition from the disk table.
3. Select the SCSI bus on which the new disk resides.
4. Type the connection address for the new device (target, LUN).
5. Select **EXECUTE**.
6. Repeat steps 2 through 5 for each new device being added to the configuration.

HP Tru64 UNIX systems

Whenever devices are newly mapped to a host system or device channel addresses are changed, you need to perform the following actions to introduce the new devices to the system.

When using Tru64 UNIX, you can introduce the new devices to the system with these steps:

1. At the prompt, type:

```
scsimgr -scan_bus bus=BUSNUM
```

2. Repeat for each LUN:

- a. Write a label to the device you are defining:

```
disklabel -rw rz<lun_letter><unitID> <label>
```

- b. Change the ownership on the device to a particular application:

```
chown <owner>:<group> *rz<lun letter><unitID>*
```

3. Follow the host documentation to introduce new devices to the host environment.

Windows systems

Whenever devices are newly mapped to a host system or device channel addresses are changed, you need to perform the following actions to introduce the new devices to the system.

Windows 2000

To add new Symmetrix devices to a Windows 2000 system while the system remains online, perform these steps:

1. From the desktop, select **Start, Settings, Control Panel, Add /Remove Hardware**. Complete the wizard to discover and add the new Symmetrix devices.
2. Partition and format the new devices as described in the *Windows 2000* chapter of the *Open Systems Host Environment Product Guide*.

Windows 2003

To add new Symmetrix devices to a Windows 2003 system while the system remains online, perform these steps:

1. From the desktop, select **Start, Settings, Control Panel, Add /Remove Hardware**. Complete the wizard to discover and add the new Symmetrix devices.
2. Partition and format the new devices as described in the Windows 2003 documentation.

This chapter describes the Symmetrix Integration Utilities (SIU) disk management operations for your storage system in a Windows Server environment.

Topics include:

◆ Introduction	362
◆ Configuring a VMware virtual server	365
◆ Viewing detailed information about disks	366
◆ Mounting and unmounting disks	367
◆ Manipulating disk signatures	369
◆ Updating partition tables	370
◆ Flushing cache buffers to disk	371
◆ Device masking operations	372
◆ Using volume flags	374
◆ Performing typical storage operations	375
◆ SIU Examples	380
◆ XML EXAMPLES	382

Introduction

The Symmetrix Integration Utilities (SIU) integrates and extends the Windows 2003 and Windows 2008 disk management functionality to better operate with and on TimeFinder and Symmetrix Remote Data Facilities (SRDF) devices.

SIU operations integrate with the management functionality of TimeFinder and SRDF with support specifically for Windows servers by extending their ability to:

- ◆ Mount and unmount hard disks and their associated file systems.
- ◆ Flush file system cache buffers to disk.
- ◆ Manipulate disk signatures.

SIU is not a replacement for the Windows Logical Disk Manager (LDM), but bridges these lacking functionalities necessary for Windows users to optimally work with EMC business continuance storage devices.

SIU capabilities

SIU enables you to perform the following actions:

- ◆ View the physical disk, volume, and VMware datastore configuration data.
- ◆ Update the partition table on a disk.
- ◆ Set and clear volume flags.
- ◆ Flush any pending cached file system data to disk.
- ◆ Show individual disk, volume, or VMware datastore details.
- ◆ Mount and unmount volumes to a drive letter or mount point.
- ◆ Manipulate disk signatures.
- ◆ Scan the drive connections and discovers any new disks available to the system.
- ◆ Mask devices to and unmask devices from the Windows host.

Privilege requirements

SIU is based on the functionality and features of associated hardware and software in an EMC Enterprise Storage environment managed and used on Windows 2003 and Windows 2008 platforms. Most command actions require administrative privileges. On Windows 2008, you must be a local administrator, or the command prompt must be `Run as administrator`.

Note: For security reasons, Windows Server 2008 does not honor the `Start In` path property of the shortcut `Run as administrator`. Therefore, the shortcut will always begin in the Windows system directory, instead of the SIU directory.

SIU commands

The SIU CLI commands are a syntactical set and form of one command (`symntctl`), which has various arguments to support each of the main functional operations or actions. A summary of the SIU command set that operate in a Windows operating

system environment is listed in [Table 24](#). These commands can be used in scripts when working on Windows 2003 or Windows 2008 platforms

Table 24 SIU command summary

Command	Argument	Action
symmctl		Performs disk management operations for Windows platforms to properly work with Business Continuity devices. Use with the following arguments for other operations.
	flag	Sets and clears volume flags.
	flush	Flushes all pending unwritten file system cache entries to disk.
	list	Lists all visible physical disks, volumes, and VMware datastores.
	mask	Removes access to the specified Symmetrix device on all HBAs of the host where the specified device is visible.
	mount	Mounts the specified volume to the specified mount point or drive letter.
	rescan	Scans the device connections and discovers any new disks available to the system.
	show	Displays individual disk, volume, or VMware datastore details.
	signature	Manipulates disk signatures.
	umount	Unmounts a volume from the drive letter and all mount points.
	unmask	Adds access to the specified Symmetrix device on all HBAs of the host that is connected to the specified Symmetrix array.
	update	Updates the partition table on a disk.

For command syntax information, refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help*.

Volume management software

Windows Server 2003 and Windows Server 2008 have two types of disks: basic and dynamic. Basic disks support primary partitions only, while dynamic disks support dynamic volumes only.

The Windows Logical Disk Manager (LDM) provides full management functionality for basic disks, but only limited management functionality for dynamic disks. Most of the advanced capabilities of dynamic disks are only available with Veritas Storage Foundation for Windows (SFW).

SFW dynamic disk groups with TimeFinder

For Windows advanced server systems with dynamic disks, SIU functionality works with Veritas Storage Foundation for Windows.

If you are going to set up and use Veritas dynamic disk groups (VDGs) on a production server, you must comply with the following requirements when configuring VDGs for use with Solutions Enabler TimeFinder/Mirror software:

- ◆ Create a TimeFinder/Mirror device group for every dynamic disk group, maintaining the one-to-one basis. To keep this relationship clear, name the dynamic disk groups the same as the TimeFinder/Mirror device groups.

- ◆ It is not necessary to create the disk group on the backup server, since the Logical Disk Manager (LDM) database is resident on every dynamic disk, as it is copied over to the backup server on each of the BCVs. All disk group definitions are stored in this database and will immediately appear on the backup server (in an offline state). You can create the TimeFinder/Mirror device groups on either the production server or the backup server.

Be aware of the following when creating TimeFinder/Mirror device groups:

- Do not mix STD and BCV devices in the same VDG.
- Do not have Symmetrix and non-Symmetrix storage type devices in the same VDG.
- Do not mix R1 and R2 devices in the same VDG.
- Do not add gatekeeper devices to any VDG.

Note: SFW 5.1 on windows server 2008 is not supported.

Configuring a VMware virtual server

In the case where the SIU server is a VMware virtual server (VM), there are additional setup requirements and steps. The additional setup requirements are:

- ◆ VMware ESX Server 3.5 Update 2.
- ◆ The SPC-2 flag must be enabled on the Symmetrix ports being used.

Once these requirements are met, the following steps must be followed:

1. Install VMware Tools on the backup/recovery server.
2. Open a command prompt to the Solutions Enabler `shlib` directory which is typically located at:

`C:\Program Files\EMC\SYMCLI\shlib`
3. Run the following command, `vicfg` setup. This utility is designed to configure access to the ESX Server by enabling SSL ports on the firewall, configuring certificates, and encrypting and storing the login credentials:
 - When prompted, please enter the username and password to be used as credentials for the ESX Server. These credentials are encrypted and stored in the Solutions Enabler authorization database, so there is no concern over their security.
4. Assign at least 2 Symmetrix gatekeeper devices to the VM. These gatekeepers will be required in order for Solutions Enabler to discover the Symmetrix from the VM:
 - Verify that the proper zoning is in place to connect the HBAs in the ESX Server to the Symmetrix.
 - Using Symmetrix device masking (`symmask.exe`), assign access for the ESX Server to the gatekeeper devices.
 - Configure the VM to have Raw Device Mappings (RDM) for each of the gatekeeper devices. The VM should now see the gatekeepers in a SCSI inquiry (`syminq.exe`).
5. Run the command `symcfg discover` to verify that Solutions Enabler can discover the Symmetrix.

SIU does not automatically interoperate with the VMware high availability software such as Vmotion, or the VMware load-balancing software such as Distributed Resource Scheduler (DRS). If the VM moves to a different ESX Server, whether through manual intervention or through one of these products, the configuration and authorization needs to be performed again. In this case, please complete step 2 again to configure the VM to run on the new ESX Server. Furthermore, you will need to verify that the new ESX Server has visibility to all of the Symmetrix devices that SIU is using.

Viewing detailed information about disks

The SIU list action provides a consolidated view for displaying and managing physical disk, volume, and VMware datastore configuration data.

To view disk numbers in your storage configuration, type:

```
symntctl list -disk
```

To view the volume configuration, type:

```
symntctl list -volume
```

To view VMware datastores in the virtual infrastructure, type:

```
symntctl list -datastore
```

The `show` action provides detailed information about an individual disk, volume, or VMware datastore.

To target a specific disk enter:

```
symntctl show -pd diskN
```

where *diskN* is the disk number.

To target a specific volume enter:

```
symntctl show -vol VolName [-g VolGroup]
```

To target a specific VMware datastore enter:

```
symntctl show -datastore DatastoreName
```

Specifying the output type

SIU allows you to specify the output format for the `symntctl list` and `symntctl show` commands, as follows:

- ◆ XML_ELEMENT — Provides the output using individual XML elements.
- ◆ XML_ATTRIBUTE — Provides the output using concatenated XML attributes.

Samples of XML output are provided in [“SIU Examples” on page 380](#).

Mounting and unmounting disks

After splitting BCV or SRDF volumes, you can reassign the drive letters for those volumes using the SIU mount action. The SIU mount action lets you mount hard disks and their associated file systems in your storage environment.

The mount action mounts the specified volume name, GUID, or Symmetrix device to the specified drive letter or mount point:

```
symntctl mount -drive DriveLetter | -path MountPnt
-vol VolName [-g VolGroup] | -guid VolGuid |
-sid SymId | -symdev SymDev | -part Partition#
```

SIU utilizes Microsoft Virtual Disk Service (VDS) functionality to make a volume permanently dismounted (offline) while performing unmount operations. Other applications are prevented from accessing the volumes in this state, thus ensuring data integrity during TimeFinder/Mirror establish and split operations. The volume can be brought back online with the SIU mount operation

SIU and SYMAPI database

When performing mount operations on the Symmetrix device number (`-symdev` option), SIU relies on the SYMAPI database to obtain physical drive information. The physical drive information within the SYMAPI database is created and updated following a Symmetrix discover command `symcfg discover`. Therefore, if any device changes are made to the mount host, it is recommended to run the `symcfg discover` command in order to update the SYMAPI database to reflect those changes. If device changes are made frequently to the mount host, then the `discover` should be scripted as a part of the mount process.

When `symntctl mount` or `umount` commands are called, SIU takes an exclusive lock against the SYMAPI database. If other Solutions Enabler commands occur in parallel to the SIU operations, SIU waits on any locks against the SYMAPI database and continues after the locks are released.

Because SIU mount and unmount operations can interact with the SYMAPI database, it is recommended that at least 2 gatekeeper devices are presented to the SIU mount host.

Disk unmount operations

On the BCV or SRDF target (R2) side, establishing a BCV with its standard volume makes the BCV unavailable to the Windows system. Windows systems generate error messages if a mounted volume is referenced while unavailable.

The SIU unmount action provides the ability to unmount hard disks and their associated file systems, and also removes the drive letter from the specified disk volume.

You can target a drive letter or mount point:

```
symntctl umount -drive DriveLetter
symntctl umount -path MountPnt
```

Or, you can target a volume or global ID:

```
symntctl umount -vol VolName [-g VolGroup]
symntctl umount -guid VolGuid
```

Or, you can target a Symmetrix device or physical device, or device partition:

```
symmtctl umount -sid SymID -symdev SymDev [-part Partition#]  
symmtctl umount -pd Pdev [-part Partition#]
```

The `-force` option can be used when unmounting to ignore and break any handles open to the volume by other applications.

Unmount operations affect the current drive letter mapping. The command performs a file system flush as part of the dismount process, and the drive is flagged as “permanently dismounted” (offline) until a subsequent `mount` operation. This ensures that no other applications can access the volume while it is being unmounted, thus ensuring data integrity during TimeFinder/Mirror establish and split operations.

When an unmount operation is performed, SIU and VDS does the following:

1. Gets a handle to the volume being unmounted from the operating system.
2. Obtains an exclusive lock on that volume from the operating system (unless `-force` is specified).
3. Flushes all pending file system cached data to disk for that volume.
4. Dismounts the volume by calling `DeviceIoControl()` with the `FSCTL_DISMOUNT_VOLUME` control code.
5. Offlines the volume by calling `DeviceIoControl()` with the `IOCTL_VOLUME_OFFLINE` control code.
6. Deletes all drive letter or mount points associated with the volume and releases the lock.

Manipulating disk signatures

In the Windows operating system, disks are identified by disk signatures (also known as labels).

Note: In Windows Server 2003/2008, GPT (GUID Partitioned) disks are identified by disk ID. The disk signature operation is not applicable to GPT disks.

Using a signature action, you can assign, change, initialize, and erase a signature from a disk.

Specify the physical disk for a signature operation using the `-pd` option, or use the Symmetrix ID and Symmetrix device options:

```
symmtctl signature -pd diskN ...
```

where *diskN* is the physical disk number.

or

```
symmtctl signature -sid SymID -symdev SymDev
```

To view disk numbers in your storage configuration, enter:

```
symmtctl list -signature
```

Using the `-sig` option, the signature of a disk is assigned/changed to any desired label specified in this command option.

Using the `-initialize` option, SIU assigns signatures to any disks discovered without signatures.

Using the `-erase` option, the signature of the specified disk is erased. This option is only used when setting up shared disks and TimeFinder configurations.

Note: With Windows 2003/2008, this option will erase the signature, only to have Windows immediately re-generate a new signature. Therefore, the volume will appear with a different signature, instead of an empty signature.

Updating partition tables

SIU allows the Symmetrix device partition tables to be updated using the `symntctl update` command with the following options:

- ◆ Use the `-all` option to update the partition tables of each physical disk.
- ◆ Use the `-pd PhysicalDrive#` option to update the partition table for the specified physical drive, for example:

```
symntctl update -pd \\.\PhysicalDrive#
```

- ◆ Use the Symmetrix ID and device number to update the partition table for the specified Symmetrix device, for example:

```
symntctl update -sid 309 -symdev 01e3
```

Flushing cache buffers to disk

TimeFinder/Mirror operations such as an on-the-fly BCV split require that the file system cache buffers be flushed to disk for a clean and complete split. The SIU flush action does this without removing the drive letter or volume association. This flush action is needed where the standard/BCV or SRDF R1/R2 pair is split without taking down an application.

You can target the standard or R1 device to flush with a drive letter or a mount point:

```
symntctl flush -drive DriveLetter
symntctl flush -path MountPnt
```

Or, you can target the standard device to flush with a volume name or global ID (GUID):

```
symntctl flush -vol VolName -g VolGroup
symntctl flush -guid VolGUID
```

Because this command only flushes the Windows file system, applications that handle their own cache management (such as SQL Server or Exchange) are not affected. Flush or checkpoint mechanisms embedded in the application must flush its own application data. The application data must be flushed first, followed by a file system flush.

For example, to flush a volume with the volume GUID of
Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}, enter:

```
symntctl flush -guid Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}
```

Device masking operations

It is often necessary to add and remove access to a given Symmetrix device for the current host. This is known as *device masking*. Device masking typically requires several steps to make the device visible or hidden to the host, especially if multiple HBAs and PowerPath are being used. SIU provides the ability to perform device masking operations with a single command by leveraging some of its internal functionality, like system rescan. When using SIU to mask or unmask a device, SIU does the following:

1. Obtains a list of all of the HBAs connected to the specified Symmetrix array on the current host.
2. Determines the current state of the specified device on each of the HBAs (whether it is currently masked or unmasked).
3. Either adds or removes access to the specified device (depends on whether it is a mask or unmask operation) for each of the HBAs.
4. Performs a system rescan of the host so that the newly masked devices will be removed from the host, and the newly unmasked devices will appear on the host.

For SIU device masking to work correctly, the Symmetrix array must be configured for device masking. The following requirements exist:

- ◆ There must be a valid VC MDB created on the Symmetrix array.
- ◆ The VC MDB must be mapped to all of the Fiber Adapters (FA) where the device masking is to take place.
- ◆ The VCM flag must be set on all of the FAs where the device masking is to take place.
- ◆ The Symmetrix device that is to be masked or unmasked must be mapped to all of the FAs where the device masking is to take place.

Note: SIU device masking does not perform any checking to determine if the Symmetrix device is currently accessible to other hosts on the SAN. It is your responsibility to prevent multiple hosts from using the same device.

If connected to a Symmetrix running Enginuity 5874, to perform Device Masking/Unmasking, at least one masking view must be present. Refer to [Chapter 2, “Masking Devices with Autoprovisioning Groups,”](#) for details about creating and using masking views.

Device masking is very useful when attempting to break open handles to a current volume. By masking the device from the host, Windows will be forced to completely unload the I/O stack for that device, thus removing all open handles. The device can then be unmasked and it will appear on the host without any open handles.

Note: In Windows 2008 devices must be properly uninstalled before being masked from the system. SIU performs the necessary uninstall process, however, the process can fail if there are open handles to a volume.

If you experience trouble masking devices on Windows 2008 use the `-force` flag to break any open handles. It is best to manually close any programs with open handles to the volume first, but specifying `-force` will break the handles automatically.

Using device masking to break the open handles to a volume may be necessary when using TimeFinder/Mirror in particular environments. Currently, it is recommended

that the BCV be unmounted by SIU before establishing it. In most cases, this is sufficient to prevent the volume's cache from getting corrupted during the TimeFinder/Mirror establish. However, if the host is running monitoring agents or anti-virus software that maintains open handles to the BCV, then the unmount will not be enough. Instead, you must mask the BCV prior to the TimeFinder/Mirror establish in order to remove the open handles. After the split operation, the BCV can be unmasked so that it is accessible to the host. This procedure is explained in [“For basic disks with open handles” on page 376](#).

Using volume flags

SIU provides three flag options that can be set on volumes, Symmetrix devices (and partitions), physical devices (and partitions), drive letters, and volume GUIDs using the `symntctl flag -set Flag` command.

Where *Flag* can be:

- ◆ READONLY — Sets the volume as read only.
- ◆ HIDDEN — Sets the volume as hidden. Volumes flagged as HIDDEN cannot be identified by `-guid` for any SIU command.
- ◆ NO_DEFAULT_DRIVE_LETTER — Sets the volume to have no default drive letter.

For example, to set the HIDDEN flag on partition 4 of Symmetrix device 01e3, in Symmetrix array 309, enter:

```
symntctl flag -set hidden -sid 309 -symdev 01e3 -part 4
```

To clear an existing flag, use the `symntctl flag -clear` option, as follows:

```
symntctl flag -clear hidden -sid 309 -symdev 01e3 -part 4
```

Use the `symntctl flag -clear` command with the `-all` option to clear the flags for all disks or volumes, as shown:

```
symntctl flag -clear hidden -all
```

Refer to the *EMC Solutions Enabler V7.0 SYMCLI Command Reference HTML Help* for the complete syntax of the `symntctl` command.

Performing typical storage operations

The following are typical storage operations that use both SIU and TimeFinder/Mirror commands to manage the storage environment. The `sleep` and `timeout` utilities can be used interchangeably to cause a delay in the execution of scripts

Synchronizing the BCV

Using the `symntctl` and the `symmir` commands, the following procedures for basic and dynamic disks work with `devgroup1` to synchronize the BCV device (drive *x*) with the standard on primary and secondary servers. For more information on the `symmir` command, refer to *EMC Solutions Enabler Symmetrix TimeFinder Family CLI Product Guide*.

For Windows Basic or Dynamic Disks

1. Unmount the volume on the secondary system with the following command:
`symntctl umount -drive x:`
2. Establish the standard and BCV pair on the primary system with the following command:
`symmir -g devgroup1 est`
3. Verify that the device is synchronized on the primary system with the following command:
`symmir -g devgroup1 verify -synched`
4. Perform a flush on the primary system with the following command:
`symntctl flush -drive x:`
5. Split the device on the primary system with the following command:
`symmir -g devgroup1 split`
6. Perform a rescan on the secondary system with the following command:
`symntctl rescan`

7. Mount the volume on the secondary system with the following command:

```
symntctl mount -drive x: -vol volumex
```

For Veritas Dynamic Disks

1. Unmount the volume on the secondary system with the following command:

```
symntctl umount -drive x:
```

2. Deport the dynamic disk group on the secondary system and rescan with the following commands:

```
vxdg deport -g devgroup1  
symntctl rescan
```

3. Perform an establish on the primary system with the following command:

```
symmir -g devgroup1 est
```

4. Rescan the storage bus on the backup server to assure that the disk has been dropped and will not be accessed by Windows Plug and Play with the following command:

```
symntctl rescan
```

5. Verify that the device is synchronized on the primary system with the following command:

```
symmir -g devgroup1 verify -synched
```

6. Perform a flush on the primary system with this command:

```
symntctl flush -drive x:
```

7. Split the device on the primary system with the following command:

```
symmir -g devgroup1 split
```

8. Perform a rescan on the secondary system with the following command:

```
symntctl rescan
```

9. Perform an import of the disk group on the secondary system and rescan with the following commands:

```
vxdg import -g devgroup1 -C  
symntctl rescan
```

10. Mount the volume on the secondary system with the following command:

```
symntctl mount -drive x: -vol volumex
```

For basic disks with open handles

To ensure that cache corruption does not occur when a BCV has open handles, use the following procedure:

Note: This procedure should be used for all TimeFinder operations if the BCV volume has open handles (usually due to a monitoring agent or anti-virus software).

1. Unmount the volume on the secondary system with the following command:

```
symntctl umount -drive x: -force
```

2. Establish the standard and BCV pair on the primary system with the following command:


```
symmir -g devgroup1 est
```

3. Verify that the device is synchronized on the primary system with the following command:

```
symmir -g devgroup1 verify -synched
```

4. Perform a flush on the primary system with the following command:

```
symntctl flush -drive x:
```

5. Split the device on the primary system with the following command:

```
symmir -g devgroup1 split
```

6. Mount the volume on the secondary system with the following command:

```
symntctl mount -drive x: -vol volumex
```

Restoring Veritas dynamic disks with TimeFinder

The following procedure restores a BCV of a dynamic disk that uses both SIU and TimeFinder/Mirror commands.

1. Unmount any drive letters or repare points currently assigned to the BCVs in the dynamic disk group on the backup server with the following command:

```
symntctl umount -drive x:
```

2. Deport the dynamic disk group on the backup server and rescan with the following commands:

```
vxdg deport -g DgName  
symntctl rescan
```

3. Unmount any drive letters or repare points currently assigned to the standard devices in the dynamic disk group on the production server with the following command:

```
symntctl umount -drive x:
```

This step is necessary to ensure that the standard device is not written to during the restore (since all writes are committed to the BCV). Do not use `mountvol` to perform this dismount.

4. Deport the dynamic disk group on the production server and rescan with the following commands:

```
vxdg deport -g DgName  
symntctl rescan
```

Failure to perform this step may leave the dynamic disk group in an unrecoverable state on the production server.

5. Restore the TimeFinder/Mirror device group with the following command:

```
symmir -g DgName restore
```

6. Rescan the storage bus on the backup server with the following command.

```
symntctl rescan
```

The dynamic disk group and all of its dynamic disks are listed as unreadable, and are therefore, offline.

7. Split the TimeFinder/Mirror device group with the following commands:

```
symmir -g DgName verify -synched  
symmir -g DgName split
```

8. Rescan the storage bus on the backup server with the following command:

```
symmtctl rescan
```

The dynamic disk group and all of its dynamic disks are readable, but are still offline and foreign.

9. Rescan the storage bus on the production server with the following command:

```
symmtctl rescan
```

The dynamic disk group and all of its dynamic disks are present, but offline and foreign on the production server.

10. Import the dynamic disk group on the backup server with the following command:

```
vxvg import -g DgName -C
```

11. Rescan the storage bus on the backup server with the following command:

```
symmtctl rescan
```

12. Import the dynamic disk group on the production server with the following command:

```
vxvg import -g DgName -C
```

The dynamic disk group and all of its dynamic disks are online and healthy.

13. Rescan the storage bus on the production server with the following command:

```
symmtctl rescan
```

The dynamic disk group and all of its dynamic disks are present, but offline and foreign on the production server.

14. Remount each of the volumes in the disk group on the production server with the following command:

```
symmtctl mount -drive x: -vol volumex
```

Use the LDM snap-in, SIU CLI, for this mount.

Creating a backup using TimeFinder/Snap

This example starts with the VDEVs unmasked (available) to the backup/reporting host. *Automount* should be disabled. Disabled is the default and can be checked through the *diskpart* CLI.

1. Create the snap devices:

```
symsnap -g test create -nop
```

2. Activate the snap devices:

```
symsnap -g test activate -nop
```

3. Plug and Play rescan to discover the VDEVs in their Ready state:

```
symntctl rescan  
timeout 30
```

4. Update the SYMAPI database for SIU mount operations:

```
symcfg dis
```

5. The VDEVs should now be in their Ready state. Force Windows to rescan the device layout of the VDEVs:

```
symntctl update -sid 54 -symdev 3C0
```

6. Mount the snap devices to desired mount points:

```
symntctl mount -drive w: -symdev 3C0 -sid 54
```

7. When the backup or reporting is finished, unmount the snap devices:

```
symntctl umount -drive w:
```

8. Terminate the snap devices:

```
symsnap -g test terminate -nop
```

9. The VDEVs should now be in their terminated state. Force Windows to rescan the device layout of the VDEVs:

```
symntctl update -sid 54 -symdev 3C0
```

Creating a backup using TimeFinder/Clone

This example starts with the BCVs unmasked (available) to the backup/reporting host. Automount should be disabled. Disabled is the default and can be checked through the "diskpart" CLI. Any volumes on the BCVs should have been previously unmounted by SIU.

1. Create or recreate the clone session using one or the other of these commands:

```
symclone create -g test  
symclone recreate -g test
```

2. Activate the clone devices:

```
symclone activate -g test
```

3. Mount the clone devices:

```
symntctl mount -drive w: -sid 54 -symdev 456
```

4. When backup or reporting is finished, unmount the clone devices:

```
symntctl umount -drive w
```

SIU Examples

The following are examples using the `symntctl` command:

To flush any unwritten file system buffers to drive E, enter:

```
symntctl flush -drive E:
```

To flush any unwritten file system buffers to mount point E:\drivex, enter:

```
symntctl flush -path E:\drivex
```

To flush any unwritten file system buffers to volume volumex, enter:

```
symntctl flush -vol volumex
```

To flush a volume with a GUID of

Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}, enter:

```
symntctl flush  
-guid Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}
```

To show the physical configuration for disk 3, enter:

```
symntctl show -pd disk3
```

To show the following example output enter:

```
symntctl show -vol HarddiskVolume7
```

To mount a basic volume called volumex to drive x, enter:

```
symntctl mount -drive x -vol volumex
```

To mount a dynamic volume called volumex of volumegroupx to mount point x:\volumex, enter:

```
symntctl mount -path x:\volumex -vol volumex  
-g volumegroupx
```

To mount a volume with the a GUID of

Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303} to drive x, enter:

```
symntctl mount -drive x  
-guid Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}
```

To unmount a volume associated with drive x, enter:

```
symntctl umount -drive x
```

To unmount a volume called volumex of volumegroupx, enter:

```
symntctl umount -vol volumex -g volumegroupx
```

To unmount a volume with a GUID of

Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303} from drive x, enter:

```
symntctl umount  
-guid Volume{ecba40b5-d522-11d5-98cb-00b0d0b03303}
```

To change the signature of disk 4 to 1ED67A70, enter:

```
symntctl signature -pd disk4 -sig 1ED67A70
```

To erase the signature of disk 4, enter:

```
symntctl signature -pd disk4 -erase
```

To initialize the signature of all disks that have 0 for their signatures, enter:

```
symmtctl signature -initialize
```

To remove a Symmetrix device from the host, enter:

```
symmtctl mask -pd disk4
```

or

```
symmtctl mask -sid 000190300186 -symdev 3ED
```

To present a Symmetrix device to the host, enter:

```
symmtctl unmask -sid 000190300186 -symdev 3ED
```

XML EXAMPLES

SIU provides the ability to specify the output format for the `symntctl list` and `symntctl show` commands. Here are examples of XML output.

`symntctl.exe show -vol HarddiskVolume2 -output XML_ATTRIBUTE`

```
<?xml version="1.0" standalone="yes" ?>
<SymCLI_ML>
  <Volume name="HarddiskVolume2" guid="e79fbcba-cdae-11dc-a419-005056b841fa" flag="PERMANENTLY
DISMOUNTED">
    <Extent name="\\.\PhysicalDrive3" starting="32256" length="526385664" symid="000190300186"
symdev="3DC" />
  </Volume>
</SymCLI_ML>
```

`symntctl.exe show -pd \\.\PhysicalDrive3 -output XML_ELEMENT`

```
<?xml version="1.0" standalone="yes" ?>
<SymCLI_ML>
  <Disk>
    <name>\\.\PhysicalDrive3</name>
    <disktype>basic</disktype>
    <partitiontype>MBR</partitiontype>
    <size>4314</size>
    <symid>000190300186</symid>
    <symdev>3DC</symdev>
    <signature>5C02D0EB</signature>
    <status>online</status>
    <diskaddress>Port1Path0Target3Lun0</diskaddress>

<devicepath>\\?\scsi#disk&ven_emc&prod_symmetrix&rev_5771#4&3a739529&0&030#{53f56307-b6bf-11d
0-94f2-00a0c91efb8b}</devicepath>
    <adapter>LSI Logic PCI-X Ultra320 SCSI Host Adapter</adapter>
    <Volume>
      <name>HarddiskVolume2</name>
      <starting>32256</starting>
      <length>526385664</length>
    </Volume>
  </Disk>
</SymCLI_ML>
```

Symbols

137

A

- abort argument 26
- access control, and device masking 149
- add RDF mirror 107
- adding devices 155, 156
- advise, reservations 77
- AIX systems
 - introducing new devices 358
- alias
 - creating 169
 - renaming 169
- allocated space
 - freeing up 86
- ASCII World Wide Names (AWWN)
 - establishing 153
 - format 153
 - usage 154
- attributes, setting 35
- audit log
 - viewing 250
- authentication
 - CHAP 181, 183
 - CHAP two-way 184
 - iSCSI 180
- auto meta feature
 - setting 35
- Autoprovisioning Groups
 - displaying views 137
 - examples 139
 - initiator groups 123
 - masking devices using 117
 - masking view 124
 - port groups 122
 - storage groups 121

B

- backup file
 - creating a database 160
- basic disks

- synchronizing the BCV for 375
- BCV
 - configurations 42
 - devices 42
 - synchronizing steps 376
- between 71
- binding devices to a pool 48
- Brocade switch
 - finding the FCID for 172

C

- cache buffer flush 371
- cache partitions 273
 - adding devices to 275
 - creating 275
 - listing 276
 - modifying 276
 - removing devices from 276
- Cascaded RDF 107
- change sessions
 - aborting 26
 - monitoring changes 25
- CHAP
 - authentication 181, 183
 - authentication with symaccess 131
 - configuring 188
 - for iSCSI 147
 - two-way authentication 184
 - two-way configuration 195
- checksum
 - benefits of using 255
 - checking databases 254
 - configuration parameter setting for 256
 - license requirements 256
 - validation list 258
 - where it occurs 254
- checksum testing
 - implementing 256
 - validating your system 258
 - verifying checksum is enabled 258
 - within the Symmetrix 255
- CKD devices 36
 - assigning PAV aliases 99

- copying a CU image 99
 - mapping a range 98
 - removing PAV aliases 100
 - unmapping a range 101
- code examples, Quality of Service 278
- command file
 - mandatory actions to perform 233
 - parameters 238
 - policy-scripted 217
- concurrent configuration change sessions 24
- concurrent RDF, enabling 35
- configuration control
 - operations 30
 - operations, by Enginuity version 31
- configuration verification 28
- Connectrix switch
 - finding the FCID for 172
- converting databases 161
- copy operation
 - for BCV 269
 - for Clone 269
 - for mirrors 269
 - for RDF 269

D

- data corruption
 - between host and storage 254
 - traditional checking 254
- DATA devices
 - configuration rules 82
 - creating 81
 - draining 92
- data preservation 59
- data, refreshing 110
- device
 - HP ID 71
 - identifiers 71
 - VMS ID 71
- device geometry
 - output example 69
- device identifiers
 - output example 72
- device locking 66
- device LUN visibility 173
- device masking 372
 - and access control 149
 - architecture 142
 - configuration steps 150
 - functionality 142
- device masking database
 - activating a new configuration 151
 - backup file for 160
 - blocking direct writes to 162
 - converting 161
 - discover 145
 - examining 163
 - external lock, releasing 145
 - initializing 159
 - maintenance 159
 - restore and convert 161
 - restoring 161
 - security for 148
 - types 160
 - using syscalls with 148
- device migration
 - examples 251
- device migrations
 - requirements 240
 - using virtual LUN 240
- device name 71
- device reservations 76
 - output example 78
- device type
 - conversion 44
 - creating 44
- devices
 - adding 155, 156
 - binding to a pool 48
 - capacity of 165
 - converting 61
 - creating (adding) 46
 - deleting 74
 - mapping 96
 - migrating 236
 - removing 157
 - setting characteristics 67
 - unmapping 101
- directors
 - refresh 151, 170
- discover
 - using 145
- disk groups
 - configuring SFW 363
- disk management
 - adding a spare disk 60
 - deleting a spare disk 60
- disk signatures 369
- disk space 28
 - configuring into devices 49
- draining DATA devices 92
- dyn_rdf, enabling 67
- dyn_rdf1_only, enabling 67
- dyn_rdf2_only, enabling 67
- dynamic cache memory 273
- dynamic concurrent RDF, enabling 35
- dynamic disk groups 363
- dynamic disks
 - restoring with TimeFinder 377
 - synchronizing the BCV for 376
- dynamic LUN addressing, adding devices with 156
- dynamic RDF 103
 - devices, configuring 103
 - enabling 35, 103
- dynamic reallocation volumes (DRVs) 222
- dynamic swap 111

E

- enabling
 - concurrent RDF 35

- dyn_rdf 67
- dynamic RDF 35
- FBA multi access cache 35
- RAID 5 support 36
- RDB_Cksum 67
- VCMDB 67
- enabling checksum 258
- enforce, reservations 77
- Enhanced Virtual LUN Technology
 - overview of 240
- environment option, non visible devices 29
- external lock
 - on VCMDB 145

F

- FBA multi access cache, enabling 35
- features 362
- Fibre Channel ID (FCID)
 - finding the FCID 172
 - for Brocade switch 172
 - for Connectrix switch 172
 - format 173
 - lock down 131, 171
 - lock down effects on commands 172
 - lock down procedure 131, 172
- fibre protocols 112
- file
 - masking devices using 156
 - removing devices using 158
- flush cache 371
- force 372
- free disk space 28

G

- Generic SafeWrite 260
 - device flag 262
 - disabling 263
 - where to enable 261
- groups
 - dynamic 363

H

- HBA
 - alias names 153
 - discovering 120
 - management 136, 169
 - supported initiators 147
 - swapping 136, 169
- HBA, deleting associations 170
- held devices 66
- host
 - updating 279, 355
- host updating 29
- hot swap policy, setting 35
- HP ID
 - device identifier 71
- HP Tru64 UNIX systems
 - introducing new devices 359
- HP-UX systems

- introducing new devices 357

I

- initiator groups
 - cascaded 126
 - creating 123
 - renaming 126
- IPsec
 - configuration component 207
 - iSCSI link 206
 - network tunnel 205, 206
 - policies 204, 208, 212, 217
 - SRDF link 205
- iSCSI
 - authentication 180
 - configuring the software driver for 187
 - topology 147
 - with CHAP 131
 - with CHAP authentication 181, 183, 188
 - without CHAP authentication 196

L

- latency time 225
- listing extents
 - output 257
- lock
 - on VCMDB 145
 - releasing 145
- Logical Disk Manager 364
- login history table
 - contents 166
 - usage 153
- LUN
 - offset, hexadecimal value 173
 - skip hole 173
 - visibility 173

M

- mapping to front-end ports 96
- masked channel 142
- masking
 - devices 372
- masking view
 - creating 124
 - deleting 126
- masking views
 - renaming 126
- maximum hypers per disk, setting 35
- metadevice
 - thin 58
- metadevices
 - adding 155
 - autocreate 53
 - configuration restrictions 56
 - converting 64
 - creating 52
 - dissolving 57
 - metahead 51
 - performance 51

- size requirements 51
- metamembers
 - removing 57
 - removing from the database 157
- metrics, viewing 36
- migrating devices 236
- migration sessions
 - listing 247
 - managing 247
 - terminating 249
 - verifying 248
- mirror set protections 37
- mirrors
 - RAID-S 41
 - two way 40
- mounting BCV or SRDF volumes 367
- MVS subsystem IDs 36, 47, 50

N

- new devices, introducing to
 - AIX systems 358
 - HP Tru64 UNIX systems 359
 - HP-UX systems 357
 - Solaris systems 356
 - Windows systems 360
- non visible devices 29

O

- Optimizer
 - best swaps with 225
 - command file 238
 - configuration requirements 222
 - copying volumes to DRVs 227
 - legal swaps 228
 - performance metrics 224
 - rollbacks 235
 - service times 225
 - setting parameters for 232
 - setting performance time windows 234
 - setting user approved mode 233
 - starting 231
 - stopping 231
 - swap procedure 226
 - time windows, showing 232

P

- parity RAID 41
- PAV alias addresses
 - adding to mapped devices 99
 - removing from mapped devices 100
- PAV alias limit, setting 35
- PAV mode, specifying 36
- performance time windows, setting 234
- policies
 - IPsec 204, 208, 212, 217
- port characteristics
 - copying from one port to another 112
 - setting 112
- port flags 112

- viewing 116
- port groups
 - creating 122
 - renaming 126
- preallocate size
 - thin devices 58
- profile tables 151, 170
- protocols
 - network connection 205
- provisioning
 - configuring 80

Q

- QoS 268
 - dynamic cache partitions 273
- Quality of Service
 - code examples 278
 - Engenuity version requirement 270
 - parameters for 278
 - setting the copy pace level 269
- query argument 25

R

- R1/R2 devices, swapping RDF personalities 111
- R21 device 107
- RA groups, swapping 110
- RADIUS server 201
- RAID 5 support, enabling 36
- RAID-S 41
- RAID-S support, enabling 36
- RDB_Cksum, enabling 67
- RDF metadevices 55
- refresh directors 151, 170
- refresh option 110
- remove RDF mirror 107
- removing devices 157
- removing masked devices 157
- requirements
 - SIU 362
- reservations
 - committing changes to 77
 - device 76
 - list output example 78
 - releasing 79
 - viewing 78
- restoring a database 161
- restrictions
 - for configuring metadevices 56
 - for configuring SAVE devices 65
 - for configuring virtual devices 65
 - for converting devices 63
 - for creating devices 49
 - for deleting devices 74
 - for setting device characteristics 67
 - for swapping RA groups 110
- rollbacks
 - performing 235

S

- safe disciplines 28
- SAVE device 90
- SAVE device pool, types 91
- SCSI protocols 112
- SCSI writes
 - blocking from the database 162
- SCSI3_persist_reserv 67
- SCSI3_persist_reserv, enabling 67
- SDR
 - mapping 96
 - unmapping 101
- security
 - CHAP protocol 147
- seek time 225
- service attributes, setting 67
- session ID 26
- signatures, manipulating 369
- SIU
 - features 362
 - overview 362
 - requirements 362
 - viewing disk information 366
- skip hole
 - hexidecimal value 173
 - LUN base/offset 173
- Snap pool 91
- Solaris systems
 - introducing new devices 356
- spare disks
 - adding 60
 - removing 60
 - viewing 60
- SRDF
 - adding RDF mirrors 107
- SRDF configurations 42
- SRDF devices 42
 - converting from static to dynamic 61
- SRDF group attributes
 - setting 105
- SRDF/A
 - assigning session priority 105
 - setting host throttle time 36
- SRDF/A DSE
 - enabling autostart 105
 - naming pools 105
 - pool 91
 - setting threshold 105
- SSIDs 36
- storage group
 - adding devices to 128
 - creating 121
 - deleting 129
 - displaying details 137
 - removing devices from 129
 - renaming 126
- striped data 52
- striped meta
 - protect_data 57
- subsystem IDs 36

- suppress feature option 257
- swap RDF devices 103
- swaps
 - Optimizer 228
- symaccess
 - masking devices using 118
- symipsec command 207
- symmask actions
 - list hba 150
 - list logins 151
- symmaskdb actions
 - backup 160
 - init 159
 - list database 163
- Symmetrix
 - device locks 66
 - Priority Controls 271
 - setting attributes 35
- Symmetrix Integration Utilities (see SIU) 279, 361
- symmigrate
 - validating migration sessions 243
- symmtctl
 - flush 371
 - mount 367
- symoptmz
 - command features 231
- symqos 268
- symqos, command description 268

T

- thin devices
 - allocating space on 86
 - binding 83
 - binding to pool 58
 - configuration rules 83
 - converting 63
 - creating 82
 - data allocation 81
 - meta 58
- thin pools
 - creating 84
 - viewing 86
- threshold
 - setting for SRDF/A DSE 105
- time windows
 - clearing 235
 - setting 234
 - showing 232
- TimeFinder
 - restoring dynamic disks 377
- transfer time 225
- transmit idle, enabling 105

U

- unmount operations 367
- Updating the host 279, 355
- user approved mode, for Optimizer 233

V

VCMDB

- converting 161
- create a record 189
- enabling 67
- setting restricted access 36
- types 160

See also device masking database

Veritas Storage Foundation for Windows

- dynamic disk groups (VDGs) 363

virtual LUN migration

- examples 251
- file format 242
- to configured space 244
- to unconfigured space 246

Virtual Provisioning

- configuring 80
- creating data devices 81
- creating thin devices 82

visibility

- devices 173

VMS ID

- device identifier 71

VMware server

- configuring for SIU 365

W

Windows systems

- introducing new devices 360

World Wide Name (WWN) 153

- profile tables 151, 170

See also ASCII World Wide Names (AWWN)