EMC® Solutions Enabler Symmetrix® Array Controls CLI

Version 6.4

Product Guide

P/N 300-002-940
REV A04

EMC Corporation
Corporate Headquarters:
Hopkinton, MA 01748-9103
1-508-435-1000
www.EMC.com

Copyright © 2005 - 2007 EMC Corporation. All rights reserved.

Published March, 2007

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 listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

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

Contents

Preface 13

Part 1 Concepts and Procedures

Chapter 1 Managing Configuration Changes

Introduction 20
Devices and system configurations 21
- Configurations 21
- SRDF and BCV configurations 25
- Metadevices 27
- Configuration Manager operations 29
- Configuration Manager device emulation 32

Device creation strategy 34
- Device creation restrictions 35
- Meta configuration restrictions 36
- Virtual device configuration restrictions 36
- SAVE device configuration restrictions 37
- RDF configuration changes 38
- Changes to mirror protection 38

Configuration control operations overview 39
- Device locking 39
- Held devices 39
- Operations classes 40
- Configuration change safety 41
- The command file 42

Stages of the configuration change action 43
Creating devices 45
Configuring disk space 48
Deleting devices 50
Converting devices 52
Setting device attributes 55
Managing device reservations 57
- Reserving devices 57
- Committing changes on reserved devices 58
- Viewing existing reserved devices 59
- Releasing reserved devices 60

Mapping devices 61
Unmapping devices 63
Configuring dynamic RDF-capable devices 64

Enabling dynamic RDF on the Symmetrix array 64
Enabling dynamic RDF on the device 66
Setting RDF group attributes 67
Configuring metadevices 68
Creating a metadevice 68
Expanding an existing metadevice 70
Removing metamembers 70
Dissolving a metadevice 71
Converting a metadevice 71
Data preservation concerns 72
Managing CKD devices 73
Mapping a range of devices 73
Unmapping a range of devices 73
Copying the mapping of a range of devices 74
Assigning PAV alias addresses to mapped devices 74
Removing PAV alias addresses from mapped devices 75
Managing SAVE device pools 76
Creating a SAVE device pool 77
Deleting a SAVE device pool 77
Adding SAVE devices to a pool 77
Removing SAVE devices from a pool 78
Enabling/disabling SAVE devices 78
Setting port characteristics 79
Setting Symmetrix metrics 84
Swapping RA groups 86
Managing spare disks 87
Adding a spare disk 87
Deleting a spare disk 87
Viewing existing spares 88
Swapping R1 devices with R2 devices 89
Monitoring the status of the change session 90
Aborting the change session 91
Subsystem IDs for mainframe systems 92

Chapter 2 Device Masking

Device masking overview 94
How Symmetrix device masking works 94
Host access example 94
Managing access for unsupported host platforms 95
Device masking controls overview 96
Database device locking 96
Mapping volume logix commands 96
Supported topologies 97
Fibre Channel topology 97
Native iSCSI topology 98
Device masking VCMDB security enhancement 99
VCMDB management updates 99
Initial device masking setup 100
Access control environment setup 100
Device masking configuration 101
Identify configuration components 101
Initialize and update the database 102
Enable authentication 102
Recommendations for activating the configuration 102

Discovering host HBAs 104
Using alias names 104
How to add and remove masked devices 106
Adding devices 106
Using reserved devices 106
Adding device group devices 106
Adding devices from a device file 107
Adding devices with dynamic LUN addressing 107
Removing devices 108
Removing device group devices 108
Removing devices using a device file 109
Device masking VCMDB maintenance 110
Initializing the database 110
Preventing unauthorized modification of the VCMDB 110
Managing a backup VCMDB file 111
VCMDB database types 111
Converting a VCMDB type 112
Restoring a backup VCMDB 112
Restoring a backup and converting its type 112
Blocking direct writes to the VCMDB 113
How to view the VCMDB 114
Viewing the database 114
Viewing device capacity 116
Viewing the login history table 117
HBA initiator management 118
Creating and renaming aliases 118
Swapping a new HBA for a failed HBA 118
Deleting HBA associations 119
Fibre Channel-to-host interface management 120
Locking down a Fibre Channel ID 120
Setting device LUN visibility 122
Setting the LUN base/offset skip adjustment 122
Setting the HBA port flags 124
Setting the heterogeneous host configuration 125

Chapter 3 Device Masking: iSCSI Setup

iSCSI authentication overview 130
iSCSI CHAP authentication: Enginuity version 56xx 131
Requirements 131
Setting iSCSI authentication via the SYMCLI 131
Showing authentication information 131
Restoring authentication data and updating the iSCSI driver 132
iSCSI CHAP authentication: Enginuity Version 57xx 133
Setting one-way CHAP authentication 133
Setting two-way CHAP authentication 134
Initializing an authentication database 134
Displaying authentication information 135
Backing up and restoring authentication information 135
iSCSI software driver configuration 137
Requirements 137
Installing the iSCSI software initiator 137
Configuring iSCSI with CHAP authentication 138
Configuring iSCSI for 2-way chap authentication 145
Configuring iSCSI without CHAP authentication 146

RADIUS server configuration 151

Chapter 4 Optimizing Array Performance

What Is Symmetrix Optimizer? 154
How Optimizer works 154
Supported environments 155
Configuration requirements 156
Dynamic reallocation volumes (DRVs) 156
Open mirror position requirement 156
Open configuration lock requirement 156
How does Symmetrix Optimizer work? 157
The analysis 157
Performance metrics 157
Modeling service time 158
Finding the best swap 158
The swap procedure 159
Legality of swaps 160
Impact of swaps 162
Symmetrix Optimizer operations via SYMCLI 163
Starting and stopping Optimizer 163
Committing command file changes 163
Setting user approved mode 164
Setting performance time windows 164
Showing time windows 165
Clearing a time window 165
Setting swap priority 165
Performing rollbacks 166
Setting manual swap lists 166
Optimizer logs and swap history 166
Migrating devices 167
Command file general parameters 168
Example 168

Chapter 5 Managing Quality of Service

Overview 170
Setting QoS copy parameters and values 171
Copy pace to certain devices 171
Setting QoS LRU parameters and values 172
Using Symmetrix Priority Controls 173
Enabling and disabling priority of service 173
Setting priority of service 173
Listing priority of service 174
Using dynamic cache partitions 175
Enabling cache partitions 175
Creating cache partitions 175
Adding devices to a cache partition 176
Modifying a cache partition 176
Removing devices from a cache partition 177
Listing cache partitions 177
QoS examples 179

Chapter 6 Performing Double Checksum Operations

Overview 182

Traditional methods of preventing data corruption 182
Data corruption between host and conventional storage 182
Benefits of checking within the Symmetrix array 183
Implementing EMC Double Checksum for Oracle 184
Other Checksum operations 184
Suppressing default features 185
Enabling checksum options 185
Verifying checksum is enabled 186
Validating for Checksum operations 186
Disabling Checksum 187
Functionality by Enginuity level 187
Implementing Generic SafeWrite for generic applications 188
Torn pages: using generic safewrite to protect applications 188
Why generic? 188
Where to enable Generic SafeWrite 189
Configuring Generic SafeWrite 190
How to disable Generic SafeWrite 191
Listing Generic SafeWrite devices 192
Performance considerations 192
Syntax and examples 193

Part 2 Operational Examples

Chapter 7 Configuration Change Examples

Example 1: Creating devices 198
Example 2: Mapping devices 204
Example 3: Mapping a single device on multiple EA ports 209
Example 4: Mapping a range of devices and adding aliases 215
Example 5: Unmapping devices 220
Example 6: Setting front-end port attributes 224
Example 7: Forming an FBA metadevice 229
Example 8: Configuring RDF dynamic devices 235
Example 9: Configuring concurrent RDF devices 246

Chapter 8 Device Masking Examples

Example 1: Adding masked devices for HBA access 254
Example 2: Using Fibre Channel ID lockdown 256
Example 3: Options when displaying the VCMDB 258

Appendix A Updating the Host

Sun Solaris systems 262
HP-UX systems 263
IBM AIX systems 264
HP Tru64 UNIX systems 265
Windows systems 266
Windows 2000 266
Windows 2003 266

Appendix B Device Masking: Volume Logix Conversion

Volume Logix conversion 268

Index 269

Figures

	Title	Page
1	Symmetrix concatenated metadevice two-way mirror scheme	22
2	Symmetrix three-way mirror scheme	23
3	Symmetrix parity RAID mirror scheme	24
4	Simple SRDF device configuration	25
5	Various SRDF-connected device configurations with BCVs	26
6	Concatenated metadevices	27
7	Striped data	28
8	Device masking solution with WWN initiators	95
9	Point-to-point and multi-initiator topologies	97
10	Native iSCSI topology	98
11	Device masking syscalls	99
12	Device masking components	101
13	Identify volumes to swap	159
14	Copy volume to DRV	159
15	Copy DRVs to new locations	160
16	DRVs are split	160
17	Generic SafeWrite list output	192

Tables

	Title	Page
1	Mirror set protection schemes	21
2	Metavolume stripe sizes	28
3	Configuration Manager operations	29
4	Configuration Manager device emulation	32
5	Steps to create a device	34
6	Steps to create a metadevice	35
7	Metadevice data preservation	72
8	SCSI protocol port flags	80
9	Fibre protocol port flags	81
10	LUN base/offset scenarios for multiple hosts with skip holes	123
11	Host platforms and interface configuration flags	125
12	Optimizer strategies for determining swaps	157
13	Symmetrix Optimizer command	163
14	Parameters in the command file	168
15	SYMAPI Quality of Service command	170
16	EMC Double Checksum for Oracle functionality by Enginuity level	187
17	symchksum syntax and descriptions	193
18	Volume Logix to SYMCLI conversion	268

Preface

As part of its effort to continuously improve and enhance the performance and capabilities of the EMC product line, EMC periodically releases new versions of both the EMC Enginuity Operating Environment and EMC Solutions Enabler. Therefore, some functions described in this document may not be supported by all versions of Enginuity or Solutions Enabler currently in use. For the most up-to-date information on product features, see your product release notes.

If a Solutions Enabler feature does not function properly or does not function as described in this guide, please contact the EMC Customer Support Center for assistance.

Audience

This document provides a guide for advanced command-line users and script programmers by describing how 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 Symmetrix CLI Command Reference*
- *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, environment variables, option file parameters, and error codes can now be found in the companion *EMC Solutions Enabler Symmetrix CLI Command Reference*.

Conventions used in this manual

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 3*xxx* and 5*xxx* model numbers.

Every occurrence of *Symmetrix 5*, refers to Symmetrix 8*xxx* 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 font	In running text: • Interface elements (for example, button names, dialog box names) outside of procedures • Items that user selects outside of procedures • Java classes and interface names • Names of resources, attributes, pools, Boolean expressions, buttons, DQL statements, keywords, clauses, environment variables, filenames, functions, menu names, utilities • Pathnames, URLs, filenames, directory names, computer names, links, groups, service keys, file systems, environment variables (for example, command line and text), notifications
Bold	In procedures: • Names of dialog boxes, buttons, icons, menus, fields • Selections from the user interface, including menu items and field entries • Key names • Window names In running text: • Command names, daemons, options, programs, processes, notifications, system calls, man pages, services, applications, utilities, kernels.
Italic	Used for: • Full publications titles referenced in text • Unique word usage in text
Bold Italic	Anything requiring extra emphasis
`Courier`	Used for: • System output • Filenames • Complete paths • command-line entries • URLs
`Courier, bold`	Used for: • User entry • Options in command-line syntax
`Courier, italic`	Used for: • Arguments used in examples of command-line syntax • Variables in examples of screen or file output • Variables in path names
`Courier, bold, italic`	Variables used in a command-line sample
< >	Angle brackets enclose parameter or variable values supplied by the user

[ ]	Square brackets indicate 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:

`techpub_comments@EMC.com`

PART 1

Concepts and Procedures

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, "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 3, "Device Masking: iSCSI Setup," describes how to configure your iSCSI driver software and authentication information.

Chapter 4, "Optimizing Array Performance," describes the Symmetrix Optimizer concepts and how to improve array performance using the Symmetrix Optimizer commands of the SYMCLI.

Chapter 5, "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.

Chapter 6, "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.

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

---

1 Managing Configuration Changes

This chapter describes configuration concepts and how to manage various types of configurations using the Configuration Change component of the SYMCLI. The chapter covers the following topics:

- Introduction 20
- Devices and system configurations 21
- Device creation strategy 34
- Configuration control operations overview 39
- Stages of the configuration change action 43
- Creating devices 45
- Configuring disk space 48
- Deleting devices 50
- Converting devices 52
- Setting device attributes 55
- Managing device reservations 57
- Mapping devices 61
- Unmapping devices 63
- Configuring dynamic RDF-capable devices 64
- Setting RDF group attributes 67
- Configuring metadevices 68
- Managing CKD devices 73
- Managing SAVE device pools 76
- Setting port characteristics 79
- Setting Symmetrix metrics 84
- Swapping RA groups 86
- Managing spare disks 87
- Swapping R1 devices with R2 devices 89
- Monitoring the status of the change session 90
- Aborting the change session 91
- Subsystem IDs for mainframe systems 92

Introduction

The SYMCLI Configuration Change component extends the basic SYMCLI command set to include the `symconfigure` command on the following platforms:

- Sun Solaris
- Hewlett Packard HP-UX
- Windows 2000
- Tru64 UNIX
- IBM AIX
- Linux
- AS/400 (via client/server)

This command allows you to perform powerful control operations that manage various types of configuration changes on devices in your Symmetrix® environment. 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.

The `symconfigure` command enables you to manage change sessions with the following control operations:

- Creates new Symmetrix devices
- Creates/deletes SAVE device pools
- Converts devices to other configuration types by adding/removing RDF, BCV, or DRV attributes
- Copies device mappings from one EA/EF port to another
- Increases device protection by adding additional mirrors
- Decreases device protection by removing existing mirrors
- Deletes devices
- Disables SRDF/A on an RA group
- Sets device attributes (for example, RDB_Cksum)
- Sets selective Symmetrix metrics (for example, parity RAID)
- Sets RA group parameters
- Swaps RDF source/target attributes for an RA group
- Converts static RDF devices to dynamic RDF devices
- Forms/dissolves metadevices
- Adds/removes metadevice members
- Maps/unmaps a range of Symmetrix devices that form a mainframe CU image to a front-end port
- Maps/unmaps a Symmetrix device to front-end ports
- Sets/clears SCSI or fibre port flags
- Adds/removes dynamic (hot) spare
- Adds/removes PAV aliases from mainframe devices

Devices and system 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* or a *hyper*.

Configurations

There are various Symmetrix device configurations that involve various mirror set protection possibilities. Table 1 lists the most typical mirror set protection schemes.

Table 1 Mirror set protection schemes

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			
DRV	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			
BCV+R-5[a]	Data and Parity	Data and Parity		
BCV+R-6[a]	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		
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]	Data and Parity	Data and Parity		

Table 1 Mirror set protection schemes (continued)

Device/mirror configuration	M1	M2	M3	M4
RDF2+R-6	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	
Unprotected	Data			

a. This table 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.

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

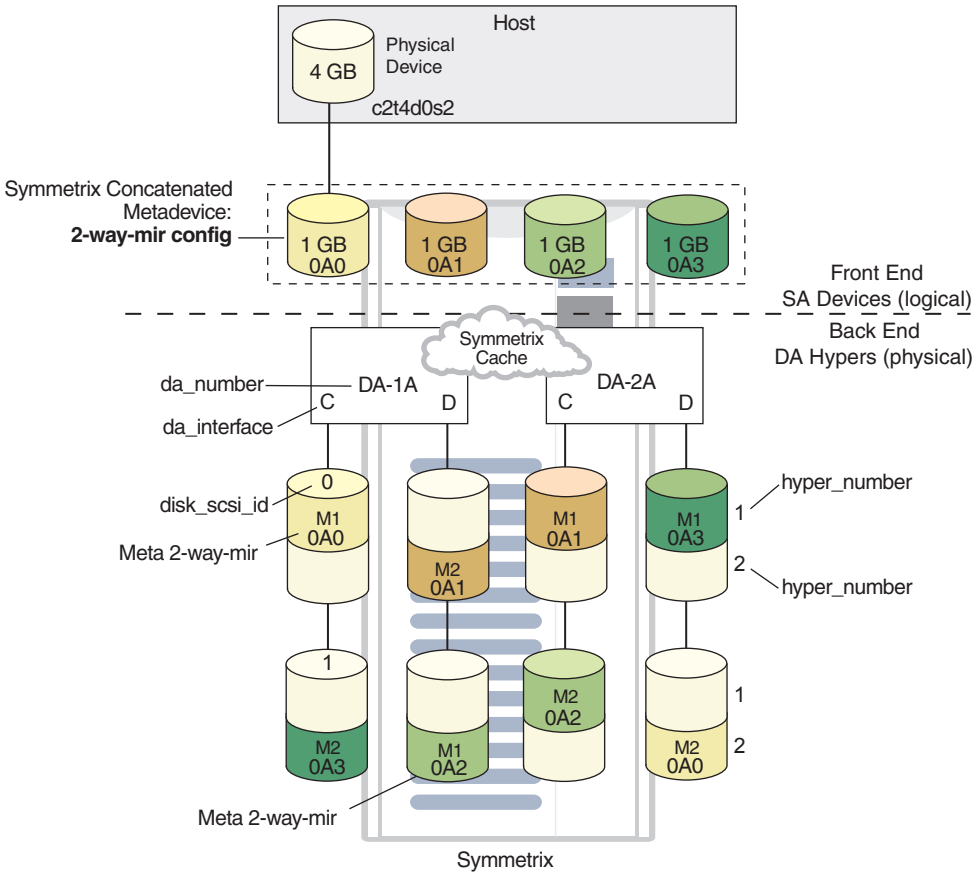


Figure 1 Symmetrix concatenated metadevice two-way mirror scheme

Figure 2 illustrates the devices for two three-way mirror configurations (3-Way-Mir).

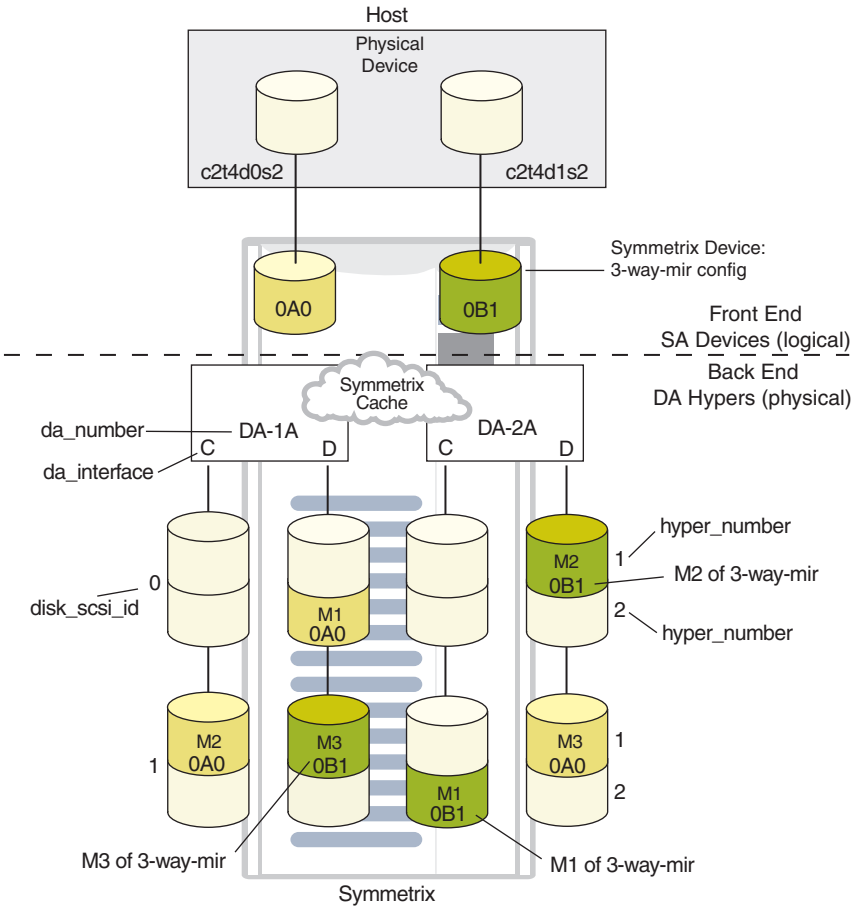


Figure 2 Symmetrix three-way mirror scheme

Parity RAID

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

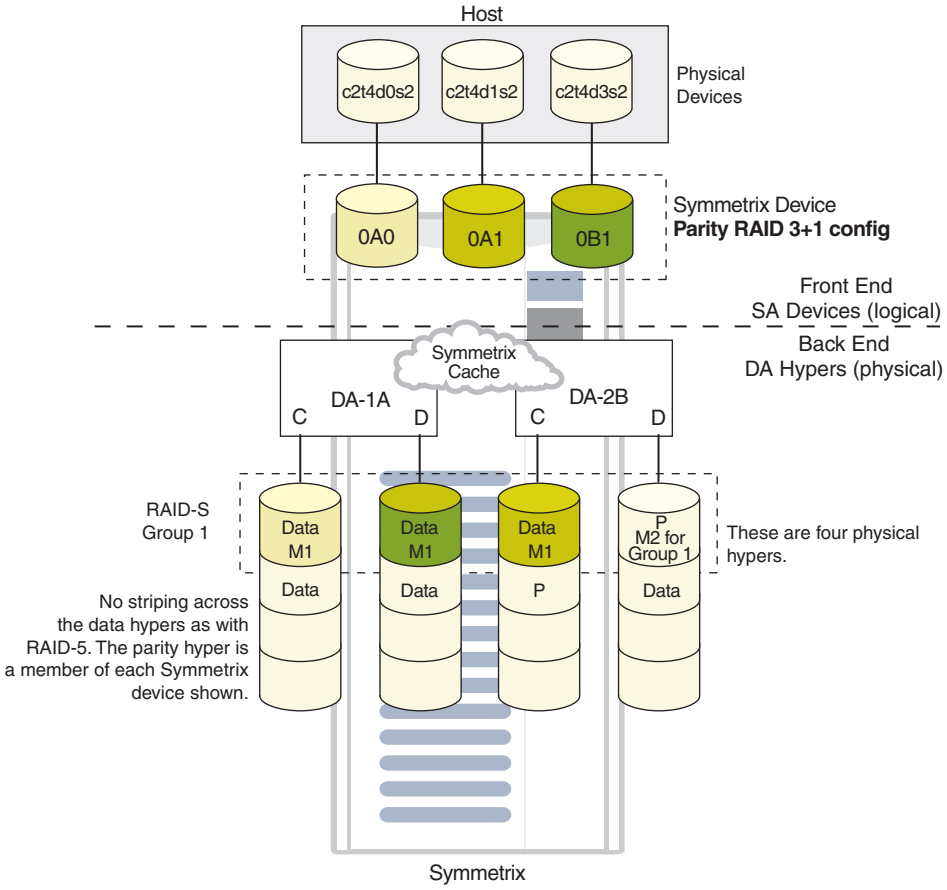


Figure 3 Symmetrix parity RAID mirror scheme

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 and BCV configurations

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) devices to synchronize and coordinate SRDF transfer activity. As shown in Figure 4, 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`.

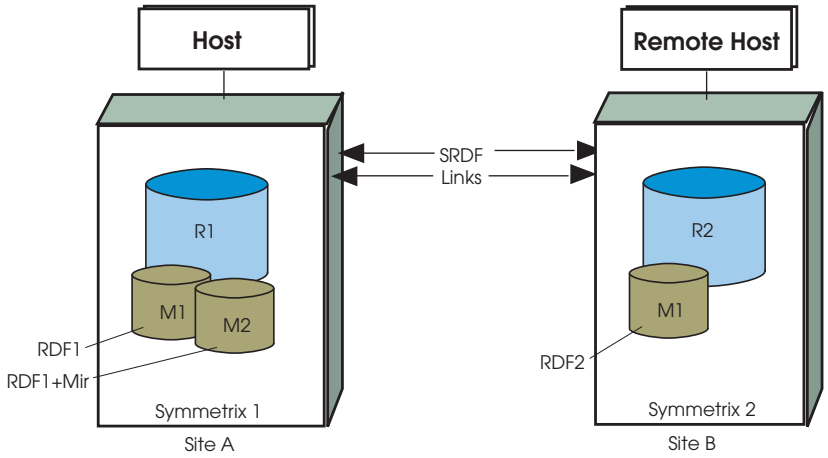


Figure 4 Simple SRDF device configuration

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 5 on page 26, 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, also can be designated R1-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.

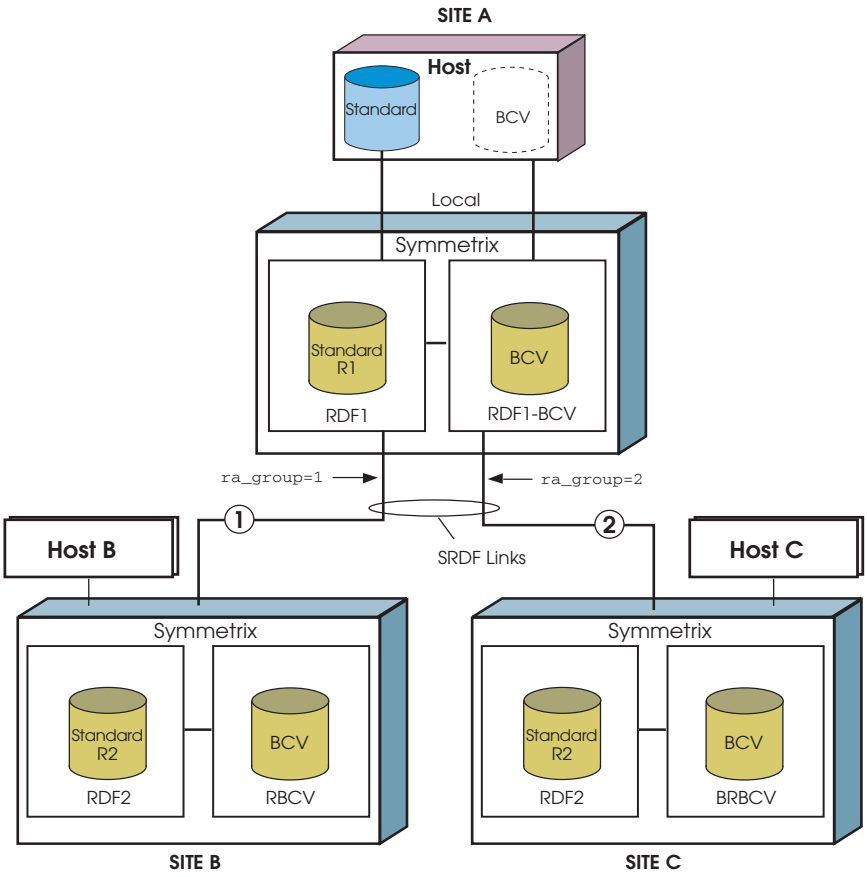


Figure 5 Various SRDF-connected device configurations with BCVs

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 6).

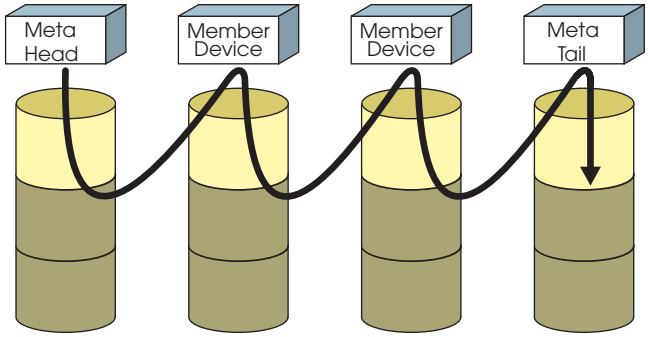


Figure 6 Concatenated metadevices

Metadevice size requirements

Metadevices may contain as many as 255 members. The maximum size metadevice that has been tested by EMC is 3.825 terabytes, which is the maximum size supported at this time. Metadevices can consist of nonsequential and nonadjacent volumes.

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 6). 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 7). When writing to a striped volume, equal size stripes of data from each participating drive are written alternately to each member of the set.

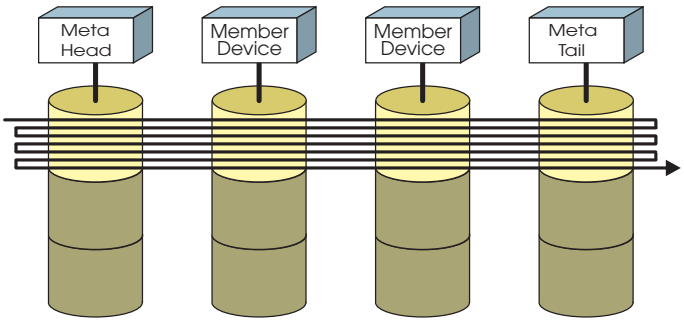


Figure 7 **Striped data**

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.

Consider the following when creating a striped metadevice:

Note: This information is important if you plan on creating a new device that will be used as a BCV or the target of a clone operation of a striped meta formed this way.

- A stripe size of 2 cylinders (which is the requirement for Enginuity Version 5*x*69 and higher) is recommended. **If you chose a device to be a striped metamember and it has an odd number of cylinders, the last cylinder will not be used.**
- When creating a striped metadevice, each member should be the same size. **Once a member is full, no further stripes will be written to any member.**

Table 2 defines the data stripe sizes and capacities for Symmetrix metavolumes.

Table 2 **Metavolume stripe sizes**

Stripe size	Stripe capacity (512 byte blocks)
2 Cylinders[a]	1,920
4 Cylinders[b]	3,840
8 Cylinders[b]	7,680
16 Cylinders[b]	15,360
32 Cylinders[b]	30,720
64 Cylinders[b]	61,440

a. The recommended default stripe size is 2 cylinders for best performance. Enginuity Version 5669 and above requires a stripe size of 2 cylinders.

b. Available with Enginuity Version 5x68 or earlier.

Configuration Manager operations

Table 3 lists supported Solutions Enabler V6.4 Configuration Manager operations for each Enginuity version.

If you are using Enginuity 5*x*68 to 5669, you must upgrade your Enginuity version as follows:
5568: Upgrade to 5568.56.22 or higher
5669: Upgrade to 5669.47.25 or higher
Failure to upgrade may result in serious side effects

Table 3 Configuration Manager operations

Configuration Manager operation	5*x*68	5669	5670	5671	5771	5772
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)		✓	✓	✓	✓	✓
Delete Device			✓	✓	✓	✓

Table 3 Configuration Manager operations (continued)

Configuration Manager operation	5x68	5669	5670	5671	5771	5772
Convert Device:						
Adding or removing BCV/DRV	✓	✓	✓	✓	✓	✓
Adding or removing RDF	✓	✓	✓	✓	✓	✓
Dynamic RDF[a]	✓	✓	✓	✓	✓	✓
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	✓	✓	✓	✓	✓	✓
Set Device Attributes:						
VCMdb	✓	✓	✓	✓	✓	✓
RDB_Cksum	✓	✓	✓	✓	✓	✓
Worm	✓	✓	✓	✓	✓	✓
dyn_rdf[a]	✓	✓	✓	✓	✓	✓
dyn_rdf1_only[a]	✓	✓	✓	✓	✓	✓
dyn_rdf2_only[a]	✓	✓	✓	✓	✓	✓
SCSI3_persist_reserv	✓	✓	✓	✓	✓	✓
Map Device:	✓	✓	✓	✓	✓	✓
AS/400	✓	✓	✓	✓	✓	✓
CKD				✓	✓	✓

Table 3 Configuration Manager operations (continued)

Configuration Manager operation	5x68	5669	5670	5671	5771	5772
Unmap Device:	✓	✓	✓	✓	✓	✓
AS/400	✓	✓	✓	✓	✓	✓
CKD			✓	✓	✓	✓
Manage PAV Alias			✓	✓	✓	✓
Form Meta	✓	✓	✓	✓	✓	✓
Add Meta Member:				✓		
Concatenated	✓	✓	✓	✓	✓	✓
Striped[a]	✓	✓	✓	✓	✓	✓
Remove Meta Member:				✓	✓	✓
Concatenated	✓	✓	✓	✓	✓	✓
Striped[b]						
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				✓	✓	✓

Table 3 Configuration Manager operations (continued)

Configuration Manager operation	5x68	5669	5670	5671	5771	5772
rdfa_host_throttle_time				✓	✓	✓
hot_swap_policy					✓	✓
Add Dynamic Spare	✓	✓	✓	✓	✓	
Remove Dynamic Spare	✓	✓	✓	✓	✓	
Enable/Disable RA Group for RDFA Operation			✓			
Create/Delete Save Pools				✓	✓	✓
Add/Remove/Set Snap Pool Members				✓	✓	✓

a. Symmetrix 8000-series or higher.

b. A striped metamember cannot be removed.

Configuration Manager device emulation

Table 4 lists supported Solutions Enabler V6.4 Configuration Manager operations that you can perform on a device with various emulation types

Table 4 Configuration Manager device emulation

Configuration Manager operation	FBA FBA Celerra® VME 512	CKD	AS/400
Create Device	✓	✓	✓
Create CKD Meta Device		✓	
Convert Device:			
Adding or removing BCV/DRV/Standard	✓	✓	✓
Adding or removing RDF	✓	✓	✓
Adding another mirror	✓	✓	✓
Removing a mirror	✓	✓	✓
Set Device Emulation	✓		
Set Device Attributes:			
VCMdb	✓		✓
RDB_Cksum	✓		
Worm	✓		
dyn_rdf	✓	✓	✓
dyn_rdf1_only	✓	✓	✓
dyn_rdf2_only	✓	✓	✓
SCSI3_persist_reserv	✓		

Table 4 Configuration Manager device emulation (continued)

Configuration Manager operation	FBA FBA Celerra® VME 512	CKD	AS/400
Map Device	✓	✓	✓
Unmap Device	✓	✓	✓
PAV Aliases	✓	✓	✓
Form Meta	✓		✓
Add Meta Member:			
Concatenated	✓		
Striped	✓		
Dissolve Meta	✓		✓
Convert Meta	✓		✓

Device creation strategy

Creation of certain device (configuration) types may require more than one class of command execution. Table 5 describes the classes or steps 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

Desired device type	Class - step 1 `create`	Class - step 2 `convert`
2-Way-BCV-Mir	2-Way-BCV-Mir	
2-Way-Mir	2-Way-Mir	
3-Way-Mir	3-Way-Mir	
4-Way-Mir [a]	4-Way-Mir	
BCV	BCV	
DRV	DRV	
Parity RAID	RAID-S[b]	
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	→RDF1-BCV
RDF1-BCV+R-5	RDF1+R-5	→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

Table 5 Steps to create a device (continued)

Desired device type	Class - step 1 `create`	Class - step 2 `convert`
RDF2-Mir	RDF2-Mir	
Unprotected	Unprotected	
Virtual	VDEV	

a. 4-Way-Mir devices cannot have any BCV devices.

b. In multiples of three or seven.

For information on creating dynamic-RDF capable devices, refer to "Configuring dynamic RDF-capable devices" on page 64.

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 are created following the same procedure as for standard protected devices, with the additional specification of `attribute=savedev`.

Device creation restrictions

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.

The creation of metadevices often requires more than one configuration change session. Some require three sessions (`create`, `form`, and `convert`) as shown in Table 6.

Table 6 Steps to create a metadevice

Desired device type	Session 1 `create`	Session 2 `form`	Session 3 `convert`
Meta 2-Way-BCV-Mir	2-Way-BCV-Mir	→Meta 2-Way-BCV-Mir	
Meta 2-Way-Mir	2-Way-Mir	→Meta 2-Way-Mir	
Meta 2-Way-Mir-RDF	2-Way-Mir	→Meta 2-Way-Mir	→Meta 2-Way-Mir-RDF
Meta 3-Way-Mir	3-Way-Mir	→Meta 3-Way-Mir	
Meta 4-Way-Mir	4-Way-Mir	→Meta 4-Way-Mir	
Meta BCV	BCV	→Meta BCV	

Table 6 Steps to create a metadevice (continued)

Desired device type	Session 1 `create`	Session 2 `form`	Session 3 `convert`
Meta Parity RAID	RAID-S[a]	→Meta Parity RAID	
Meta RAID 5	RAID 5	→Meta RAID 5	
Meta RAID 6	RAID 6	→Meta RAID 6	
Meta RAID 5 BCV	RAID 5 BCV	→Meta RAID 5 BCV	
Meta RAID 6 BCV	RAID 6 BCV	→Meta RAID 6 BCV	
Meta RDF1	Unprotected	→Meta Unprotected	→Meta RDF1
Meta RDF1+R-5	RAID 5[b]	→Meta RAID 5	→Meta RDF1+R-5
Meta RDF1+R-6	RAID 6[b]	→Meta RAID 6	→Meta RDF1+R-6
Meta RDF1+R-S	RAID-S[a]	→Meta Parity RAID	→Meta RDF1+R-S
Meta RDF1-BCV	BCV	→Meta BCV	→Meta RDF1-BCV
Meta RDF2	Unprotected	→Meta Unprotected	→Meta RDF2
Meta RDF2+R-S	RAID-S[a]	→Meta Parity RAID	→Meta RDF2+R-S
Meta RDF2-BCV	BCV	→Meta BCV	→Meta RDF2-BCV
Meta VDEV	VDEV	→Meta VDEV	

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` command 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 6 on page 35), 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 converted to a non-RDF device first.

Virtual device 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 metadevice. 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 unprotected[1], mirrored, or RAID 5.[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.

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.

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

Changes to mirror protection

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 via 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.

Configuration control operations overview

The `symconfigure` command and the `symrdf swap` command allow the *advanced* user more powerful control to manage various types of configuration changes affecting Symmetrix devices than otherwise provided by the SYMCLI set.

Most SYMCLI commands, including the `symconfigure` command, can operate on Symmetrix systems and devices that are either directly attached to the user's host, or are accessible via an RDF link.

CAUTION

Before issuing a `symconfigure` command to a Symmetrix 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 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.

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 "Stages of the configuration change action" on page 43.

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.

Operations classes

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

Note: For information on the Enginuity requirements for the configuration control operations, refer to Table 3 on page 29.

Creating devices
- Creates new Symmetrix devices.
- Configures new Symmetrix devices by copying existing devices.

Converting a device's configuration
- Converts a device configuration by adding or removing BCV or DRV attributes.
- Converts an RDF device from static RDF to dynamic RDF.

Changing a device's protection
Increases or decreases device protection by adding or removing a mirror(s).

Deleting devices
Deletes Symmetrix devices.

Configuring RDF
- Adds/removes RDF attributes.
- Swaps RDF source/target attributes for an RA group.
- Enables/disables SRDF/A on an RA group.
- Sets RA group attributes.
- Converts static RDF device to dynamic.

Managing metadevices
- Forms/dissolves metadevices.
- Adds/removes metamembers.
- Converts from concatenated to striped and vice versa.
- Modifies stripe size.

Managing SAVEDEV pools
- Adds/deletes pools
- Adds/removes pool members
- Enables/disables pool members

Managing spare disks
- Adds a new spare disk.
- Removes an existing spare disk.

Mapping devices to ports
- Maps/unmaps a Symmetrix device to a front-end port.
- Maps/unmaps a range of Symmetrix devices that form a mainframe CU image to a front-end port.
- Copies device mappings from one EA/EF port to another.
- Add or remove PAV aliases from mainframe devices.

Setting device attributes
- Sets the devices as available for use as dynamic RDF devices, worm, VCM database devices, or RDB checksum devices.
- Enables the SCSI3 persistent reservation option for clustered devices.

Setting device emulation
Sets the device emulation type.

Setting front-end port attributes

- Sets/resets SCSI or fibre port flags.
- Sets fibre FA loop ID.
- Sets hostnames of port connections.
- Sets Gig-E front-end port IP addresses and netmasks

Setting RDF (RA) group parameters

- Sets the session priority.
- Sets the minimum cycle time.
- Enables/disables RDF/A transmit idle function.
- Enables/disables RDF/A DSE autostart.
- Sets RDF/A DSE pool emulation type.
- Sets RDF/A DSE threshold.

Setting the Symmetrix metrics

Sets the metrics of Symmetrix configuration.

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.

Configuration change safety

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:

1. 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
   ```

2. 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
   ```

3. 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 device(s) 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.

4. 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.

5. 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 A, "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.

The command file

For the configuration change session, you must define a set of changes within a specified command file. The `symconfigure` command file format contains various command entries terminated with a semicolon (;). Multiple changes can be made is one session, but all changes must fall into one operation class. Command file examples are provided with the actions described later in this chapter.

Note: Starting with Enginuity Version 5669, changes for multiple operation classes can be executed within the same session, except for dynamic RDF changes, SAVE device pool changes, and expanding stripe metadevices with protection.

Stages of the configuration change action

A lock on the Symmetrix configuration is engaged by the Symmetrix configuration server during the configuration change session. Note that it does not lock out normal I/O activity, it simply blocks others from attempting to change the configuration.

A session acting on a specified `symconfigure` command file is processed in a series of progressive stages that can be monitored using polling options:

- preview
- prepare
- commit

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 stage

The `preview` argument (stage) 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).

To preview the command file that creates a concatenated metadevice in Symmetrix `12345`, using device `0010` as the metahead, and `0011` - `0013` as members, enter:

```
symconfigure -sid 12345 preview -file add_meta.cmd -noprompt -v
```

where `add_meta.cmd` contains:

```
form meta from dev 0010, config = concatenated;

add dev 0011:0013 to meta 0010;
```

Prepare stage

The `prepare` argument (stage) will perform the preview checks and also verify the appropriateness of the resulting configuration definition against the current state of the Symmetrix arrays; the argument then terminates the session without change execution (correct within the realm of both the host and the Symmetrix array). The `prepare` argument has no function for SAVEDEV pool sessions.

To prepare the command file that adds four new Symmetrix devices to Symmetrix array 000000012345 as two-way mirrored devices with a size of 1100 cylinders, using FBA emulation, enter:

```
symconfigure -sid 12345 prepare -file add_new_symdevs.cmd -noprompt -v
```

where `add_new_symdevs.cmd` contains:

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

Commit stage

The `commit` argument (last stage) completes all stages and activates the changes in the specified Symmetrix array.

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, 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. (For specific impact, see the reference to this option in each of the Class descriptions.)

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
```

Commit example

To `commit` the command file that creates a concatenated metadevice in Symmetrix 12345, using device 0010 as the metahead, and 0011 - 0013 as members, enter:

```
symconfigure -sid 12345 commit -file add_meta.cmd -noprompt -v
```

where `add_meta.cmd` contains:

```
form meta from dev 0010, config = concatenated;

add dev 0011:0013 to meta 0010;
```

Creating 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,
   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
    [in pool PoolName] [member_state=ENABLE | DISABLE], ]
   [, disk_group_num=nnn, remote_disk_group_num=nnn];
```

where:

count = The number of devices to create.

size = The size of the device needed in number of cylinders. In Symmetrix DMX models, a cylinder of an FBA device contains 1920 512-byte blocks, or .98 megabytes. In earlier Symmetrix models, a cylinder of an FBA device contains 960 512-byte blocks or, .468 megabytes.

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

blocks ÷ 960
or
(devicesize in megabytes) x 1024^2 ÷ (960 x 512 bytes)
or
(size in megabytes) x 2.133

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 = The device emulation type, which currently must be Fixed Block Architecture (FBA), CELERRA FBA, VME512 FBA, or CKD-3380 or CKD-3390 for z/OS environments. For a list of supported AS/400 environments, refer to the *symconfigure* description in the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

config = The desired device configuration type. Possible values are shown in the second column of Table 5 on page 34.

data_member_count=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 = The desired remote SRDF configuration (if any) from Table 5 on page 34.

remote_data_member_count =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 = 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 = 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.

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
dyn_rdf	dyn_rdf
dyn_rdf1_only	dyn_rdf2_only
dyn_rdf2_only	dyn_rdf1_only

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 92.

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

```
symcfg -ssid list
```

attribute = The desired device attribute. Possible values are:

`ckd_meta` — When creating a device with emulation type of CKD-3380 or CKD-3390, this indicates that the device should be a striped metadevice. CKD metadevices 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.

PoolName = The name of the SAVE device 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* devices.

member_state = Whether the SAVE device(s) being added should be enabled or disabled in the pool.

disk_group_num = When creating a device, this option allows you to specify a disk group. 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_num = When creating a device, this option allows you to specify a remote disk group.

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 hypers to support the new devices in disk group 3, create a command file 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 SymmID -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 44.

Restrictions on creating devices

There are restrictions on the use of unprotected devices created by the `symconfigure` utility, 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.

Configuring disk space

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=Cylinders] [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.

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 = The size in cylinders of the new devices.

emulation = The emulation type of the new devices.

config = The configuration type of the new devices. Possible values are shown in the second column of Table 5 on page 34.

data_member_count =The 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 92.

disk_group = 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;
```

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.

Example: deleting devices

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, cmd -v -noprompt commit
```

Example: deleting a RAID-S set

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.

- 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 device[1]
  - WORM-protected
  - The VCM database device
  - Masked by VCM
- 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.

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=CELERRA_FBA,]
   [ra_group=n, remote_dev=SymDevName,
   invalidate=invalidate_option,
   [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 = 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 = The desired device configuration type. Possible values are shown the first column of Table 5 on page 34.

emulation=CELERRA_FBA = 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.

ra_group = The RA group number in the SRDF environment.

remote_dev = 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 = The RDF device (R1 source or R2 target) to invalidate. An invalidated copy requires a full refresh (synchronization of all tracks) at some point. If NONE is selected, the RDF device must be specified later using the symrdf command.

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 = 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.

Example: Converting BCV Devices

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;
```

Example: Converting RDF devices

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

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

Example: Converting 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.

Example: Converting a 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.

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 44.

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 Enginuity Version 5669, changes for multiple operation classes can be executed within the same session, except for dynamic RDF changes and SAVEDEV 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.
- Enginuity Versions below 5669 cannot create R1/R2 pairs if the R2 is larger than the R1.

  Enginuity 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.

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 = The device emulation type, which can be fixed block architecture selections: FBA, CELERRA FBA or VME512 FBA.

attribute = A device attribute that restricts how a device can be accessed. Possible values include:

- RDB_Cksum
- VCMdb (for device masking)
- WORM (can be enabled only)*
- 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**

* To enable WORM on an individual device, it must first be enabled on the Symmetrix array. Contact an EMC representative for assistance.

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

Example: Setting emulation

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

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

Example: Setting attribute

To enable five devices (0015 to 0019) for WORM protection, enter the following in the command file:

```
set device 0015:0019 attribute=Worm;
```

Example: Resetting attribute

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 44.

You can set emulation and attributes in the same session (command file). For more information on classes, refer to "Operations classes" on page 40.

Note: If you are running Enginuity Version 5668 or lower, you cannot execute changes for multiple operation classes within the same session.

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.

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, providing 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 Symmetrix CLI Command Reference*.

Reserving devices

To reserve devices, use the following form:

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

where:

ExpirationDate = 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 = 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 = The name of the owner of the reservation (up to 31 characters long).

Usercomment = 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[ile] 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  Date         Flags
ID       Reserved     TM    Owner          Devices    Port Addresses
-------  -----------  ----- -------------  ---------  ---------------------------------
000001   09/27 11:51 E-    Kevin          01A5            -
  Comment: Testing metadevice 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 = The device reservation ID.

Example To release the set of devices reserved in the example on page 58, enter:

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

Mapping devices

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

where:

awwn = The user-given name or alias WWN of a host HBA port, if updating a VCM database.

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.

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.

iscsi = The iSCSI name, if updating a VCM database.

lun = The SCSI logical unit number (hex value).

starting = The starting address for the range of devices.

target = The SCSI target ID (hex value).

vbus = The virtual bus (vbus) address for mapping to an FA port if using volume set addressing.

wwn = The unique 64-bit World Wide Name (WWN) identifier for an HBA port, if updating a VCM database.

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 "Updating the Host" on page 261. 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 44.

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.

Example: Unmapping class

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 "Updating the Host" on page 261. After the host environment is updated, I/O activity can resume with the Symmetrix device.

Configuring dynamic RDF-capable devices

With Enginuity Version 5568 and higher, devices can be configured via `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.

The *dynamic RDF* feature must be enabled in the Symmetrix array. If using Enginuity Version 5568, configuration options can be done using the `symconfigure` command to mark standard devices as being capable of supporting dynamic RDF operations.

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.

If using Enginuity Version 5567, dynamic RDF operations are limited to swap only and the Symmetrix configuration can be changed only by EMC.

Note: Dynamic swap is not supported if Enginuity Versions 5567 and 5568 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 Symmetrix array

To enable dynamic RDF in a Symmetrix array, create a command file containing the following option:

```
set symmetrix dynamic_rdf = enable;
```

Then activate the option using the following format:

```
symconfigure -sid SymmID -file set_sym_dyn_rdf.cmd -noprompt -v
   commit
```

The current setting of this metric can be viewed using the following format:

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

Sample output

```
Symmetrix ID: 000387940054
Time Zone   : EDT

    Product Model                          : 800-M2
    Symmetrix ID                           : 000387940054

    Microcode Version (Number)             : 5671 (16270000)
    Microcode Date                         : 10.11.2005
```

```
Microcode Patch Date                  : 10.11.2005
Microcode Patch Level                 : 42

Cache Size                            :   32768 (MB)
# of Available Cache Slots            :  961074
# of PermaCache Slots In Use          :       3
Max # of System Write Pending Slots   :  160000
Max # of DA Write Pending Slots       :   80000
Max # of Device Write Pending Slots   :    4706

Symmetrix Total Operating Time        : 219 days, 01:22:37
Symmetrix Power ON Time               : Mon Mar 14 10:35:06 2005
Symmetrix Last IPL Time (Cold)        : Mon Jun 27 14:31:29 2005
Symmetrix Last Fast IPL Time (Hot)    : Mon Oct 17 12:37:53 2005

Host DB Sync Time                     : Wed Oct 19 12:43:44 2005
Symmetrix CLI (SYMCLI) Version        : T6.1.0.35 (Edit Level: 691)
Built with SYMAPI Version             : T6.1.0.35 (Edit Level: 691)
SYMAPI Run Time Version               : T6.1.0.35 (Edit Level: 691)

Number of Configured (Sym) Devices    : 446
Number of Visible (Host) Devices      :  91
Number of Configured Actual Disks     :  55
Number of Configured Hot Spares       :   0
Number of Unconfigured Disks          :   0
Maximum number of hypers per disk     :  74

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      : 148A84
Switched RDF Configuration State      : Enabled
Concurrent RDF Configuration State    : Disabled
Dynamic RDF Configuration State       : Enabled
Concurrent Dynamic RDF Configuration  : Disabled
RDF Data Mobility Configuration State: Disabled
Access Control Configuration State    : Disabled
Device Masking (VCM) Config State     : Mixed
VCMdb Access Restricted State         : Disabled
Multi LRU Device Assignment           : BY_NUMBER
Disk Group Assignments                : In Use

Parity Raid Configuration             : RAID-S (7+1)
Raid-5 Configuration                  : RAID-5 (7+1)
PAV Mode                              : NoPAV
PAV Alias Limit                       : 15

SRDF/A Maximum Host Throttle (Secs)   : 0
SRDF/A Maximum Cache Usage (Percent)  : 94
```

Enabling dynamic RDF on the device

Next, identify the nonRDF 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.

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.

Example 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;
```

Configuring metadevices

You can form or dissolve a metadevice, or add or remove a metamember. To initially create a metadevice, the head and members must be specified in the same command file. You can add or remove metamembers from an existing concatenated metadevice, or add members to a striped metadevice. You can also dissolve a specified metadevice, which subsequently releases all its members for normal use as stand-alone devices.

Creating a metadevice

As mentioned in the introduction, 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.

First, to form a metadevice, use the following command:

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

where:

`config` = The meta configuration type. Possible values are `concatenated` or `striped`.

`stripe_size` = The size of the striped metadevice. 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 Enginuity Version 5x69 and later, which is the default for all versions of Enginuity. 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 28.

`cyl` = If the `stripe_size` is expressed in cylinders, include `cyl` with the *`MetaStripeSize`* value. The size of a cylinder for FBA emulation is 960 512-byte blocks.

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

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), since metadevices must be unmapped for this operation. For information about enabling this option, refer to "Enable SYMAPI environment option" on page 44.

Adding metamember(s)

Next, add the command specifying the new members.

Note: Omit this step if the `count` option was used previously with the `form meta` command.

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

Example: Forming a concatenated meta

To form a concatenated metadevice 0010 and add metamembers, enter:

```
form meta from dev 0010, config = concatenated;

add dev 0011:0012 to meta 0010;
```

Example: Forming a striped meta

To form a striped metadevice 0010 and add metamembers, enter:

```
form meta from dev 0010, config = striped,
   stripe_size=1920;

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

Note: It is recommended that you add all the metamembers in the same session (same command file) where you form a striped metadevice. You must add at least one metamember in the same session where you form a concatenated device.

To create a striped metadevice by requesting the configuration server to form the metadevice with four members; the metahead 0030 and three other members selected from the pool of unmapped devices, enter the following:

```
form meta from dev 0030, config = striped,
   stripe_size=1920, count=4;
```

Restrictions

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 6 on page 35 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 metadevice, 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 metadevice, 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 via 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 metadevice, if the data on the metadevice is to be protected, you must specify the name of a `bcv_meta` that matches the original metadevice 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 metadevice, 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 metadevice

To dissolve a metadevice, use the following form:

```
dissolve meta dev SymDevName;
```

Example To dissolve metadevice 0010, enter:

```
dissolve meta dev 0010;
```

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

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` = 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` = The Symmetrix device name.

Metadevices can be converted from concatenated metadevices to striped devices, and vice versa 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.

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 meta.

Table 7 lists the possible data management operations and their effects on data integrity.

Table 7 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]
Metadevice striped	Converting to concatenated	Configurable[c]

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 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` = 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;
```

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];
```

Example 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;
```

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, etc.). 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 the following forms, according to the version of Enginuity:

- For Enginuity Version 5671, use the following form:

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

- For Enginuity 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;
```

Managing SAVE device pools

Beginning with Solutions Enabler V6.4, there are two types of SAVE device pools, Snap pools and SRDF/A DSE 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 offloading some or all of the cycle data from cache to pre-configured disk storage pool. These pools are configured the same as Snap pools.

The following are attributes of device pools:

- When a pool is created, it will be created as either an SRDF/A DSE pool or a Snap pool.
- SRDF/A DSE pools or Snap pools are made up of SAVE devices which are specifically configured to be used for 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.
- SAVE devices that do not belong to any pools are available to be added to a SRDF/A DSE pool or a Snap pool. SAVE devices in a pool are available for use once they are enabled.
- SAVE devices can be removed from an SRDF/A DSE pool but must be disabled and "drained." Drained devices will become inactive within their pool and can then be moved out of the Pool to be available for other pools.
- An SRDF/A DSE pool can contain only one type of device from the list of FBA, CKD3390, CKD3380, and AS400.
- 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.

This section explains how to create and delete Snap and SRDF/A DSE pools, add and remove devices from pools, as well as 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*.

Creating a SAVE device pool

To create a SAVE device pool, use the following form:

```
create pool PoolName, type = snap | rdfa_dse;
```

where:

type = The type of SAVE device pool. Snap pools are for TimeFinder/Snap operations, SRDF/A DSE pools are for SRDF/A operations.

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

When creating SAVE device pools, follow these guidelines:

- Pools are made up of SAVE devices that are specifically configured to be used for pools.
- A special pool DEFAULT_POOL exists by default and contains SAVE devices that have not been assigned to any named pools, but are available for operations. The devices in the DEFAULT_POOL are either enabled or disabled, but have used tracks.
- An SRDF/A DSE pool cannot have the same name as a Snap pool for the same Symmetrix array.
- An SRDF/A DSE pool can only contain one type of device emulation, either FBA, CKD3390, CKD3380, or AS400.
- SRDF/A DSE pools can be shared with multiple SRDF/A groups.
- A single SRDF/A group can have up to 4 SRDF/A DSE pools associated with it (one for each device emulation type).

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

The maximum number of SAVE device pools per Symmetrix is 256.

Example To create a new pool named *Finance*, enter:

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

Deleting a SAVE device pool

You can only delete a SAVE 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;
```

Example To delete a pool named *Finance*, enter:

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

Adding SAVE devices to a pool

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

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

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

where:

member_state = Whether the SAVE device(s) being added should be enabled or disabled in the pool.

Example To add SAVE devices to an SRDF/A DSE pool named *ACT20*, enter:

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

Removing SAVE devices from a pool

You can only remove disabled and inactive SAVE devices from a pool. Once you remove a device from a pool, the device is available to be added to any Snap or SRDF/A DSE pool.

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

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

Example To remove a SAVE device from a pool named *ACT20*, enter:

```
remove dev 001D from pool ACT20, type=rdfa_dse;
```

Enabling/disabling SAVE devices

You can enable or disable SAVE 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.

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

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

where:

PoolName = The name of the SAVE 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 SAVE devices in a pool named *Finance*, enter:

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

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
   [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, etc.). 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 8 on page 80, and the values for the fibre protocol flags are in Table 9 on page 81.

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.

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 8 SCSI protocol port flags

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	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[a,b]	When this flag is enabled, the Symmetrix via 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[a,b]	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[a, b]	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[a,b]	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	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 a xx3C error occurs.
Linked_Commands[a,b]	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[c]	When enabled for AS/400 hosts, this flag forces a SCSI negotiation by the Symmetrix array after a: • SCSI reset • Error • Bus device reset This flag is used for AS/400 systems only (default is off).
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.

Table 8 SCSI protocol port flags (continued)

SCSI protocol flags	Description
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_Support1	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	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	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[d]	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	When enabled for Sun PDB clusters, this flag enables the Sunapee option on the port.
Sync_Transfer[a,b]	When enabled, this flag enables SCSI synchronous negotiations. (Default is enabled, and currently, you **cannot change this flag**.)
Tagged_Commands[a,b]	When enabled, this flag enables support for tagged commands. (Default is enabled, and currently, you **cannot change this flag**.)
Wide_Transfer[a,b]	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. Can only be set on SA ports with Enginuity Version 5568 and higher.

d. Requires Enginuity Version 5670 and higher.

Table 9 Fibre protocol port flags

Fibre protocol flags	Description
AS400	This flag should be enabled for any AS400 (iSeries) hosts connecting to the port.
Auto_Negotiate	When enabled, allows two fibre ports to *handshake* 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.

Table 9 Fibre protocol port flags (continued)

Fibre protocol flags	Description
Generic_VSA[a,b]	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_Addressing[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_Addressing 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	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.)
Volume_Set_Addressing	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 Enginuity Version 68 and higher.

b. Not available for host-based configuration changes.

c. Not available for Gig-E ports.

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_Addressing=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 44.

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
```

Setting Symmetrix metrics

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

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

where:

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

`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`.

`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.

`pav_alias_limit` — If PAV is enabled, specifies the maximum number of aliases that can be assigned to a device. Possible values for a Symm 6 are 1-7 using 5670, 1-15 using 5671.

`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` — The percentage of write pending cache that can be used by RDFA. This is a value from 0 to 100 percent.

`rdfa_host_throttle_time` — 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.

Example To change the maximum number of hypers that can be created on a Symmetrix physical disk from 32 to 128, enter in the command file:

```
set Symmetrix max_hypers_per_disk=128;
```

Viewing the Symmetrix metrics

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

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

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.

For information about swapping R1/R2 device designations, refer to "Swapping R1 devices with R2 devices" on page 89.

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 44.

Managing 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 hypers of a Symmetrix disk that has failed.

Adding a spare disk

A spare disk can only be created from a new disk containing no hypers. 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. Select a format based on the type of disk it should be able to replace.

- For Symmetrix DMX models:
  - for CKD and FBA use 512
  - for AS/400 and Tandem use 520
- For Symmetrix 5 models:
  - for CKD and FBA use 512
  - for AS/400 and Tandem use 520
- For Symmetrix 4 models:

  With selective LLF enabled:
  - for CKD and FBA use 512
  - for AS/400 and Tandem use 520

  If no AS/400 or Tandem devices:
  - for CKD and FBA use 512

  If AS/400 or Tandem devices:
  - for CKD use 512
  - for FBA, AS/400, and Tandem use 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.

Swapping R1 devices with R2 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: 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 44.

Monitoring the status of the change session

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.

Example 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.

Aborting the change session

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

Example To abort a change session on Symmetrix 12345, enter:

```
symconfigure -sid 12345 abort
```

Note: 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.

Note: 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.

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
```

2

Device Masking

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 94
- Supported topologies 97
- Device masking VCMDB security enhancement 99
- Initial device masking setup 100
- Device masking configuration 101
- Discovering host HBAs 104
- How to add and remove masked devices 106
- Device masking VCMDB maintenance 110
- How to view the VCMDB 114
- HBA initiator management 118
- Fibre Channel-to-host interface management 120

Device masking overview

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 64 unique WWNs or 128 iSCSIs (beginning with Enginuity Version 5670). 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 example

When a host attempts to access a Symmetrix storage device, as shown in In Figure 8, 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.

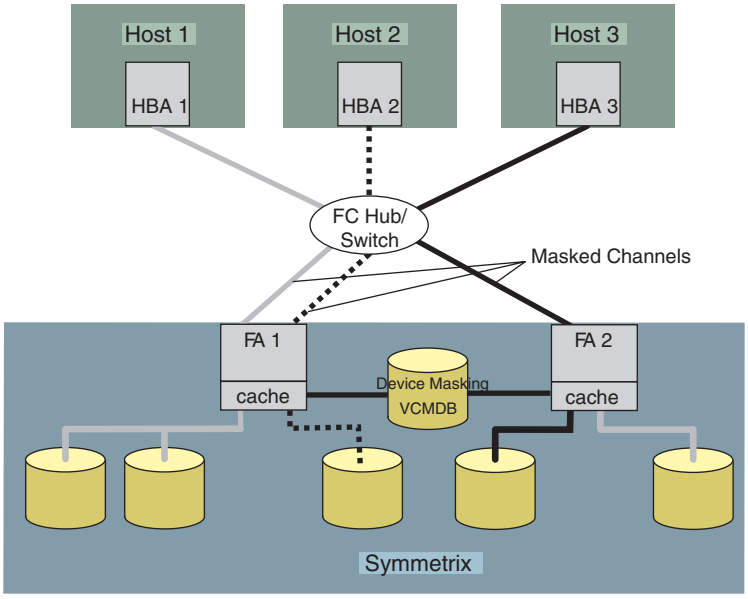


Figure 8 **Device masking solution with WWN initiators**

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 8, 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.

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 optons, refer to the *EMC Solutions Enabler Symmectrix CLI Command Reference*.

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
```

Mapping volume logix commands

If you have been using EMC's Volume Logix and need to convert to the SYMCLI device masking command set, refer to Appendix B, "Device Masking: Volume Logix Conversion."

Supported topologies

Device masking supports Fibre Channel (point-to-point, arbitrated loop, and switch fabric) and native iSCSI topologies.

Fibre Channel topology

Figure 9 illustrates two different Fibre Channel network configurations that device masking supports:

- Point-to-Point — Network A (left) is configured with a 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 via a common fabric.

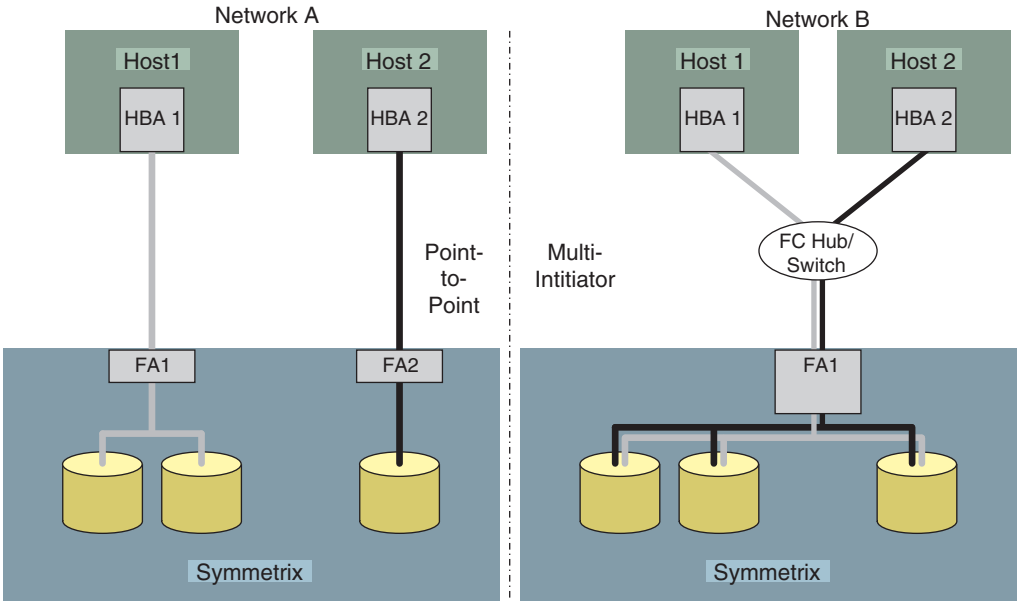


Figure 9 Point-to-point and multi-initiator topologies

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 10.

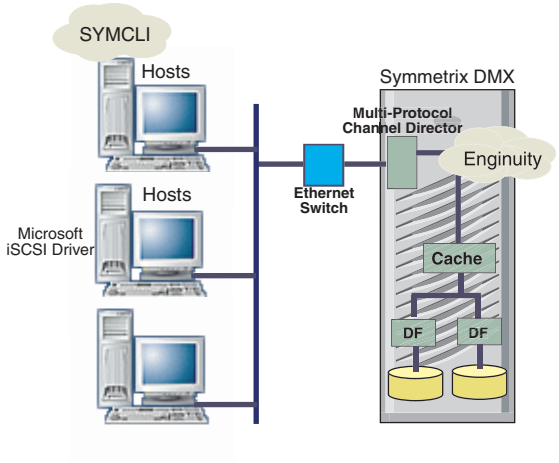


Figure 10 Native iSCSI topology

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 137 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 104 for more information about WWN and iSCSI initiators.

Device masking VCMDB security enhancement

By default, the device masking VCMDB is accessible to all HBAs that log into 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 to 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 11). Any host with access to the VCMDB could update the database.

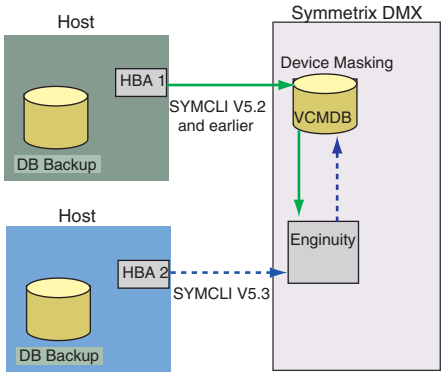


Figure 11 Device masking syscalls

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 setup your device masking environment, you should initialize the device masking VCMDB (see "Initializing the database" on page 110). This should only be done on an inital 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 VCMDB would fail.

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
2. Initialize and update the database
3. Enable authentication
4. Recommendations for activating the configuration

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 12 shows device masking components to identify.

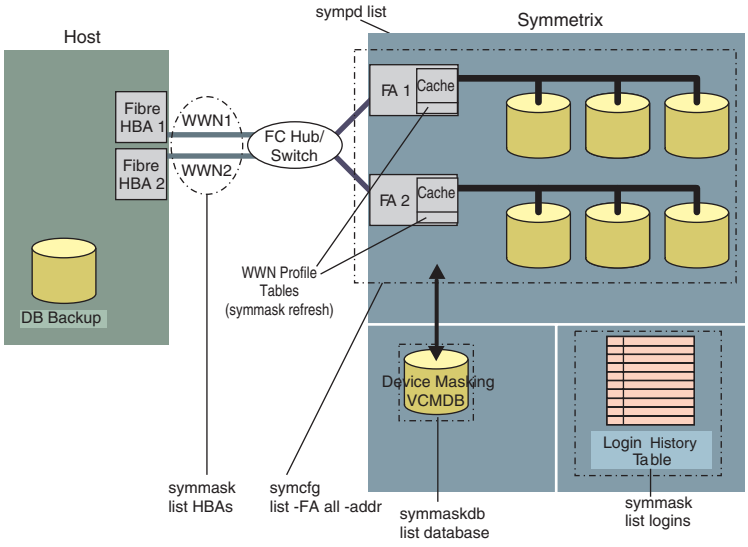


Figure 12 Device masking components

The following commands can be used to obtain the dentifiers 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 110; and for more information on supports database types, refer to "VCMDB database types" on page 111.

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 3, "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. Backup 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, 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/AISCSI 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/AISCSI) 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/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 118.

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
add devs startSymDevname:endSymDevname|SymDevname|SymDevname,,,...\
-dir # -p # [-noprompt]
```

For example, to add devices 0014 and 0015 on Symmetrix 0128 for access to Host3b using director 16a, port 0, enter:

```
symmask -sid 0128 -awnn Host3b/4a add devs 0014,0015 -dir 16a -p 0
```

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 -awnn Host3b/4a add dev 0014 -dir 16a -p 0 -reserve_id 5
```

"Managing device reservations" on page 57 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:

```
symmask -sid SymmID -wwn wwn|-awwn awwn|-iscsi iscsi |-aiscsi aiscsi
add -g DgName [-std] [-bcv] [-vdev] [-tgt] [-noprompt] -dir # -p # [-noprompt]
```

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 -awnn 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 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 startSymDevname:endSymDevname|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` using LUN addresses 0 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 startSymDevname:endSymDevname|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 VCMDB, 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 -awnn 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

This 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 57*xx* 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 Enginuity 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 Enginuity Version 5671, a Type 5 device masking VCMDB is supported, and Enginuity 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 VVCMDB 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 `-vcm_type` option and specify the type: 4 or 5. For example , enter:

```
symmaskdb -sid 0128 restore -file MyCvrtBU -vcm_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 Enginuity 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 you 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,@2      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,@2      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

                                        User-generated
Identifier        Type   Node Name        Port Name         Devices
----------------  -----  ---------------------------------  ---------
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

                                  User-generated
Identifier        Type   Node Name        Port Name         Devices
----------------  -----  ---------------------------------  ---------
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                                                  LUN
Name    Dir:P  Physical Device Name     VBUS  TID  SYMM HOST  Attr  Cap(MB)
------  -----  ----------------------  ----  ---  ---- ----  ----  -------
0040     2A:1  /dev/rdsk/c1t0d1s2          0    0    1    1          187
0041     2A:1  /dev/rdsk/c1t0d2s2          0    0    2    2          187
0042     2A:1  /dev/rdsk/c1t0d3s2          0    0    3    3          187
0043     2A:1  /dev/rdsk/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 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

                              User-generated                        Logged On
Identifier       Type  Node Name         Port Name         FCID    In     Fabric
---------------- ----- ---------------------------------- ------ ------ -----
10000000c9238053 Fibre api145            i@1f,4000,@2      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.

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 VCMDB.

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 VCMDB 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 VCMDB:

```
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` *`SymmID`* `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` command, 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 VCMDB 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 VCMDB. 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 VCMDB, 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 VCMDB is visible, run `symmask list hba` to find the device path of the HBA you want to protect.

   Note: If the VCMDB 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" later on this page).
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" section.

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 5*x*68 minimum) provides the ability to specify a LUN *base* and an *offset* hexidecimal 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, (via 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 10 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 10 LUN base/offset scenarios for multiple hosts with skip holes

	Host A LUNs	Host B LUNs	*base*	*offset*
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 11 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.

To set (or reset) the HBA flags, use the following form:

```
symmask -sid SymmID set hba_flags
<on <flag,flag,flag...> <-enable | -disable> | off>
-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 overrided 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:

```
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 superceeded by Setting the HBA port flags. The heterogeneous host configuration types listed in Table 11 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 11, (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 11.

Table 11 Host platforms and interface configuration flags

Host platform	Requirements	Bit[a]	Host configuration flag (*HostConfigFlag*)
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
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

Table 11 Host platforms and interface configuration flags (continued)

Host platform	Requirements	Bit[a]	Host configuration flag (*HostConfigFlag*)
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_EMC
IBM AIX with EMC Fibre Channel	PowerPath V1.5.x or earlier	D	IBM_EMC_PP15
Linux			LINUX
Linux		C	LINUX_DMP
Linux			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
Sun	PowerPath V1.5.x or earlier	D	SOLARIS_PP15
Sun	VERITAS DMP	C, D	SOLARIS_DMP

Table 11 Host platforms and interface configuration flags (continued)

Host platform	Requirements	Bit[a]	Host configuration flag (*HostConfigFlag*)
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

d. 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 5670 or lower
0S2007	Enable SCSI_Support1

3

Device Masking: iSCSI Setup

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 130
- iSCSI CHAP authentication: Enginuity version 56xx 131
- iSCSI CHAP authentication: Enginuity Version 57xx 133
- iSCSI software driver configuration 137
- RADIUS server configuration 151

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 56*xx***: For details on configuring iSCSI authentication, refer to "iSCSI CHAP authentication: Enginuity version 56xx" on page 131.
- **Enginuity Version 57*xx***: For details on configuring iSCSI authentication, refer to "iSCSI CHAP authentication: Enginuity Version 57xx" on page 133.

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 151.

iSCSI CHAP authentication: Enginuity version 56*xx*

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 higher)

  Note: If you are running Enginuity Version 57*xx* (5771 and higher), refer to "iSCSI CHAP authentication: Enginuity Version 57xx" on page 133.

- 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 137.

Setting iSCSI authentication via 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 57*xx* (5771 and higher), requires the use of the `symconnect` command instead of the `symmask set authentication` command, which is valid only for Enginuity Version 56*xx* (5670 and higher). For information about using `symconnect` commands, refer to "iSCSI CHAP authentication: Enginuity Version 57xx" on page 133.

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 137.

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 57*xx*

Beginning with Solutions Enabler version 6.0.2 running on Symmetrix arrays using Enginuity Version 5771, connection security for iSCSI ports is set via 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 151.

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" 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 2-way chap authentication" on page 145.

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 via 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 via iSCSI between the host computer and the designated Symmetrix director port with/without authentication.
   - Have the host computer logon to the Symmetrix via 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 138, or without authentication as described in "Configuring iSCSI without CHAP authentication" on page 146. 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.

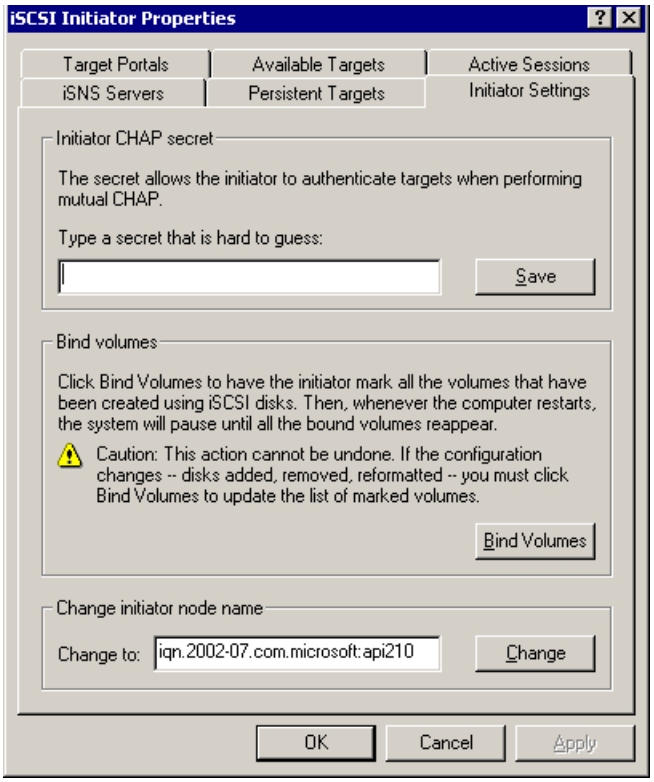
Configuring iSCSI with CHAP authentication

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_name -dir # -p # add dev #
```

where:

- *SymmID* = The Symmetrix ID.
- *iscsi_name* = The iSCSI name (from step 3).
- -dir # = Symmetrix director number.
- -p # = Symmetrix port number.
- add dev # = Symmetrix device number(s).

For 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_name
   set chap -cred Credential -secret Secret
```

where:

  - *SymmID* = The Symmetrix ID.
  - *iscsi_name* = The host iSCSI name.
  - *Credential* = 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_name
   set authentication -type CHAP
   -credential Credential -secret Secret
```

where:

  - *SymmID* = The Symmetrix ID.
  - *iscsi_name* = The host iSCSI name.
  - CHAP = The authentication type.
  - *CHAPcredential* = Eight alphanumeric characters.
  - *CHAPsecret* = 12-16 alphanumeric characters.

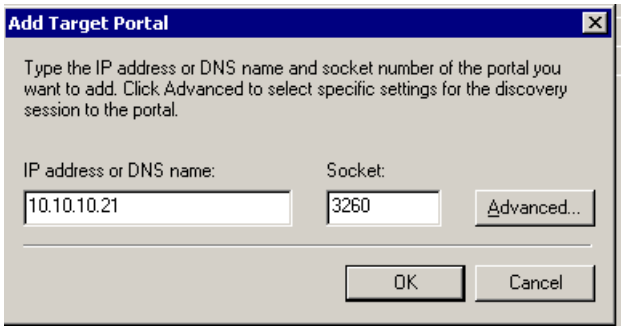
Note: For details about setting iSCSI Authentication, refer to "iSCSI CHAP authentication: Enginuity version 56xx" on page 131 or "iSCSI CHAP authentication: Enginuity Version 57xx" on page 133.

7. Refresh the VCMDB, as shown in the following example:

```
symmask -sid 6208 refresh
```

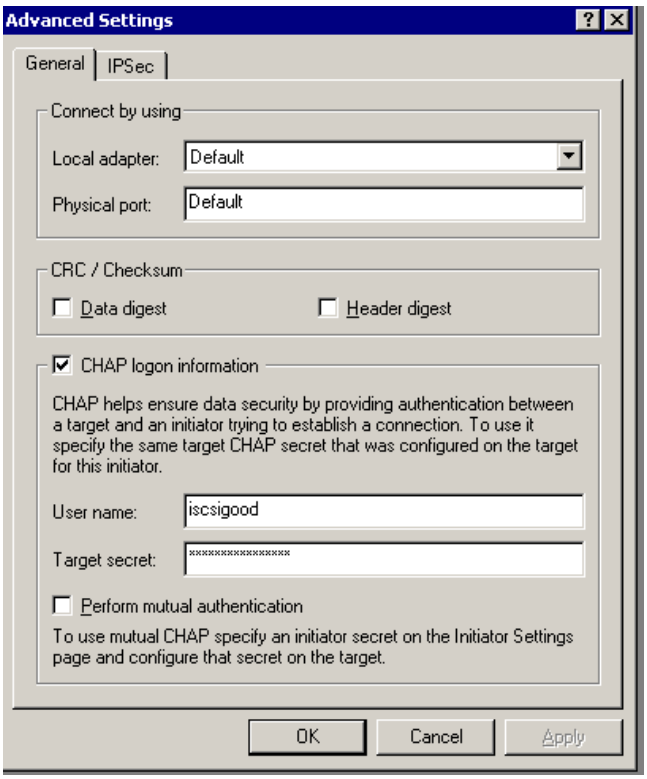
Update iSCSI initiator with Symmetrix information

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.



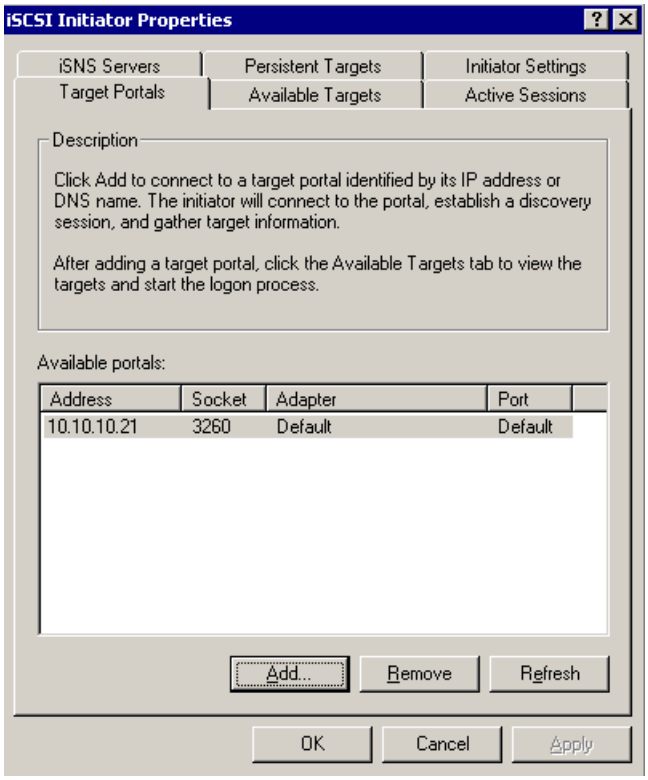
10. Check the box labeled **CHAP logon information**. Change the CHAP credential in the **User name** field, and enter a CHAP secret in the **Target secret** box. Click **OK**.

    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.

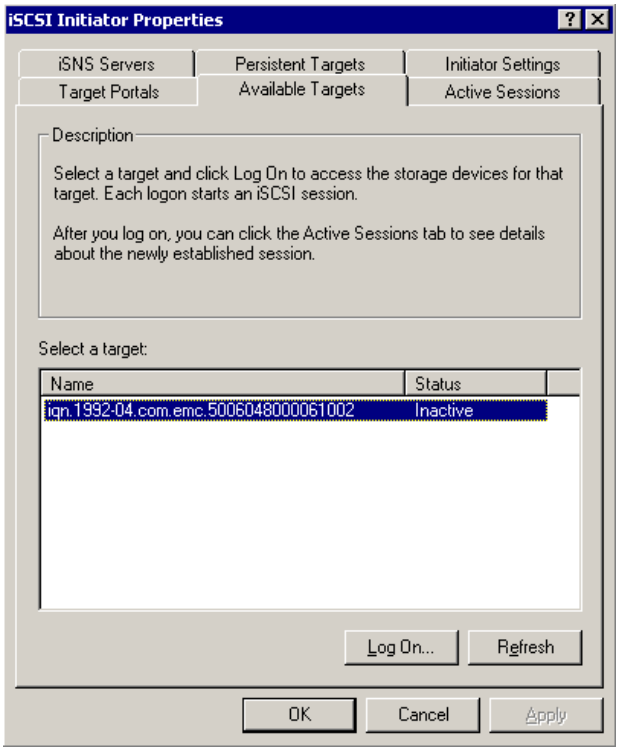
11. Click **OK** to close the **Advanced Settings** dialog box, and click **OK** to close the **Target Portal** dialog box.

    The multi-protocol director IP address should appear in the **Available portals** list in the **Target Portals** window.

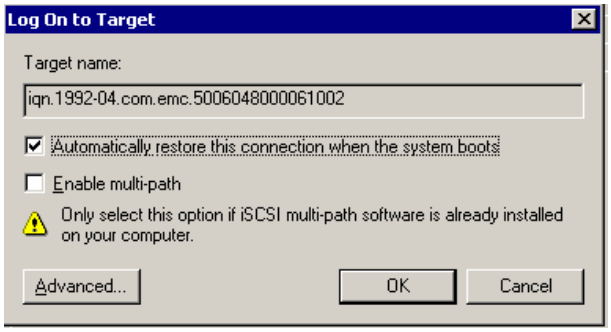


Note: If an error displays, select the IP address from the **Available portals** list and click **Remove**. Begin the configuration procedure again.

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.

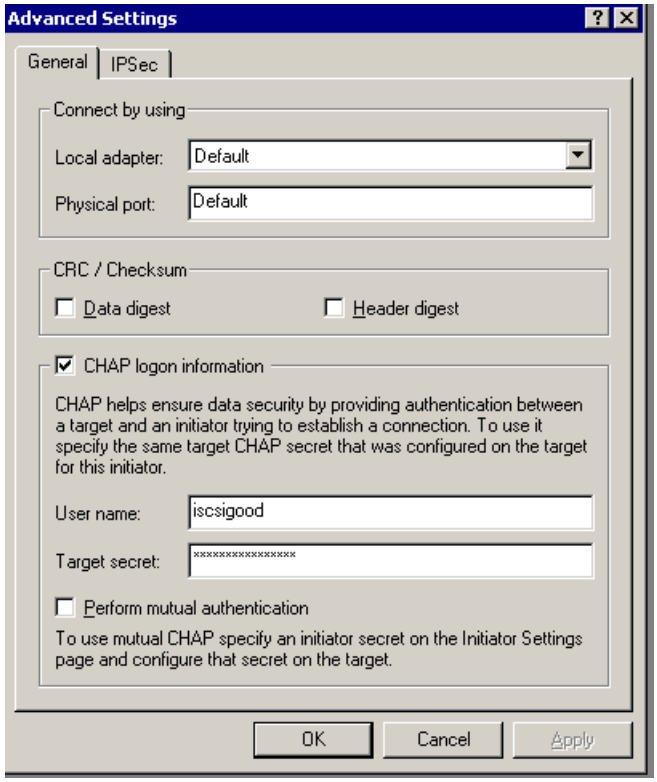


13. Click **Log On**. The **Log On to Target** window dialog box appears.



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.

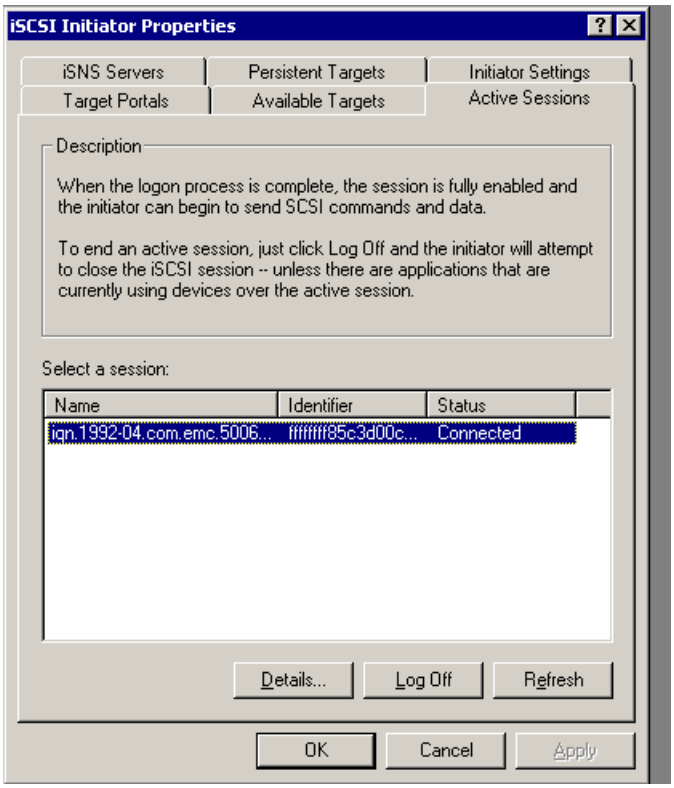


15. Select the box labeled **CHAP logon information**. Change the CHAP credential in the **User name** field, and enter a CHAP secret in the **Target secret** box. Click **OK**.

    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 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.

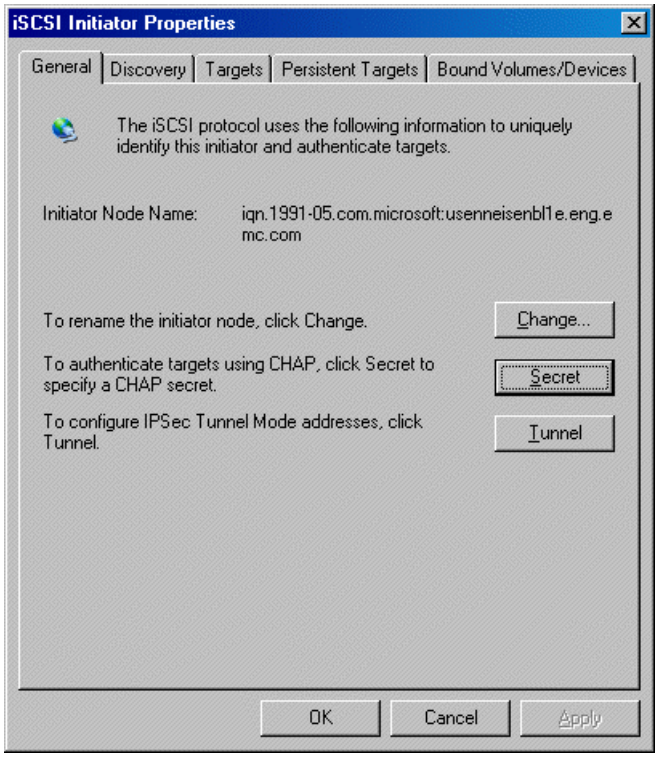
Perform disk administration on the Symmetrix devices

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 2-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 134.



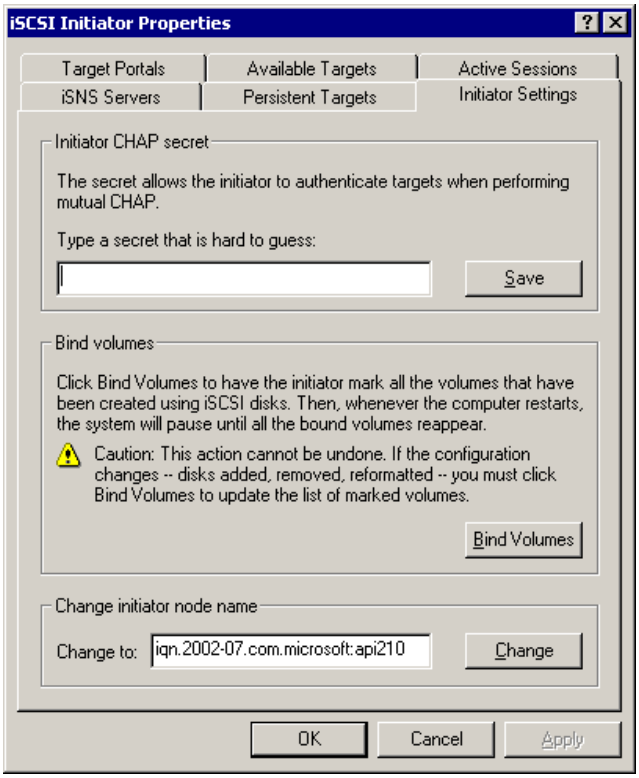
Configuring iSCSI without CHAP authentication

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_name -dir # -p # add dev #
   ```

where:

- *SymmID* = The Symmetrix ID.
- *iscsi_name* = The iSCSI name copied in step 3.
- `-dir` # = Symmetrix director number.
- `-p` # = Symmetrix port number.
- `add dev` # = Symmetrix device number.

For example:

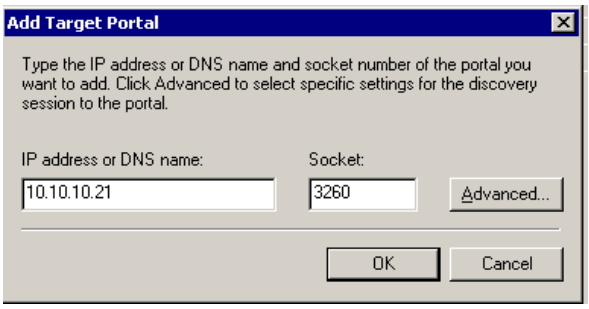
```
symmask -sid 6208 -iscsi iqn.2002-07.com.microsoft:api210 -dir 3a -p 0
      add dev 0023
```

6. Refresh the VCMDB, as shown in the following example:

   ```
   symmask -sid 6208 refresh
   ```

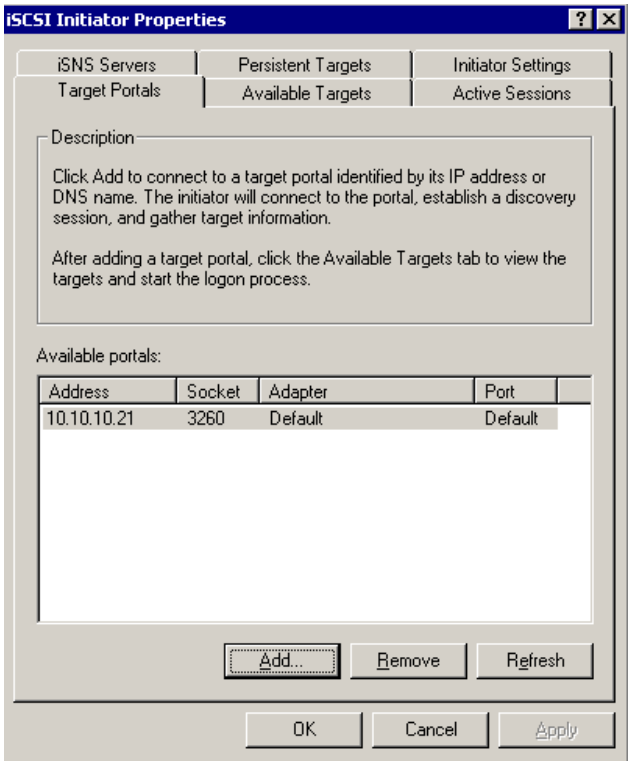
7. From the iSCSI Initiator window, select the **Target Portals** tab and click **Add**.

   The **Add Target Portal** dialog box appears.



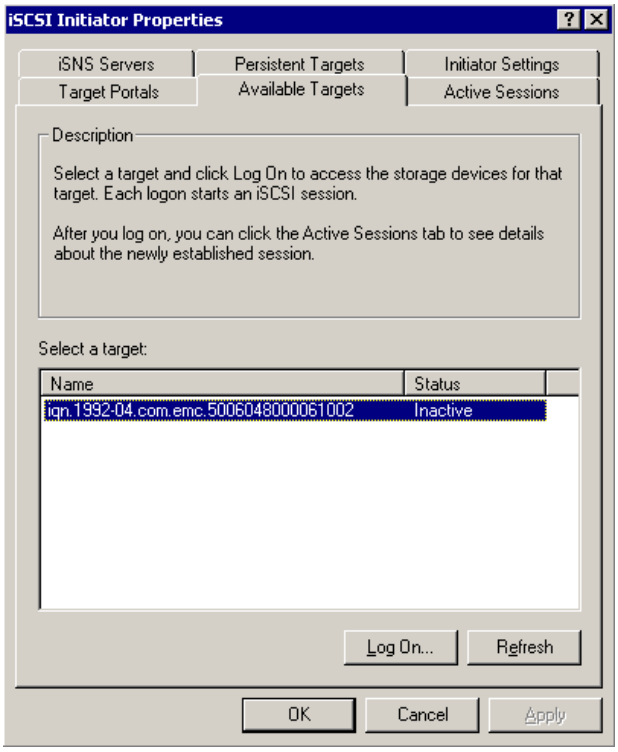
8. Enter the iSCSI IP address (from step 4) of the Symmetrix multi-protocol director in the **IP address or DNS name** box and click **OK**.

The multi-protocol director IP address should appear in the **Available portals** list in the **Target Portals** window.

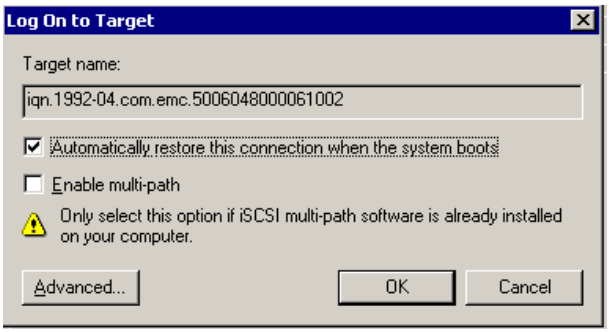


Note: If an error displays, select the IP address from the **Available portals** list and click **Remove**. Begin the configuration procedure again.

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.

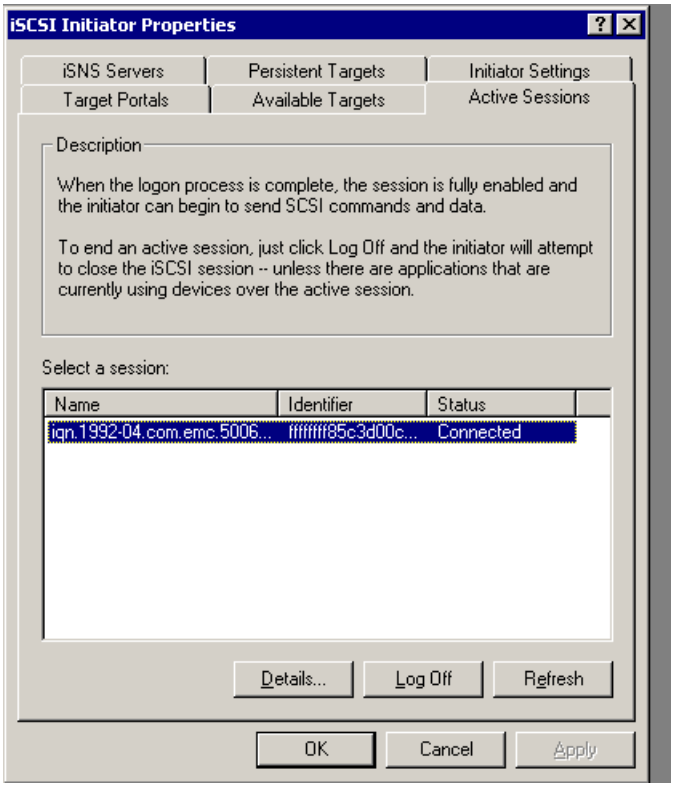


10. Click **Log On**. The **Log On to Target** dialog box appears.



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
```

4

Optimizing Array Performance

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? 154
- Configuration requirements 156
- How does Symmetrix Optimizer work? 157
- Symmetrix Optimizer operations via SYMCLI 163
- Command file general parameters 168

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 via 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 disks or 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, 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) 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 160 for a complete list of valid swaps).

Note: Optimizer does not support parity RAID (parity 3+1, 7+1) and RAID 1/0 devices.

Configuration requirements

Symmetrix Optimizer must be enabled by EMC from the service processor and DRVs must be configured on the array. In addition, Optimizer requires an open mirror position to perform its swapping function.

Dynamic reallocation volumes (DRVs)

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.

As a minimum, two DRVs 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 DRVs must be configured. If more than one simultaneous swap is desired, then additional DRVs are needed (two for each swap). If there are volumes of mixed size or emulation within the same array, then DRVs are required for each size and type to be swapped.

Open mirror position requirement

Optimizer requires an open mirror position to be able to perform a swap, however, Symmetrix arrays 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 BCVs 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

Optimizer also must hold a Symmetrix External Lock (SEL) to perform a swap. 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 (such as SDR) 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, so other applications will not have the ability to perform configuration changes during this time.

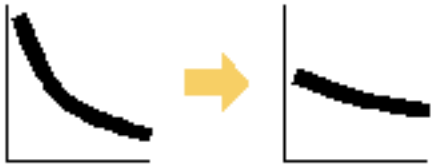
How does Symmetrix Optimizer work?

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 12 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.

The swap procedure

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.

Hypervolumes are swapped using a four-step process. To swap two hypervolumes, the hypervolumes must be the same size, the same emulation, and, based on the architecture, on the same system bus or in the same power zone in the Symmetrix DMX architecture. At least two DRVs must be configured per swap. With Version 5.1.1 of Symmetrix Optimizer, up to eight simultaneous swaps can happen at one time.

Step 1: Identify volumes to swap

Symmetrix Optimizer identifies a pair of hypervolumes to swap by recognizable patterns of hypervolume activity and criteria. Assume the red volumes (on A) have high activity and the blue volumes (on B) have low activity.

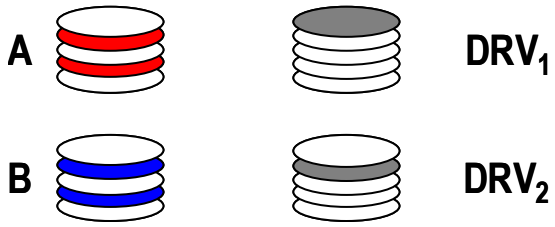


Figure 13 Identify volumes to swap

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 14). Both hypervolumes still have two (or more) physical mirrors. Host activity to the hypervolumes is now directed to DRVs and the other mirror.

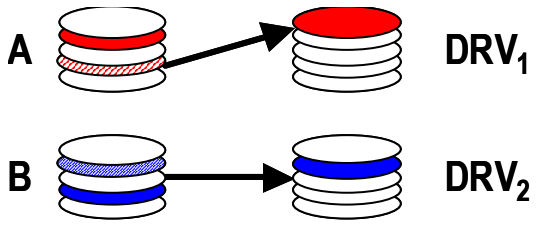


Figure 14 Copy volume to DRV

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 15). This is similar to a BCV restore.

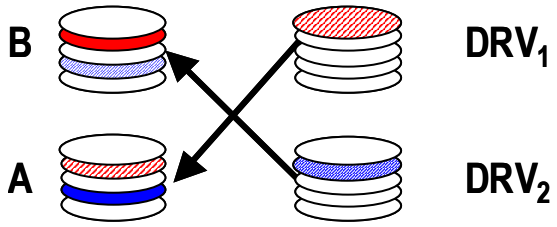


Figure 15 Copy DRVs to new locations

Step 4: DRVs are split

The final step is to split the DRVs from their standard and mirror hypervolumes after synchronization completes (see Figure 16). The drive's balance improves and the DRVs are now available for the next swap.

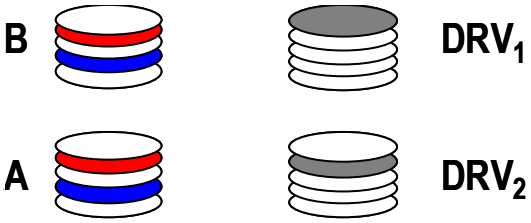


Figure 16 DRVs are split

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
- Not swappable devices – The following devices are not swappable:
  - Parity RAID (parity and data)
  - CKD Stripe (RAID 1/0)
  - BCV, DRV, SFS

- WORM
- 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

- RAID 1 (mirrored volumes)
- RDF devices with two local copies (default, can be controlled from SP)
- Same size and emulation

◆ Not swappable devices — The following devices are not swappable:

- Parity RAID (parity and data)
- CKD Stripe (RAID 1/0)
- BCV, DRV, SFS
- WORM
- AS400
- SAVE devices and VDEVs
- Standard, unprotected volumes (default, can be controlled from the Service Processor)
- Targets of Snap sessions

◆ 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

Each swap performed by Optimizer is the equivalent of two TimeFinder operations. 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 via SYMCLI

The Symmetrix Optimizer is a component of the EMC ControlCenter® Symmetrix Manager. 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 13 describes the main features of the Symmetrix Optimizer command:

Table 13 Symmetrix Optimizer command

Command	Description
symoptmz	Performs operations on the Symmetrix Optimizer. Possible operations are: • Lists information about the Optimizer-specific device attributes of Symmetrix devices that are configured in one or more Symmetrix units connected to this host. • Shows specific information about the current Optimizer parameters. • Enables the Optimizer algorithm processing. • Disables the Optimizer algorithm processing. • Queries Symmetrix Optimizer and displays the current state and version information. • Clears disk statistics maintained by the Optimizer. • Controls the Optimizer session and releases the Optimizer's API lock. • Rolls back the Symmetrix configuration to that of a previous point-in-time. • Verifies, performs range checks, and commits a command file.

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 (ControlCenter or SYMCLI), but it will not collect statistics or initiate any swaps until it is enabled.

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 "Command file general parameters" on page 168); 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.

Setting user approved mode

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=<time_val>,]
  TIMESTAMP=<time_val>;
```

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 performance time windows

To set the performance time windows, use the `symoptmz` 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=<date_time>,
  ending=<date_time>,
  [days=<day_list>,
  start_time=<hh:mm>,
  end_time=<hh:mm>];
```

where:

<tw_id> is 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 r*ange*. 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.

<*date_time*> is in the form of *MMDDYYYY:HHMMSS*.

<*day_list*> is any comma-separated combination of MON, TUE, WED, THU, FRI, SAT, or SUN. For the case of WEEKLY, <*day_list*> 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.

Showing time windows

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).

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 <no_swap|normal|high>
   [-range Startdevname:Enddevname][-n NumDevs]
```

Use the following command to list the priority settings for all the devices or a range of devices:

```
symoptmz list [-range Startdevname:Enddevname][-n NumDevs]
```

Performing rollbacks

Rollbacks are used to undo swaps that conflict with your business rules. Rollbacks are performed by first placing the Symmetrix Optimizer in rollback mode by calling `symoptmz rollback -sid SymmID` on a specified Symmetrix. To return a list of possible rollback points use the `symoptmz -rollback_list -sid SymmID` command.

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 "Setting user approved mode" on page 164 to undo specific swaps. Rollback swaps can be scheduled for execution according to normal Optimizer policy.

Setting manual swap lists

To set a manual swap list, use the following form:

```
set swap_list {Hyper1} with {Hyper2} [, {Hyper3} with {Hyper4},... ]
[begin_at=<time_val>];
```

where:

Hypern 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

time_val is in the form of `MMDDYYYY:HHMMSS`.

Optimizer logs and swap history

To get a list of all swaps that have been completed, use the following command:

```
symoptmz show -swap_hist
```

To retrieve the Optimizer activity log, including swaps that have been completed, use the following command:

```
symoptmz read -log_type RUNTIME [-start <date_time>] [stop <date_time>]
```

To retrieve the Optimizer error log, use the command:

```
symoptmz read -log_type ERROR [-start <date_time>] [stop <date_time>]
```

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.

Migrating devices

Optimizer with Enginuity version 5772 introduces the ability to move existing devices from their current locations to a new set of disks or to a predefined location on another disk. 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.

To initiate a migration, use the `symoptmz` command with the following syntax in the command file:

```
migrate
          dev[s] <Startdevname1>[:<Enddevname1>]
            [,<Startdevname2>[:<Enddevname2>],...]
          TO disk[s] {diskn} [,{diskn},...]
          [begin_at=<time_val>];
```

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 `time_val` is in the form of `MMDDYYYY:HHMMSS`.

For example:

```
migrate devs 240, 245:247 to disks {16B, C, 0}, {4D, C, 0};
```

Command file general parameters

General Optimizer parameters are set using the following syntax in the command file:

```
set control_parms [start_mode=<AUTO | MANUAL>,]
  [min_perf_period=<min_perf>,]
  [workload_period=<workload>,]
  [swap_mode=<AUTO | USER_OK>,]
  [max_simult_swaps=<max_simult>,]
  [swap_rate=<max_swaps>];
```

Table 14 describes each parameter.

Table 14 Parameters in the command file

Parameter	Description
start_mode	Determines whether Optimizer is enabled or disabled whenever Optimizer is launched, such as after the service processor is rebooted.
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;
```

5

Managing Quality of Service

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 170
- Setting QoS copy parameters and values 171
- Setting QoS LRU parameters and values 172
- Using Symmetrix Priority Controls 173
- Using dynamic cache partitions 175
- QoS examples 179

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 15 lists the features of the QoS command:

Table 15 SYMAPI Quality of Service command

Command	Description
symqos	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 10, 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

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 <startDevRange:endDevRange>| -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 <startDevRange:endDevRange>]
   list -g DgName [-std] [-bcv]
```

For example, to list the priority of service time for Symmetrix 131, devices 0002 and 0003, enter:

```
symqos -pst -sid 131 list -hostio_priority -range 0002:0003
```

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 Symmetrix CLI Command Reference*.

Using dynamic cache partitions

With Enginuity 5772, 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 maximun percentage. In addition, you can donate unused cache to other partitions after a a specified donation time.

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 cach partitions on Symmetrix 301, enter:

```
symqos -cp -sid 301 enable
```

To disable cach partitions on Symmetrix 301, enter:

```
symqos -cp -sid 301 disable
```

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 meta and CKD devices.
- Do not add VAULT 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 0001 to partition TestP on Symmetrix 310, enter:

```
symqos -cp -name TestP -sid 310 add dev 0001
```

To add device range 0002 through 0012 to partition TestP on Symmetrix 310, enter:

```
symqos -cp -name TestP -sid 310 -range 0002:0012 addall
```

To add the BCV devices in device group Finance to the partition TestP on Symmetrix310, 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 Symmetrix310, 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 <startDevRange:endDevRange> [-all]| -rdfg 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         : Enabled
    Number of Partitions       : 2
    Max Num of Partitions      : 8
    Min Allowed Target %       : 10
    Max Allowed Target %       : 90
    Time of Last Modification  : 01:12:02 PM on Tue Mar 13,2007


                                  Min  Tgt  Max  WP   Time
Partition Name                    (%)  (%)  (%)  (%)  (s)
--------------------------------  ---  ---  ---  ---  -----
DEFAULT_PARTITION                   0   65  100   80    300
Lab2                               35   35   35   50    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
    Time of Last Modification   : 01:12:02 PM on Tue Mar 13,2007


                                                          Cache Slots
                    Min  Tgt  Max  WP   Time   Device  ------------------  Used
Partition Name      (%)  (%)  (%)  (%)  (s)    Count      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 Symmetrix CLI Command Reference*.

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 that if you omit the `-sid` option, the specified range of devices will apply to all Symmetrix units.

Note that 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
```

6

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 182
- Implementing EMC Double Checksum for Oracle 184
- Implementing Generic SafeWrite for generic applications 188
- Syntax and examples 193

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 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 184 for details on this feature.

For generic RDBMS applications, **EMC Double Checksum for Generic Applications** provides the Generic SafeWrite[1] 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 184 for details on this feature.

This chapter contains overview and concept information, see the `symchksum` command in the *EMC Solutions Enabler Symmetrix CLI Command Reference* 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 and/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.

The Oracle instance being checked must be started with the following `init.ora` configuration parameter set to true: db_block_checksum=true.

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.)
- NonzeroDba — 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
                                        Num Blk           o c e  u i B a l B c r
Device Name                        Dev Exts Siz Type      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

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 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 16 shows the **EMC Double Checksum for Oracle** functionality provided in each Enginuity version

Table 16 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]
5*x*68	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

a. Symmetrix extents are internal to the Symmetrix array and are used to map Oracle tablespace locations; they are not the same as Oracle extents

b. 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.

c. 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 Checkum 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 184.

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

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. All device types are supported including devices replicated using the SRDF and the TimeFinder family of products. 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 diskpar and diskpart to Align Partitions on Windows Basic and Dynamic Disks* available on EMC® Powerlink™.

Configuring Generic SafeWrite

To use Generic SafeWrite, you must:

- 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 via the symchksum command

Setting the RDB_cksum Symmetrix device flag

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 metadevice, 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 via 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 metadevice, 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. 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.

```
cmd
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
                                        Num Blk                  o c e  u i B a l B c r
Device Name                        Dev Exts Siz Type             g t H  m c A d k A t d
-------------------------------------------------------------------------------------------

\\.\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 17 **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

Performance testing was done with Microsoft Exchange, Microsoft SQL Server and Oracle on standard devices, and in the case of Microsoft Exchange, also on SRDF/S and SRDF/A devices. For the Microsoft SQL Server and Oracle performance tests, a TPCC workload was used. For Microsoft Exchange, the Jetstress performance tool was used. The results of these tests showed no performance degradation from an application perspective. The reason application performance should remain unaffected is because database log writes, as well as database reads, will be performed normally. When considering log write response times and database read response times as the main determinates of application performance with respect to storage, it is expected that client and application response times will not be greatly affected.

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 Symmetrix CLI Command Reference* for the complete list of syntax and options.

Table 17 symchksum syntax and descriptions

Command	Argument	Description
`symchksum`	`list`	Lists all devices that currently have checksum checking enabled.
	`show`	Displays the extents of a specified device that are having checksum checking performed.
	`enable`	Enables checksum checking on the extents of the specified devices.
	`disable`	Disables checksum checking on the extents of the specified devices.
	`validate`	Validates that a specified database or tablespace is able to have checksum checking enabled.
	`verify`	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
```

PART 2

Operational Examples

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:

Chapter 7, "Configuration Change Examples," provides detailed examples on using the Configuration Change component.

Chapter 8, "Device Masking Examples," provides detailed examples on using the Device Masking component.

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.

7

Configuration Change Examples

This chapter provided detailed examples on using the Configuration Change component.

The following examples illustrate Configuration Manager functionality included in EMC Solutions Enabler Version 6.0 running on Symmetrix arrays using Enginuity Version 5*x*71. Although these examples illustrate basic configuration concepts, only advanced users or system administrators should change a Symmetrix configuration.

- Example 1: Creating devices 198
- Example 2: Mapping devices 204
- Example 3: Mapping a single device on multiple EA ports 209
- Example 4: Mapping a range of devices and adding aliases 215
- Example 5: Unmapping devices 220
- Example 6: Setting front-end port attributes 224
- Example 7: Forming an FBA metadevice 229
- Example 8: Configuring RDF dynamic devices 235
- Example 9: Configuring concurrent RDF devices 246

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     Unformatted
                                  (Cyl)           (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 `symdisk list` command displays available space (in cylinders) on physical disks and whether those disks can hold additional hypers. 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.

  symdisk -sid 120 list

  ```
  Symmetrix ID: 000184500120

  Symmetrix ID                  : 000184500120
  Disks Selected                : 60

                                                                    Capacity(Cyl)
  Ident   Symb Int TID Vendor      Type        Hypers    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
------------------------- ------------ ------------------------------------
                                                                       Cap
Sym  Physical               SA :P DA :IT Config        Attribute    Sts  (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 220 and "Example 6: Setting front-end port attributes" on page 224).

- 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
---------------------------- ------------ ------------------------------------
                                                                          Cap
Sym  Physical                SA :P DA :IT Config        Attribute    Sts   (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) running Enginuity version 5568. 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
---------------------------- ------------ ------------------------------------
                                                                        Cap
Sym  Physical                SA :P DA :IT Config        Attribute    Sts   (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
---------------------------- ------------ ------------------------------------
                                                                          Cap
Physical                Sym  SA :P DA :IT Config        Attribute    Sts   (MB)
---------------------------- ------------ ------------------------------------

/dev/rdsk/c1t0d1s2      0029 04B:0 16B:C3 RDF2          N/Grp'd      WD     103
/dev/rdsk/c1t0d2s2      0033 04B:0 15A:C3 RDF1          N/Grp'd      RW     103
/dev/rdsk/c1t0d3s2      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/rdsk/c1t0d1s2              0    0  001
                           0033  /dev/rdsk/c1t0d2s2              0    0  002
                           003D  /dev/rdsk/c1t0d3s2              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
---------------------------- ------------ ---------------------------------------
                                                                            Cap
Sym  Physical                SA :P DA :IT Config        Attribute    Sts   (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
---------------------------- ------------ ---------------------------------------
                                                                            Cap
Physical                Sym  SA :P DA :IT Config        Attribute    Sts   (MB)
---------------------------- ------------ ---------------------------------------

/dev/rdsk/c1t0d1s2      0029 04B:0 16B:C3 RDF2          N/Grp'd      WD     103
/dev/rdsk/c1t0d2s2      0033 04B:0 15A:C3 RDF1          N/Grp'd      RW     103
/dev/rdsk/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
---------------------------- ------------ ------------------------------------
                                                                          Cap
Physical                Sym  SA :P DA :IT Config        Attribute    Sts   (MB)
---------------------------- ------------ ------------------------------------

/dev/rdsk/c1t0d1s2      0029 04B:0 16B:C3 RDF2          N/Grp'd      WD     103
/dev/rdsk/c1t0d2s2      0033 04B:0 15A:C3 RDF1          N/Grp'd      RW     103
/dev/rdsk/c1t0d3s2      003D 04B:0 02B:D2 Unprotected   N/Grp'd      RW       3
/dev/rdsk/c1t0d32s2     009B 04B:0 01A:C0 2-Way Mir     N/Grp'd      RW     103
/dev/rdsk/c1t0d33s2     009C 04B:0 16A:D4 2-Way Mir     N/Grp'd      RW     103
/dev/rdsk/c1t0d34s2     009D 04B:0 01A:D2 2-Way Mir     N/Grp'd      RW     103
/dev/rdsk/c1t0d35s2     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 215), 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
.....................................................................

                                 : 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 `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.

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/rdsk/c1t0d36s2

    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/rdsk/c1t0d36s2      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) running Enginuity version 5568. The host is connected to front-end director FA-4A, which is a "Fibre Channel" type 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: Forming an FBA metadevice

This example creates two 2-way mirrored devices on the Symmetrix array (sid 40) and then forms them into a concatenated metadevice. The metadevice is mapped to (and then unmapped from) a front-end port. The example illustrates the use of input redirection, rather than a command file, to input change commands and parameters to the `symconfigure commit` command. Although not used in this example, it is recommended that you use `symconfigure verify` prior to a commit action (see previous examples).

- 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.

```
symconfigure -sid 40 list -freespace

Symmetrix ID: 000184600040

    Emulation               Formatted    Unformatted
                                (Cyl)          (Cyl)
    ---------------------   ---------    -----------

    FBA                       2732468              0
```

- The `symdev list` command displays existing devices. Notice that the last device in the display is 012.

```
symdev list -sid 40

Symmetrix ID: 000184600040

        Device Name          Directors                    Device
--------------------------- ------------ ---------------------------------------
                                                                            Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts     (MB)
--------------------------- ------------ ---------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD         8
001 /dev/rdsk/c1t0d1s2      03B:0 02B:C1 Unprotected   N/Grp'd      RW         3
002 /dev/rdsk/c1t0d2s2      03B:0 01A:C2 Unprotected   N/Grp'd      RW         3
003 Not Visible             ???:? 01A:C0 Unprotected   N/Grp'd      RW      1875
004 Not Visible             ???:? 02B:C3 Unprotected   N/Grp'd      RW      1875
005 Not Visible             ???:? 02A:C0 Unprotected   N/Grp'd      RW      1875
006 Not Visible             ???:? 01B:C3 Unprotected   N/Grp'd      RW      1875
007 Not Visible             ???:? 01B:C0 2-Way Mir     N/Grp'd  (M) RW      7500
008 Not Visible             ???:? 02B:C0 2-Way Mir     N/Grp'd  (m) RW         -
009 Not Visible             ???:? 01A:D0 2-Way Mir     N/Grp'd  (m) RW         -
00A Not Visible             ???:? 02A:D0 2-Way Mir     N/Grp'd  (m) RW         -
00B Not Visible             ???:? 01B:D0 BCV           N/Asst'd     RW      1875
00C Not Visible             ???:? 02A:D2 BCV           N/Asst'd     RW      1875
00D /dev/rdsk/c1t0d13s2     03B:0 02B:D0 2-Way BCV Mir N/Asst'd     RW      1875
00E /dev/rdsk/c1t0d14s2     03B:0 01A:C1 2-Way BCV Mir N/Asst'd     RW      1875
00F Not Visible             ???:? 02A:C1 DRV           N/Grp'd      RW      1875
010 Not Visible             ???:? 01B:C2 DRV           N/Grp'd      RW      1875
011 Not Visible             ???:? 01A:D1 2-Way Mir     N/A      (FS) RW      2878
012 Not Visible             ???:? 02A:D1 2-Way Mir     N/A      (FS) RW      2878
```

- On UNIX platforms, you can use input redirection, rather than a command file, to input the `create dev` command and its parameters to the `symconfigure commit` command. Use a beginning delimiter (like <<EOF) and a corresponding end delimiter to frame the change command. The size of each device will be 1024 cylinders (or 480 MB). The ellipsis (…) represents output that was omitted for brevity.

```
symconfigure -sid 40 commit <<EOF
     create dev count=2, size=1024, emulation=FBA, config=2-Way-Mir;
     EOF

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

    Establishing a configuration change session.....Established.
    Processing symmetrix 000185400123
    Submitting configuration changes.....Submitted.
    Validating configuration changes.....Validated.
    Initiating PREPARE of configuration changes.....Queued.
    PREPARE requesting required resources.....Obtained.
    Step 006 of 010 steps.....Executing.
    ..........................................................................................
    Step 009 of 010 steps.....Executing.
    Local:  PREPARE.....Done.
    Initiating COMMIT of configuration changes.....Queued.
    COMMIT requesting required resources.....Obtained.
    Step 004 of 097 steps.....Executing.
    ..........................................................................................
    Step 097 of 097 steps.....Executing.
    Local:  COMMIT.....Done.
    Terminating the configuration change session.....Done.

The configuration change session has successfully completed.
```

- When you add new devices and form metadevices, 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.

  The `symdev list` command displays the two new devices, 013 and 014. Each device is 480 MB.

```
symdev list -sid 40

Symmetrix ID: 000184600040

         Device Name          Directors                  Device
--------------------------- ------------ ---------------------------------------
                                                                            Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts     (MB)
--------------------------- ------------ ---------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD         8
..........................................................................................
012 Not Visible             ???:? 02A:D1 2-Way Mir     N/A      (FS) RW      2878
013 Not Visible             ???:? 02B:C3 2-Way Mir     N/Grp'd      RW       480
014 Not Visible             ???:? 01A:C0 2-Way Mir     N/Grp'd      RW       480
```

- The following command forms the two new devices into a metadevice. Device 013 is the metahead and will represent the metadevice. Device 014 is the metatail (last metamember added to a metadevice).

```
symconfigure -sid 40 commit <<EOF
   > form meta from dev 13, config=concatenated;
   > add dev 14 to meta 13;
   > EOF

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

    Establishing a configuration change session..............Established.
    ..........................................................................
    Local:  COMMIT.....................................Done.
    Terminating the configuration change session..............Done.

The configuration change session has successfully completed.
```

- The `symdev list` command displays the devices again to check that the two new devices have been converted into a metadevice. For device 013, the (M) after the "N/Grp'd" attribute denotes that it is the metahead. The corresponding (m) for device 014 denotes that it is a metamember. The capacity (size) of device 013 has increased from 480 MB to 960 MB to show the total metadevice size, and the capacity of device 014 is not reported.

```
symdev list -sid 40

Symmetrix ID: 000184600040

        Device Name          Directors                  Device
--------------------------- ------------ ---------------------------------------
                                                                          Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts     (MB)
--------------------------- ------------ ---------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD        8
..........................................................................
013 Not Visible             ???:? 02B:C3 2-Way Mir     N/Grp'd  (M) RW      960
014 Not Visible             ???:? 01A:C0 2-Way Mir     N/Grp'd  (m) RW        -
```

- In this example, the ???:? displayed in the SA :P column for device 013 indicates that the metadevice is not yet mapped to a port. Subsequent commands will map the metadevice to a new host connected to front-end director 03A, port 0. The `symcfg list` command displays the current port settings (flags) for this port.

```
symcfg -sa 03A -p 0 list -v -sid 40

Symmetrix ID: 000184600040 (Local)

    Product Model                       : 8130
    Symmetrix ID                        : 000184600040

    Microcode Version (Number)          : 5568 (15BFAA01)
    Microcode Date                      : 06.11.2001

    Microcode Patch Date                : 06.11.2001
    Microcode Patch Level               : 30
    ..........................................................................

          Fibre Specific Flags
            {
              Disk_Array                : Enabled
              Volume_Set_Addressing     : Enabled
              Hard_Addressing           : Disabled
              Non_Participating         : Disabled
```

```
    Global_3rdParty_Logout       : Disabled
    Init_Point_to_Point          : Enabled
    Unique_WWN                   : Enabled
    Generic_VSA                  : Disabled
    VCM_Enabled                  : Enabled
    Class_2_Service              : Disabled
    OpenVMS                      : Disabled
  }
```

- The old host connected to director/port 03A:0 was running HP-UX and required special port settings (Disk Array enabled and Volume Set Addressing enabled). The new host does not require these settings. The following command disables these port flags.

```
symconfigure -sid 40 commit <<EOF
   > set port 03A:0 Disk_Array=disable, Volume_Set_Addressing=disable;
   > EOF

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

    Establishing a configuration change session..............Established.
    Processing symmetrix 000184600040
    ..........................................................................................................
    Local:  COMMIT.....................................Done.
    Terminating the configuration change session..............Done.

The configuration change session has successfully completed.
```

- Another `symcfg list` command confirms that these port settings have been reset to disabled.

```
symcfg -sa 03A -p 0 list -v -sid 40

Symmetrix ID: 000184600040 (Local)
..........................................................................

          Fibre Specific Flags
            {
              Disk_Array                 : Disabled
              Volume_Set_Addressing      : Disabled
              Hard_Addressing            : Disabled
              Non_Participating          : Disabled
              Global_3rdParty_Logout     : Disabled
              Init_Point_to_Point        : Enabled
              Unique_WWN                 : Enabled
              Generic_VSA                : Disabled
              VCM_Enabled                : Enabled
              Class_2_Service            : Disabled
              OpenVMS                    : Disabled
            }
```

- Now that port settings are correct for the new host, the metadevice can be mapped to the port. However, the mapping operation requires specifying a LUN address. The `symcfg list -addresses` command displays the Fibre Channel addresses to determine the next available LUN address.

```
symcfg -sid 40 -sa 3A -addresses list

Symmetrix ID: 000184600040 (Local)

          Director             Device  Name              Address
    ----------------------  ----------------------------  --------------
    Ident   Symbolic  Port  Sym   Physical                VBUS  TID  LUN
    ------  --------  ----  ----  ----------------------  ----  ---  ---

    FA-3A   03A       0     000   Not Visible               0    0  000
                            001   Not Visible               0    0  001
                            002   Not Visible               0    0  002
                            003   Not Visible               0    0  003
                            004   Not Visible               0    0  004
                            005   Not Visible               0    0  005
                            006   Not Visible               0    0  006
                            007   Not Visible               0    0  007
                            00B   Not Visible               0    0  00B
                            00C   Not Visible               0    0  00C
                            00D   Not Visible               0    0  00D
                            00E   Not Visible               0    0  00E
```

- The example uses the LUN address 00F to map the metadevice.

```
symconfigure -sid 40 commit <<EOF
   > map dev 13 to dir 03A:0 lun=00F;
   > EOF

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

    Establishing a configuration change session..............Established.
```

- The `symdev list` command displays that the mapping operation was successful, indicated by the presence now of 03A:0 in the SA: P column of the metahead (013) and metatail (014). The metamembers will continue to be marked "Not Visible" until you execute the host update utilities that assign the host names. Instead of updating the host, this example now proceeds directly to unmapping the metadevice.

```
symdev list -sid 40

Symmetrix ID: 000184600040

        Device Name          Directors                  Device
--------------------------- ------------ ------------------------------------
                                                                          Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts    (MB)
--------------------------- ------------ ------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD       8
.....................................................................................
013 Not Visible             03A:0 02B:C3 2-Way Mir     N/Grp'd  (M) RW     960
014 Not Visible             03A:0 01A:C0 2-Way Mir     N/Grp'd  (m) RW       -
```

- The second part of this example unmaps the metadevice from the director/port. Before unmapping any device, you need to ensure that there is no host I/O to the device. You can use the `symdev` command to make the metadevice Not Ready for any I/O. Recall that the metahead (device 013) represents the entire metadevice.

```
symdev -sid 40 not_ready 013 -noprompt

Not Ready operation successfully completed for the device.
```

- A `symdev list` command shows that the status of the metadevice has changed to Not Ready (NR).

```
symdev list -sid 40

Symmetrix ID: 000184600040

        Device Name          Directors                   Device
--------------------------- ------------ ---------------------------------------
                                                                            Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts     (MB)
--------------------------- ------------ ---------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD        8
..........................................................................................
013 Not Visible             03A:0 02B:C3 2-Way Mir     N/Grp'd  (M) NR      960
014 Not Visible             03A:0 01A:C0 2-Way Mir     N/Grp'd  (m) NR        -
```

- The following command uses input redirection to unmap the metadevice.

```
symconfigure -sid 40 commit <<EOF
   > unmap dev 13 from dir 03A:0;
   > EOF

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

Establishing a configuration change session..............Established.
```

- The SA :P column displayed by the `symdev list` command confirms that the metadevice has been unmapped (denoted when `???:?` appear in the column). Note that a device whose status is NR (Not Ready) can be mapped again and that a side effect of the mapping operation is to make the device Ready (RW).

```
symdev list -sid 40

Symmetrix ID: 000184600040

        Device Name          Directors                   Device
--------------------------- ------------ ---------------------------------------
                                                                            Cap
Sym Physical                SA :P DA :IT Config        Attribute    Sts     (MB)
--------------------------- ------------ ---------------------------------------

000 /dev/rdsk/c1t0d0s2      03B:0 01B:C1 2-Way Mir     N/Grp'd      WD        8
..........................................................................................
013 Not Visible             ???:? 02B:C3 2-Way Mir     N/Grp'd  (M) NR      960
014 Not Visible             ???:? 01A:C0 2-Way Mir     N/Grp'd  (m) NR        -
```

Example 8: Configuring RDF dynamic devices

This example is performed using Solutions Enabler version 6.0. You can use the SYMCLI Configuration Manager to configure dynamic RDF devices, making it possible to create dynamic SRDF pairs while the Symmetrix array is running. In this example, a Solaris host (api60) is connected to a local Symmetrix array (sid 605). The host is connected to front-end director FA-4A, which is a Fibre Channel type director. The local Symmetrix array is connected via RDF links to a remote Symmetrix (sid 85). The example creates six devices on the local Symmetrix, sets the dynamic attribute for them, and maps them to the director/port. It omits the same steps that were used to create the dynamic RDF devices on the remote Symmetrix for the subsequent R1/R2 matching of dynamic SRDF pairs.

- The following command uses the vi text editor to create a text file named set_sym.cmd. As was done here, you can enter into the file the `set symmetrix` entry that enables dynamic RDF in the Symmetrix array attached to your host. (Dynamic RDF must also be enabled on the remote Symmetrix.)

  vi set_sym.cmd

  set symmetrix dynamic_rdf=enable;

- The `symconfigure commit` command executes the command file and initiates setting the Symmetrix `dynamic_rdf` parameter to enable. The ellipsis (.....) indicates output that was omitted for brevity.

```
symconfigure -sid 605 -file set_sym.cmd -v -noprompt commit

    Establishing a configuration change session..............Established.
    Processing symmetrix 000000005605
    {
      set symmetrix dynamic_rdf_configuration = 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 following command uses the vi text editor to create a text file named `create_std.cmd`. As was done here, you can enter into the file the `create dev` entry that defines six new devices as RAID 5 type devices. Each device will have a size of 1100 cylinders.

  vi create_std.cmd

  create dev, count=6, size=1100, emulation=fba, config=RAID-5;

- The `symconfigure commit` command executes the command file and initiates the processing to create six new standard devices.

```
symconfigure -sid 605 -file create_std.cmd -v -noprompt commit

    Establishing a configuration change session..............Established.
    Processing symmetrix 000000005605
    {
      create dev count=6, size=1100, emulation=FBA, config=RAID-5;
    }

    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.
    Step 010 of 011 steps....................................Executing.
    Local:  PREPARE..........................................Done.
    Initiating COMMIT of configuration changes...............Queued.
    COMMIT requesting required resources.....................Obtained.
    Step 011 of 102 steps....................................Executing.
    ..........................................................................
    Step 089 of 102 steps....................................Executing.
    Local:  COMMIT...........................................Done.
    Terminating the configuration change session..............Done.
```

- The `symdev list` command displays the six new devices (091 through 096).

```
symdev list -sid 605

Symmetrix ID: 000000005605

        Device Name           Directors                  Device
---------------------------- ------------ ------------------------------------
                                                                          Cap
Sym  Physical                SA :P DA :IT Config        Attribute    Sts   (MB)
---------------------------- ------------ ------------------------------------

0000 Not Visible             ???:? 01B:C0 2-Way Mir     Grp'd        RW     516
0001 /dev/rdsk/c1t0d1s2      04A:0 16B:C1 2-Way Mir     Grp'd        RW     516
..........................................................................................
0091 Not Visible             ???:? 16B:C0 RAID-5        N/Grp'd      RW     516
0092 Not Visible             ???:? 02B:C1 RAID-5        N/Grp'd      RW     516
0093 Not Visible             ???:? 15B:C1 RAID-5        N/Grp'd      RW     516
0094 Not Visible             ???:? 16B:D0 RAID-5        N/Grp'd      RW     516
0095 Not Visible             ???:? 02B:D1 RAID-5        N/Grp'd      RW     516
0096 Not Visible             ???:? 15A:D1 RAID-5        N/Grp'd      RW     516
```

- The following command uses the vi text editor to create a text file named set_dyn_devices.cmd. As was done here, you can enter into the file the `set dev` entry that sets the dynamic RDF attribute for the six new devices. The `dyn_rdf` attribute allows a device to be dynamically converted to either an R1 or an R2 device.

  vi set_dyn_devices.cmd

  set dev 0091:0096 attribute=dyn_rdf;

- The `symconfigure commit` command executes the command file and initiates the processing to set the dynamic RDF-capable attribute for the six devices.

symconfigure -sid 605 -file set_dyn_devices.cmd -v -noprompt commit

```
Establishing a configuration change session..............Established.
Processing symmetrix 000000005605
{
  set dev 0091:0096 attribute = DYN_RDF;
}

Submitting configuration changes.........................Submitted.
Validating configuration changes.........................Validated.
Initiating PREPARE of configuration changes..............Queued.
PREPARE requesting required resources....................Obtained.
Step 007 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 014 of 041 steps....................................Executing.
.....................................................................................
Step 037 of 041 steps....................................Executing.
Local:  COMMIT...........................................Done.
Terminating the configuration change session..............Done.
```

- The `symdev show` command confirms that device 091 was set to have a "Dynamic RDF Capability" of either an R1 or an R2 device. The example assumes that the configuration change session also set this attribute for the other devices specified by the `set dev` command file entry.

```
symdev show 0091 -sid 605

Symmetrix ID: 000000005605

    Device Physical Name     : Not Visible

    Device Symmetrix Name    : 0091
    Device Serial ID         : N/A
    Symmetrix ID             : 000000005605

    Attached BCV Device      : N/A

    Attached Snap TGT Device : N/A

    Vendor ID                : EMC
    Product ID               : SYMMETRIX
    Product Revision         : 5671
.....................................................................................................
Device Configuration     : RAID-5

    Device is WORM Enabled   : No
    Device is WORM Protected : No

    Dynamic Spare Invoked    : No

    Dynamic RDF Capability   : RDF1_OR_RDF2_Capable
.....................................................................................................
```

- 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 (api60) uses to reach the Symmetrix (sid 605).

```
symcfg list -sid 605 -connections

Symmetrix ID : 000000005605

  Symmetrix                                  Host
-------------  ------------------------------------------------------------
Director Port  Node Name     IP Address      HW Type  OS Name  OS Revision
-------- ----  ------------- --------------- -------- -------- -----------

FA-4A     0    api60         172.23.65.60    sun4u    SunOS    5.6
```

- The following `symcfg list` command with the `-addresses -available` options displays the VBUS, TID, and LUN addresses associated with front-end director 04A, port 0, and indicates the next available address in the run. To decide on a LUN value, consider the LUN conventions for your host platform. Because there is a large run of available LUN addresses between 004 and 060, the example uses LUN addresses 004 through 009 for the six new devices.

```
symcfg list -sid 605 -sa 04A -p 0 -addresses -available

Symmetrix ID: 000000005605 (Local)

          Director                 Device Name          Attr     Address
    ---------------------- ------------------------------ ---- --------------
    Ident   Symbolic  Port Sym   Physical                     VBUS  TID  LUN
    ------  --------  ---- ----  -----------------------      ----  ---  ---

    FA-4A   04A       0    -     AVAILABLE                       0    0  000 *
                           0001  /dev/rdsk/c1t0d1s2              0    0  001
                           0002  /dev/rdsk/c1t0d2s2              0    0  002
                           0003  /dev/rdsk/c1t0d3s2              0    0  003
                           -     AVAILABLE                       0    0  004 *
                           0040  /dev/rdsk/c1t0d96s2             0    0  060
                           0041  /dev/rdsk/c1t0d97s2             0    0  061
                           0042  /dev/rdsk/c1t0d98s2             0    0  062
                           0043  /dev/rdsk/c1t0d99s2             0    0  063
                           -     AVAILABLE                       0    0  064 *


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 following command uses the vi text editor to create a text file named map_dyn_devices.cmd. As was done here, you can enter into the file the `map dev` entries that map the six new devices.

```
# vi map_dyn_devices.cmd

map dev 091 to dir 04A:0, lun=004;
map dev 092 to dir 04A:0, lun=005;
map dev 093 to dir 04A:0, lun=006;
map dev 094 to dir 04A:0, lun=007;
map dev 095 to dir 04A:0, lun=008;
map dev 096 to dir 04A:0, lun=009;
```

- The `symconfigure commit` command executes the command file and initiates processing of the mapping operations.

```
symconfigure -sid 605 -file map_dyn_devices.cmd -v -noprompt commit

    Establishing a configuration change session..............Established.
    Processing symmetrix 000000005605
    {
      map dev 0091 to dir 4A:0, lun=004;
      map dev 0092 to dir 4A:0, lun=005;
      map dev 0093 to dir 4A:0, lun=006;
      map dev 0094 to dir 4A:0, lun=007;
      map dev 0095 to dir 4A:0, lun=008;
      map dev 0096 to dir 4A:0, lun=009;
    }

    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 046 steps....................................Executing.
    ..........................................................................
    Step 041 of 046 steps....................................Executing.
    Step 041 of 046 steps....................................Executing.
    Local:  COMMIT...........................................Done.
    Terminating the configuration change session..............Done.
```

- The `symcfg discover` command updates the SYMAPI database on the host.

```
symcfg discover

This operation may take up to a few minutes. Please be patient...
```

- The `sympd list` command shows that the newly-mapped devices are now visible to the host.

```
sympd list -sid 605

Symmetrix ID: 000000005605

        Device Name            Directors                   Device
---------------------------- ------------ ------------------------------------
                                                                          Cap
Physical                 Sym  SA :P DA :IT Config        Attribute    Sts   (MB)
---------------------------- ------------ ------------------------------------

/dev/rdsk/c1t0d1s2       0001 04A:0 16B:C1 2-Way Mir     Grp'd        RW     516
/dev/rdsk/c1t0d2s2       0002 04A:0 02B:C0 2-Way Mir     Grp'd        RW     516
/dev/rdsk/c1t0d3s2       0003 04A:0 15A:C0 2-Way Mir     Grp'd        RW     516
/dev/rdsk/c1t0d4s2       0091 04A:0 16B:C0 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d5s2       0092 04A:0 02B:C1 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d6s2       0093 04A:0 15B:C1 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d7s2       0094 04A:0 16B:D0 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d8s2       0095 04A:0 02B:D1 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d9s2       0096 04A:0 15A:D1 RAID-5        N/Grp'd      RW     516
/dev/rdsk/c1t0d96s2      0040 04A:0 02B:C1 Unprotected   N/Grp'd      RW       3
/dev/rdsk/c1t0d97s2      0041 04A:0 15A:C1 Unprotected   N/Grp'd      RW       3
/dev/rdsk/c1t0d98s2      0042 04A:0 01B:C1 Unprotected   N/Grp'd      RW       3
/dev/rdsk/c1t0d99s2      0043 04A:0 16A:C1 Unprotected   N/Grp'd      RW       3
```

- At this point, the example needs to return to the beginning and repeat the same steps to create dynamic RDF devices (010 through 015) on the *remote* RDF-linked Symmetrix array (085). For the sake of brevity, the example does not repeat these steps. Now that the devices are created, available for use as dynamic RDF devices, and mapped, subsequent processing is now handled using the `symrdf` utility.

  The following command uses the vi text editor to create a text file named create_pair.cmd. As was done here, you can enter into the file those device names that will constitute the dynamic SRDF pairs. The R1 devices are listed in first column, and the R2 devices created on the remote Symmetrix are listed in the second column on the same line as their respective R1 source. For more information about creating and using dynamic SRDF pairs, refer to the *EMC Solutions Enabler Symmetrix SRDF Family CLI Product Guide*.

```
vi create_pair.cmd

0091  0010
0092  0011
0093  0012
0094  0013
0095  0014
0096  0015
```

- If your initial operation is to establish the pairs, use the `-establish` option or `-invalidate R2` option when performing the create operation. The example uses the `-invalidate` option, which allows the creation of dynamic SRDF pairs without bringing up the RDF links and initiating the copying of data. However, when invalidating, you need to first write-disable the devices on *both* local and remote Symmetrix arrays. The following commands show only the write disable operation for the six R1 devices on the *local* Symmetrix array (sid 605). Assume that the example performs these steps also for the six R2 devices (010 through 015) on the remote Symmetrix array (sid 085).

  The following commands create a Regular type device group (new_dyn_rdf_grp), add the range of devices (091 through 096) to the device group, and place all devices in the device group in the Write Disable state.

  ```
  symdg create new_dyn_rdf_grp -type regular
  symld -g new_dyn_rdf_grp -sid 605 addall dev -range 091:096
  symld -g new_dyn_rdf_grp write_disable
  ```

- The `symdg list` command displays the new device group (new_dyn_rdf_grp) among those device groups defined on this host.

```
symdg list -sid 605
```

```
                                D E V I C E      G R O U P S

                                                                    Num of   Num of  Num of
    Name                  Type      Valid  Symmetrix ID            Devices  GK's    BCV's

    oracle_1              REGULAR   Yes    000000005605               3        0       0
    doc                   REGULAR   Yes    000000005605               5        0       0
    payroll               RDF2      Yes    000000005605               3        0       0
    records               RDF1      Yes    000000005605               4        0       0
    new_dyn_rdf_grp       REGULAR   Yes    000000005605               6        0       0
```

- The `symrdf createpair` command parses the file called create_pair.cmd 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 (sid 605). The device group containing these devices will be changed from a Regular device group to an RDF device group. Communication is via RDF group 1 (`-rdfg 1`). The `-invalidate r2` option invalidates all tracks on the R2 devices in preparation for a subsequent establish operation that will copy data from the R1 devices to the R2 devices.

```
symrdf createpair -file create_pair.cmd -sid 605 -rdfg 1 -type RDF1 -invalidate R2
-noprompt
```

```
An RDF 'Create Pair' operation execution is in progress for device file
   'create_pair.cmd'. Please wait...

    Create RDF Pair.....................................................Done.
    Mark target device(s) in (5605,01) for full copy from source....Started.
    Device: 0091 ..................................................... Marked.
    Device: 0092 ..................................................... Marked.
    Device: 0093 ..................................................... Marked.
    Device: 0094 ..................................................... Marked.
    Device: 0095 ..................................................... Marked.
    Device: 0096 ..................................................... Marked.
    Mark target device(s) in (5605,01) for full copy from source....Done.

The RDF 'Create Pair' operation successfully executed for device file
   'create_pair.cmd'.
```

- The `symdg list` command displays the new RDF device group (new_dyn_rdf_grp) among those device groups defined on this host. Note that the new_dyn_rdf_grp device group has been changed from a Regular type group to an RDF1 type device group.

symdg list -sid 605

```
                                D E V I C E     G R O U P S

                                                                  Num of   Num of  Num of
 Name                Type      Valid  Symmetrix ID    Devices  GK's
BCV's

 oracle_1           REGULAR  Yes    000000005605      3       0      0
 doc                REGULAR  Yes    000000005605      5       0      0
 payroll            RDF2     Yes    000000005605      3       0      0
 records            RDF1     Yes    000000005605      4       0      0
 new_dyn_rdf_grp    RDF1     Yes    000000005605      6       0      0
```

- The `symrdf query` command shows the configuration and status of the new dynamic SRDF pairs within this device group. The new SRDF pairs are in the Suspended state and, as expected, the RDF links are Not Ready (NR).

symrdf -g new_dyn_rdf_grp query

```
Device Group (DG) Name: new_dyn_rdf_grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000005605

       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  0091 WD       0    16500 NR 0010 WD       0    16500 S..   Suspended
DEV002  0092 WD       0    16500 NR 0011 WD       0    16500 S..   Suspended
DEV003  0093 WD       0    16500 NR 0012 WD       0    16500 S..   Suspended
DEV004  0094 WD       0    16500 NR 0013 WD       0    16500 S..   Suspended
DEV005  0095 WD       0    16500 NR 0014 WD       0    16500 S..   Suspended
DEV006  0096 WD       0    16500 NR 0015 WD       0    16500 S..   Suspended

Total            ------   ------             ------   ------
  Track(s)            0    99000                  0    99000
  MB(s)             0.0   3093.6                0.0   3093.6

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
```

- The `symrdf establish` command initiates copying R1 data to R2 devices. The `-invalidate r2` option from the previous command invalidated the R2 devices, a step that is usually carried out during a full establish operation. Consequently, you do not need the `-full` option here. The invalidate step is not repeated, regardless of whether you use the `-full` option or not. If subsequently you re-establish or restore the dynamic SRDF pairs, omitting or including the `-full` option will affect how the copy occurs (either incremental copy or full copy, respectively). The output below says "Incremental Establish" because the `-full` option was omitted. However, because all tracks on the R2 devices were previously invalidated, the result is a full copy of all R1 tracks to the R2 tracks.

```
symrdf -g new_dyn_rdf_grp establish -noprompt -sid 605

An RDF 'Incremental Establish' operation execution is in progress for
   device group 'new_dyn_rdf_grp'. Please wait...

    Suspend RDF link(s).....................................Done.
    Read/Write Enable device(s) on SA at source (R1).......Done.
    Resume RDF link(s)......................................Not Done.
    Merge device track tables between source and target.....Started.
    Devices: 0091-0096 ..................................... Merged.
    Merge device track tables between source and target.....Done.
    Resume RDF link(s)......................................Done.

The RDF 'Incremental Establish' operation successfully initiated for
   device group 'new_dyn_rdf_grp'.
```

- The following query displays the status of the dynamic SRDF pairs. The pairs are currently in the process of synchronizing (pair state is SyncInProg).

```
symrdf -g new_dyn_rdf_grp query -sid 605

Device Group (DG) Name: new_dyn_rdf_grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000005605

       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  0091 RW       0    16500 RW 0010 WD       0    16500 S..   SyncInProg
DEV002  0092 RW       0    16500 RW 0011 WD       0    16500 S..   SyncInProg
DEV003  0093 RW       0    16500 RW 0012 WD       0    16500 S..   SyncInProg
DEV004  0094 RW       0    16500 RW 0013 WD       0    16500 S..   SyncInProg
DEV005  0095 RW       0    16500 RW 0014 WD       0    16500 S..   SyncInProg
DEV006  0096 RW       0    16500 RW 0015 WD       0    16500 S..   SyncInProg

Total            ------   ------             ------   ------
  Track(s)            0    99000                  0    99000
  MB(s)             0.0   3093.6                0.0   3093.6

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
```

- The `symrdf verify` command checks at 30-second intervals and verifies when the dynamic SRDF pairs have reached the Synchronized state. The ellipsis (...) represents repetitive output that was omitted.

```
symrdf verify -g new_dyn_rdf_grp -i 30 -sid 605

NONE of the mirrored pairs are in the 'Synchronized' state

NONE of the mirrored pairs are in the 'Synchronized' state
.....
All devices in the RDF group 'new_dyn_rdf_grp' are in the 'Synchronized' state.
Another query confirms the SRDF pairs are now in the Synchronized state.
symrdf -g new_dyn_rdf_grp query -sid 605

Device Group (DG) Name: new_dyn_rdf_grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000005605

       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  0091 RW       0         0 RW 0010 WD       0         0 S..   Synchronized
DEV002  0092 RW       0         0 RW 0011 WD       0         0 S..   Synchronized
DEV003  0093 RW       0         0 RW 0012 WD       0         0 S..   Synchronized
DEV004  0094 RW       0         0 RW 0013 WD       0         0 S..   Synchronized
DEV005  0095 RW       0         0 RW 0014 WD       0         0 S..   Synchronized
DEV006  0096 RW       0         0 RW 0015 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 9: 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


                                                Remote        Local   Remote
  Ident   Symb    Num   Slot   Type       Attr  SymmID        RA Grp  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


                                                Remote       Local   Remote
  Ident   Symb    Num   Slot   Type       Attr  SymmID       RA Grp  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


                                                Remote       Local   Remote
  Ident   Symb    Num   Slot   Type       Attr  SymmID       RA Grp  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.

```
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


                                                 Remote       Local   Remote
  Ident   Symb    Num   Slot   Type       Attr   SymmID       RA Grp  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
--------------------------- ------------ ------------------------------------
                                                                        Cap
Sym  Physical               SA :P DA :IT  Config        Attribute    Sts   (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 `OEA2O6A2.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 OEA2O6A2.list

300  150
301  151
302  152
303  153
304  154
305  155
```

- The `symrdf createpair` command parses the file called `OEA2OEB2.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 OEA2OEB2.list -sid 11 -rdfg 58 -type rdf1 -establish
   -noprompt

An RDF 'Create Pair' operation execution is in progress for device
file 'OEA2OEB2.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 'OEA2OEB2.list'.
```

- A second `symrdf createpair` command parses the file called `OEA2O6A2.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 OEA2O6A2.list -sid 11 -rdfg 51 -type rdf1 -establish -nop

An RDF 'Create Pair' operation execution is in progress for device
file 'OEA2O6A2.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 'OEA2O6A2.list'.
```

- The `symdg` 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).

  ```
  symdg 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
```

8 Device Masking Examples

This chapter provides detailed examples on using the device masking component.

- Example 1: Adding masked devices for HBA access 254
- Example 2: Using Fibre Channel ID lockdown 256
- Example 3: Options when displaying the VCMDB 258

Example 1: Adding masked devices for HBA access

The hardware setup consists of a single Symmetrix array (sid 313) connected to a Solaris controlling host.

The following command lists host HBAs that have a channel to a Symmetrix array and generates an AWWN (the User-generated Name column) for any HBA that does not have an AWWN assigned yet. This information is stored in the login history table. The Identifier column displays an HBA's unique WWN.

```
symmask discover hba


Symmetrix ID          : 000184500313
Device Masking Status : Success

  Identifier        Type   User-generated Name
  ----------------  -----  --------------------------------
  10000000c9274156  Fibre  api140/i@1f,4000,@2
  10000000c92741a1  Fibre  api140/i@1f,4000,@4
```

The following command lists the FA director/port (Dir:P) to which each HBA connects and the various physical device paths of the VCMDB device on the Symmetrix array. Notice that one of the director/ports (14D, port 0) is connected to both HBAs. Either HBA can access devices that are available to 14D:0.

```
symmask list hba

Identifier        Type   Adapter           Physical Device Path      Dir:P
----------------  -----  ----------------  ------------------------  -----
10000000c9274156  Fibre  i@1f,4000,@2      /dev/rdsk/c1t0d0s2        14B:0
                                           /dev/rdsk/c1t1d0s2        14D:0
10000000c92741a1  Fibre  i@1f,4000,@4      /dev/rdsk/c2t0d0s2        14B:0
                                           /dev/rdsk/c2t2d80s2       14C:0
                                           /dev/rdsk/c2t1d0s2        14D:0
```

The `symcfg list` command with the `-addr` option displays a list of devices that are available to a specified director/port (in this case, FA director 14D). This example uses devices 0019, 0060, 00A0, and 00D4. Notice that devices 0060 and 00A0 are metadevices (M).

```
symcfg list -FA 14D -add

Symmetrix ID: 000184500313 (Local)

          Director                  Device Name          Attr     Address
    ---------------------- ------------------------------ ---- --------------
    Ident   Symbolic  Port Sym   Physical                     VBUS  TID  LUN
    ------  --------  ---- ----  -----------------------      ----  ---  ---

    FA-14D  14D       0    0402  /dev/rdsk/c1t1d0s2      VCM     0    0  000
                           0019  Not Visible                     0    0  001
                           001A  Not Visible                     0    0  002
                           001B  Not Visible                     0    0  003
                           001C  Not Visible                     0    0  004
                           001D  Not Visible                     0    0  005
                           001E  Not Visible                     0    0  006
                           001F  Not Visible                     0    0  007
                           0060  Not Visible               (M)   0    0  010
                           00A0  Not Visible               (M)   0    0  011
                           00D4  Not Visible                     0    0  019
                           00D5  Not Visible                     0    0  01A
                           00D6  Not Visible                     0    0  01B
```

```
00D7  Not Visible                    0     0  01C
00D8  Not Visible                    0     0  01D
00D9  Not Visible                    0     0  01E
00DA  Not Visible                    0     0  01F
00DB  Not Visible                    0     0  020
0018  Not Visible                    0     0  030
03FD  Not Visible                    0     0  060
```

The `symmask add` command adds devices 0019, 0060, 00A0, and 00D4 to the HBA whose WWN identifier is 10000000c92741a1 and whose director/port channel is director 14D, port 0.

```
symmask add dev 19,60,A0,D4 -wwn 10000000c92741a1 -dir 14D -p 0
```

The following command checks the device masking database (VCMDB) to determine that the addition was correct. For the two metadevices, only the metaheads will be added, device 0060 and 00A0, to the database and listed in the output. Any attempt to add a metamember device will fail with an error.

```
symmaskdb list database


Symmetrix ID            : 000184500313

Last updated at         : N/A

Director Identification : FA-14D
Director Port           : 0

                                 User-generated
Identifier        Type   Node Name        Port Name          Devices
----------------  -----  ---------------------------------  ---------
10000000c92741a1  Fibre  api140           i@1f,4000,@4       0019
                                                            0060:0067
                                                            00A0:00A7
                                                            00D4
```

EMC recommends backing up the VCMDB after *all* additions are made (not after each `symmask add` command). Since these are the only additions that this example makes, the following command backs up the VCMDB to a file called backup0723.

```
symmaskdb backup -file backup0723

Backup Symmetrix SymMask database on Symmetrix 000184500313 to backup file
   backup0723 (y/[n])? y

Symmetrix SymMask database backed up to file backup0723 from Symmetrix 000184500313
```

After completing your additions, refresh the WWN-related profile tables in Symmetrix director cache with the latest VCMDB data. The following command updates the cache on the FA director with the contents of the VCMDB.

```
symmask refresh

Refresh Symmetrix FA directors with contents of SymMask database 000184500313
   (y/[n]) ? y

Symmetrix FA directors updated with contents of SymMask Database 000184500313
```

Reboot the host whose HBA entry you added to the VCMDB and then issue the `symcfg discover` command to scan the Symmetrix devices and refresh the SYMAPI database.

```
symcfg discover
```

Example 2: Using Fibre Channel ID lockdown

This example uses the same hardware setup as Example 1 and demonstrates the security feature that limits host access to devices by adding the Fibre Channel ID (FCID) value to an HBA entry in the VCMDB.

This command allows you to determine the physical device path for each HBA you want to lock down and the director/port on which it will be locked down. Only the paths for each HBA to the VCMDB will be displayed; if no path to the VCMDB exists for a given device, then no physical device path will appear. The example uses the first WWN on director/port 14B:0, which has a physical device path of `/dev/rdsk/c1t0d0s2`.

```
symmask list hba

Identifier        Type   Adapter           Physical Device Path      Dir:P
----------------  -----  ----------------  ------------------------  -----
10000000c9274156  Fibre  i@1f,4000,@2      /dev/rdsk/c1t0d0s2        14B:0
                                           /dev/rdsk/c1t1d0s2        14D:0
10000000c92741a1  Fibre  i@1f,4000,@4      /dev/rdsk/c2t0d0s2        14B:0
                                           /dev/rdsk/c2t2d80s2       14C:0
                                           /dev/rdsk/c2t1d0s2        14D:0
```

The `symmask list logins` command allows you to determine the FCID value (100200) of the Fibre Channel to be locked down (director/port 14B:0, and physical device path `/dev/rdsk/c1t0d0s2`).

```
symmask list logins -dir 14B -p 0 -pdev /dev/rdsk/c1t0d0s2

Director Identification : FA-14B
Director Port           : 0

                            User-generated                        Logged On
Identifier       Type  Node Name         Port Name         FCID   In     Fabric
---------------- ----- --------------------------------- ------ ------ ------
10000000c920c484 Fibre NULL              NULL              100001 Yes    Yes
10000000c9274156 Fibre api140            i@1f,4000,@2      100200 Yes    Yes
10000000c92741a1 Fibre api140            i@1f,4000,@4      100500 Yes    Yes
210000e08b046406 Fibre NULL              NULL              100800 Yes    Yes
210000e08b046506 Fibre NULL              NULL              100a00 Yes    Yes
210100e08b246506 Fibre NULL              NULL              100b00 Yes    Yes
```

The following command locks downs the Fibre Channel specified by the FCID 100200.

```
symmask set lockdown on 100200 -wwn 10000000c9274156 -dir 14B -p 0
```

The following command with the verbose (-v) option provides a detailed display of HBA-to-FA channel entry in the VCMDB and verifies that the correct FCID value was added to this entry.

```
symmaskdb list database -dir 14B -p 0 -v

Symmetrix ID            : 000184500313
Last updated at         : N/A
Director Identification : FA-14B
Director Port           : 0

   Originator Port wwn  : 10000000c9274156
   Type                 : Fibre
   User-generated Name  : api140/i@1f,4000,@2
   Visibility           : No
   FCID Lockdown        : Yes
     FCID Value         : 100200
   Lun Offset           : No
   Heterogeneous Host   : No
   Devices              : 0008 000A:000F 00D5
```

The following command backs up the VCMDB to a file named backup0723.

```
symmaskdb backup -file backup0723

Backup Symmetrix SymMask database on Symmetrix 000184500313 to backup file
   backup0723 (y/[n])? y

Symmetrix SymMask database backed up to file backup0723 from Symmetrix 000184500313
```

If you have completed your entire set of changes for this session, refresh the VCMDB in the Symmetrix cache.

```
symmask refresh

Refresh Symmetrix FA directors with contents of SymMask database 000184500313
   (y/[n]) ? y

Symmetrix FA directors updated with contents of SymMask Database 000184500313
```

Example 3: Options when displaying the VCMDB

The following `symmaskdb list` commands are available with Solutions Enabler.

The following `symmaskdb list capacity` command displays the capacity of all devices assigned through an HBA-to-FA channel to host api140. The ellipsis (…) indicates output that was omitted.

```
symmaskdb list capacity -host api140

Symmetrix ID      : 000184500313

Host Name         : api140

Identifiers Found : 10000000c9274156
                    10000000c92741a1

Device  Cap(MB)  Attr  Dir:P
------  -------  ----  ----
0009       1031        14B:0
000A       1031        14B:0
000B       1031        14B:0
000C       1031        14B:0
000D       1031        14B:0
000E       1031        14B:0
000F       1031        14B:0
0011       1031        14C:0
0012       1031        14C:0
0013       1031        14C:0
0014       1031        14C:0
0015       1031        14C:0
0016       1031        14C:0
0017       1031        14C:0
0050       8250   (M)  14B:0
0051          -   (m)  14B:0
0052       1031        14B:0
0053          -   (m)  14B:0
0054          -   (m)  14B:0
0055          -   (m)  14B:0
0056          -   (m)  14B:0
0057          -   (m)  14B:0
0090       8250   (M)  14B:0
0091          -   (m)  14B:0
0092          -   (m)  14B:0
0093          -   (m)  14B:0
0094          -   (m)  14B:0
0095          -   (m)  14B:0
0096          -   (m)  14B:0
0097          -   (m)  14B:0
.....................................................
0402          7        14C:0

-----------------------------

MB Total:  42282
GB Total:   41.3
```

The `symmaskdb list assignment` command displays which HBAs are assigned to a set of devices. The range of devices specified is 0001 through 000A.

```
symmaskdb list assignment -dev 0001:000A

Symmetrix ID : 000184500313

Device  Identifier        Type   Dir:P
------  ----------------  -----  -----
0001    10000000c9229db5  FIBRE  14A:0
0002    10000000c9229db5  FIBRE  14A:0
0003    10000000c9229db5  FIBRE  14A:0
0004    10000000c9229db5  FIBRE  14A:0
0005    10000000c9229db5  FIBRE  14A:0
0006    10000000c9229db5  FIBRE  14A:0
0007    10000000c9229db5  FIBRE  14A:0
0008    -                 -      -
0009    10000000c9274156  FIBRE  14B:0
000A    10000000c9274156  FIBRE  14B:0
```

A

Updating the Host

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 262
- HP-UX systems 263
- IBM AIX systems 264
- HP Tru64 UNIX systems 265
- Windows systems 266

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.

B

Device Masking: Volume Logix Conversion

This appendix provides a mapping of EMC's previously supported Volume Logix commands to the SYMCLI device masking command.

- Volume Logix conversion..... 268

Volume Logix conversion

If you have been using EMC's Volume Logix and need to convert to the SYMCLI device masking command set, Table 18 lists equivalent SYMCLI commands to replace the Volume Logix commands.

Table 18 Volume Logix to SYMCLI conversion

VCM command	SYMCLI command
vcmfind	symmask discover hbas
fpath lshbawwn	symmask list hbas
fpath lshosts -d	symmask -sid list logins
fpath adddev -d [-w|-u] -f -r	symmask -sid [-wwn|-awwn] -dir -p add devs <r>
fpath rmdev -d [-w|-u] -f -r	symmask -sid [-wwn|-awwn] -dir -p remove devs <r>
fpath chgname -d -w -n	symmask -sid -wwn rename <awwn>
fpath clrwwn -d [-w|-u]	symmask -sid [-wwn|-awwn] delete
fpath swaphba -d [-w|-u] -n	symmask -sid [-wwn|-awwn] replace <wwn>
fpath chgattr -d -h	symmask -sid [-wwn|-awwn] -dir -p set visibility <on|off>
fpath chgattr -d -s -v	symmask -sid [-wwn|-awwn] -dir -p set lockdown <on <fcid>|off>
fpath chgattr -d -l -o -b	symmask -sid [-wwn|-awwn] -dir -p set lunoffset <on <offset><base>|off>
fpath chgattr -d -m -c	symmask -sid [-wwn|-awwn] -dir -p set heterogeneous <on <host>|off>
fpath refresh -d	symmask -sid refresh
fpath lsdb -d	symmaskdb -sid list database
fpath lsbackup -o	symmaskdb -file list database
fpath lshbavols [-d|-o] [-w|-u]	symmaskdb -sid [-wwn|-awwn] list devs
fpath initdb -d	symmaskdb -sid -file init
fpath backupdb -d	symmaskdb -sid -file backup
fpath restoredb -d	symmaskdb -sid -file restore
fpath lshostdev	sympd list -sid
fpath lssymmdev -d	symcfg list -sid -FA all -address
fpath lssymmfas -d	symcfg list -sid -FA all
fpath lsstatus -d	symcfg list -sid -v

Index

A

abort argument 91
access control, and device masking 100
adding devices 106, 107
advise, reservations 58
alias
  creating 118
  renaming 118
ASCII World Wide Names (AWWN)
  establishing 104
  format 104
  usage 105
attributes, setting 84

B

backup file
  creating a database 111
BCV configurations 25
BCV devices 25
Brocade switch
  finding the FCID for 121
business continuance
  operations 27

C

cache partitions 175
  adding devices to 176
  creating 175
  listing 177
  modifying 176
  removing devices from 177
change sessions
  aborting 91
  monitoring changes 90
CHAP
  configuring 138
  for iSCSI 98
checksum
  benefits of using 183
  checking databases 182
  configuration parameter setting for 184
  license requirements 184
  validation list 186
  where it occurs 182
checksum testing
  implementing 184
  validating your system 186
  verifying checksum is enabled 186
  within the Symmetrix 183
CKD devices 92
  assigning PAV aliases 74
  copying a CU image 74
  mapping a range 73
  removing PAV aliases 75
  unmapping a range 73
classes 40
code examples, Quality of Service 179
command file 42
  mandatory actions to perform 164
  parameters 168
Comments 15
commit argument 43
concurrent RDF, enabling 84
configuration component 20
configuration verification 41
Connectrix switch
  finding the FCID for 121
converting databases 112
copy operation
  for BCV 171
  for Clone 171
  for mirrors 171
  for RDF 171

D

data corruption
  between host and storage 182
  traditional checking 182
data preservation 72
data, refreshing 86
device locking 39
device LUN visibility 122
device masking
  architecture 94
  configuration steps 101

functionality 94
device masking database
activating a new configuration 102
backup file for 111
blocking direct writes to 113
converting 112
discover 96
examining 114
external lock, releasing 96
initializing 110
maintenance 110
restore and convert 112
restoring 112
security for 99
types 111
using syscalls with 99
device reservations 57
device type
conversion 34
creating 34
devices
adding 106, 107
capacity of 116
converting 52
creating (adding) 45
deleting 50
mapping 61
mapping to front-end ports 61
removing 108
setting characteristics 55
unmapping 63
directors
refresh 102, 119
discover
using 96
disk management
adding a spare disk 87
deleting a spare disk 87
disk space 41
disk space, configuring into devices 48
dyn_rdf, enabling 55
dyn_rdf1_only, enabling 55
dyn_rdf2_only, enabling 55
dynamic cache memory 175
dynamic concurrent RDF, enabling 84
dynamic LUN addressing, adding devices with 107
dynamic RDF 64
devices, configuring 64
enabling 66, 84
dynamic reallocation volumes (DRVs) 156
dynamic swap 89

E

enabling checksum 186
enforce, reservations 58
environment option, non visible devices 44
external lock
on VCMDB 96

F

FBA multi access cache, enabling 84
Fibre Channel ID (FCID)
finding the FCID 121
for Brocade switch 121
for Connectrix switch 121
format 122
lock down 120
lock down effects on commands 121
lock down procedure 121
fibre protocols 79
file
masking devices using 107
removing devices using 109
free disk space 41

G

Generic SafeWrite 188
device flag 190
disabling 191
where to enable 189

H

HBA
alias names 104
management 118
supported initiators 98
swapping 118
HBA, deleting associations 119
held devices 39
host updating 42
hot swap policy, setting 84

I

iSCSI
configuring the software driver for 137
topology 98
with CHAP authentication 138
without CHAP authentication 146

L

latency time 158
listing extents
output 185
lock
on VCMDB 96
releasing 96
login history table
contents 117
usage 104
LUN
skip hole 122
visibility 122
LUN offset
hexidecimal value 122

M

masked channel 94
maximum hypers per disk, setting 84
meta devices 27, 106
  converting 71
  creating 35
metadevices
  configuration restrictions 69
  configuring 68
  converting 71
  creating 68
  dissolving 71
  metahead 27
  performance 27
  size 27
metamembers
  adding concatenated 68
  adding striped 68
  removing 70
  removing from the database 108
metrics, viewing 85
migrating devices 167
mirror set protections 21
mirrors
  RAID-S 24
  three way 23
  two way 22
MVS subsystem IDs 46, 49, 92

N

new devices, introducing to
  AIX systems 264
  Digital UNIX systems 265
  HP-UX systems 263
  Solaris systems 262
  Windows systems 266
non visible devices 44

O

operations classes 40
Optimizer
  best swaps with 158
  command file 168
  configuration requirements 156
  copying volumes to DRVs 159
  legal swaps 160
  migrating devices 167
  performance metrics 157
  rollbacks 166
  service times 158
  setting parameters for 163
  setting performance time windows 164
  setting user approved mode 164
  starting 163
  stopping 163
  swap history 166
  swap procedure 159
  time windows, showing 165

P

parity RAID 24
PAV alias addresses
  adding to mapped devices 74
  removing from mapped devices 75
PAV alias limit, setting 84
PAV mode, specifying 84
performance time windows, setting 164
port characteristics
  copying from one port to another 79
  setting 79
port flags 79
  viewing 83
prepare argument 43
preview argument 43
profile tables 102, 119

Q

QoS
  dynamic cache partitions 175
Quality of Service
  code examples 179
  Enginuity version requirement 172
  parameters for 179
  setting the copy pace level 171
query argument 90

R

R1/R2 devices, swapping RDF personalities 89
RA groups, swapping 86
RAID 5 support, enabling 84
RAID-S 24
RAID-S support, enabling 84
RDB_Cksum, enabling 55
refresh directors 102, 119
refresh option 86
removing devices 108
removing masked devices 108
reservations
  committing changes to 58
  device 57
  list output example 59
  releasing 60
  viewing 59
restoring a database 112
restrictions
  for configuring metadevices 36, 69
  for configuring SAVE devices 37
  for configuring virtual devices 36
  for converting devices 54
  for creating devices 47
  for deleting devices 50
  for setting device characteristics 56
  for swapping RA groups 86
rollbacks
  performing 166

S

safe disciplines 41
safety considerations 41
SAVE device 76
SAVE device pool, types 77
SAVE device pools
  adding devices 77
  enabling/disabling devices 78
  removing devices 78
SCSI protocols 79
SCSI writes
  blocking from the database 113
SCSI3_persist_reserv, enabling 55
SDR
  mapping 61
  unmapping 63
security
  CHAP protocol 98
seek time 158
service attributes, setting 55
skip hole
  hexidecimal value 122
  LUN base/offset 122
Snap pool 77
spare disks
  adding 87
  removing 87
  viewing 88
SRDF configurations 25
SRDF devices 25
  converting from static to dynamic 52
SRDF group attributes
  setting 67
SRDF/A
  assigning session priority 67
  setting host throttle time 84
SRDF/A DSE
  enabling autostart 67
  naming pools 67
  pool 77
  setting threshold 67
SSIDs 92
stripe sizes 28
striped data 28
striped meta
  protect_data 70
subsystem IDs 92
suppress feature option 185
swap RDF devices 64
swaps
  impact of 162
  Optimizer 160
symmask actions
  list hba 101
  list logins 102
symmaskdb actions
  backup 111
  init 110
  list database 114
Symmetrix
  device locks 39
  model numbers 13
  setting attributes 84
symoptmz 163
  command features 163
symqos 170
symqos, command description 170

T

time windows
  clearing 165
  showing 165
transfer time 158
transmit idle, enabling 67

U

Updating the host 261
user approved mode, for Optimizer 164

V

VCMDB
  converting 112
  enabling 55
  See also device masking database
  setting restricted access 85
  types 111
visibility
  devices 122
Volume Logix
  command conversion 96, 268

W

World Wide Name (WWN) 104
  profile tables 102, 119
WORM attribute, enabling 55