EMC® Solutions Enabler TimeFinder® Family

Version 8.0.1

CLI User Guide

VMAX® and VMAX3™ Family

REV 01

EMC²

Copyright © 2003 - 2014 EMC Corporation. All rights reserved. Published in the USA.

Published November, 2014

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.

EMC², EMC, and the EMC logo are registered trademarks or trademarks of EMC Corporation in the United States and other countries. All other trademarks used herein are the property of their respective owners.

For the most up-to-date regulatory document for your product line, go to the technical documentation and advisories section on the EMC online support website.

CONTENTS

Preface

Chapter 1 Introduction to the TimeFinder CLI

Introduction to TimeFinder 18
TimeFinder SnapVX (HYPERMAX OS) 18
TimeFinder Enginuity (5773 - 5876) 19
Copy Session Limits 19
TimeFinder CLI overview 20
VMAX3 arrays (HYPERMAX OS) 20
VMAX arrays (Enginuity 5773 - 5876) 20
TimeFinder emulations (HYPERMAX OS) 21
Command summary 22

Chapter 2 Performing TimeFinder SnapVX Operations (HYPERMAX OS)

TimeFinder SnapVX Overview 26
SnapVX supported configurations 26
SnapVX operating modes 26
Establishing a snapshot 27
Consistent snapshots 27
Time to live 28
Restoring a snapshot 28
Linking to a snapshot 29
Relinking 30
Unlinking 30
Displaying link status 30
Displaying snapshot and link summaries 31
Terminating a snapshot 32
Cascading snapshots 33
Modified point-in-time copies 34
SnapVX behavior when the number of source devices changes 35
Restore operations 35
Link operations 36
Relink operations 36

Chapter 3 Performing TimeFinder/Clone Operations

TimeFinder/Clone overview 38
Creating a clone copy session 38
Fully copying to a clone 39
Creating a session in nocopy mode 40
Starting to copy data before activating the session 40
Copying only changed data to a clone 41
Copying from a source device to a larger target device 41
Using the establish command 42
Pairing an additional target device with each source device in a group 42
Performing operations on devices in a clone target list 42
Activating a clone copy session 43
Precopying data before activating the session 43

Making the target device not ready to the host 44
Activating copy sessions consistently 44
Activating a clone session for an additional pair in a group 45
Modifying a clone copy session 46
Recreating a clone copy device 46
Starting to copy data before activating the session 47
Using the establish command 47
Recreating a clone session for each pair in a group 47
Restoring data from a target device 47
Splitting a clone device pair 48
Terminating a clone copy session 48
Querying clone pairs 49
Using the -summary option 49
Verifying clone pair states 50
Cascaded clone 52
Restore to Target 52
Persistent Restore to Target 52
Configuration Rules 53
Cascaded clone with VP Snap 55
Using a BCV as the clone source 55
Pair states ruling clone operations 56
Example: Creating a clone from a source device 57
Creating multiple clone copies from a standard device 58
Cloning a copy on a remote array 58
Cloning a local R1 standard device 59
Cloning a copy of a local BCV device on a remote array 60
Cloning a copy on a hop-2 array 61
Cloning copies of the same data locally and remotely 62
Cloning multiple copies locally and remotely 64
Cloning a copy at the tertiary site of a cascaded SRDF configuration 65
Using composite groups to manage clone pairs across arrays 67
Command options with device groups 69
Command options with composite groups 70
Command options with device files 72

Chapter 4 Performing TimeFinder/Snap Operations (Enginuity versions 5773 through 5876)

TimeFinder/Snap overview 76
Creating a virtual copy session 76
Multivirtual snaps 77
Specifying a SAVE device pool 77
Monitoring SAVE device usage 78
Pairing an additional target device with each source device in a group .. 78
Copying a virtual device to another virtual device (duplicate snap) 78
Activating a virtual copy session 79
Using the establish command 79
Making the target device not ready to the host 79
Activating copy sessions consistently 80
Activating an additional copy session for each device pair in a group 81
Activating a duplicate snap session 81
Recreating a virtual copy session 82
Restoring data from virtual devices 83
Incrementally restoring to a source 83
Incrementally restoring to a BCV 84

Fully restoring to anywhere 86
Terminating a virtual copy session 87
Terminating a duplicate snap session 87
Querying snap pairs 87
Using the -summary option 88
Verifying snap pair states 88
Using a BCV as the snap source 90
Pair states ruling snap operations 90
Example: Creating a virtual copy from a BCV 90
Creating multiple virtual copies 92
Creating multiple virtual copies from a standard device 92
Creating multiple virtual copies from a BCV device 94
Attaching source and target virtual devices 95
Using composite groups to manage snap pairs across arrays 96
Snapping a copy on a remote array 99
Snapping a copy from a remote BCV 100
Snapping copies of a source device's data locally and remotely 101
Snapping multiple copies 102
Snapping a copy at the tertiary site of a cascaded SRDF configuration 104
Snapping a copy from a clone target device 105
Command options with device groups or composite groups 106
Command options with device files 107

Chapter 5 Performing TimeFinder VP Snap Operations

TimeFinder VP Snap overview 110
Creating a VP Snap copy session 110
Activating a VP Snap session 111
Using the establish command 111
Restoring a VP Snap session 111
Terminating a VP Snap copy session 112

Chapter 6 Performing TimeFinder/Mirror Operations

Clone Emulation mode and TimeFinder/Mirror 114
TimeFinder/Mirror 114
SRDF-connected sites 115
TimeFinder/Clone Emulation 117
TimeFinder operations overview 119
Device external locks 119
Disallow synchronization actions 119
Wait for synchronization actions to complete 120
Listing BCV devices 120
Associating BCV devices with a device group 121
Disassociating BCV devices from a device group 125
Moving BCV devices from one device group to another device group 126
Managing BCV devices with composite groups 127
Establishing BCV pairs 128
Specifying the default method for establishing BCV pairs 131
Incrementally establishing BCV pairs 140
Splitting BCV pairs 142
Performing a reverse split 145
TimeFinder consistent split 149
Fully restoring BCV pairs 153
Incrementally restoring BCV pairs 156

Protecting BCV data during full or incremental restores 160
Cancelling BCV pairs 160
Querying BCV pairs 161
Using the -summary option 162
Verifying BCV pair states 163
Using composite groups to manage BCV pairs across arrays 165
Preferred attachment of BCVs (optional operations) 166
Script summary for typical TimeFinder operations 171
Script example for multi-BCV environment 171
BCV pair states 172
Transient BCV pair states 173
BCV actions and applicable states 173
Command options with device groups 174
Command options with composite groups 176
Command options with device files 178
Various remote multihop configurations 180
Commands to various multihop devices and links 180
Second-level controls for multihop SRDF environments 182
Using the -remote option on multihop split actions 183

Appendix A TimeFinder State Rules Reference

TimeFinder SnapVX operations 186
Using the tables for Clone, VP Snap, Snap, and Mirror 187
TimeFinder/Clone operations 189
Basic TimeFinder/Clone operations 189
Concurrent TimeFinder/Clone operations 190
Cascaded TimeFinder/Clone operations 194
VP Snap operations 201
TimeFinder/Snap operations 207
Basic TimeFinder/Snap Operations 207
Concurrent TimeFinder/Snap operations 208
Cascaded TimeFinder/Snap operations 212
TimeFinder/Mirror operations 215
Basic TimeFinder/Mirror operations 215
Concurrent TimeFinder/Mirror operations 216
Cascaded TimeFinder/Clone operations 219

Appendix B SRDF State Rules Reference

SnapVX-SRDF Interaction Rules State Determination 224
SnapVX State Determination Examples 224
SRDF pair states 226
State rules for TimeFinder SnapVX operations 228
TimeFinder SnapVX R1 source 228
TimeFinder SnapVX R2 source 228
TimeFinder SnapVX R1 target 229
TimeFinder SnapVX R2 target 230
State rules for TimeFinder/Clone operations 231
TF/Clone operations R1 source 232
TF/Clone operations R1 target 233
TF/Clone operations R2 source 233
TF/Clone operations R2 target 234
State rules for TimeFinder/Snap operations 235
TimeFinder/Snap operations R1 source 235
TimeFinder/Snap operations R1 target 236

TimeFinder/Snap operations R2 source 236
TimeFinder/Snap operations R2 target 237

Appendix C Rcopy State Rules Reference

TimeFinder SnapVX operations 240
SnapVX source with Rcopy push 240
SnapVX source with Rcopy pull 240
TimeFinder SnapVX target with Rcopy push 241
TimeFinder SnapVX target with Rcopy pull 241
Rcopy push with TimeFinder SnapVX source 242
Rcopy push with TimeFinder SnapVX target 242
Rcopy pull with TimeFinder SnapVX source 243
Rcopy pull with TimeFinder SnapVX target 244
TimeFinder/Snap and TimeFinder/Clone operations 245

FIGURES

	Title	Page
1	Cascading snapshots	33
2	Copying a modified point-in-time back to the source	34
3	Storage group with two snapshots	35
4	Source devices after a restore operation	35
5	Target devices after a link operation	36
6	TimeFinder/Clone copy of a standard device	39
7	TimeFinder/Clone consistent activate using ECA	45
8	Clone from clone target (both sessions are cascaded clone)	52
9	Creating a TimeFinder/Clone from a BCV source	56
10	Creating multiple clone copies from a standard device	58
11	Cloning a copy of a local R1 standard device on a remote array	59
12	Cloning a copy of a local BCV device on a remote array	60
13	Cloning a copy of a local R1 standard device on a Hop-2 array	61
14	Cloning copies of a local R1 standard to a local target and to a remote BCV target	62
15	Cloning copies of a local device to a local BCV target and to a remote BCV target	63
16	Cloning multiple copies on local and remote arrays	64
17	Cloning a copy at the tertiary site of a cascaded SRDF configuration	65
18	Using a composite group when a set of devices spans two arrays	67
19	Copy of a standard device to a virtual device (VDEV)	76
20	TF/Snap consistent activate using ECA	81
21	Incremental restore to a BCV	85
22	Creating a virtual copy from a BCV	91
23	Creating multiple virtual copies from a standard device	93
24	Creating multiple virtual copies from a BCV device	94
25	Using a composite group when a set of devices spans two arrays	97
26	Snapping a copy on a remote array	99
27	Snapping from a remote BCV source device	100
28	Snapping copies on local and remote arrays	101
29	Snapping multiple copies on local and remote arrays	103
30	Snapping a copy at the tertiary site of a cascaded SRDF configuration	104
31	VP Snap sessions sharing allocations within a thin pool	110
32	Establishing a BCV pair	115
33	SRDF: Mirroring the local standard	116
34	Mirror configuration types	117
35	Compounded remote configuration example	124
36	Initial BCV configuration	129
37	Fully establishing a BCV pair	130
38	Establishing a multi-BCV environment	132
39	Canceling a multi-BCV	133
40	Cascaded Clone to a cascaded Clone Emulation session	135
41	Establish concurrent BCV pairs	138
42	Establishing two-way BCV mirrors with protected establish	139
43	Establishing a multi-BCV environment	141
44	Split the BCV pair	143
45	Two-way mirror BCV establish/split normal behavior	146
46	Two-way mirror reverse establish/split behavior	146
47	Practical use of a reverse split	147
48	Splitting concurrent BCV pairs	148
49	Split behavior on two-way BCV mirrors	149
50	ECA consistent split	150

51 Consistent splits on both SRDF sides using ECA 153
52 Full restore of the BCV pair 153
53 Incremental restore the STD 157
54 Restoring a BCV in a multi-BCV environment 158
55 Using a composite group when a set of devices spans two arrays 165
56 Control operations on multihop SRDF configurations 182
57 The -remote option on multihop configurations 184
58 Concurrent copy operation 187
59 Cascaded copy operation 187
60 Cascaded copy example (Snap off Clone) 188
61 Basic TimeFinder/Clone device relationship 189
62 Concurrent TimeFinder/Clone device relationships 190
63 Concurrent TimeFinder/Clone device relationships: Clone and Mirror 191
64 Concurrent TimeFinder/Clone device relationships: Clone and Snap 192
65 Clone off Clone device relationship: A -› B 194
66 Clone off Clone device relationship: B -› C 195
67 Mirror off Clone device relationship 196
68 Clone off Mirror device relationship 198
69 Snap off Clone device relationship 199
70 Basic VP Snap device relationship 201
71 Concurrent VP Snap device relationships 202
72 Concurrent VP Snap: VP Snap with additional TimeFinder/Clone 203
73 Concurrent VP Snap: TimeFinder/Clone with additional VP Snap 204
74 VP Snap off Clone device relationship: A -› B 205
75 VP Snap off Clone device relationship: B -› C 206
76 Basic snap device relationship 207
77 Concurrent TimeFinder/Snap device relationships 208
78 Concurrent TimeFinder/Snap device relationships: Snap and Mirror 209
79 Concurrent TimeFinder/Snap device relationships: Snap and Clone 210
80 Snap off Mirror device relationship 212
81 Snap off Clone device relationship 213
82 Basic TimeFinder/Mirror device relationship 215
83 Concurrent TimeFinder/Mirror device relationships 216
84 Concurrent TimeFinder/Mirror device relationships: Mirror and Clone 217
85 Concurrent TimeFinder/Mirror device relationships: Mirror and Snap 218
86 Clone off Mirror device relationship 219
87 Mirror off Clone device relationship 220
88 Snap off Mirror device relationship 221
89 R1 device as TimeFinder/Clone copy source. 231
90 R1 device as TimeFinder/Clone copy target. 232

TABLES

	Title	Page
1	Number of session slots used per operation	19
2	TimeFinder/Clone commands mapped to SnapVX operations	21
3	TimeFinder command summary	22
4	Using options to verify a clone pair state	51
5	Clone from clone target session states (both sessions are cascaded clone)	54
6	symclone -g control arguments and possible options	69
7	symclone -cg control arguments and possible options	70
8	symclone -file control arguments and possible options	72
9	Using options to verify a snap pair state	89
10	symsnap -g and -cg control arguments and possible options	106
11	symsnap -file control arguments and possible options	107
12	TimeFinder commands mapped to clone operation	118
13	BCV state	133
14	Clone and B to C TimeFinder/Clone Emulation states	136
15	Clone Emulation and clone target session states	137
16	Using options to verify a BCV mirror state	164
17	Using options to verify a BCV pair state	164
18	BCV pair states	172
19	Actions for BCV devices	173
20	BCV control actions and applicable states	173
21	symmir -g control arguments and possible options	174
22	symmir -g view arguments and possible options	175
23	symmir -cg control arguments and possible options	176
24	symmir -cg view arguments and possible options	177
25	symmir -file control arguments and possible options	178
26	symmir -file view arguments and possible options	179
27	Prerequisites and states for SnapVX control operations	186
28	Basic TimeFinder/Clone operations	189
29	Concurrent TimeFinder/Clone operations	190
30	Concurrent TimeFinder/Clone operations: Clone and Mirror	192
31	Concurrent TimeFinder/Clone operations: Clone and Snap	193
32	Clone off Clone operations: A -> B	194
33	Clone off Clone operations: B -> C	196
34	Mirror off Clone operations: Source -> STD	197
35	Clone off Mirror operations: BCV -> Target	199
36	Snap off Clone operations: Source -> Target	200
37	Basic VP Snap operations	201
38	Concurrent VP Snap: VP Snap with additional VP Snap	202
39	Concurrent VP Snap: VP Snap with additional TimeFinder/Clone	203
40	Concurrent VP Snap: TimeFinder/Clone with additional VP Snap	204
41	VP Snap off Clone operations: A -> B	205
42	VP Snap off Clone operations: B -> C	206
43	Basic TimeFinder/Snap operations	207
44	Concurrent TimeFinder/Snap operations	208
45	Concurrent TimeFinder/Snap operations: Snap and Mirror	209
46	Concurrent TimeFinder/Snap operations: Snap and Clone	210
47	Snap off Mirror operations: BCV -> VDEV	213
48	Snap off Clone operations: B -> VDEV	214
49	Basic TimeFinder/Mirror operations	215
50	Concurrent TimeFinder/Mirror operations	217

51 Concurrent TimeFinder/Mirror operations: Mirror and Clone 218
52 Concurrent TimeFinder/Mirror operations: Mirror and Snap 219
53 Clone off Mirror operations: STD -> BCV..... 220
54 Mirror off Clone operations STD -> BCV..... 221
55 Snap off Mirror operations: STD -> BCV 222
56 SnapVX State Determination 224
57 SRDF pair states 226
58 SnapVX control operations allowed on R1 source device by SRDF state 228
59 SnapVX control operations allowed on R2 source device by SRDF state 228
60 SnapVX control operations allowed on R1 target device by SRDF state..... 229
61 SnapVX control operations allowed on R2 target device by SRDF state..... 230
62 TF/Clone operations allowed when R1 is source of clone copy data..... 232
63 TF/Clone operations allowed when R1 is target of clone copy data 233
64 TF/Clone operations allowed when R2 is source of clone copy data..... 233
65 TF/Clone operations allowed when R2 is target of clone copy data 234
66 TimeFinder/Snap operations allowed when R1 is source of the snap..... 235
67 TimeFinder/Snap operations allowed when R1 is target of the snap 236
68 TimeFinder/Snap operations allowed when R2 is source of snap..... 236
69 TimeFinder/Snap operations allowed when R2 is target of the snap 237
70 SnapVX control operations allowed on source device with Rcopy push..... 240
71 SnapVX control operations allowed on source device with Rcopy pull..... 240
72 SnapVX control operations allowed on target device with Rcopy push 241
73 SnapVX control operations allowed on target device with Rcopy pull..... 241
74 Rcopy push operations allowed on SnapVX source control devices 242
75 Rcopy push operations allowed on SnapVX target control devices..... 242
76 Rcopy pull operations allowed on SnapVX source control devices 243
77 Rcopy pull operations allowed on SnapVX target control devices..... 244
78 TimeFinder/Snap and TimeFinder/Clone operations..... 245

PREFACE

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

Contact your EMC representative if a product does not function properly or does not function as described in this document.

Note: This document was accurate at publication time. New versions of this document might be released on the EMC online support website. Check the EMC online support website to ensure that you are using the latest version of this document.

Purpose

This document is part of the EMC Solutions Enabler documentation set, and describes how to use TimeFinder.

Audience

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

Related documentation

- *EMC VMAX3 Family Documentation Set* — Contains the product guide, physical planning guide, and power documentation for the VMAX3 arrays.
- *EMC VMAX Family Viewer for Desktop and iPad®* — Illustrates system hardware and system configurations offered for VMAX and VMAX3 arrays.
- *EMC Solutions Enabler Documentation Set* — Contains all the product guides and installation manuals needed to manage your array using the Solutions Enabler SYMCLI mechanisms.
- *EMC Solutions Enabler Release Notes* — Describes the contents of your kit and how to prepare for an installation. These release notes identify any known functionality restrictions and performance issues that may exist with the current version and your specific storage environment.
- *EMC Unisphere for VMAX Documentation Set* — Explains how to use EMC Unisphere for VMAX for storage system configuration, management, and monitoring.
- *EMC Unisphere for VMAX Release Notes* — Describe the contents of your kit and how to prepare for an installation. These release notes identify any known functionality restrictions and performance issues that may exist with the current version and your specific storage environment.
- *EMC VMAX3 Family HYPERMAX OS Release Notes* — Identifies any known functionality restrictions and performance issues that may exist with the current version and your specific storage environment.

- *SolVe Desktop* —Provides procedures for common tasks and supported SRDF features and applicable limitations for 2-site and 3-site solutions.

  To download the SolVe desktop tool go to EMC Online Support and search for SolVe Desktop. Download the SolVe Desktop and load the VMAX Family and DMX procedure generator.

  Note: You need to authenticate (authorize) your SolVe Desktop. Once installed, please familiarize yourself with the information under Help tab.

- E-Lab™ Interoperability Navigator (ELN) — Provides a web-based interoperability and solution search portal. You can find the ELN at https://elabnavigator.EMC.com.
- Detailed man page descriptions of all SYMCLI commands, environment variables, option file parameters, and error codes are documented in the companion *EMC Solutions Enabler CLI Command Reference*.

ProtectPoint documentation

The following guides provide additional information on EMC ProtectPoint solution:

- *EMC ProtectPoint Implementation Guide*
- *EMC ProtectPoint Solutions Guide*
- *EMC ProtectPoint File System Agent Command Reference*
- *EMC ProtectPoint Release Notes*

Conventions used in this document

EMC uses the following conventions for special notices:

⚠CAUTION

CAUTION, used with the safety alert symbol, indicates a hazardous situation which, if not avoided, could result in minor or moderate injury.

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

IMPORTANT

An important notice contains information essential to software or hardware operation.

Typographical conventions

EMC uses the following type style conventions in this document:

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

Note: To open a service request through EMC Online Support, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or to answer any questions about your account.

Product information

For documentation, release notes, software updates, or for information about EMC products, licensing, and service, go to EMC Online Support (registration required) at:

https://support.EMC.com

Technical support

EMC offers a variety of support options.

Support by Product — EMC offers consolidated, product-specific information on the Web at:

https://support.EMC.com/products

The Support by Product web pages offer quick links to Documentation, White Papers, Advisories (such as frequently used Knowledgebase articles), and Downloads, as well as more dynamic content, such as presentations, discussion, relevant Customer Support Forum entries, and a link to EMC Live Chat.

EMC Live Chat — Open a Chat or instant message session with an EMC Support Engineer.

eLicensing support

To activate your entitlements and obtain your license files, visit the Service Center on http://support.EMC.com, as directed on your License Authorization Code (LAC) letter e-mailed to you.

For help with missing or incorrect entitlements after activation (that is, expected functionality remains unavailable because it is not licensed), contact your EMC Account Representative or Authorized Reseller.

For help with any errors applying license files through Solutions Enabler, contact the EMC Customer Support Center.

If you are missing a LAC letter, or require further instructions on activating your licenses through EMC Online Support, contact EMC's worldwide Licensing team at licensing@emc.com or call:

- North America, Latin America, APJK, Australia, New Zealand: SVC4EMC (800-782-4362) and follow the voice prompts.
- EMEA: +353 (0) 21 4879862 and follow the voice prompts.

Your comments

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

VMAXContentFeedback@emc.com

CHAPTER 1
Introduction to the TimeFinder CLI

This chapter introduces the SYMCLI commands for the EMC Solutions Enabler TimeFinder components.

- Introduction to TimeFinder 18
- TimeFinder CLI overview 20
- Command summary 22

Introduction to TimeFinder

The TimeFinder Family of products are local replication solutions, designed to non-disruptively create point-in-time copies (snapshots) of critical data. TimeFinder provides a business continuance solution for customers requiring constant uptime for business applications. TimeFinder is used to initiate replicas, offload backups, and refresh test and development environments from any host running EMC Symmetrix Solutions Enabler (SE) software.

TimeFinder SnapVX (HYPERMAX OS)

In VMAX3 arrays running HYPERMAX OS, TimeFinder SnapVX lets you non-disruptively create point-in-time copies (snapshots) of critical data. SnapVX creates snapshots by storing changed tracks (deltas) directly in the Storage Resource Pool (SRP) of the source device.

With SnapVX, you do not need to specify a target device and source/target pairs when you create a snapshot. If there is ever a need for the application to use the point-in-time data, you can create links from the snapshot to one or more target devices. If there are multiple snapshots and the application needs to find a particular point-in-time copy for host access, you can link and relink until the correct snapshot is located.

SnapVX supports TimeFinder/Clone, TimeFinder VP Snap, and TimeFinder/Mirror via emulations that convert commands to SnapVX commands. This conversion happens within the HYPERMAX OS and there is no visible change to the end user. You can still run scripts that use TimeFinder/Clone, TimeFinder VP Snap, and TimeFinder/Mirror commands; they operate in the same way, although the underlying mechanism is SnapVX.

Note: TimeFinder Emulation sessions and SnapVX snapshots cannot coexist on the same device.

For a high-level overview of TimeFinder SnapVX functionality, refer to the *EMC VMAX3 Family with HYPERMAX OS Product Guide*.

EMC ProtectPoint and TimeFinder SnapVX Technology

EMC ProtectPoint integrates primary storage on VMAX3 arrays running HYPERMAX OS with protection storage for backups on EMC Data Domain systems. The primary storage sends changed blocks directly to Data Domain, which uses the changed blocks to create full backups in native format.

This is accomplished using Federated Tiered Storage (FTS) and the targetless snapshot technology - SnapVX - to encapsulate Data Domain devices and copy data to/from these devices. In this configuration, backups are archived in permanent storage within the Data Domain unit.

Encapsulated Data Domain devices are fully recognized by VMAX3 arrays running the HYPERMAX OS; appear in `symsnapvx` device displays; and can be used as a filter when listing devices.

- For a description of EMC ProtectPoint backup and restore operations with VMAX3, refer to the *EMC VMAX3 Family with HYPERMAX OS Product Guide.*

- The EMC ProtectPoint documentation set contains all of the instructions for setting up, configuring, and using the integrated ProtectPoint/VMAX3 solution.

TimeFinder Enginuity (5773 - 5876)

In VMAX arrays running Enginuity 5773 - 5876, the TimeFinder family of replication products allow you to nondisruptively create and manage point-in-time copies of data, enabling simultaneous action of business tasks that were previously sequential. For example, TimeFinder allows you to create a point-in-time copy of critical data while this data continues to be used in production operations.

The ability to access source data during the TimeFinder copy operation eliminates the backup window and provides benefits such as accelerated upgrades and high availability. TimeFinder can also shorten the maintenance window, minimize infrastructure costs, and improve service levels.

For a comprehensive description of TimeFinder replication concepts and devices, refer to the *EMC Symmetrix TimeFinder Product Guide*.

Copy Session Limits

In VMAX3 arrays running HYPERMAX OS, SnapVX supports up to 256 snapshots per source device (including any emulation mode snapshots). The legacy session limits still apply to the emulations.

VMAX family arrays running Enginuity 5773 - 5876 support up to 16 sessions per source device, which can be used for TimeFinder/Clone, TimeFinder/Snap, TimeFinder VP Snap, SRDF/Star, Solutions Enabler Open Replicator (ORS), or Symmetrix Differential Data Facility (SDDF) operations. This limits the number of available copies that can be created.

Note: TimeFinder VP Snap allows an additional 32 sessions per device.

Table 1 lists the number of copy sessions required for each operation.

Note: Not all VMAX family arrays support all of these technologies. In addition, the number of supported copy sessions per source device may vary on certain platforms. Refer to your product guide for your specific VMAX series for details about supported features.

Table 1 Number of session slots used per operation (page 1 of 2)

Operation	Session slots
TimeFinder SnapVX	• Supports 256 snapshots per source device (including any emulation mode snapshots). • Supports up to 1,024 linked targets per source device.
TimeFinder/Snap	• One session slot per snap session, plus an additional session slot for restore operations. • For arrays running Enginuity 5875 and higher: One session slot per snap session, plus two additional session slots for restore operations.
Multivirtual snap	• One session slot (regardless of the number of sessions), plus an additional session slot reserved for restore operations. • For arrays running Enginuity 5875 and higher: One session slot per session, plus two additional session slots for restore operations.

Table 1 Number of session slots used per operation (page 2 of 2)

Operation	Session slots
TimeFinder/Clone	• Two session slots per copy session, unless using the `-nodifferential` option, in which case one session slot is used per copy session.
TimeFinder/Clone Emulation mode	• Two session slots.
ORS	• One session slot per ORS session, and one session slot for each additional session unless the session was created with the `-nodifferential` option.
SRDF/Star	• Two session slots.
SRDF/A	• One session slot.
SDDF	• One session slot per SDDF session

TimeFinder CLI overview

This section provides an overview of the TimeFinder Command Line Interface (CLI).

VMAX3 arrays (HYPERMAX OS)

The symsnapvx operations are: `establish`, `link`, `list`, `relink`, `rename`, `restore`, `set mode`, `set ttl`, `terminate`, `unlink`, and `verify`.

Chapter 2 describes the `symsnapvx` command and the SnapVX operations in greater detail.

Note: On VMAX3 arrays, SnapVX supports TimeFinder/Clone, TimeFinder VP Snap, and TimeFinder/Mirror using emulations that convert the respective commands to SnapVX commands. For more information, see "TimeFinder emulations (HYPERMAX OS)" on page 21.

VMAX arrays (Enginuity 5773 - 5876)

In VMAX arrays running Enginuity 5773 - 5876, TimeFinder commands include: `symclone`, `symsnap`, `symbcv`, and `symmir`.

- The TimeFinder/Clone `symclone` command creates a point-in-time copy.

  The TimeFinder/Clone operations are `create`, `activate`, `recreate` and `activate` (or `establish` that combines the two procedures into one operation), `restore`, and `terminate`. Chapter 3 describes the `symclone` command and the TimeFinder/Clone operations in greater detail.

- The TimeFinder/Snap `symsnap` command creates virtual device copy sessions between a source device and multiple *virtual* (VDEV) target devices. These virtual devices only store pointers to changed data blocks from the source device, rather than a full copy of the data.

  The TimeFinder/Snap operations are `create`, `activate`, `recreate`, `restore` and `terminate`. Chapter 4 describes the `symsnap` command and the TimeFinder/Snap operations in greater detail.

- TimeFinder VP Snap utilizes the `symclone` command to create space-efficient snaps for thin devices. VP Snap provides the efficiency of Snap technology with improved cache utilization and simplified pool management. Chapter 5 describes how to use the `symclone` command to perform VP Snap operations.

TimeFinder emulations (HYPERMAX OS)

TimeFinder automatically maps TimeFinder/Clone, TimeFinder VP Snap, and TimeFinder/Mirror commands to the executable of the appropriate SnapVX command.

Table 2 describes the mapping of TimeFinder/Clone operations to their SnapVX operational equivalents. (VP Snap is included in this table because it symclone CLI with the `-vse` flag.)

In some cases, there is no exact equivalent because SnapVX snapshots are targetless. The plus sign (+) indicates that the command is followed by the next command listed in that table cell. The letters A and B indicate devices, and the arrow symbols indicate data direction.

Table 2 TimeFinder/Clone commands mapped to SnapVX operations

symclone command	symsnapvx command
symclone establish -full A -> B	symsnapvx establish A + symsnapvx link -copy A -> B
symclone establish A -> B	symsnapvx establish A + symsnapvx link -copy A -> B
symclone create A -> B + symclone activate A -> B	symsnapvx establish A + symsnapvx link -copy A -> B
symclone restore A <- B	symsnapvx restore A

TimeFinder/Mirror commands are first converted to TimeFinder/Clone using legacy Clone Emulation, and the TimeFinder/Clone commands are then converted to SnapVX as described in Table 2.

Command summary

Note: The *EMC Solutions Enabler CLI Command Reference* provides complete details about the syntactical form of the commands.

Table 3 lists TimeFinder commands, the VMAX array in which the command is supported, and a description of the tasks performed by each command.

Table 3 TimeFinder command summary (page 1 of 2)

Command	Description
`symsnapvx`[1]	• Creates and activates snapshots. • Name or rename snapshots. • Sets snapshots to automatically terminate in a specified number of days. • Creates links from snapshot data to host mapped target devices. • Automatically unlinks existing links and relinks to a different snapshot. • Unlinks a snapshot without relinking it to a different snapshot. • Removes the snapshot from the system. • Restores snapshots by copying point-in-time data from a snapshot back to the original source device. • Verifies whether one or more devices are in the requested state.
`symclone`[2,4]	Performs TimeFinder/Clone control operations on standard or BCV devices: • Creates a copy session for making multiple data copies between a source device and target devices. • Creates and activates a copy session • Modifies the mode in which a copy session is operating. • Activates a copy session to make data instantly accessible to multiple target hosts. • Copies (incrementally) all subsequent changes made to a source device to a target device, after a clone session is fully copied. • Restores data from a target device back to a source device or to another device. • Terminates a copy session to remove holds on target devices and delete device pair information from the array. • Queries information about the state of mirroring for multiple copy sessions. • Verifies the state for selected devices. • Lists all copy sessions that have been created on the array. Performs TimeFinder VP Snap control operations: • Creates space-efficient snaps for thin devices • Incrementally restores data back to the original source device without requiring that the source device is fully copied

Table 3 TimeFinder command summary (page 2 of 2)

Command	Description
`symsnap`[3]	Performs Snap control operations for virtual copy sessions from normal devices to virtual devices. The source device can be either a standard or a BCV device and the target device must be a virtual device (VDEV). • Creates a virtual copy session for making multiple data copies between a source device and up to 15 target devices - HYPERMAX OS supports up to 15 target devices - Enginuity versions 5875 - 5876 support up to 14 target devices. - Multivirtual snap supports up to 128 target devices. • Specifies a particular SAVE pool for use in a virtual copy session. • Activates a virtual copy session to make data instantly accessible to multiple target hosts. • Recreates a snap session on existing VDEVs to prepare to active a new point-in-time image and is only valid when issued against previously activated sessions. • Terminates a virtual copy session to remove holds on target devices and delete device pairing information from the array. • Queries information about the state of mirroring for multiple copy sessions. • Verifies device states. • Attaches and detaches target devices as the preferred devices to use in a requested Snap operation. • Restores a virtual device to another device, or to the original device. • Duplicates a point-in-time copy of a virtual device, which is paired in a previously activated snap session, to another virtual device. • Lists all virtual copy sessions that have been created on the array.
`symbcv`[2,4]	Performs operations on one or more BCV devices: • Associates a device pair. • Disassociates a device pair. • Lists all BCV devices in the array. • Moves a BCV device from one group to another. • Removes all BCV devices from the specified device group.
`symmir`[2,4]	Performs control operations on BCV device pairs including: • Establishes (mirror) one or all standard devices with one or more BCV devices. The operation can be a full or incremental establish. • Restores one or all standard devices from one or more BCV devices that are associated locally or remotely. The operation can be a full or incremental restore. • Splits one or all BCV devices from one or more standard devices. • Returns information about the state of mirroring of one or all BCV device pairs. • Cancels the existing internal SDDF session between the specified standard and BCV devices. • Lists all BCV sessions created on an array.

1. Supported only in VMAX3 arrays with HYPERMAX OS.
2. Supported in VMAX3 arrays with HYPERMAX OS in emulation mode.
3. Supported only in VMAX arrays with Enginuity 5773 - 5876.
4. Supported in VMAX arrays with Enginuity 5773 - 5876.

CHAPTER 2
Performing TimeFinder SnapVX Operations (HYPERMAX OS)

This chapter describes how to perform Timefinder SnapVX operations on VMAX3 arrays running HYPERMAX OS.

- TimeFinder SnapVX Overview 26
- Restoring a snapshot 28
- Linking to a snapshot 29
- Displaying snapshot and link summaries 31
- Terminating a snapshot 32
- Cascading snapshots 33
- Modified point-in-time copies 34
- SnapVX behavior when the number of source devices changes 35

TimeFinder SnapVX Overview

SnapVX operations are performed using the `symsnap` command to create point-in-time copies (snapshots) of critical data. SnapVX creates snapshots by storing changed tracks (deltas) directly in the Storage Resource Pool (SRP) of the source device.

In VMAX3 arrays running HYPERMAX OS, SnapVX supports up to 256 snapshots per source device (including any emulation mode snapshots). The legacy session limits still apply to the emulations.

For a high-level overview of TimeFinder SnapVX functionality, refer to the *EMC VMAX3 Family with HYPERMAX OS Product Guide*.

SnapVX supported configurations

SnapVX is supported in the following configurations:

- On a local VMAX3 array
- On a remote VMAX3 array
- On a VMAX3 array two hops away

Local operations will act upon the selected device itself, remote operations will act on the device's remote partner, and two hop operations will act on the device's remote-remote partner.

SnapVX operating modes

SnapVX supports the following modes of operation:

- Device list
- Device ranges
- DG group mode[1]
- CG group mode[1]
- SG group mode[2]

1. DG/CG group modes use STD devices as sources and TGT devices as link targets.
2. SG group mode will only select device which are not considered to be gatekeepers.

Display examples throughout this chapter show how to execute SYMCLI commands using each mode of operation.

For further details, the man page descriptions of all SYMCLI commands, environment variables, option file parameters, and error codes are documented in the companion *EMC Solutions Enabler CLI Command Reference*.

Establishing a snapshot

The `symsnapvx establish` command creates and activates a SnapVX snapshot with the name that you supply.

Note: Snapshot names are case sensitive and can be up to 32 alphanumeric characters long. They can include the special characters dash (-) and underscore (_).

Example To establish a SnapVX snapshot named 600am for the source devices specified in Storage Group1, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -name 600am establish
```

Every time a new snapshot with the same name is created (on the same source device), the new snapshot becomes generation 0, and all of the previous generation numbers are incremented.

SnapVX supports up to 256 snapshots per source device.

Consistent snapshots

By default, all SnapVX snapshots are consistent with the database when created and activated using the `symsnapvx establish` command. Depending on the state of the devices at the time of the snapshot, SnapVX pauses I/O to ensure there are no writes to the source device while the snapshot is created.

When the activation completes, writes are resumed and the target device contains a consistent production database copy of the source device at the time of activation.

Source device requirements

To ensure consistent snapshots, source devices must meet the following criteria:

- All devices must be R1 or R2; a mixture of R1 and R2 devices is not supported.
- A mixture of R1 and non-SRDF devices is allowed. The R1 devices can be a mixture of RW and NR on the SRDF link.
- A mixture of R2 and non-RDF devices is allowed if the R2 devices are NR on the SRDF link.
- A mixture of R1 and R21 devices is allowed if the R1->R21 is NR on the SRDF link.
- A mixture of R2 and R21 devices is allowed, if the following criteria is met:
  - R1->R2 and R1->R21 devices are all RW on the SRDF link.
  - R1->R2 and R1->R21 are all NR on the SRDF link.

If the source devices do not meet these criteria then the snapshot cannot be created unless the `-force` flag is used.

Note: Using the `-force` flag forces the snapshot to be created; however, the consistency of the snapshot cannot be guaranteed.

Time to live

You can set snapshots to automatically terminate after a specified number of days or at a specified date and time. HYPERMAX OS will only terminate the snapshot if does not have any links. If it does have links, HYPERMAX OS will terminate the snapshot when the last link has been unlinked.

Example To establish a SnapVX snapshot named 600am that will automatically terminate after two days, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -name 600am establish -ttl
-delta 2
```

Restoring a snapshot

SnapVX allows you to restore the snapshot's point-in-time data back to the original source device. SnapVX restore operations are inherently differential. In other words, only the tracks that have changed need to be copied back to the source device.

Note: You need to terminate a restore session before terminating the original snapshot session.

Example To restore generation 5 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -snapshot_name 600am -gen 5
restore
```

If the number of source devices in a storage group changes, a snapshot (or a generation of a snapshot) might not exist for that device for a particular point-in-time. If you restore to a source device that does not have the specified snapshot, SnapVX makes the source device not ready. This is because there was not any data on that source at that time, and the application should not be using that device. "SnapVX behavior when the number of source devices changes" on page 35 provides additional details.

Linking to a snapshot

To access a point-in time-copy, you must create a link from the snapshot data to a host mapped target device. You can create links in Copy mode for a permanent copy on the target device, or in NoCopy mode for temporary use. Copy mode links create full-volume copies of the data by copying it to the target device's SRP. NoCopy mode links are space-saving snapshots that only consume space for the changed data that is stored in the source device's SRP. NoCopy mode links will not retain point-in-time data once the link is removed.

If you do not specify Copy mode when you link the snapshot, HYPERMAX OS creates a NoCopy link. After you have created a link, you can change the copy mode by using the `set mode` command.

SnapVX supports up to 1,024 linked targets per source device.

The following restrictions apply when linking to a target:

- You can only link to devices that are not the target of another link copy or another TimeFinder session.
- Devices with persistent track allocations cannot be used as link targets.
- Any pre-existing data that was exclusive to the target will be lost during a link or relink.
- The target device must be of equal or greater size than the source device.

Note: By default, SnapVX allows the target device to be larger than the source device. If necessary, you can disable this setting. The *EMC Solutions Enabler CLI Command Reference* contains information on changing the Option file parameters.

Example To create a Copy mode link to generation 5 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -lnsg StorageGroup2
-snapshot_name 600am -gen 5 link -copy
```

If the number of source devices in a storage group changes, a snapshot (or a generation of a snapshot) might not exist for that device for a particular point-in-time. If you link from a source device that does not have the specified snapshot, SnapVX makes the corresponding target device not ready. This is because there is no data on that target for that time, and the application should not be using that device. "SnapVX behavior when the number of source devices changes" on page 35 provides additional details.

Note: If your environment supports EMC ProtectPoint Data Domain systems, before ProtectPoint can be used to backup the production database, the TimeFinder SnapVX link copy command must be run between the production and the backup devices. Full instructions are contained in the EMC ProtectPoint documentation set.

Relinking

SnapVX allows you to automatically unlink an existing link and relink to a different snapshot.

You can only use the relink command if a link already exists on at least one of the targets.

If the relink is in Copy mode, the copy will be differential between the original linked snapshot and the newly linked snapshot.

The relink command may also be used to relink to the same snapshot. This has the effect of refreshing the point-in-time copy on the link target if it has been modified by host writes.

Example To relink (in Copy mode) to generation 6 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -lnsg StorageGroup2
-snapshot_name 600am -gen 6 relink -copy
```

Unlinking

The unlink command allows you to unlink a snapshot without relinking to a different snapshot.

Example To unlink generation 6 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -lnsg StorageGroup2
-snapshot_name 600am -gen 6 unlink
```

Displaying link status

The list command allows you to display the state of all snapshots and links on individual devices or on the entire VMAX3 array. To display link status for a group, devices must have previously been associated with the group.

You can display two types of link status:

- **Basic Link Display** (`symsnapvx list -sid`) - This link display is source oriented, showing only those snapshots that have linked targets. Lets you list all snapshots for a device, a device file, device ranges, device group, composite group, storage group, or for the entire VMAX3 array.
- **Target Oriented Link Display** (`symsnapvx list -linked - tgt)` - Lets you display the state of all snapshots and links on individual devices, a device file, device ranges, device group, composite group, or storage group oriented and ordered by link target device.

When using the `symsnapvx` list command in environments with ProtectPoint Data Domain systems, if a link is in a force failed state due to non-supported configuration, the Failed column contains an **"F" as shown in the examples below.**

Example To display a list a list of snapshots that contain only linked targets:

```
symsnapvx list -f snapvx_devices -linked
```

The resulting display shows only those snapshots that have linked targets.

```
Device File Name          : snapvx_devices
Device's Symmetrix ID     : 000197100001    (Microcode Version: 5977)

--------------------------------------------------------------------------------
Sym                                          Link  Flgs
Dev   Snapshot Name                     Gen  Dev   FCMD Snapshot Timestamp
----- ------------------------------- ---- ----- ---- ------------------------
00123 Tuesday_account_400pm                1 01122 .C.X Tue Feb 12 16:00:15 2013
      Tuesday_account_400pm                1 01126 .D.X Tue Feb 12 16:00:15 2013
00124 Tuesday_account_400pm                1 01123 .IX. Tue Feb 12 16:00:15 2013
00125 Tuesday_account_400pm                1 01124 FI.. Tue Feb 12 16:00:15 2013

Flgs:
  (F)ailed  :  F = Force Failed, X = Failed,. = No Failure
  (C)opy    :  I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odifed :  X = Modified Target Data, . = Not Modified
  (D)efined :  X = All Tracks Defined, . = Define in progress
```

Example To display a list a list of snapshots that are ordered by the link target device:

```
symsnapvx list -g snapvx_dg -linked -by_tgt
```

The resulting display shows only those snapshots with linked targets, listed by link target device.

```
Device Group (DG) Name    : snapvx_dg
DG's Type                 : REGULAR
DG's Symmetrix ID         : 000197100001    (Microcode Version: 5977)

--------------------------------------------------------------------------------
Link  Sym                                          Flgs
Dev   Dev   Snapshot Name                     Gen  FCMD Snapshot Timestamp
----- ----- ------------------------------- ---- ---- ------------------------
01122 00123 Tuesday_account_400pm               1 .C.X Tue Feb 12 16:00:15 2013
01123 00124 Tuesday_account_400pm               1 .IX. Tue Feb 12 16:00:15 2013
01124 00125 Tuesday_account_400pm               1 FI.. Tue Feb 12 16:00:15 2013
01126 00123 Tuesday_account_400pm               1 .D.X Tue Feb 12 16:00:15 2013

Flgs:
  (F)ailed  :  F = Force Failed, X = Failed,. = No Failure
  (C)opy    :  I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odifed :  X = Modified Target Data, . = Not Modified
  (D)efined :  X = All Tracks Defined, . = Define in progress
```

Displaying snapshot and link summaries

The `symsnapvx -summary` is used with `symsnapvx list` or `symsnapvx verify` to view a summary of the state of selected devices. When used in conjunction with the –i or –c switches, the remaining tracks and estimated time to completion are displayed.

The total remaining time may also be displayed as megabytes, gigabytes, or terabytes with the use of the optional –mb, -gb, and –tb switches.

Example To display a summary of the state of selected file devices:

```
symsnapvx list -f snapvx_devices -summary
```

```
Device File Name          : snapvx_devices
Device's Symmetrix ID     : 000197100001    (Microcode Version: 5977)

  Snapshot State               Count
  -----------------------      ------
  Restored                          1
  RestInProg                        0
  Established                       3
  EstablishInProg                   0
  Linked                            2
  CopyInProg                        0
  Copied                            0
  NoSnapshot                        0
  Failed                            0
  -----------------------      ------
  Total                             6

                              Track(s)
                              -----------
Total Remaining                    132000
```

Example To display a summary of the state of selected storage groups that includes the total remaining storage space in terabytes:

```
symsnapvx list -sg snapvx_source -summary -tb
```

```
Storage Group (SG) Name   : snapvx_source
SG's Symmetrix ID         : 000197100001    (Microcode Version: 5977)

  Snapshot State               Count
  -----------------------      ------
  Restored                          1
  RestInProg                        0
  Establihed                        3
  EstablishInProg                   0
  Linked                            2
  CopyInProg                        0
  Copied                            0
  NoSnapshot                        0
  Failed                            0
  -----------------------      ------
  Total                             6

                              TB(s)
                              -------
Total Remaining                  2.01
```

Terminating a snapshot

Terminating a snapshot removes the it from the system. Note that you cannot terminate a snapshot if it has any links or restore sessions. Linked devices should be unlinked, and then the snapshot can be terminated. Likewise, a restore session must first be terminated, and then the original snapshot can be terminated.

Example To terminate generation 5 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -snapshot_name 600am -gen5
terminate
```

Example To terminate the restore of generation 5 of the 6:00 AM snapshot, enter:

```
symsnapvx -sid 001 -sg StorageGroup1 -snapshot_name 600am -gen 5
terminate -restored
```

Cascading snapshots

The target device of a snapshot can be used as the source device for additional snapshots.

Figure 1 shows an example of a cascading snapshot. In this example, Snapshot B is a cascading snapshot off of the target of Snapshot A. In other words, the target of Snapshot A is the source for Snapshot B.

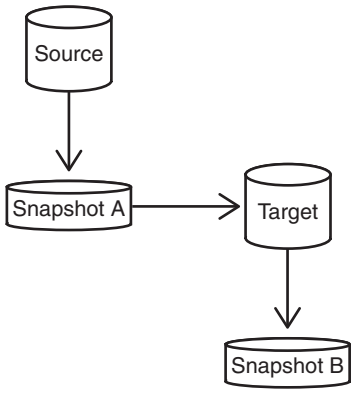

Figure 1 Cascading snapshots

There are no architectural restrictions on the number of cascading hops, and you can link to multiple targets from any of the snapshots.

If the linked target is in NoCopy mode, the cascading snapshot is considered a dependent snapshot. Target devices cannot be unlinked without removing dependent snapshots. In other words, the snapshots must be removed in the reverse of the order of their creation. However, if you change the copy mode to Copy, you can unlink the target once the link is fully copied.

A relink operation to a target with dependent snapshot is allowed, but if the relink is in NoCopy mode, the unlink is allowed only if there are no new snapshots off of that target device.

A linked target must be fully defined before you can create snapshots off of it. When a target is first linked, all of the tracks are undefined. This means that the target does not know where in the SRP the track is located, and host access to the target must be derived from the SnapVX metadata. A background process eventually defines the tracks and updates the thin device to point directly to the track location in the source device's SRP.

Example To determine if a linked target is fully defined, enter:

```
symsnapvx list -sid 001 -sg StorageGroup1 -linked
```

The resulting display shows only those snapshots that have fully defined linked targets. The (D) flag shows whether not all of the tracks for a linked target are defined.

```
Device File Name          : snapvx_devices
Device's Symmetrix ID     : 000197100001    (Microcode Version: 5977)

--------------------------------------------------------------------------------
Sym                                          Link  Flgs
Dev   Snapshot Name                    Gen   Dev   FCMD Snapshot Timestamp
----- -------------------------------- ---- ----- ---- ------------------------
00123 Tuesday_account_400pm               1 01122 .C.X Tue Feb 12 16:00:15 2013
00123 Tuesday_account_400pm               1 01126 .D.X Tue Feb 12 16:00:15 2013
00124 Tuesday_account_400pm               1 01123 .IX. Tue Feb 12 16:00:15 2013
00125 Tuesday_account_400pm               1 01124 .I.. Tue Feb 12 16:00:15 2013

Flgs:
  (F)ailed  :  X = Failed,. = No Failure
  (C)opy    :  I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odifed :  X = Modified Target Data, . = Not Modified
  (D)efined :  X = All Tracks Defined, . = Define in progress
```

Modified point-in-time copies

Host writes to a linked target device will not change the point-in-time copy, so only the linked target device will contain a modified point-in-time copy. To copy a modified point-in-time copy back the original source device, you must establish a snapshot from the target device and then create a Copy mode link of that snapshot to the source device.

Figure 2 shows an example of copying a modified point-in-time copy back to the source by linking (in Copy mode) a snapshot of the linked target back to the source device.

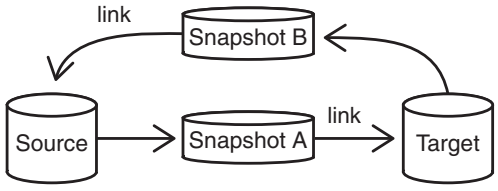

Figure 2 Copying a modified point-in-time back to the source

SnapVX behavior when the number of source devices changes

Over time, an application may grow (or shrink) the number of devices that it uses. SnapVX automatically adjusts restore, link, and relink operations to accommodate extra or missing snapshot names and generations.

Figure 3 shows a storage group with three source devices. Source device 3 was added to the storage group after the first snapshot (Mon 12pm) was established, so that snapshot does not exist for that device.

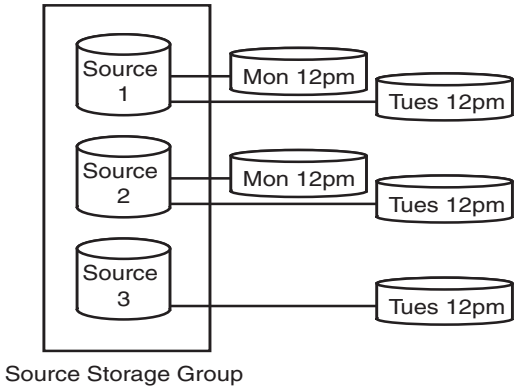

Figure 3 Storage group with two snapshots

Restore operations

If you restore the Mon 12pm snapshot, the point-in-time data will be copied back to source device 1 and source device 2, but source device 3 will be marked NR because there is no data on that device for that point-in-time. Figure 4 shows the state of the source devices after the restore operation.

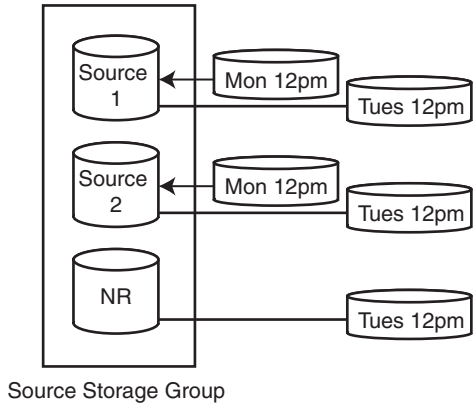

Figure 4 Source devices after a restore operation

Link operations

If you link target devices to the Mon 12pm snapshot, target device 1 and target device 2 will be linked to the snapshot, but target device 3 will be marked NR because there is no data on that target for that point-in-time. Figure 5 shows the state of the target devices after the link operation.

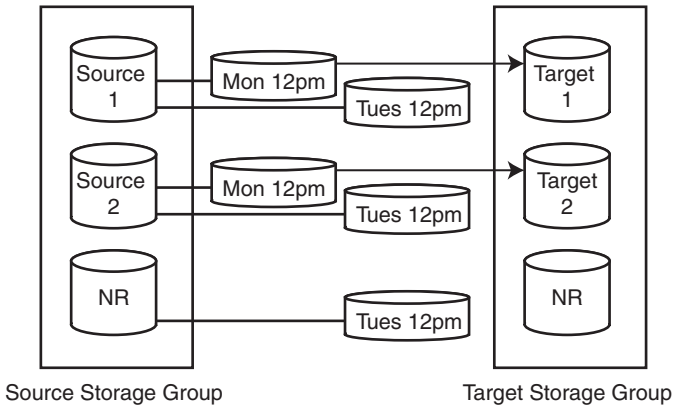

Figure 5 Target devices after a link operation

Note: You can only use the link command when there are no links to any of the targets. In Copy mode, the result will be a full copy.

Relink operations

You can only use the relink command if a link exists on at least one of the targets. In Copy mode, if a link already exists for a particular target, the result will be an incremental copy. If there is no existing link for a target, then the result will be a full copy.

If you relink from a source device that has the selected snapshot and generation, but the target device does not have an existing link, a new link will be created and the result will be a full copy.

If you relink from a source device that does not have the selected snapshot and generation, the target device will be marked NR. If the target has an existing link, it will be unlinked.

If you relink from a source device that has the selected snapshot and generation, and the target device has a previous link, the relink will result in an incremental copy.

CHAPTER 3
Performing TimeFinder/Clone Operations

This chapter describes how to perform TimeFinder/Clone operations using the SYMCLI `symclone` command.

- TimeFinder/Clone overview 38
- Creating a clone copy session 38
- Activating a clone copy session 43
- Modifying a clone copy session 46
- Recreating a clone copy device 46
- Restoring data from a target device 47
- Splitting a clone device pair 48
- Terminating a clone copy session 48
- Querying clone pairs 49
- Verifying clone pair states 50
- Cascaded clone 52
- Using a BCV as the clone source 55
- Creating multiple clone copies from a standard device 58
- Cloning a copy on a remote array 58
- Cloning copies of the same data locally and remotely 62
- Cloning multiple copies locally and remotely 64
- Cloning a copy at the tertiary site of a cascaded SRDF configuration 65
- Using composite groups to manage clone pairs across arrays 67
- Command options with device groups 69
- Command options with composite groups 70
- Command options with device files 72

TimeFinder/Clone overview

For a high-level overview of TimeFinder/Clone functionality, refer to the *EMC Symmetrix TimeFinder Product Guide*.

TimeFinder/Clone operations are performed using the `symclone` command to create clone copies of a source device on one or multiple target devices.

A single source device can have up to 16 clone copy sessions. You can copy up to eight full data copies simultaneously, without disruption to database production activity. You can create up to eight more copies once the first eight are completed. "Copy Session Limits" on page 19 provides additional details about the number of copies that you can create.

Note: Data Domain devices are not supported as TimeFinder/Clone source or target devices.

Creating a clone copy session

Initially, you must create a clone copy session that defines and sets up the cloning devices you have selected for the clone operation.

To begin a clone copy session and define a specified target device `DEV005` to be the clone of source device `DEV001` in group `ProdDB`:

```
symclone -g ProdDB create DEV001 sym ld DEV005
```

You can specify `-nocopy` and `-nodifferential` flags on the `symclone create` command:

```
symclone -g ProdDB create DEV001 sym ld DEV005 -nocopy
```

- The `-nocopy` flag creates a session without a background copy. The `-nocopy` flag is not allowed with the `-differential` flag.
- The `-nodifferential` flag creates a nondifferential session, which cannot be recreated or restored.

The `symclone create` action defines the clone copy session requirements and sets the track protection bitmap on the source device to detect which tracks are being accessed by the target host or written to by the source host, as shown in Figure 6.

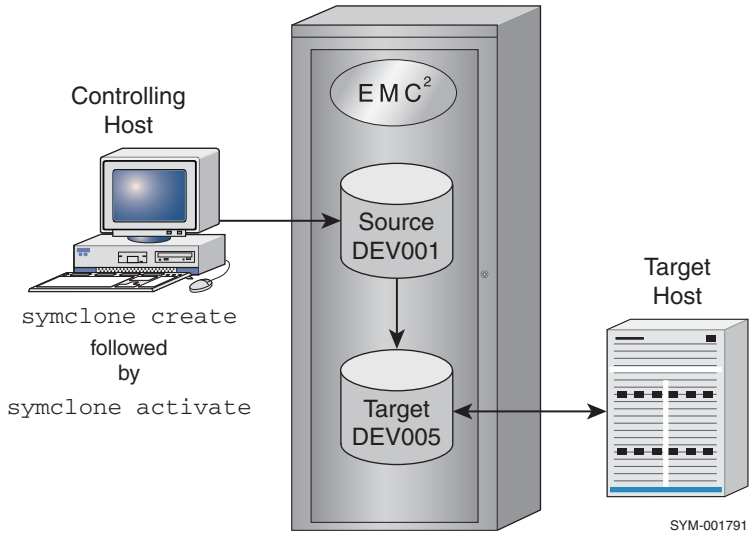

Figure 6 TimeFinder/Clone copy of a standard device

- The target device is made Not Ready to its host and placed on hold status for clone copy session activity. This prevents other control operations from using the device.
- The device pair state will transition from CreateInProgress to Created when complete.
- The clone copy does not become accessible to its host until the copy session is activated. "Activating a clone copy session" on page 43 contains greater detail.

Note: You cannot verify that a device pair is in the CreateInProgress state. After the copy session completes, you can issue a `symclone verify -created` command to verify that the clone pair was successfully created. If a copy session is created and not activated, it can be terminated. Although, the data on the target device should then be considered invalid.

Fully copying to a clone

Solutions Enabler specifies the `-copy` option by default. When the copy session is activated, data begins background copying so that a full copy of the data will become available on the target device. While background copying, the state of the device pair is CopyInProgress; when the operation completes, the state goes to Copied. The copy session must be activated before the target host can access the data. However, once the session is activated, the data is available immediately to the target host.

You can use the `SYMCLI_CLONE_COPY_MODE` environment variable to change the default copy mode. The possible values for this variable are `NOCOPY_NODIFF`, `COPY_NODIFF`, `COPY_DIFF`, `PRECOPY_DIFF`, and `VSE_NODIFF`. These copy modes are described in detail in the following sections.

To see a list of sessions that are using a particular copy mode, combine the list command with any of the copy options (`-copy`, `-nocopy`, `-precopy`, or `-vse`).

Example To see a list of sessions with the background copy active, enter:

```
symclone list -copy
```

Creating a session in nocopy mode

To create a copy session without a full background copy, you must use the `-nocopy` option. When activating a copy session in nocopy mode, the default device pair state is CopyOnAccess. This means that after activating the copy session, only those tracks that have been written to the source or written/read from the target will be copied to the target device. A full data copy to the target device will not occur unless all of the device tracks are accessed or written to while participating in the active session.

Note: If a write occurs to the source device, old data is copied to the target device. If a write occurs to the target device, new data is written to the target device.

When Enginuity detects that a source-protected track was written, it copies the track to the target device and unprotects the track before accepting the new write. Data from the source then becomes available to a target-connected host during the active session.

You can modify this default device pair state to CopyOnWrite by setting the following parameter in the `options` file to `ENABLE`.

```
SYMAPI_CLONE_COPY_ON_WRITE = ENABLE | DISABLE
```

You can also modify the default device pair state by setting the following SYMCLI environment variable:

```
SYMCLI_CLONE_COPY_ON_WRITE = ENABLE | DISABLE
```

The environment variable will override the default settings specified in the options file.

Once you have enabled CopyOnWrite as the default pair state and activated a copy session, all reads will be handled from the source device and writes to the source device or target device during the active copy session will result in the data being copied to a target device.

Starting to copy data before activating the session

You can use the `-precopy` option with the `create` argument to start copying tracks in the background, before activating the copy session. This allows the early movement of data before the point-in-time clone copy is established.

Note: When using `-precopy`, the target device is not ready to the host until the session is activated.

Precopy can occur in the Created or Recreated state. While in this state, the precopy process keeps checking for new writes to be precopied to the target device until the copy session is activated. Once activated, the normal background copy mechanism starts and the precopy operation ends. You can set the precopy mode in one of two ways:

- Include the `-precopy` option with the `symclone create` command. Using `-precopy` will result in a full copy. For example:

```
symclone -g ProdDB create -precopy DEV001 sym ld DEV005
```

- Use the `symclone set mode` command as described in "Modifying a clone copy session" on page 46. For example:

```
symclone -g ProdDB set mode precopy DEV001 sym ld DEV005
```

Copying only changed data to a clone

Note: The `-differential` option is specified by default. Use the `-nodifferential` flag to create a nondifferential session, which cannot be recreated or restored.

Subsequent cloning to the same target can be performed as differential copying. A differential clone operation copies only those device tracks that have changed since the full clone was performed (that is, only new writes to the source device will be copied). This requires that the copy session that existed for the full clone still exists. Because a full clone is required, the `-differential` option must be used with the `-copy` (or `-precopy`) option. The `-nocopy` option is not allowed.

For example:

```
symclone -g ProdDB create -copy -differential DEV001 sym ld DEV005
```

The `-differential` option creates an SDDF session for the source.

The `symclone activate` command initiates a full copy of the source device to the target device. For example:

```
symclone -g ProdDB activate DEV001 sym ld DEV005
```

To capture subsequent writes to the source during this clone pair session, use `symclone recreate` to clone just those tracks that have changed since the full copy completed:

```
symclone -g ProdDB recreate DEV001 sym ld DEV005
symclone -g ProdDB activate DEV001 sym ld DEV005
```

To save some steps, you can use the `symclone establish` command in the same way. For example, after the first full clone operation, you can capture any new writes with a subsequent clone operation that is automatically an incremental copy:

```
symclone -g ProdDB establish -full DEV001 sym ld DEV005
:
:
symclone -g ProdDB establish DEV001 sym ld DEV005
```

Copying from a source device to a larger target device

Clone source and target devices no longer have the requirement to be the same size. Now the size of a clone target device can be larger than the source device. This support requires the following SYMCLI environment variable be set:

`SYMCLI_CLONE_LARGER_TGT = ENABLED`

The following limitations apply:

- Restore is not allowed.
- Full copy support only; must use `-nodifferential`.
- VP Snap is not supported.

- Concatenated metadevices are not supported.
- When using this feature on striped metadevices you have to preserve the metageometry. That is, the source and target devices should contain the same number of metamembers. However, the target device members can be larger than the source device members.
- Exact pairing are the only operations allowed, as follows:

  `-file`

  `-g` or `-cg` with `-exact`

  `-g` or `-cg` with source and target ldev name supplied

Using the establish command

To create and then immediately activate a copy session with a single command, you can use the `symclone establish` command.

To create and then activate the copy session shown in the example on page 38, enter:

```
symclone -g ProdDB establish DEV001 sym ld DEV005 -full
```

Note: The `symclone establish` command sets the target device to Not Ready for a short time. If you are using a filesystem, unmount the target host before performing the establish command.

Pairing an additional target device with each source device in a group

When working with either a composite or device group, you can use the `-concurrent` option with the `create` or full `establish` action to pair an additional target device with each source device in a group.

To pair an additional target device with each source device in group `ProdDB`, enter:

```
symclone -g ProdDB create -concurrent
```

When the copy session is created, an additional target device will be paired with each source device in the group. For example, if there were two target devices paired with each source device in the group before creating the session, there will be three target devices paired with each source device after the session is created.

To verify that each source device in the group has multiple targets, enter:

```
symclone -g ProdDB verify -created -concurrent
```

Performing operations on devices in a clone target list

When working with either a composite or device group, you can use the `-tgt` option to indicate that devices from a local target list are to be used as targets for the specified action. You can also use the `-tgt` option together with the following options to indicate that devices from a remote target list are to be used as targets for the specified action:

- `-rdf` specifies remote attached devices (RTGTs).
- `-hop2` specifies devices that are remotely associated on the second hop of a cascaded SRDF configuration (2TGTs).

The `-tgt` option will work with all `symclone` actions, except for `query` and `verify`, as these actions are source device oriented and will select all target devices paired with source devices by default, including TGT and RTGT.

When working with specific pairs, the following `symclone` syntax will support the target devices:

```
sym ld LdevName
sym dev SymDevName
sym pd PdevName
```

Note: The *EMC Solutions Enabler Array Management CLI Product Guide* contains information on creating and managing clone target lists.

In the following example, *`action`* can be any `symclone` action, except for `query` and `verify`.

Copying from a local STD to a local TGT

To copy from a local standard device to a local target, use the following syntax:

For device groups:

```
symclone -g DgName action [-tgt [-bcv] | -rdf [-bcv | -tgt] |
           -rbcv -tgt | -rrbcv | -hop2 [-tgt]]
```

For composite groups:

```
symclone -cg CgName action [-tgt [-bcv] | -rdf [-bcv | -tgt] |
           -rbcv -tgt | -rrbcv | -hop2 [-tgt]]
```

Activating a clone copy session

To activate the copy session created in "Creating a clone copy session" on page 38, enter:

```
symclone -g ProdDB activate DEV001 sym ld DEV005
```

This activates the copy operation from the source device to the target device. Activating the copy session places the target device in the Read/Write state. The target host can access the cloned data and has access to data on the source host until the copy session is terminated.

Note: Cloned data is made available as a point-in-time copy at the time of activation and not at the time that the session was created.

Precopying data before activating the session

The `-precopy` option can be used with the `create` or `recreate` actions to start copying tracks in the background, before the copy session is activated. When using this option, a point-in-time copy will be established when the session is activated.

While in the Created state, the pre-copy process never actually completes. Instead, the process keeps checking for new writes to be pre-copied to the target device until the session is activated. Once activated, the normal background copy mechanism finishes copying the remaining tracks and the pre-copy operation ceases.

Making the target device not ready to the host

The `-not_ready` option can be used with the `activate` action to cause the target device to remain not ready to its host, as follows:

```
symclone -g ProdDB activate DEV001 sym ld DEV005 -not_ready
```

The copy session will be activated and the target device will be placed in the Not Ready state. The clone copy can later be read/write enabled to the host using either the `symdg ready` or `symdev ready` command.

Activating copy sessions consistently

The `symclone activate` command can be used with the `-consistent` option to create clone copies that are consistent with the database up to the point in time that the activation occurs. This feature can be implemented using either the Consistency Assist feature[1] or SRDF/A.

This feature can be used to create clone copies that are consistent with the database up to the point in time that the activation occurs. The feature suspends writes to the source devices during the activation.

When the activation has completed, writes are resumed and the target device contains a consistent production database copy of the source device at the time of activation.

You can use the Enginuity Consistency Assist (ECA) feature to consistently activate copy sessions across multiple, heterogeneous hosts. To consistently activate copy sessions using ECA, you must have either a control host with no database or a database host with a dedicated channel to the gatekeeper devices. In a SAN environment, gatekeepers and data devices may share the same FA port on the array but the gatekeepers must be available on a separate host HBA than the data devices. This means that there must be a dedicated channel from a host HBA to the switch that can be used to access only gatekeepers and not devices that contain host data. This will ensure that in write intensive environments SYMAPI will be able to freeze and then thaw I/O to the devices in the device group within the ECA window, regardless of the number of outstanding I/Os held by the HBA.

For detailed information on gatekeeper management, refer to the *EMC Solutions Enabler Installation Guide*.

Note: When activating a copy session from an R2 device, if the SRDF pair is in adaptive copy mode, the `-consistent` option is not allowed.

1. If the R2 is in a consistent state and the copy session is pre-copying data, setting the consistent option invokes SRDF/A to maintain consistency, instead of ECA.

Figure 7 illustrates how a control host can consistently activate a copy session involving three database hosts that access devices on an array.

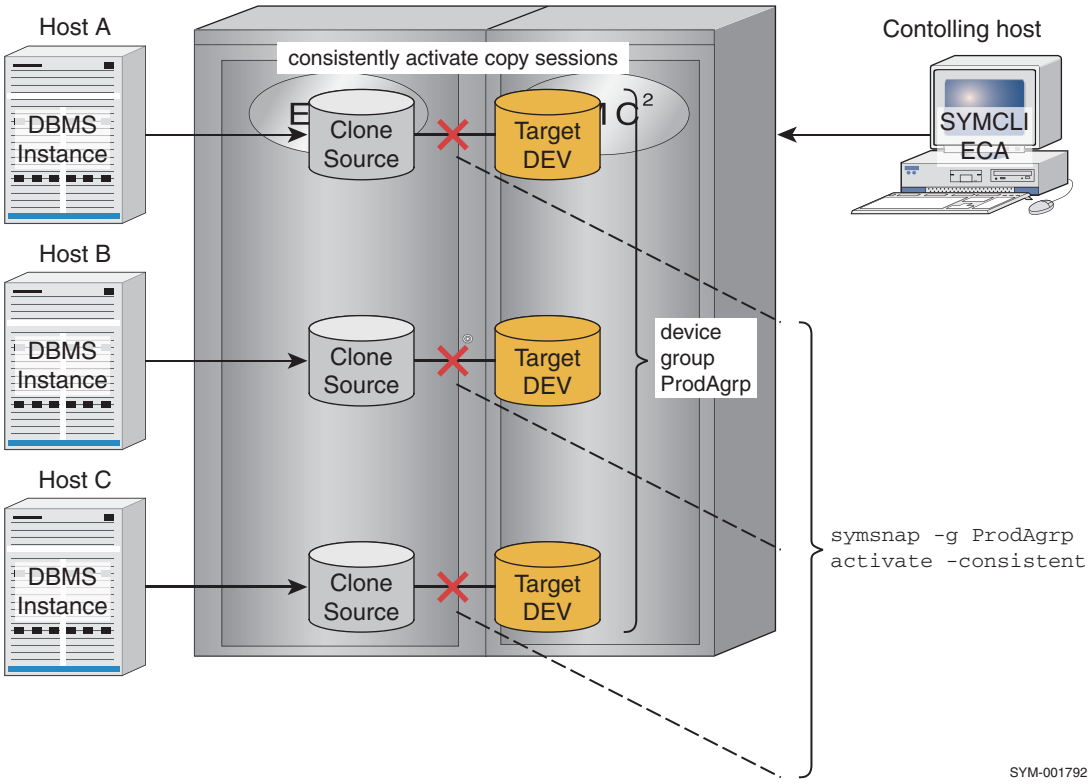

Figure 7 TimeFinder/Clone consistent activate using ECA

A device group, composite group, or a device file must be created on the controlling host for the target database to be consistently activated. Device groups can be created to include all of the devices being accessed or defined by database host access.

For example, if you define a device group that includes all of the devices being accessed by Hosts A, B, and C (see Figure 7), then you can consistently activate all of the copy sessions related to those hosts with a single command. However, if you define a device group that includes only the devices accessed by Host A, then you can activate those copy sessions related to Host A without affecting the other hosts.

Activating a clone session for an additional pair in a group

When working with either a composite or device group, you can use the `-concurrent` option with the `activate` action to activate a copy session for an additional clone pair in a group.

Note: The copy session must exist prior to issuing the command.

To activate a copy session for an additional clone pair in group `ProdDB`, enter:

```
symclone -g ProdDB activate -concurrent
```

Modifying a clone copy session

Use the `set mode` command to modify the mode in which a copy session is operating.

Possible values are:

- Copy — If a session was created without the `-copy` option, a copy will initiate once the session is activated, or the copy will start immediately if the session is already activated.
- Nocopy — If a session was created with the `-nocopy` option, the session will become CopyOnWrite/CopyOnAccess once the session is activated and no full device copy will initiate, or the copy will stop if the session is already activated.

  Note: Do not attempt to change a session created with the -differential option to the nocopy mode, as the session will fail. The -differential option is specified by default.

- Precopy — If a session was created without the `-precopy` option, setting the mode to `precopy` will cause a Pre-Copy to take effect. Once you are in Pre-Copy mode, you cannot change to No-Copy mode. Once the session is activated, the session changes to Copy mode. For more information on Precopy, refer to "Starting to copy data before activating the session" on page 40.

Example To change a copy session from Copy mode to Nocopy mode, enter:

```
symclone -g ProdDB set mode nocopy
```

Recreating a clone copy device

Use the `symclone recreate` command to incrementally copy all subsequent changes made to the source device (made after the point-in-time copy initiated) to the target device:

- With Enginuity 5876.159.102 and higher (excluding 5977), you can recreate a clone copy without terminating TimeFinder/Snap or VP Snap sessions that are cascading off of the clone target. "TimeFinder State Rules Reference" on page 185 provides additional details.
- You can use `symclone recreate` once a clone device is in the CopyInProg state to incrementally copy all subsequent changes made to the source device.

To recreate a clone copy:

- The copy session must not have been created with the `-nocopy` or `-nodifferential` option.
- The session must have been activated to establish the new point-in-time copy.

While in the Recreated state, the target device will remain Not Ready to the host.

Example To recreate the copy session created in the example on page 38, enter:

```
symclone -g ProdDB recreate DEV001 sym ld DEV005
```

Starting to copy data before activating the session

The `-precopy` option can be used with the `recreate` argument to start copying tracks in the background, before the copy session is activated.

When using this option, a point-in-time copy will be established when the session is activated.

While in the Recreated state, the pre-copy process never actually completes. Instead, the process keeps checking for new writes to be pre-copied to the target device until the session is activated. Once activated, the normal background copy mechanism takes over and the pre-copy operation ceases.

You must use the `-precopy` option with the `recreate` argument if the session was initially created as a precopy session.

Using the establish command

You can recreate and then immediately activate a clone session using the `symclone establish` command. To recreate and then activate the copy session shown in the example on page 38, enter:

```
symclone -g ProdDB establish DEV001 sym ld DEV005
```

Note: The `symclone establish` command sets the target device to Not Ready for a short time. Therefore, you may want to unmount the target device before issuing the command.

You can also use the `-concurrent` option with the `establish` command to activate an existing clone session for an additional clone pair in a group.

Example To recreate and then activate a copy session for an additional clone pair in group `ProdDB`, enter:

```
symclone -g ProdDB establish -concurrent
```

Recreating a clone session for each pair in a group

When working with either a composite or device group, you can use the `-concurrent` option with the `recreate` action to recreate a clone session for each clone pair in a group.

Example To recreate a copy session for each clone pair in group `ProdDB`, enter:

```
symclone -g ProdDB recreate -concurrent
```

Restoring data from a target device

Use the `symclone restore` command to copy target data to another device (full restore), or back to the original source device (incremental restore).

Restore operations require that the session be differential and the device pair be in the Copied state.

In the case of a full restore (`-full`), the original session will terminate and a copy session to the target of the restore will start.

In the case of an incremental restore, the original session copy direction is reversed and changed data is copied from the target device to the source device.

Example To fully restore data from the original target device (`DEV005`) created in the example on page 38 to a device (`DEV006`) that was not involved in the original clone session, enter:

```
symclone -g ProdDB restore -full DEV006 sym ld DEV005
```

Note: When constructing a `symclone restore` command, the device receiving the data always appears first in the command, followed by the device from which the data is being copied. Therefore, in the above command, `DEV006` is actually the target of the data being copied from `DEV005`.

Splitting a clone device pair

Use the `symclone split` command to *split* a clone device pair that is in the Restored state. This command changes the direction of the clone relationship (that is, the original source device becomes the source device for a future copy), which enables you to use either the `establish` or `recreate` command.

Example To split the pair created in the example on page 38, enter:

```
symclone -g ProdDB split DEV001 sym ld DEV005
```

Terminating a clone copy session

Terminating a copy session deletes the pairing information in the array and removes any hold on the target device.

Terminating a session while the device pairs are in the CopyOnAccess, CopyOnWrite, or CopyInProg state will cause the session to end. If the application has not finished accessing all of the data, the target copy is not a full copy.

The `symclone terminate` command is allowed for all TimeFinder/Clone pair states.

Example To terminate the copy session in the example in page 38 using the `symclone` command, enter:

```
symclone -g ProdDB terminate DEV001 sym ld DEV005
```

Note: A created and activated copy session may be terminated, but the data on the target device is not valid unless the state had previously been COPIED.

If the state is CopyInProg, then the `-symforce` option must be applied to terminate the session. This will also leave the target copy as an incomplete copy.

If you apply the `-not_ready` option, Solutions Enabler leaves the target devices in their prior ready or not ready state at the completion of the terminate operation. If you do not apply the `-not_ready` option, Solutions Enabler makes the target devices ready at the completion of the terminate, regardless of their state prior to the terminate operation.

Querying clone pairs

You can perform a query to determine the state of a clone pair or all clone pairs in a device group, composite group, or device file. The query is sent via the gatekeeper device to the array, returning with information about the state of the clone pair(s).

The following forms enable you to target devices in a device group, composite group, or device file:

```
symclone -g DgName query
symclone -cg CgName query
symclone -f[ile] FileName query
```

Examples To query the state of the clone pairs in the `prod` device group, enter:

```
symclone -g prod query
```

To query the state of SRDF-connected clone pairs in the `prod` device group, enter any of the following:

```
symclone -g prod query -rdf
symclone -g prod query -rdf -bcv
symclone -g prod query -rrbcv
symclone -g prod query -hop2
```

You can also obtain results using the `-offline` option, which looks at your configuration based on the host database.

The results of the query include the following information for each member of a clone pair in a device group:

- Logical device name
- Device name
- Number of invalid tracks
- Clone pair state

Using the -summary option

If you use the `-summary` option with the `query` argument, the results of the query will include the following information:

- Number of clone pairs in each clone pair state
- Number of invalid tracks
- Synchronization rate
- Estimated time to completion

The synchronization rate and estimated time to completion are shown only when `-i` or `-c` is specified and their has been a change in the number of invalid tracks since the previous iteration.

The `-summary` option also works with the `verify` argument.

Example To view the number of clone pairs in the `prod` device group that are in each state, and to view the estimated time to completion, enter:

```
symclone -g prod query -summary -i 60
```

Verifying clone pair states

You can use the `symclone verify` command to verify whether one or all clone pair(s) in a device group, composite group, or device file are in a particular state. The command can be used in scripts to guarantee that the clone device pair(s) are in a particular state prior to executing subsequent SYMCLI commands. If you do not specify any qualifiers with `symclone verify`, the default is to check for the Copied state.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symclone -g DgName verify
symclone -cg CgName verify
symclone -f[ile] FileName verify
```

The following options qualify the `symclone verify` command. If you need to verify a concurrent clone pair, include `-concurrent` with the option (for example, `-copied -concurrent`):

- `-copied` verifies that the copy sessions are in the Copied state.
- `-copyinprog` verifies that the copy sessions are in the CopyInProgress state.
- `-copyonaccess` verifies that the copy sessions are in the CopyOnAccess state.
- `-copyonwrite` verifies that the copy sessions are in the CopyOnWrite state.
- `-created` verifies that the copy sessions are in the Created state.
- `-cycled` verifies that the copy sessions have completed one precopy cycle. This options requires the `-precopy` option.
- `-precopy` verifies that the copy sessions are in the Precopy state.
- `-recreated` verifies that the copy sessions are in the Recreated state.
- `-restored` verifies that the copy sessions are in the Restored state.
- `-split` verifies that the copy sessions are in the Split.

Examples

For a multi-clone or concurrent clone device group, specifying the clone on the command line ensures that the verify operation checks the status of the clone. Otherwise, the verify operation checks the status of the standard device, which may no longer be established with the clone that you want to verify. For example, the following command returns the status of standard device DEV001 with its last paired clone:

```
symclone -g ProdBgrp verify DEV001
```

The following command returns the status of a specific clone pair (`DEV001` with `DEV005`):

```
symclone -g ProdBgrp verify DEV001 sym ld DEV005
```

The following command checks status every 30 seconds until all clone pairs in the device group (ProdBgrp) or composite group (MyConGrp) are in the Copied state (the default when no state is specified on the command line):

```
symclone -g ProdBgrp -i 30 verify
symclone -cg MyConGrp -i 30 verify
```

Possible outputs at 30-second intervals can be that none, not all, or all devices are copied.

The verify action returns a value of zero (code symbol CLI_C_SUCCESS) if the verify criteria are met, or one of the unique codes in Table 4 if the verify criteria are not met.

Table 4 lists the options to verify a clone pair state:

Table 4 Using options to verify a clone pair state

Options used with Verify	Code number	Code symbol
-copied	55	CLI_C_NOT_ALL_COPIED
-copied	56	CLI_C_NONE_COPIED
-copyinprog	53	CLI_C_NOT_ALL_COPYINPROG
-copyinprog	54	CLI_C_NONE_COPYINPROG
-copyonaccess	57	CLI_C_NOT_ALL_COPYONACCESS
-copyonaccess	58	CLI_C_NONE_COPYONACCESS
-copyonwrite	66	CLI_C_NOT_ALL_COPYONWRITE
-copyonwrite	67	CLI_C_NONE_COPYONWRITE
-created	60	CLI_C_NOT_ALL_CREATED
-created	61	CLI_C_NONE_CREATED
-cycled	75	CLI_C_NOT_ALL_PRECOPY_CYCLED
-cycled	76	CLI_C_NONE_PRECOPY_CYCLED
-precopy	73	CLI_C_NOT_ALL_PRECOPY
-precopy	74	CLI_C_NONE_PRECOPY
-recreated	68	CLI_C_NOT_ALL_RECREATED
-recreated	69	CLI_C_NONE_RECREATED
-restored	12	CLI_C_NOT_ALL_RESTORED
-restored	13	CLI_C_NONE_RESTORED
-split	25	CLI_C_NOT_ALL_SPLIT
-split	26	CLI_C_NONE_SPLIT

Cascaded clone

In environments running Enginuity 5874 through 5876, the target device of a clone session can be used as the source for one or more clone sessions. This cascading clone capability allows a clone operation to take place with a device that is already involved in a clone operation without ending the first clone session.

As Figure 8 shows, cascaded sessions are accepted from left to right. This means you can use TimeFinder to clone device A to device B. Then, while the relationship between A and B is preserved, you can clone device B to device C. If you have session A→B→C, then session B→C can only be activated after session A→B has been activated. Precopy sessions are allowed.

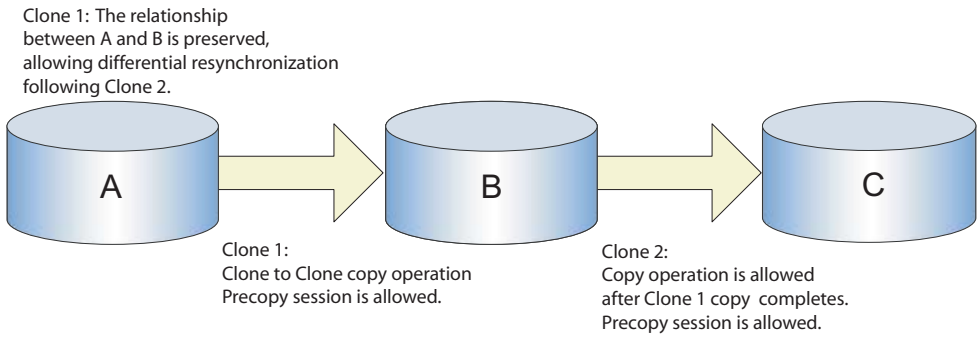

Figure 8 Clone from clone target (both sessions are cascaded clone)

Cascaded clone on thin devices is supported in environments running Enginuity 5875 through 5876.

Incremental restores of clone targets to source devices with active TimeFinder/Snap or VP Snap sessions is supported with Enginuity 5876.159.102 and higher (excluding 5977) with Solutions Enabler version 7.5 and higher.

Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.

Restore to Target

Restore to Target (RTT) allows you to perform an incremental restore to a cascaded clone target. For example, devices in an A -> B -> C cascaded clone session can copy data from device C to device A (via device B).

Persistent Restore to Target

Persistent Restore to Target (PTT) allows you to perform a TimeFinder/Snap restore to a TimeFinder/Clone target. For example, devices in an A -> B -> C cascaded session (where A -> B is TimeFinder/Clone and B -> C is TimeFinder/Snap) can copy data from device C to device A (via device B).

You can complete this operation without terminating the TimeFinder/Clone session or any existing TimeFinder/Snap sessions off of the TimeFinder/Clone target. PTT is supported in environments running Enginuity 5876.159.102 and higher (excluding 5977) with Solutions Enabler version 7.5 and higher.

Configuration Rules

When using a clone from a clone target, the following rules apply:

- If the session state is `CopyInProg`, `SyncInProg` or `RestoreInProg`, the `-symforce` flag is required.
- `Recreate` and `incremental clone establish` are allowed only on differential sessions.
- The mode `nocopy` can only be set on nondifferential sessions.
- The mode `precopy` can only be set in created and recreated states.
- Enginuity 5874 through 5876 only allows a 2 hop (device A→ target device B→target device C) cascaded relationship provided the interactions rules on page 53 are followed. Any attempt to establish a 3 hop relationship (D→C when A→B→C or Z →A when A→B→C) will fail. Although circular cascading A→B→A is not allowed, devices A and B can have additional multiple targets. For example: A→B(1)→C(1) and A→B(2) and A→B(3)→C(2). In a concurrent clone with cascaded relationship like A→B→C and A→D Enginuity 5874 will not allow incremental restore →D or B→C. Likewise, with two separate clone pairs like A-B and C-D, Enginuity 5874 will not allow a create, full establish, or full restore between B and C.

Table 5 lists the configuration rules for using a clone from a clone target. The `Terminate` and `Cancel` operations are allowed for all session states.

Table 5 Clone from clone target session states (both sessions are cascaded clone)

B→C session state	Clone A → Clone B → Clone C session state						
	A→B No session	A→B Created Recreated	A→B Precopy	A→B CopyInProg CopyOnAccess CopyOnWrite	A→B Copied Split	A→B RestoreInProg	A→B Restored
B→C No session	`Create` A→B Full Establish A→B Full Restore A→B Create B→C Full Establish B→C Full Restore B→C	Activate A→B Set Mode A→B Create B→C	Activate A→B Set Mode A→B Create B→C (precopy)	Recreate A→B Establish A→B Set Mode A→B Create B→C	Recreate A→B Establish A→B Restore A→B Set Mode A→B Create B→C Full Establish B→C	Create B→C Full Establish B→C	Split A→B Create B→C Full Establish B→C Full Restore B→C
B→C Created Recreated	Create A→B (no precopy) Full Establish A→B Full Restore A→B Activate B→C Set Mode B→C	Activate A→B Set Mode A→B (no precopy) Set Mode B→C	Not proper state	Recreate A→B (no precopy) Establish A→B Set Mode A→B (no precopy) Set Mode B→C	Recreate A→B (no precopy) Establish A→B Restore A→B Set Mode A→B (no precopy) Activate B→C Set Mode B→C	Activate B→C Set Mode B→C	Split A→B Activate B→C Set Mode B→C
B→C Precopy	Create A→B Full Establish A→B Full Restore A→B Activate B→C Set Mode B→C	Activate A→B Set Mode A→B Set Mode B→C	Activate A→B Set Mode A→B	Recreate A→B Establish A→B Set Mode A→B Set Mode B→C	Recreate A→B Establish A→B Set Mode A→B Restore A→B Activate B→C Set Mode B→C	Activate B→C Set Mode B→C	Split A→B Activate B→C Set Mode B→C
B→C CopyInProg CopyOnAccess CopyOnWrite	Full Restore A→B Recreate B→C Establish B→C Set Mode B→C	Not proper state	Not proper state	Not proper state	Restore A→B Set Mode A→B (no precopy) Recreate B→C Establish B→C Set Mode B→C	Recreate B→C Establish B→C Set Mode B→C	Split A→B Recreate B→C Establish B→C Set Mode B→C
B→C Copied Split	Create A→B Full Establish A→B Full Restore A→B Recreate B→C Establish B→C Restore B→C Set Mode B→C	Activate A→B Set Mode A→B Recreate B→C Set Mode B→C	Activate A→B Set Mode A→B Recreate B→C (precopy) Set Mode B→C	Recreate A→B Establish A→B Set Mode A→B Recreate B→C Set Mode B→C	Recreate A→B Establish AB Restore A→B Set Mode A→B Recreate B→C Establish B→C Set Mode B→C Incr. Restore B→C	Recreate B→C Establish B→C	Split A→B Recreate B→C Establish B→C Restore B→C
B→C RestoreInProg		Not proper state	Not proper state	Not proper state	Not proper state	Not proper state	
B→C Restored	Full Restore A→B Split B→C	Not proper state	Not proper state	Not proper state	Split B→C Incr. Restore B→C	Split B→C	Split B→C

Cascaded clone with VP Snap

Cascaded clone is supported with VP Snap with the following restrictions:

- Clone sessions must be in the `Copied` or `Split` state before you can create a VP Snap session from the target device.
- Incremental restore of VP Snap sessions is not permitted.
- The clone session must exist before the target device is used as a VP Snap source. Creating a clone session using a VP Snap source as the target device is not permitted.
- The target of a VP Snap session cannot be used as the source for any other clone or snap session.
- VP Snap is not supported with Clone Emulation mode.

VP Snap Restore to Target

VP Snap Restore to Target (VRTT) allows you to perform a VP Snap restore to a TimeFinder/Clone target. For example, devices in an A -> B -> C cascaded session (where A -> B is TimeFinder/Clone and B -> C is VP Snap) can copy data from device C to device A (via device B). VRTT is supported in environments running Enginuity 5876.159.102 with Solutions Enabler version 7.5.

Using a BCV as the clone source

As Figure 9 shows, you can create a copy session between a BCV device and a target device. The controlling host performs I/O to the standard device that is established with a BCV as part of a BCV pair. At some point, when the BCV is synchronized with the standard device, you can split the BCV from the standard and create a copy session between the BCV and a target device that might be accessed by Host C. The split operation must be entirely complete, including the background phase, before you can create a copy session on it.

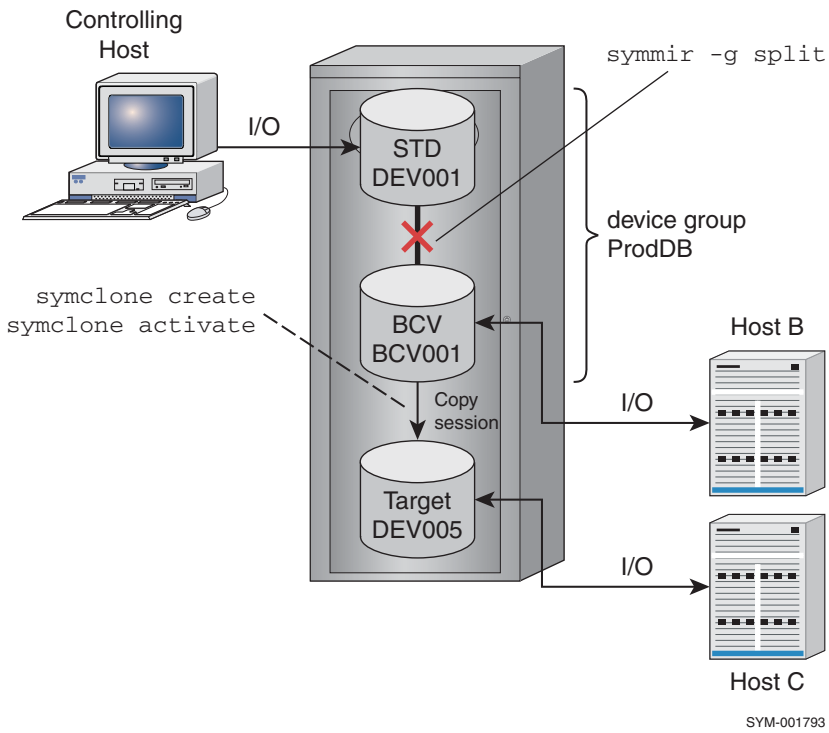

Figure 9 Creating a TimeFinder/Clone from a BCV source

For additional information on using the `symmir` command and how to perform an instant split operation, refer to Chapter 6.

Pair states ruling clone operations

Because various other ongoing operations can conflict with a clone session, certain rules must be considered. The availability of some clone copy operations depends on the current state of SRDF and BCV pairs. The following rules apply to certain BCV pair states:

- If the source or target of a `symclone create` or `activate` operation is a BCV, the BCV pair state must be split. The split must be totally complete before the operation is allowed.
- Available with Enginuity 5874, a source device A to a target device B (BCV device) clone session is in Emulation Mode state and the source device B to target device C is a clone session.
- A TimeFinder standard device cannot be created or activated in a clone session if the BCV pair state is SplitBfrRest or RestInProg.
- The `symclone terminate` command is allowed for all TimeFinder pair states.

Note: Appendix A, "TimeFinder State Rules Reference," explains the TimeFinder pair states that apply to TimeFinder/Clone copy sessions. "State rules for TimeFinder/Clone operations" on page 231 contains specific information regarding possible SRDF pair state conflicts.

Example: Creating a clone from a source device

The following example creates a copy session between source device `BCV001` in device group `ProdDB` and target device `DEV005` on the same array.

Once the copy session is activated, Host C can access target device tracks. If the accessed target tracks have not yet been copied, Enginuity software copies them for immediate access for Host C. If Host B writes to BCV device tracks that have not yet been copied, the Enginuity software immediately copies the tracks before allowing new data to overwrite those BCV tracks.

Note: In this example, where multiple hosts have access to the BCV source, consider using the `-not_ready` option with the `split` command to make the BCV not ready. This keep the same data on the BCV and clone. If you decide to use this option, you may need to release any Not Ready state imposed on any devices once the session completes.

The following steps outline the example shown in Figure 9:

1. Perform an instant split on the BCV pair. Use the `-not_ready` option to prevent the BCV's host from writing to it prior to the clone operation:

   ```
   symmir -g ProdDB split DEV001 -not_ready -noprompt
   ```

2. Verify that the background split is complete. The following command does a check every 5 seconds:

   ```
   symmir -g ProdDB verify DEV001 -split -bg -i 30
   ```

3. Begin a copy session between the BCV source device `BCV001` and the standard target device `DEV005`:

   ```
   symclone -g ProdDB create BCV001 sym ld DEV005
   ```

4. Activate the copy session to Host C:

   ```
   symclone -g ProdDB activate BCV001 sym ld DEV005
   ```

5. Make the BCV device ready to its host:

   ```
   symdg -g ProdDB ready -bcv BCV001 -noprompt
   ```

6. Query the state of the copy session and verify the CopyInProg state:

   ```
   symclone -g ProdDB query
   symclone -g ProdDB verify BCV001 -copyinprog
   ```

7. When the application has finished accessing the data, the copy session and pair relationship can be terminated:

   ```
   symclone -g ProdDB terminate BCV001 sym ld DEV005 -noprompt
   ```

8. Incrementally reestablish the BCV pair:

   ```
   symmir -g ProdDB establish DEV001 -noprompt
   ```

Creating multiple clone copies from a standard device

Figure 10 on page 58 illustrates creating multiple clone copies from a standard source device DEV001 on four standard target devices (DEV005, DEV006, DEV007, and DEV008) with various hosts accessing them.

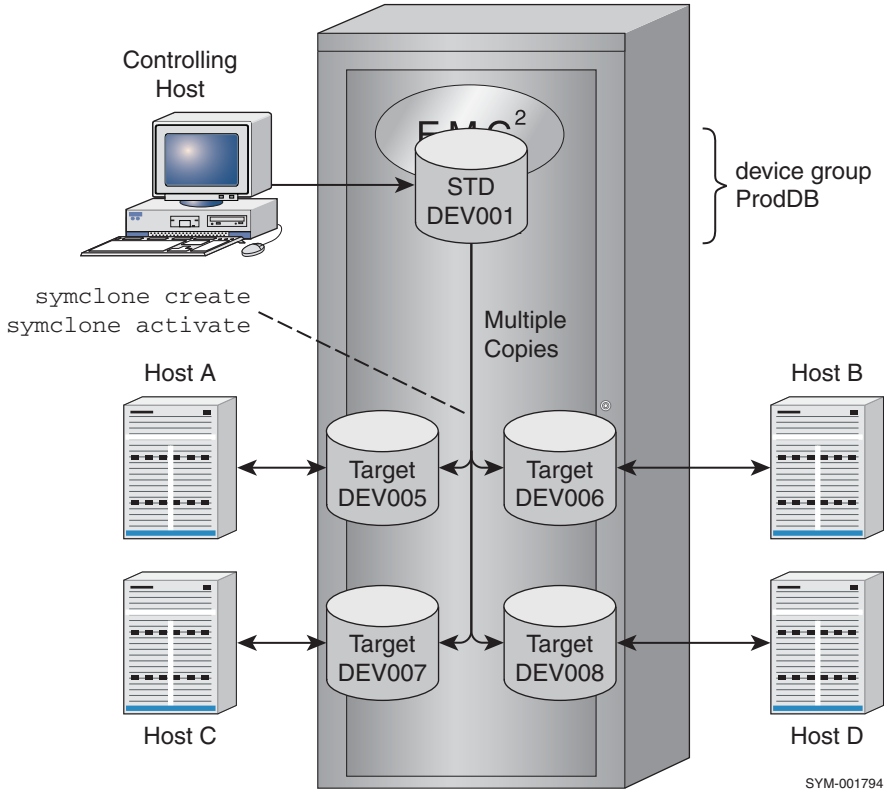

Figure 10 Creating multiple clone copies from a standard device

Note: A separate copy session must be created between the source device (DEV001) and each target device (DEV005, DEV006, DEV007, and DEV008).

Cloning a copy on a remote array

This section explains how to use SRDF to clone devices on a remote array.

Note: Refer to the *EMC Solutions Enabler SRDF Family Version CLI User Guide* for details about which SRDF features are supported with your platform.

Cloning a local R1 standard device

Figure 11 illustrates how to clone a copy of a local R1 standard device on a remote array. Performing SYMCLI commands from the controlling host allows the remote target device to receive a copy of the data from the R2 device. The cloned data can be accessed by the remote host.

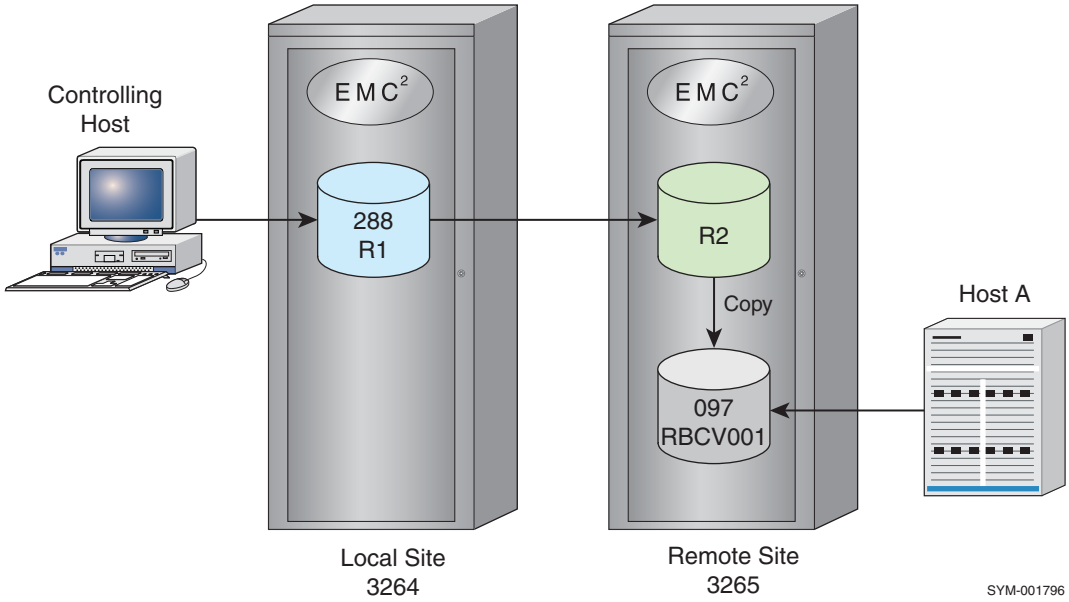

Figure 11 Cloning a copy of a local R1 standard device on a remote array

The following steps outline the example shown in Figure 11:

1. Create an RDF1 type device group:

   ```
   symdg create Rdf1Grp -type rdf1
   ```

2. Add to the device group an R1 standard device (288) on the local array (sid 3264) to be the source device. Associate a target BCV device (097) on the remote array to hold the clone copy. Note that to perform a remote operation you will need to use a remote target list (RTGT or RBCV list). This example uses an RBCV list.

   ```
   symdg -g Rdf1Grp -sid 3264 add dev 288
   symbcv -g Rdf1Grp associate dev 097 -rdf
   ```

3. Clone an immediate full copy from the source device (DEV001) to the remote BCV target device (RBCV001). DEV001 is the logical device name for device 288, and RBCV001 is the logical device name for BCV 097:

   ```
   symclone -g Rdf1Grp establish -full DEV001 bcv ld RBCV001 -rdf
   ```

4. To query the progress of the clone operation or verify when the copy is completed, issue the following commands that examine the clone pair (source and target):

   ```
   symclone -g Rdf1Grp query -rdf
   symclone -g Rdf1Grp verify -copied -rdf
   ```

Cloning a copy of a local BCV device on a remote array

Figure 12 illustrates how to clone a copy of a local BCV device on a remote array. Performing SYMCLI commands from the controlling host allows the remote target device to receive a copy of the data from the R2 device. The cloned data can be accessed by remote Host A.

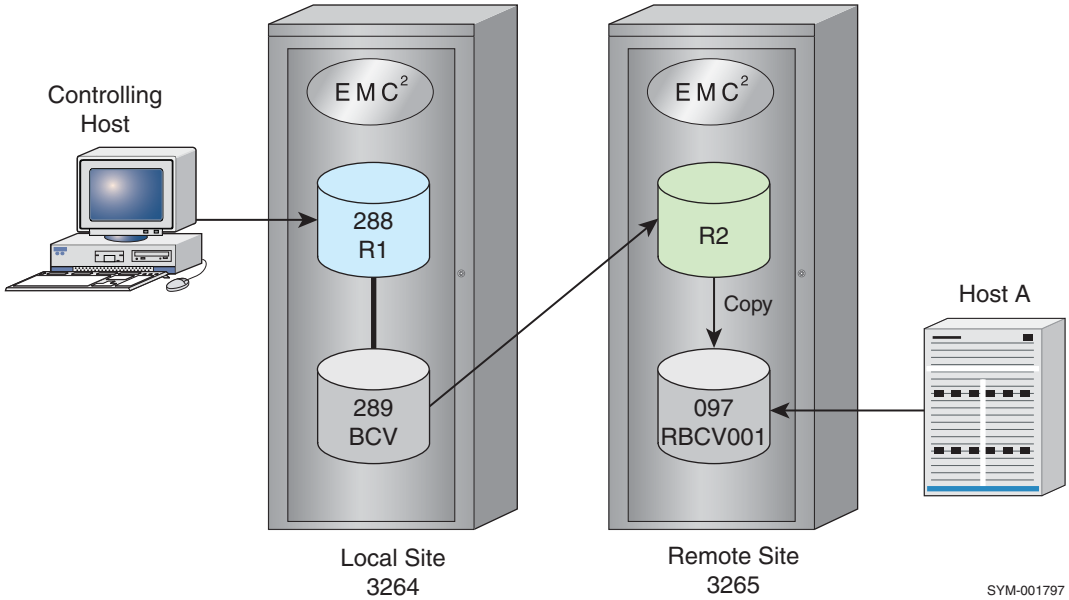

Figure 12 Cloning a copy of a local BCV device on a remote array

The following steps outline the example shown in Figure 12:

1. Create an RDF1 type device group:

   ```
   symdg create Rdf1Grp -type rdf1
   ```

2. Add to the device group a BCV device (289) on the local array (sid 3264) to be the source device. Associate a target BCV device (289) on the local array. Associate a target BCV device (097) on the remote array to hold the clone copy. In this case, the *remote* target device must be a BCV:

   ```
   symdg -g Rdf1Grp -sid 3264 add dev 288
   symbcv -g Rdf1Grp -sid 3264 associate dev 289 -rdf
   symbcv -g Rdf1Grp associate dev 097 -rdf
   ```

3. Clone an immediate full copy from the BCV device (BCV001) to the remote BCV target device (RBCV001). DEV001 is the logical device name for device 288, and RBCV001 is the logical device name for BCV 097:

   ```
   symclone -g Rdf1Grp establish -full BCV001 bcv ld RBCV001 -rdf -bcv
   ```

4. To query the progress of the clone operation or verify when the copy is completed, issue the following commands that examine the clone pair (source and target):

   ```
   symclone -g Rdf1Grp query -rdf -bcv
   symclone -g Rdf1Grp verify -copied -rdf -bcv
   ```

Cloning a copy on a hop-2 array

Figure 13 illustrates how to clone a copy of a local R1 standard device on a remote array in the second level of a multihop environment. Performing SYMCLI commands from the controlling host allows the remote target device to receive a copy of the data from the R2 device. The cloned data can be accessed by remote Host A.

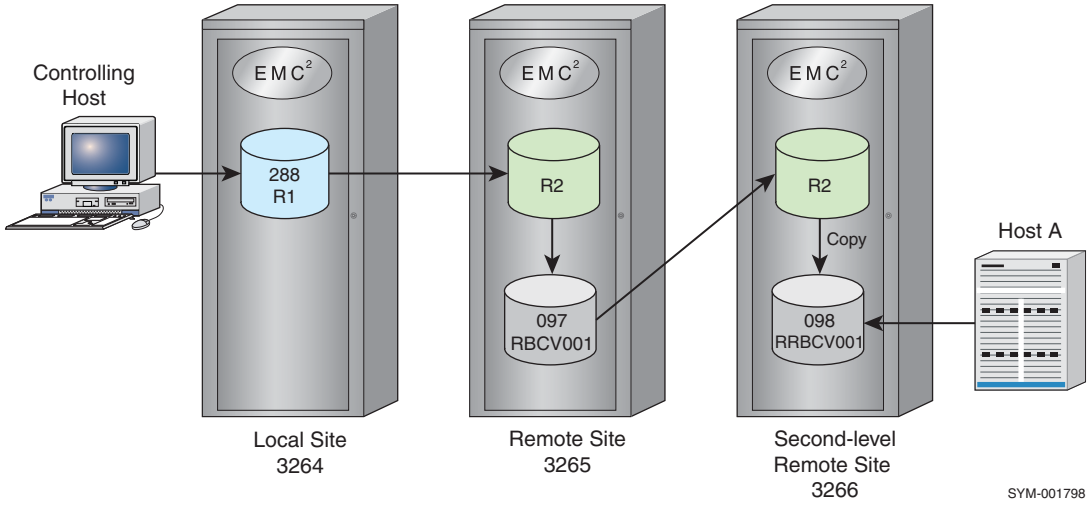

Figure 13 Cloning a copy of a local R1 standard device on a Hop-2 array

The following steps outline the example shown in Figure 13:

1. Create an RDF1 type device group:

   ```
   symdg create Rdf1Grp -type rdf1
   ```

2. Add to the device group an R1 standard device (288) on the local array (sid 3264) to be the source device. Associate a BCV device (097) on the Hop-1 array. Associate a BCV device (098) on the second-level (Hop-2) remote array to hold the clone copy. To perform a remote operation, you must use a remote target list (RTGT or RBCV list). Our example uses an RBCV list:

   ```
   symdg -g Rdf1Grp -sid 3264 add dev 288
   symbcv -g Rdf1Grp associate dev 097 -rdf
   symbcv -g RDF1GRP associate dev 098 -rrdf
   ```

3. Clone an immediate full copy from the source device (DEV001) to the remote BCV target device (RRBCV001). DEV001 is the logical device name for device 288, and RRBCV001 is the logical device name for BCV 098:

   ```
   symclone -g Rdf1Grp establish -full RBCV001 bcv ld RRBCV001 -rrbcv
   ```

   Instead of specifying the device level, you can also specify the device group:

   ```
   symclone -g Rdf1Grp establish -full -rrbcv
   ```

4. To query the progress of the clone operation or verify when the copy is completed, you can issue the following commands that examine the clone pair (source and target):

   ```
   symclone -g Rdf1Grp query -rrbcv
   symclone -g Rdf1Grp verify -copied -rrbcv
   ```

Cloning copies of the same data locally and remotely

Copies of the same data can be cloned to devices on a local and remote array at the same time so that their target devices have the same originator data.

Note that in the examples that follow, the procedures are very similar. The only differences are the device types used for the local copy (local target and local BCV target) and the command arguments used for the respective device type.

Example 1 Figure 14 illustrates how you can clone copies of a local R1 standard device to a local target device and to a remote BCV target device on a remote array.

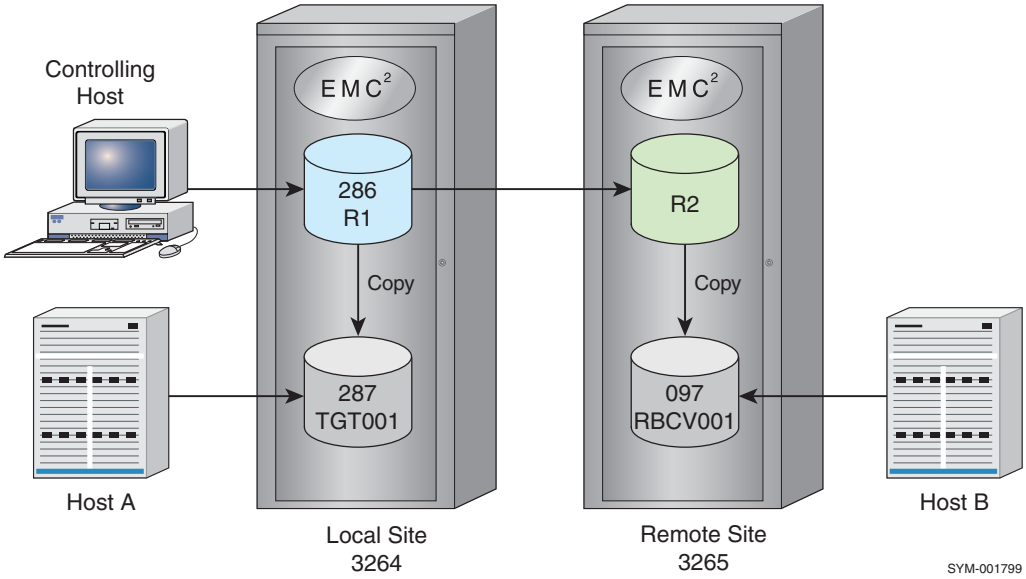

Figure 14 Cloning copies of a local R1 standard to a local target and to a remote BCV target

1. Create an RDF1 type device group:

```
symdg create Rdf1Grp -type rdf1
```

2. Add to the device group an R1 standard device (286) on the local array (sid 3264) to be the source device. Add a standard device target (287) on the local array. Associate a remote BCV target (097) on the remote array. To perform a remote operation, you must use a remote target list (RTGT or RBCV list). This example uses an RBCV list:

```
symdg -g Rdf1Grp -sid 3264 add dev 286
symdg -g Rdf1Grp -sid 3264 add dev 287
symbcv -g Rdf1Grp associate dev 097 -rdf
```

3. Clone an immediate full copy from the source device (DEV001) to the local and remote target devices (DEV002 and RBCV001, respectively). If there is no I/O to the source R1 device between these two commands, the same data will exist on both the local and remote target devices:

```
symclone -g Rdf1Grp establish -full
symclone -g Rdf1Grp establish -full -rdf
```

4. To query the progress of the local clone operation or verify when the local copy is completed, you can issue the following commands that examine the local cloned pair (source and target):

```
symclone -g Rdf1Grp query
symclone -g Rdf1Grp verify -copied
```

5. To query the progress of the remote clone operation or verify when the remote copy is completed, you can issue the following commands:

```
symclone -g Rdf1Grp query -rdf
symclone -g Rdf1Grp verify -copied -rdf
```

Example 2 Figure 15 illustrates how to clone copies of a local R1 standard device to a local BCV target device and to a BCV target device on a remote array.

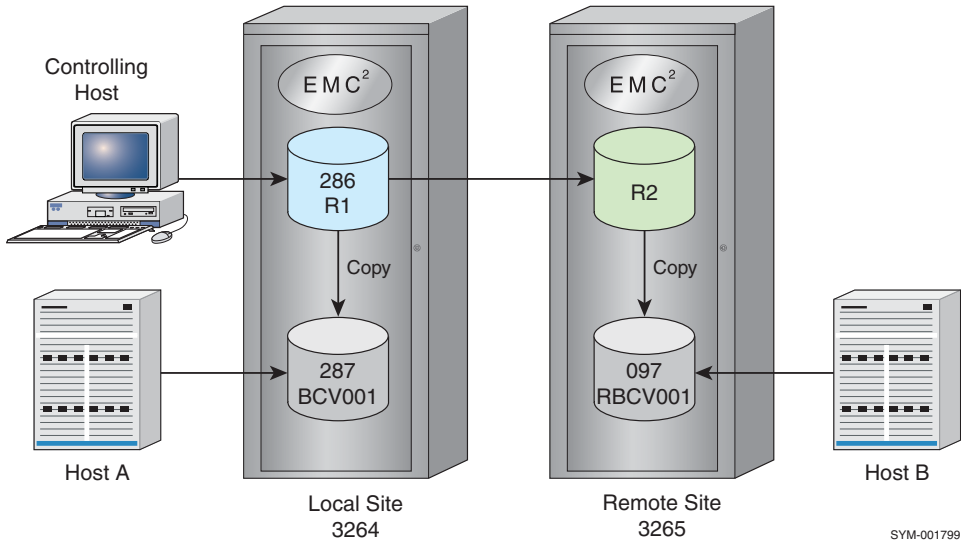

Figure 15 Cloning copies of a local device to a local BCV target and to a remote BCV target

1. Create an RDF1 type device group:

```
symdg create Rdf1Grp -type rdf1
```

2. Add to the device group an R1 standard device (286) on the local array (sid 3264) to be the source device. Add a BCV target (287) on the local array. Associate a remote BCV target (097) on the remote array. To perform a remote operation, you must use a remote target list (RTGT or RBCV list). This example uses an RBCV list:

```
symdg -g Rdf1Grp -sid 3264 add dev 286
symdg -g Rdf1Grp -sid 3264 add dev 287 -tgt
symbcv -g Rdf1Grp associate dev 097 -rdf
```

3. Clone an immediate full copy from the source device (DEV001) to the local and remote target devices (BCV001 and RBCV001, respectively). If there is no I/O to the source R1 device between these two commands, the same data will exist on both the local and remote target devices:

```
symclone -g Rdf1Grp establish -full -tgt
symclone -g Rdf1Grp establish -full -rdf
```

4. To query the progress of the local clone operation or verify when the local copy is completed, you can issue the following commands that examine the local cloned pair (source and target):

```
symclone -g Rdf1Grp query
symclone -g Rdf1Grp verify -copied
```

5. To query the progress of the remote clone operation or verify when the remote copy is completed, you can issue the following commands:

```
symclone -g Rdf1Grp query -rdf
symclone -g Rdf1Grp verify -copied -rdf
```

Cloning multiple copies locally and remotely

Multiple copies of the same data can be cloned to devices on a local and remote array at the same time so that their target devices have the same originator data.

The configuration in Figure 16 is basically the same as the configuration in Figures 14 and 15 except that this configuration uses a single `symclone` command to clone copies from a source device to *four* target devices on each array instead of cloning copies to one target on each array. In this configuration, eight hosts have access to copies of the same target data.

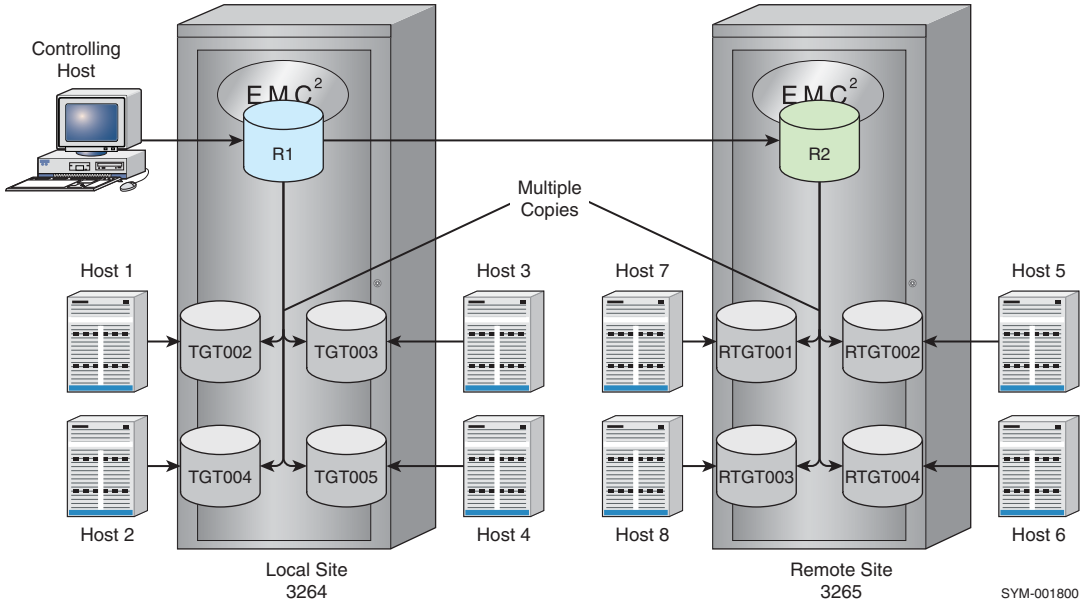

Figure 16 Cloning multiple copies on local and remote arrays

The process for building an RDF1 type device group is similar to the previous section. To add a source standard R1 device, a range of four standard device targets from the local array, and a range of four targets from the remote array:

```
symdg -g Rdf1Grp -sid 3264 add dev 100
symdg -g Rdf1Grp addall -devs 101:104 -tgt -sid 3264
symdg -g Rdf1Grp addall -devs 214:218 -rdf -tgt -sid 3265
```

What is different about this configuration is the cloning of copies from a single source to multiple target devices. To clone the four *local* target devices (TGT002 – TGT005) from the local source device (DEV001), you need to issue four `symclone establish` commands, specifying the same source device with each of the four targets:

```
symclone -g Rdf1Grp establish -full DEV001 sym ld TGT002
symclone -g Rdf1Grp establish -full DEV001 sym ld TGT003
symclone -g Rdf1Grp establish -full DEV001 sym ld TGT004
symclone -g Rdf1Grp establish -full DEV001 sym ld TGT005
```

To display the progress of *all* devices involved in the *local* clone operation, perform a clone query with the `-multi` option:

```
symclone -g Rdf1Grp query -multi
```

To verify the clone completion of one or more clone pairs in the device group specifically, or all clone devices, enter:

```
symclone -g Rdf1Grp verify -copied DEV001 sym ld DEV002
symclone -g Rdf1Grp verify -copied
```

To clone the four remote target devices (RTGT001 – RTGT004) from the source device (DEV001), issue four `symclone establish` commands with the `-rdf` option:

```
symclone -g Rdf1Grp establish -full -rdf DEV001 bcv ld RTGT001
symclone -g Rdf1Grp establish -full -rdf DEV001 bcv ld RTGT002
symclone -g Rdf1Grp establish -full -rdf DEV001 bcv ld RTGT003
symclone -g Rdf1Grp establish -full -rdf DEV001 bcv ld RTGT004
```

To display the progress of all devices involved in the *remote* clone operation, perform a clone query with the `-multi` option and the `-rdf` option:

```
symclone -g Rdf1Grp query -multi -rdf
```

To verify the clone completion of one or more remote clone pairs in the device group specifically, or all remote clone devices, add the `-rdf` option:

```
symclone -g Rdf1Grp verify -copied -rdf DEV001 sym ld RTGT004
symclone -g Rdf1Grp verify -copied -rdf
```

Cloning a copy at the tertiary site of a cascaded SRDF configuration

Using SRDF technology and the TimeFinder hop2 flag (`-hop2` option), you can clone devices on an array located at the tertiary site of a cascaded SRDF configuration (Figure 17). Performing SYMCLI commands from the controlling host allows the tertiary target device to receive a copy of the data from the R2 device. The cloned data can be accessed by remote Host A.

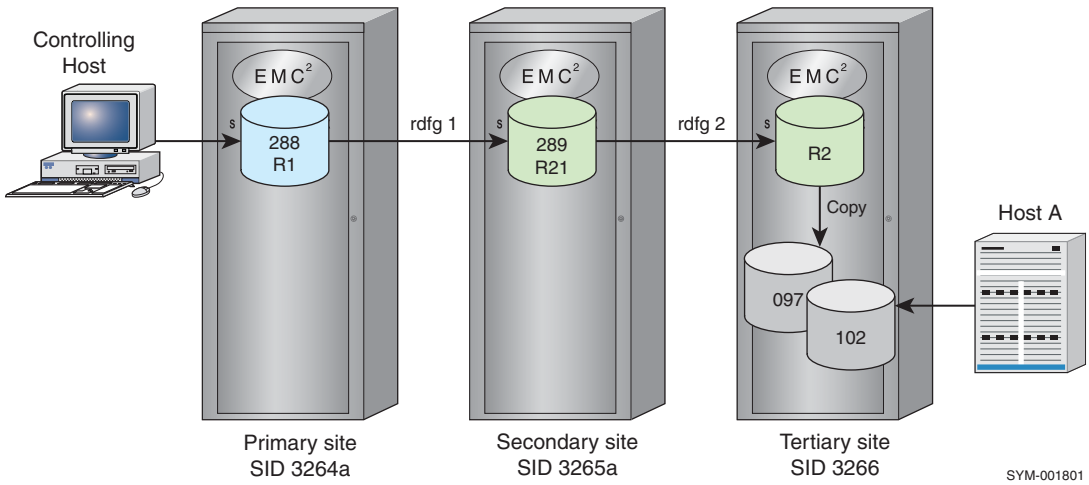

Figure 17 Cloning a copy at the tertiary site of a cascaded SRDF configuration

The following steps outline an example of cloning devices on an array located at the tertiary site of a cascaded SRDF configuration:

Note: The following procedure provides examples of device group and composite group commands.

1. Create an RDF1-type device group or composite group (for example, a group named `RDF1Grp`):

   To create an RDF1-type device group:

   ```
   symdg create Rdf1Grp –type rdf1
   ```

   To create an RDF1-type composite group:

   ```
   symcg create Rdf1Grp –type rdf1
   ```

2. Add devices to the group. From the primary site array, add an R1 standard device to be the source device. From the tertiary site array, add a target BCV device (097) to hold the clone copy:

   To add devices to a device group:

   ```
   symdg -g Rdf1Grp –sid 3264 add dev 288
   symbcv -g Rdf1Grp -hop2 -rdfg 1 -remote_rdfg 2 add dev 097
   ```

   To add devices to a composite group:

   ```
   symcg -cg Rdf1Grp –sid 3264 add dev 288
   symbcv -cg Rdf1Grp –sid 3264 -hop2 -rdfg 1 -remote_rdfg 2 add dev 097
   ```

   To add target devices to a device group using the `-tgt` option:

   ```
   symdg -g Rdf1Grp -hop2 -rdfg 1 -remote_rdfg 2 -tgt add dev 102
   ```

   To add devices to a composite group using the `-tgt` option:

```
symbcv -cg Rdf1Grp -sid 3264 -hop2 -rdfg 1 -remote_rdfg 2 -tgt add dev 102
```

3. Clone an immediate full copy from the source device to the remote BCV target device:

   ```
   symclone -g Rdf1Grp –hop2 create -precopy
   ```

   Or, you can use the `-tgt` option:

   ```
   symclone -g RdfGrp1 -hop2 create -precopy -tgt
   ```

4. To query the progress of the clone operation or verify when the copy is completed, you can issue the following commands that examine the clone pair (source and target):

   ```
   symclone -g Rdf1Grp query -hop2
   symclone -g Rdf1Grp verify –copied -hop2
   ```

Using composite groups to manage clone pairs across arrays

Figure 18 shows a production host locally connected to two arrays (A and B). A composite group is defined on the production host and includes source devices and target devices from each array. The target devices can be standard devices or BCV devices. Another locally connected host allows access to the clone targets.

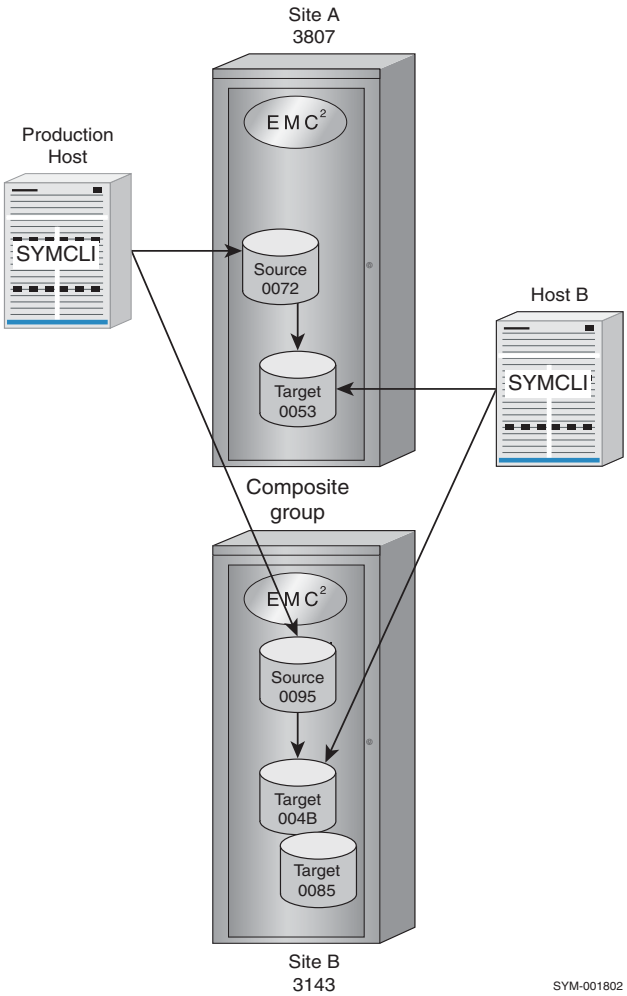

Figure 18 Using a composite group when a set of devices spans two arrays

Although clone copy operations might normally be performed from the production host (Figure 18) because the composite group is defined there in its SYMAPI database, there are methods that would allow you to control clone operations from another locally connected host like the target host. One way is to copy the composite group definition to another host. A more efficient method is to enable Group Naming Services (GNS), which automatically propagates the composite group definition to the arrays and other locally attached hosts that are running the GNS daemon.

The following steps explain how to setup a composite group that spans two arrays as shown in Figure 18 on page 67:

1. From the production host, create a Regular type composite group (for example, `MyGrp`):

   ```
   symcg create MyGrp -type regular
   ```

2. Add to the composite group those standard devices on array A (`3087`) and array B (`3143`) that are the source devices:

   ```
   symcg -cg MyGrp -sid 3087 add dev 0072
   symcg -cg MyGrp -sid 3143 add dev 0095
   ```

3. Associate a BCV target device from each array with the composite group:

   ```
   symbcv -cg MyGrp -sid 3087 associate dev 0053
   ```

4. Create clone pair sessions from those devices in the composite group:

   ```
   symclone -cg MyGrp create
   ```

5. Activate these clone pair sessions:

   ```
   symclone -cg MyGrp activate
   ```

Once you have setup the composite group, you can control specific clone pairs within it, as long as the devices reside in the same array. To create and activate only the DEV001/DEV002 clone pair from all devices in the group:

```
symclone -cg MyGrp create DEV001 sym ld DEV002
symclone -cg MyGrp activate DEV001 sym ld DEV002
```

Or using the one-step method:

```
symclone -cg MyGrp establish -full DEV001 sym ld DEV002
```

Command options with device groups

Table 6 lists the `symclone` control operations and the possible options to use when targeting a specified device group.

Table 6 symclone -g control arguments and possible options (page 1 of 2)

Option	Argument action										
	create	activate	establish -full	establish	recreate	split	restore -full	restore	terminate	query	verify
-bcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-both_sides		Y	Y	Y							
-c, -i	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-concurrent	Y	Y	Y	Y	Y						Y
-consistent		Y	Y	Y							
-copied											Y
-copy	Y										
-copyinprog											Y
-copyonaccess											Y
-copyonwrite											Y
-created											Y
-cycled											Y
-differential	Y										
-exact	Y		Y								
-force	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y
-hop2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-multi										Y	
-noprompt	Y	Y	Y	Y	Y	Y	Y	Y	Y		
-not_ready		Y	Y	Y			Y	Y			
-offline										Y	Y
-opt	Y		Y								
-preaction, -postaction		Y	Y	Y							
-precopy	Y				Y						Y
-preservetgtlocks, -lockid	Y	Y	Y	Y	Y	Y	Y	Y	Y		
-rbcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-rrbcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-rdf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 6 symclone -g control arguments and possible options (page 2 of 2)

Option	Argument action										
	create	activate	establish -full	establish	recreate	split	restore -full	restore	terminate	query	verify
-recreated											Y
-restored											Y
-restinprog											Y
-skip	Y	Y			Y	Y			Y		
-split											Y
-star	Y	Y	Y	Y	Y	Y	Y	Y	Y		
-summary										Y	Y
-symforce									Y		
-tgt	Y	Y	Y	Y	Y	Y	Y	Y	Y		
-v	Y	Y	Y	Y	Y	Y	Y	Y	Y		

Command options with composite groups

Table 7 lists the `symclone` control operations and the possible options to use when targeting a specified composite group.

Table 7 symclone -cg control arguments and possible options (page 1 of 2)

Option	Argument action									
	create	activate	establish -full	establish	recreate	restore -full	restore	terminate	query	verify
-bcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-both_sides		Y	Y	Y						
-c, -i	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-concurrent	Y	Y	Y	Y	Y					Y
-consistent		Y	Y	Y						
-copied										Y
-copy	Y									
-copyinprog										Y
-copyonaccess										Y
-copyonwrite										Y
-created										Y
-cycled										Y
-differential	Y									
-exact	Y		Y			Y	Y			

Table 7 symclone -cg control arguments and possible options (page 2 of 2)

Option	Argument action									
	create	activate	establish -full	establish	recreate	restore -full	restore	terminate	query	verify
-force	Y	Y	Y	Y	Y	Y	Y	Y		
-hop2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-multi									Y	
-noprompt	Y	Y	Y	Y	Y	Y	Y	Y		
-not_ready		Y	Y	Y		Y	Y			
-offline									Y	Y
-opt	Y		Y							
-opt_rag	Y		Y							
-preaction, -postaction		Y	Y	Y						
-precopy	Y				Y					Y
-preservetgtlocks, -lockid	Y	Y	Y	Y	Y	Y	Y	Y		
-rbcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-rrbcv	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-rdf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
-recreated										Y
-restinprog										Y
-restored										Y
-skip	Y	Y						Y		
-star	Y	Y	Y	Y	Y	Y	Y	Y		
-split										Y
-summary									Y	Y
-symforce								Y		
-tgt	Y	Y	Y	Y	Y	Y	Y	Y		
-v	Y	Y	Y	Y	Y	Y	Y	Y		

Command options with device files

Table 8 lists the `symclone` control operations and the possible options to use when targeting device pairs specified in a device file of a given array.

Table 8 symclone -file control arguments and possible options

Option	Argument Action							
	create	activate	establish	recreate	restore	terminate	query	verify
-both_sides		Y	Y					
-c, -i	Y	Y	Y		Y	Y	Y	Y
-consistent		Y	Y					
-copied								Y
-copy	Y							
-copyinprog								Y
-copyonaccess								Y
-copyonwrite								Y
-created								Y
-cycled								Y
-differential	Y							
-exact			Y					
-force	Y	Y	Y	Y	Y	Y		
-multi							Y	
-noprompt	Y	Y	Y	Y	Y	Y		
-not_ready		Y	Y		Y			
-preaction, -postaction		Y	Y					
-precopy	Y			Y				Y
-preservetgtlocks, -lockid	Y	Y	Y	Y	Y	Y		
-recreated								Y
-restinprog								Y
-restored								Y
-sid	Y	Y	Y	Y	Y	Y	Y	Y
-skip	Y	Y				Y		
-star	Y	Y	Y	Y	Y	Y		
-split								Y

Table 8 symclone -file control arguments and possible options (continued)

Option	Argument Action							
	create	activate	establish	recreate	restore	terminate	query	verify
-summary							Y	Y
-symforce						Y		
-v	Y	Y	Y	Y	Y	Y		

CHAPTER 4
Performing TimeFinder/Snap Operations (Enginuity versions 5773 through 5876)

This chapter describes how to control copy sessions for virtual devices using the SYMCLI `symsnap` command.

- TimeFinder/Snap overview 76
- Creating a virtual copy session 76
- Activating a virtual copy session 79
- Recreating a virtual copy session 82
- Restoring data from virtual devices 83
- Terminating a virtual copy session 87
- Querying snap pairs 87
- Verifying snap pair states 88
- Using a BCV as the snap source 90
- Creating multiple virtual copies 92
- Attaching source and target virtual devices 95
- Using composite groups to manage snap pairs across arrays 96
- Snapping a copy on a remote array 99
- Snapping a copy from a remote BCV 100
- Snapping copies of a source device's data locally and remotely 101
- Snapping multiple copies 102
- Snapping a copy at the tertiary site of a cascaded SRDF configuration 104
- Snapping a copy from a clone target device 105
- Command options with device groups or composite groups 106
- Command options with device files 107

TimeFinder/Snap overview

For a high-level overview of TimeFinder/Snap functionality, refer to the *EMC Symmetrix TimeFinder Product Guide*.

Note: TimeFinder/Snap is not supported on the VMAX 10K platform. Refer to your product guide for details about supported features.

Snap operations are controlled from the host by using the `symsnap` command to `create`, `activate`, `terminate`, and `restore` the snap copy sessions. The snap operations described in this chapter explain how to manage the devices participating in a copy session using the SYMCLI.

Note: Data domain devices are not supported as TimeFinder/Snap source or target devices.

Figure 19 illustrates a virtual copy session where the controlling host creates a copy of standard device `DEV001` on target device `VDEV005`.

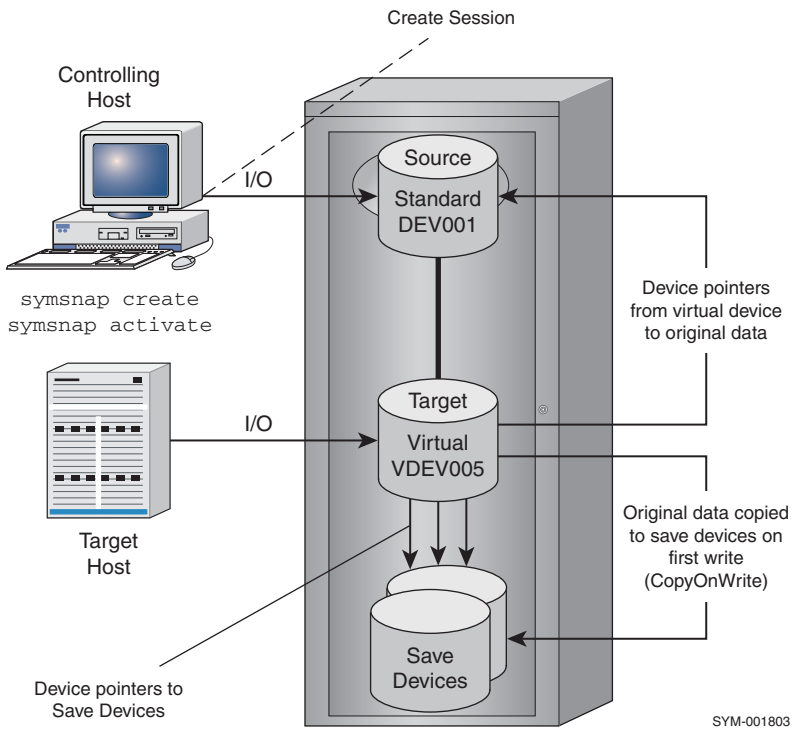

Figure 19 Copy of a standard device to a virtual device (VDEV)

Creating a virtual copy session

Initially, you must create a virtual copy session that defines and sets up the snap devices you have selected for the snap operation. For example, to begin a copy session and define a specified target device `VDEV005` to be the copy of source device `DEV001` in group `ProdDB`, enter:

```
symsnap -g ProdDB create DEV001 vdev ld VDEV005
```

The `symsnap create` action defines the copy session requirements and sets the track protection bitmap on the source device to protect all tracks and detect which tracks are being accessed by the target host or written to by the source host. The target virtual device remains Not Ready to its host and placed on hold status for copy session usage.

This prevents other control operations from using the device. The device pair state will transition from CreateInProg to Created when complete. The virtual data becomes accessible to its host when the copy session is activated. Refer to "Activating a virtual copy session" on page 79 for more information.

"Copy Session Limits" on page 19 provides details about the number of virtual copy sessions that you can create.

Multivirtual snaps

With multivirtual snaps enabled, Solutions Enabler supports up to 128 snaps from a source device. This support requires that you enable the following SYMCLI environment variable:

```
SYMCLI_MULTI_VIRTUAL_SNAP = ENABLED
```

Note: TimeFinder/Snap sessions on VMAX 40K systems always use multi-virtual mode regardless of how this variable is set.

This setting appears in the snap device output from `symdev show`. The **Snap State Flags** value will be `MultiVirtual`.

A single source device can only have snaps of one type.

The SYMCLI Auto Terminate policy does not apply to multivirtual snaps. Even if the SYMCLI_SNAP_PAIR_POLICY environment variable is set to TERM_OLDEST, you cannot create a new multivirtual snap session after the maximum number of sessions has been reached.

Note: Starting with Enginuity 5876.159.102, multivirtual snap operations are supported with CKD devices. Snap `recreate` for multivirtual snap operations is supported in environments running Enginuity 5875 and higher.

Specifying a SAVE device pool

VMAX supports the creation of multiple named SAVE device pools, allowing `symsnap` commands to use a particular pool.

The `-svp` option can be used with the `create` action to specify which SAVE device pool to use for an operation. For example, to instruct the copy session created in the example on page 76 to use the SAVE device pool `Accounting`, use the following `symsnap` command:

```
symsnap -g ProdDB create DEV001 vdev ld VDEV005 -svp Accounting
```

Note: The specified pool must exist and contain at least one enabled device before creating the copy session.

In addition to the `-svp` option, you can also set the environment variable SYMCLI_SVP to a poolname to be used when `-svp` is not present in the command line. If `-svp` and SYMCLI_SVP are not used, the operation will use the default pool, DEFAULT_POOL.

Note: SAVE devices can also be organized into *Delta Set Extension* pools for use with SRDF/A. Refer to the *EMC Solutions Enabler SRDF Family CLI User Guide* for more information on using SAVE devices in this way.

Monitoring SAVE device usage

Using virtual copies requires proper planning to prevent the SAVE devices from filling up with pre-updated data. If the SAVE devices fill up, you will begin to lose the pre-update images of the newly changed tracks in the virtual copy session, the virtual device will be set to Not Ready, and the session will fail (only sessions with I/O activity will be in a failed state, sessions without I/O activity will continue to operate normally).

Should this happen, you must terminate the failed sessions to clear the tracks on the SAVE devices. Once a session is terminated, the virtual data is lost and the SAVE device space associated with the session is freed and returned to the SAVE device pool for TimeFinder/Snap use. In addition, you should also examine how you are using the SAVE device pools and consider adding more SAVE devices.

You can monitor SAVE devices by using the `symcfg show` command to display SAVE device pool details. For more information, refer to the *EMC Solutions Enabler Array Management CLI User Guide*.

Pairing an additional target device with each source device in a group

When working with either a composite or device group, you can use the `-concurrent` option with the `create` action to pair an additional target device with each source device in a group.

To pair an additional target device with each source device in group `ProdDB`, enter:

```
symsnap -g ProdDB create -concurrent
```

When the copy session is created, an additional target device will be paired with each source device in the group. For example, if there were two target devices paired with each source device in the group before activating the session, there will be three target devices paired with each source device after the session is activated.

To verify that each source device in the group has multiple targets, enter:

```
symsnap -g ProdDB verify -created -concurrent
```

Copying a virtual device to another virtual device (duplicate snap)

Duplicate snap functionality (Enginuity 5875 and higher) allows you to duplicate a point-in-time copy of a virtual device, which is paired in a previously activated snap session, to another virtual device. This second point-in-time copy session resides with the source device of the original snap session and is charged as part of the maximum number of sessions for that source device.

Use the `-duplicate` option with the `-create` command to begin a copy session that will take one or more source virtual devices and create a copy of the point-in-time data to one or more target virtual devices. For example, to begin a copy session and define virtual device `VDEV002` to be the copy of source device `VDEV001` in group `ProdDB`, enter:

```
symsnap -g ProdDB create -duplicate VDEV001 vdev ld VDEV002
```

In this example, `VDEV001` is the target of an activated snap session and `VDEV002` is not snap paired. The resulting duplicate session is the same point-in-time as the original virtual device, but the time stamp is not inherited from the original virtual device's session. The newly created session has its own time stamp indicating the time of its creation.

The following restrictions apply to the duplicate snap functionality:

- Snap create and activate operations cannot be mixed between normal snap sessions and duplicate snap sessions within the same operation.
- Two is the maximum number of duplicated sessions in the Created state.
- When a duplicate snap session is in the Created state, the original session cannot be terminated or recreated until the duplicate session is activated.

Activating a virtual copy session

To create a point-in-time image, you must activate the create copy session. To activate the copy session created in the example on page 76, use the following `symsnap` command:

```
symsnap -g ProdDB activate DEV001 vdev ld VDEV005
```

This activates the copy operation from the source device to the virtual target device. Activating the copy session starts the copy on first write mechanism and places the target device in the Read/Write state. The target host can access the copy and has access to data on the source host until the copy session is terminated.

Note: Virtual data is made available as a point-in-time copy at the time of activation and not at the time that the session was created.

Using the establish command

With Solutions Enabler 7.4 and higher, you can use the `symsnap establish` command to create and then immediately activate a copy session with a single command.

To create and then activate the copy session shown in the example on page 76, enter:

```
symsnap -g ProdDB establish DEV001 vdev ld VDEV005 -full
```

Making the target device not ready to the host

The Not Ready (`-not_ready`) option can be used with the `activate` action to start the CopyOnWrite mechanism but causes the target device to remain not ready to its host, as follows:

```
symsnap -g ProdDB activate DEV001 vdev ld VDEV005 -not_ready
```

The copy session will be activated and the target device will be placed in the Not Ready state. The snap copy can later be Read/Write enabled to the host using the `symdg ready` command.

Activating copy sessions consistently

You can consistently activate multiple virtual copy sessions involving a database using the Enginuity Consistency Assist (ECA) feature. These features allow snap copy sessions to be activated with a consistent, restartable copy of the database.

Using ECA

You can use the Enginuity Consistency Assist (ECA) feature to activate virtual copy sessions that are consistent with the database up to the point in time that the activation occurs. The feature suspends writes to the source device during the activation.

Use the `symsnap activate` command with the consistent (`-consistent)` option to invoke ECA. When the activation has completed, writes are resumed and the target device contains pointers for a consistent production database copy of the source device at the time of activation.

To consistently activate copy sessions using ECA, you must have either a control host with no database or a database host with a dedicated channel. This will ensure that in write intensive environments SYMAPI will be able to activate the copy sessions within the ECA window, regardless of the number of outstanding I/Os held by the HBA. Refer to Figure 20, for a depiction of how a control host can consistently activate a copy session involving three database hosts that access devices on an array.

A device group, composite group, or a device file must be created on the controlling host for the target database to be consistently activated. Device groups can be created to include all of the devices being accessed or defined by database host access. For example, if you define a device group that includes all of the devices being accessed by Hosts A, B, and C (Figure 20), then you can consistently activate all of the copy sessions related to those hosts with a single command. However, if you define a device group that includes only the devices accessed by Host A, then you can activate those copy sessions related to Host A without affecting the other hosts.

Note: When activating a copy session from an R2 device, if the SRDF pair is in adaptive copy mode, the `-consistent` option is not allowed.

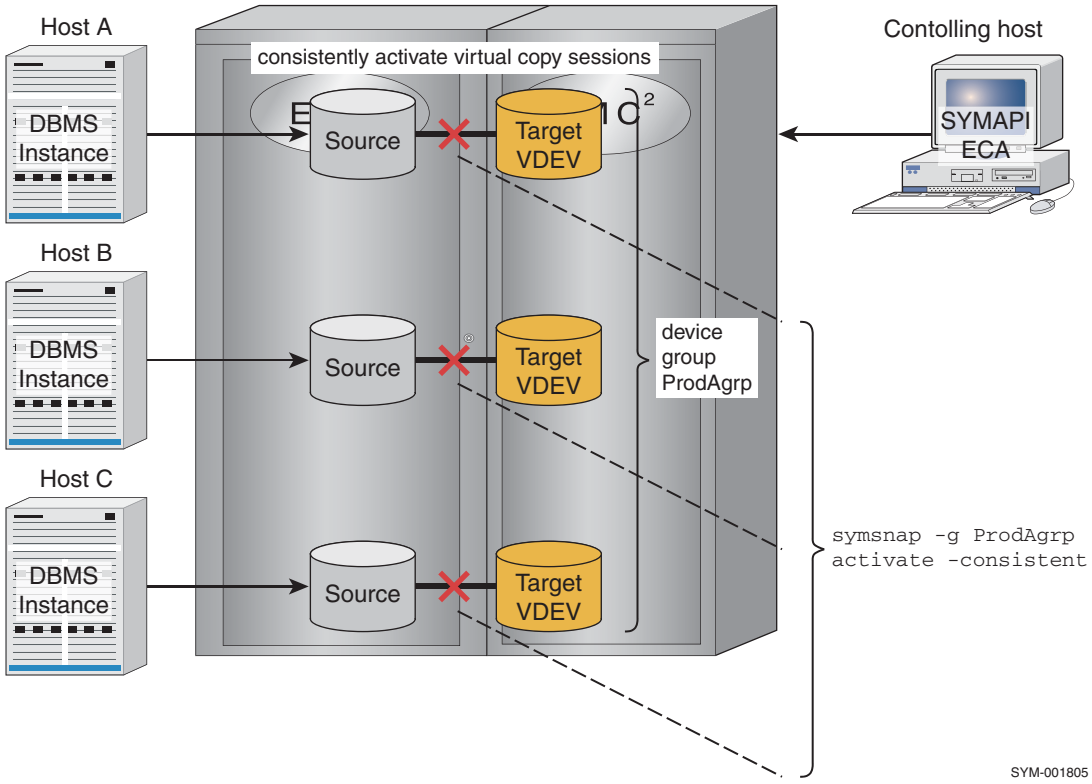

Figure 20 TF/Snap consistent activate using ECA

Activating an additional copy session for each device pair in a group

When working with either a composite or device group, you can use the `-concurrent` option with the `activate` action to activate an additional copy session for each device pair in a group.

To activate an additional copy session for each device pair in group `ProdDB`, enter:

```
symsnap -g ProdDB activate -concurrent
```

Activating a duplicate snap session

Once a duplicate snap session is activated, the copied session works like a normal snap session between the original source device and the duplicated virtual device. This session may be used for restore, recreate, and terminate operations.

Use the `-duplicate` option with the `-activate` command to activate a duplicate snap session. For example, to activate the copy session between target device `VDEV002` and source device `VDEV001` in group `ProdDB`, enter:

```
symsnap -g ProdDB activate -duplicate VDEV001 vdev ld VDEV002
```

In this example, the result is an activated session between `DEV001` and `VDEV002`, which is in addition to the original session between `DEV001` and `VDEV001`.

Combining the create and activate commands for a duplicate snap

You can combine the `symsnap create -duplicate` and `symsnap activate -duplicate` commands into a single action by using the `symsnap duplicate` command. For example, to create and activate a copy session between target device `VDEV002` and source device `VDEV001` in group `ProdDB`, enter:

```
symsnap -g ProdDB duplicate VDEV001 vdev ld VDEV002
```

Recreating a virtual copy session

Starting with Enginuity 5874, snap sessions can be recreated on an existing virtual device (VDEV) in preparation of activating a new point-in-time image. Snap `recreate` is only valid when issued against sessions that have been previously activated. This process makes it more convenient to reuse a virtual device to acquire a new point-in-time image.

Since the `recreate` operation replaces the previous point-in-time image with a new one, the used tracks in the SAVE devices that were associated with the previous session are freed during the processing of this command.

Recreating a virtual snap copy session requires the following steps:

1. Create a snap session. (Refer to page 76.)
2. Activate a snap session. (Refer to page 79.)
3. Recreate a snap session.
4. Activate a snap session. (Refer to page 79.)
5. Repeat the recreate/activate a snap session, as necessary.
6. Terminate the snap session when no longer needed. (Refer to page 87.)

To recreate the copy session activated in the example on page 79 use the following `symsnap` command:

```
symsnap -g ProdDB recreate DEV001 vdev ld VDEV005
```

With Solutions Enabler 7.4 and higher, you can use the `symsnap establish` command to recreate and then immediately activate a copy session with a single command.

To combine steps 3 and 4 described above, enter:

```
symsnap -g ProdDB establish DEV001 vdev ld VDEV005
```

The snap recreate functionality is supported with the `symsnap query` and `symsnap verify` commands. For example:

```
symsnap -g TestDg query -multi
symsnap -g TestDg verify -recreated
```

Note: You can use the `-recreated` parameter in combination with other verify states when verifying for multiple states. When multiple verify states are listed, the states are evaluated in a logical OR fashion so the result will be true if the session is in any of the listed states.

The following restrictions apply to the snap recreate functionality:

- The session to be recreated must be in a CopyOnWrite or Copied state.
- Any restore session that exists on the device must be terminated prior to issue the recreate operation.
- Snap `recreate` for multivirtual snap operations is supported in environments running Enginuity 5875 and higher.

Restoring data from virtual devices

Three types of restore operations can be performed for virtual device copy sessions:

- Incremental restore back to the original source device.
- Incremental restore to a BCV, which has been split from its original standard source device but maintains the incremental relationship with the source.
- Full restore to any standard or split BCV device outside of the existing copy session. The target device of the restore must be of the same size and emulation type as the source device.

A new restore copy session between the source device and the restore target device is created. A restore operation can only be performed if an additional copy session (two for Enginuity 5875 and higher) is available for use.

By default, any existing copy sessions persist until manually terminated by using the `symsnap terminate` command. After the virtual device has been restored to another device, the restore copy session must be terminated first, before another restore operation is allowed from that virtual device. Use the `symsnap query -multi` command to view all existing targets paired with a source device. The original snap copy session displays as being in the CopyOnWrite state, and the restore copy session displays as being either in the RestInProg or Restored state.

Incrementally restoring to a source

The following command-line entry shows an incremental restore operation back to the original source device `DEV001` from the virtual device `VDEV005`:

```
symsnap restore DEV001 vdev ld VDEV005
```

In this example, through the use of device pointers to the SAVE device, the virtual device `VDEV005` will be incrementally restored back to `DEV001`. Any changes made to the virtual device tracks during the active copy session will be restored back to the original source device.

Note: Any changes written to the original source device during the copy session will be overwritten by the virtual device tracks when the source device is restored.

The restore target device (source device `DEV001`) and virtual device (`VDEV005`) are automatically set to the Not Ready state while the track protection bitmaps are set up to copy any changed tracks. `DEV001` automatically becomes available for use (Ready state)

as soon as the track protection bitmap completes. The changed tracks then begin copying and will continue to copy in the background until all the protected tracks have been restored.

If you want to continue using the original copy session, the virtual device must be manually set to the Ready state after the restore operation has completed (all tracks are copied back to the source). Use one of the following commands to set the virtual device to the Ready state after the above restore operation:

```
symdg -g Group1 ready -vdev VDEV005
symdev -sid SymmID ready 001C
```

Once the restore completes, both the original and restore copy sessions are maintained and must be terminated manually if not needed for future use. When a original session is terminated, the device pointers are deleted from the virtual device and the SAVE device space is freed for future use.

Note: The restored copy session must be terminated first, before the original copy session is allowed to be terminated.

To terminate the restored copy session, enter:

```
symsnap terminate DEV001 vdev ld VDEV005 -restored
```

To terminate the original copy session, enter:

```
symsnap terminate DEV001 vdev ld VDEV005
```

Incrementally restoring to a BCV

The following command-line entry shows an incremental restore operation to a split BCV device `BCV001` from the virtual device `VDEV005`:

```
symsnap restore BCV001 vdev ld VDEV005
```

To restore to a BCV device, the BCV device must be split from its standard device. The point-in-time snap copy is restored to the BCV as shown in Figure 21. However, any further changes made to the point-in-time snap copy (from the attached host to the VDEV), during the active copy session, will also be included in the restore action to the split BCV device (`BCV001`). A new copy session begins between the source and target BCV and any changed tracks pointed to by the pointers are then copied to the split BCV from the source and SAVE devices. (Only the tracks that are different between the BCV and VDEV are copied.) Once the restore completes, both the original and restore copy sessions are maintained until manually terminated.

Note: You cannot restore to a BCV running in emulation mode.

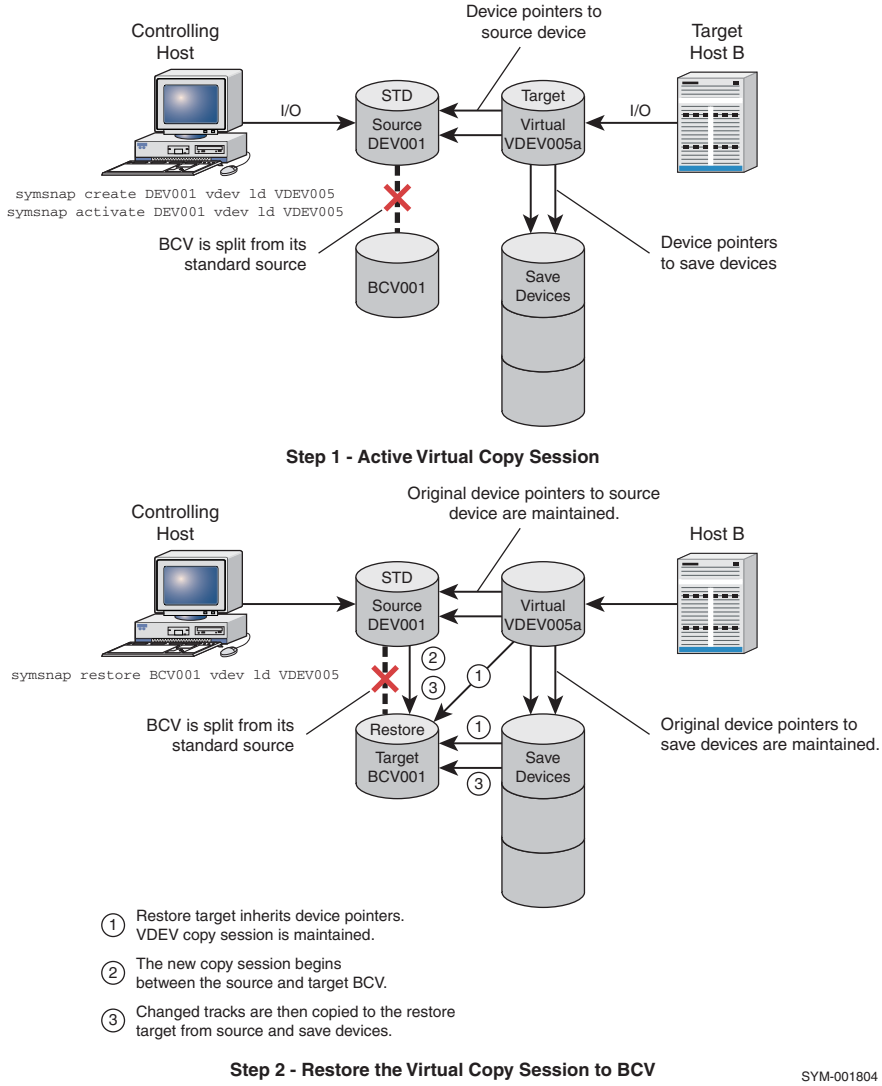

Figure 21 Incremental restore to a BCV

The target device (`BCV001`) and the virtual device (`VDEV005`) are automatically set to the Not Ready state while the track protection bitmaps are set up to copy any changed tracks. `BCV001` automatically becomes available for use (Ready state) as soon as the track protection bitmap completes. The changed tracks then begin copying and will continue to copy in the background until all protected tracks have been restored.

If you want to continue using the original copy session, the virtual device must be manually set to the Ready state after the restore operation has completed (all tracks are copied to the split BCV). Use one of the following commands to set the virtual device to the Ready state after the above restore operation:

```
symdg -g Group1 ready -vdev VDEV005
symdev -sid SymmID ready 001D
```

Note: The restored copy session must be terminated first, before the original copy session is allowed to be terminated.

To terminate the restored copy session, enter:

```
symsnap terminate DEV001 bcv ld BCV001 -restored
```

To terminate the original copy session, enter:

```
symsnap terminate DEV001 vdev ld VDEV005
```

Optionally, after terminating the copy session, you can also issue a TimeFinder `symmir restore` operation from the restored BCV back to the standard source device. For information on reestablishing and restoring BCV pairs, refer to Chapter 6.

Fully restoring to anywhere

A full device restore operation from the virtual device is allowed to any device of the same size and emulation type as the source device. The following example uses the `restore` command with the `-full` option to fully restore a virtual device `VDEV005` to another standard device `DEV004`:

```
symsnap -full restore DEV004 vdev ld VDEV005
```

Any changes made to the virtual device (`VDEV005`) during the active copy session will be restored to the specified device (`DEV004`). The restore target inherits the virtual device pointers. A new copy session begins between the source and target device and any changed tracks pointed to by the pointers are then copied to the target device from the source and SAVE devices. Upon completion of the restore operation, both the original and restore copy sessions are maintained until manually terminated.

The target device (`DEV004`) and the virtual device (`VDEV005`) are automatically set to the Not Ready state while the track protection bitmaps are set up to copy any changed tracks. `DEV004` automatically becomes available for use (Ready state) as soon as the track protection bitmap completes. The changed tracks then begin copying and will continue to copy until all protected tracks have been restored.

If you want to continue using the original copy session, the virtual device must be manually set to the Ready state after the restore operation has completed (all tracks are copied to the restore target). Use one of the following commands to set the virtual device to the Ready state after the above restore operation:

```
symdg -g Group1 ready -vdev VDEV005
symdev -sid SymmID ready 001E
```

When a session is terminated, all device pointers are deleted from the virtual device and the SAVE device space is freed for future use.

Note: The restored copy session must be terminated first, before the original copy session is allowed to be terminated.

To terminate the restored copy session, enter:

```
symsnap terminate DEV001 sym ld DEV004 -restored
```

To terminate the original copy session for a virtual device, enter:

```
symsnap terminate DEV001 vdev ld VDEV005
```

Terminating a virtual copy session

To terminate the copy session activated in the example on page 79 use the following `symsnap` command:

```
symsnap -g ProdDB terminate DEV001 vdev ld VDEV005
```

Terminating a copy session deletes the pairing information in the array and removes any hold on the target device. Terminating the session causes the target host to lose access to data pointed to by the virtual device.

Terminating a session while the device pairs are in the CopyOnWrite state will cause the session to end. Once the virtual copy session is terminated, the information is no longer available on the virtual device.

If a copy session has been restored, the restored session must be terminated first, before the original copy session is allowed to be terminated.

Note: If the state is RestInProg, then the `-symforce` option must be applied to terminate the session.

Terminating a duplicate snap session

You can terminate a duplicate snap session in the Created state using the `symsnap terminate -duplicate` command. For example, enter:

```
symsnap -g ProdDB terminate -duplicate VDEV001 vdev ld VDEV002
```

Once a duplicate snap session is activated, the session appears as normal snap session and is terminated using the `terminate` command without the `-duplicate` option.

Querying snap pairs

You can perform a query to determine the state of a snap pair or all snap pairs in a device group, composite group, or device file. The query is sent through the gatekeeper device to the array, returning with information about the state of the snap pair(s).

The following forms enable you to target devices in a device group, composite group, or device file:

```
symsnap -g DgName query
symsnap -cg CgName query
symsnap -f[ile] FileName query
```

Examples

To query the state of the snap pairs in the `prod` device group, enter:

```
symsnap -g prod query
```

You can also obtain results using the `-offline` option, which looks at your configuration based on the host database.

The results of the query include the following information for each member of a snap pair in a device group:

- Logical device name
- Device name
- Number of invalid tracks
- Snap pair state

Using the -summary option

If you use the `-summary` option with the `query` argument, the results of the query will include the following information:

- Number of snap pairs in each snap pair state
- Number of invalid tracks
- Synchronization rate
- Estimated time to completion

The synchronization rate and estimated time to completion are shown only when `-i` or `-c` is specified and their has been a change in the number of invalid tracks since the previous iteration.

The `-summary` option also works with the `verify` argument.

Example To view the number of snap pairs in the `prod` device group that are in each state, and to view the estimated time to completion, enter:

```
symsnap -g prod query -summary -i 60
```

Verifying snap pair states

You can use the `symsnap verify` command to verify whether one or all snap pair(s) in a device group, composite group, or device file are in a particular state. The command can be used in scripts to guarantee that the snap device pair(s) are in a particular state prior to executing subsequent SYMCLI commands. If you do not specify any qualifiers with `symsnap verify`, the default is to check for the Copied state.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symsnap -g DgName verify
symsnap -cg CgName verify
symsnap -f[ile] FileName verify
```

The following options qualify the `symsnap verify` command. If you need to verify a concurrent snap pair, include `-concurrent` with the option (for example, `-copied -concurrent`):

- `-copied` verifies that the copy sessions are in the Copied state.
- `-copyonwrite` verifies that the copy sessions are in the CopyOnWrite state.
- `-created` verifies that the copy sessions are in the Created state.
- `-recreated` verifies that the copy sessions are in the Recreated state.

- `-restinprog` verifies that the copy sessions are in the RestInProg state
- `-restored` verifies that the copy sessions are in the Restored state.

Examples

For a multi-snap or concurrent snap device group, specifying the snap on the command line ensures that the verify operation checks the status of the snap. Otherwise, the verify operation checks the status of the standard device, which may no longer be established with the snap that you want to verify. For example, the following command returns the status of standard device DEV001 with its last paired snap:

```
symsnap -g ProdBgrp verify DEV001
```

But the following command returns the status of a specific snap pair (`DEV001` with `DEV005`):

```
symsnap -g ProdBgrp verify DEV001 sym ld DEV005
```

The following command checks status every 30 seconds until all snap pairs in the device group (ProdBgrp) or composite group (MyConGrp) are in the Copied state (the default when no state is specified on the command line):

```
symsnap -g ProdBgrp -i 30 verify
symsnap -cg MyConGrp -i 30 verify
```

Possible outputs at 30-second intervals can be that none, not all, or all devices are copied.

The verify action returns a value of zero (code symbol CLI_C_SUCCESS) if the verify criteria are met, or one of the unique codes in Table 9 if the verify criteria are not met.

Table 9 Using options to verify a snap pair state

Options used with Verify	Code number	Code symbol
-copied	55	CLI_C_NOT_ALL_COPIED
-copied	56	CLI_C_NONE_COPIED
-copyonwrite	66	CLI_C_NOT_ALL_COPYONWRITE
-copyonwrite	67	CLI_C_NONE_COPYONWRITE
-created	60	CLI_C_NOT_ALL_CREATED
-created	61	CLI_C_NONE_CREATED
-recreated	68	CLI_C_NOT_ALL_RECREATED
-recreated	69	CLI_C_NONE_RECREATED
-restinprog	29	CLI_C_NOT_ALL_RESTINPROG
-restinprog	30	CLI_C_NONE_RESTINPROG
-restored	12	CLI_C_NOT_ALL_RESTORED
-restored	13	CLI_C_NONE_RESTORED

Using a BCV as the snap source

As shown in Figure 22 on page 91, you can create a virtual copy session between a BCV source device and a virtual target device. The controlling host performs I/O to the standard device that is established with a BCV as part of a BCV pair. At some point, when the BCV is synchronized with the standard device, you can split the BCV from the standard and create a point-in-time copy of the BCV. The split operation must be entirely complete including the background phase before you can create a copy session on it.

Chapter 6 provides additional information on using the `symmir` command and how to perform an instant split operation.

Pair states ruling snap operations

Because other operations can conflict with your copy session, certain rules must be considered. The availability of some snap copy operations depends on the current SRDF state, clone state, and BCV pair state. The following rules apply to certain BCV pair states:

- If the source of a `symsnap create` or `activate` operation is a BCV, the BCV pair state must be Split. The Split must be totally complete before the operation is allowed.

  Note: The existing BCV snap copy session must be terminated before the BCV can be reestablished or restored again with its standard device.

- A TimeFinder standard source device cannot be created, activated or restored in a copy session if the BCV pair state is SplitBfrRest or RestInProg.

  Note: If you have an active snap copy session from a standard device that also has an established BCV and you attempt to restore the standard from the BCV, the restore operation may fail due to insufficient space on the SAVE device. In this case it may be best to first restore and terminate the snap copy session, then attempt the BCV restore operation.

- The `symsnap terminate` command is allowed for all BCV pair states.

For the TimeFinder pair states that rule TimeFinder/Snap copy sessions, refer to Appendix A.

For information regarding possible SRDF pair state conflicts, refer to "State rules for TimeFinder/Snap operations" on page 235.

For a description of each BCV pair state, refer to *BCV Pair States* in Chapter 6.

Example: Creating a virtual copy from a BCV

The following example creates a copy of source device `BCV001` in device group `ProdDB` on an array to target device `VDEV005` on the same array.

Once the copy session is activated, Host C can access the virtual target device tracks containing the device pointers to the point-in-time copy. If Host B writes to BCV device tracks, TimeFinder immediately copies the original tracks to the SAVE device before allowing new data to overwrite those BCV tracks.

Note: In this example, where multiple hosts have access to the BCV source, consider using the `-not_ready` option with the `split` command to make the BCV not ready. This enables you to keep the same data on the BCV and virtual devices. If you decide to use this option, you may need to release any Not Ready state imposed on any devices after the session completes.

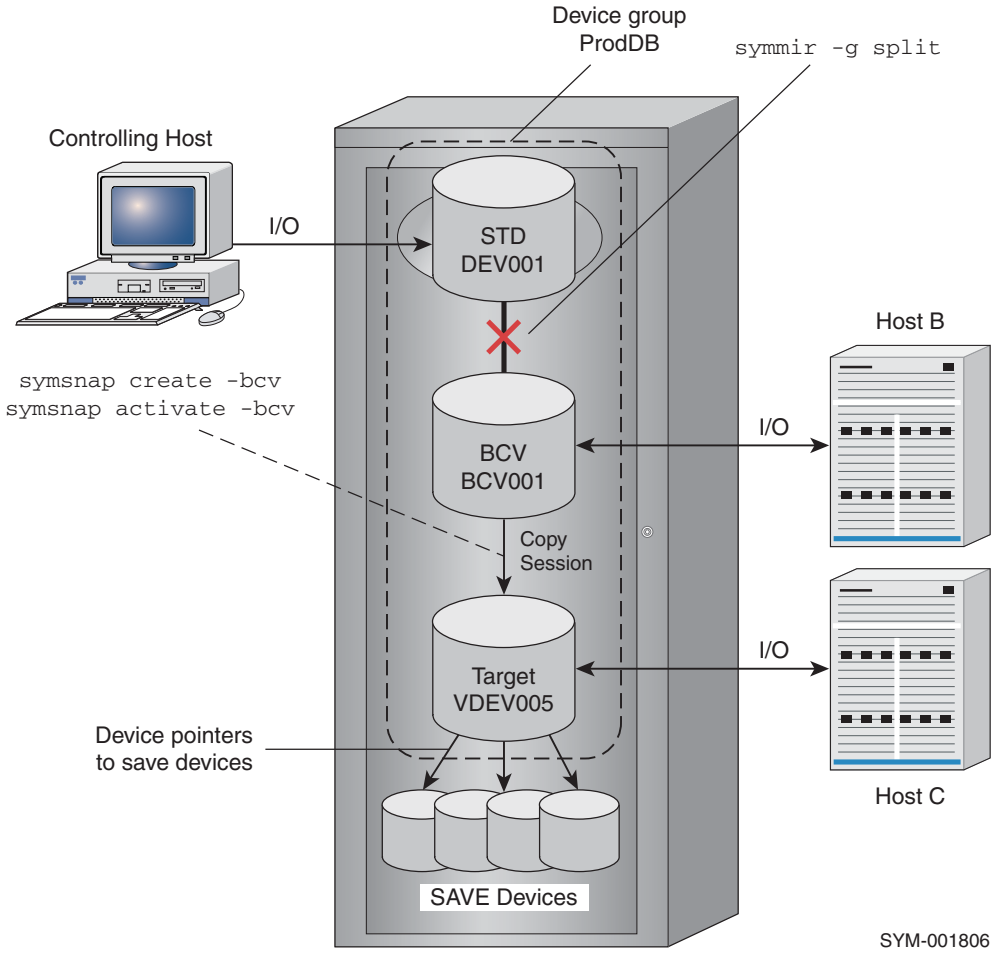

Figure 22 Creating a virtual copy from a BCV

For this example, the device pair was set in the Not Ready state.

The following steps outline the example shown in Figure 22 on page 91:

1. Perform an instant split on the BCV pair. Use the `-not_ready` option to prevent the BCV's host from writing to it:

   ```
   symmir -g ProdDB split DEV001 -not_ready -noprompt
   ```

2. Verify that the background split is complete. The following command checks every five seconds:

   ```
   symmir -g ProdDB verify DEV001 -split -bg -i 30
   ```

3. Create a copy session between the BCV source device `BCV001` and the virtual target device `VDEV005`:

   ```
   symsnap -g ProdDB create BCV001 vdev ld VDEV005
   ```

   or

   ```
   symsnap -g ProdDb create -bcv
   ```

   Note: Using the `-bcv` option applies the command within the device group to use BCV devices as the source and VDEV devices as the target.

4. Activate the copy session to Host C:

   ```
   symsnap -g ProdDB activate BCV001 vdev ld VDEV005
   ```

   or

   ```
   symsnap -g ProdDb activate -bcv
   ```

5. Query the state of the copy operation and verify the CopyOnWrite state:

   ```
   symsnap -g ProdDB query
   symsnap -g ProdDB verify BCV001 -copyonwrite
   ```

6. When the application has finished accessing the data, the copy session and pair relationship can be terminated:

   ```
   symsnap -g ProdDB terminate BCV001 vdev ld VDEV005 -noprompt
   ```

   or

   ```
   symsnap -g ProdDb terminate -bcv
   ```

7. Incrementally re-establish the BCV pair:

   ```
   symmir -g ProdDB establish DEV001 -noprompt
   ```

Creating multiple virtual copies

This section describes how to create multiple virtual copies from a standard device and a BCV device.

Creating multiple virtual copies from a standard device

Figure 23 illustrates creating copy sessions for multiple targets from a standard source device `DEV001` to four virtual target devices (`VDEV005`, `VDEV006`, `VDEV007` and `VDEV008`) with various hosts accessing them.

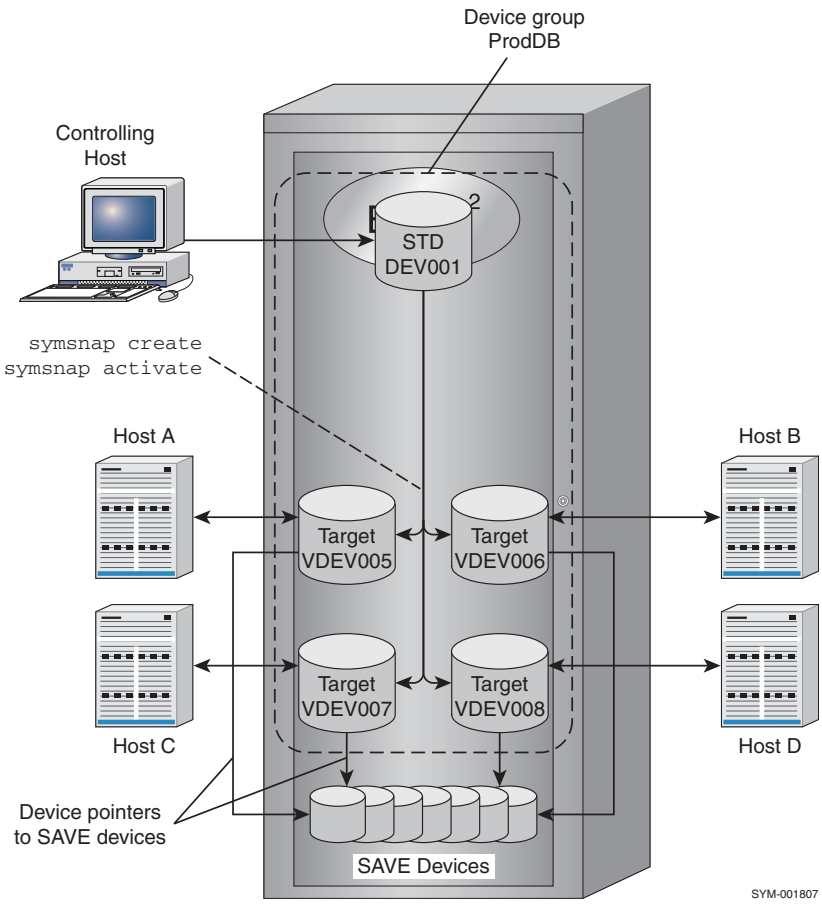

Figure 23 Creating multiple virtual copies from a standard device

Note: A separate copy session must be created between the source device (DEV001) and each target device (`VDEV005`, `VDEV006`, `VDEV007`, and `VDEV008`).

The following steps outline the example shown in Figure 23 on page 93:

1. Create a copy session between the standard source device `DEV001` and each of the four virtual target devices `VDEV005`, `VDEV006`, `VDEV007`, and `VDEV008`:

```
symsnap -g ProdDB create DEV001 vdev ld VDEV005
symsnap -g ProdDB create DEV001 vdev ld VDEV006
symsnap -g ProdDB create DEV001 vdev ld VDEV007
symsnap -g ProdDB create DEV001 vdev ld VDEV008
```

2. Activate the copy operation with one command to activate all sessions simultaneously:

```
symsnap -g ProdDB activate DEV001 vdev ld VDEV005
                           DEV001 vdev ld VDEV006
                           DEV001 vdev ld VDEV007
                           DEV001 vdev ld VDEV008
```

3. Using the `query` argument, you can display the state of all devices involved in the copy operation by including the `-multi` option:

```
symsnap -g ProdDB query -multi
```

4. When the host devices have finished using the copy sessions, the pair relationships can be terminated individually as needed:

```
symsnap -g ProdDB terminate DEV001 vdev ld VDEV005 -noprompt
symsnap -g ProdDB terminate DEV001 vdev ld VDEV006 -noprompt
symsnap -g ProdDB terminate DEV001 vdev ld VDEV007 -noprompt
symsnap -g ProdDB terminate DEV001 vdev ld VDEV008 -noprompt
```

Creating multiple virtual copies from a BCV device

Figure 24 illustrates how to create multiple copy sessions from a BCV source device `BCV001` to four virtual target devices (`VDEV005`, `VDEV006`, `VDEV007`, and `VDEV008`) with various hosts accessing them.

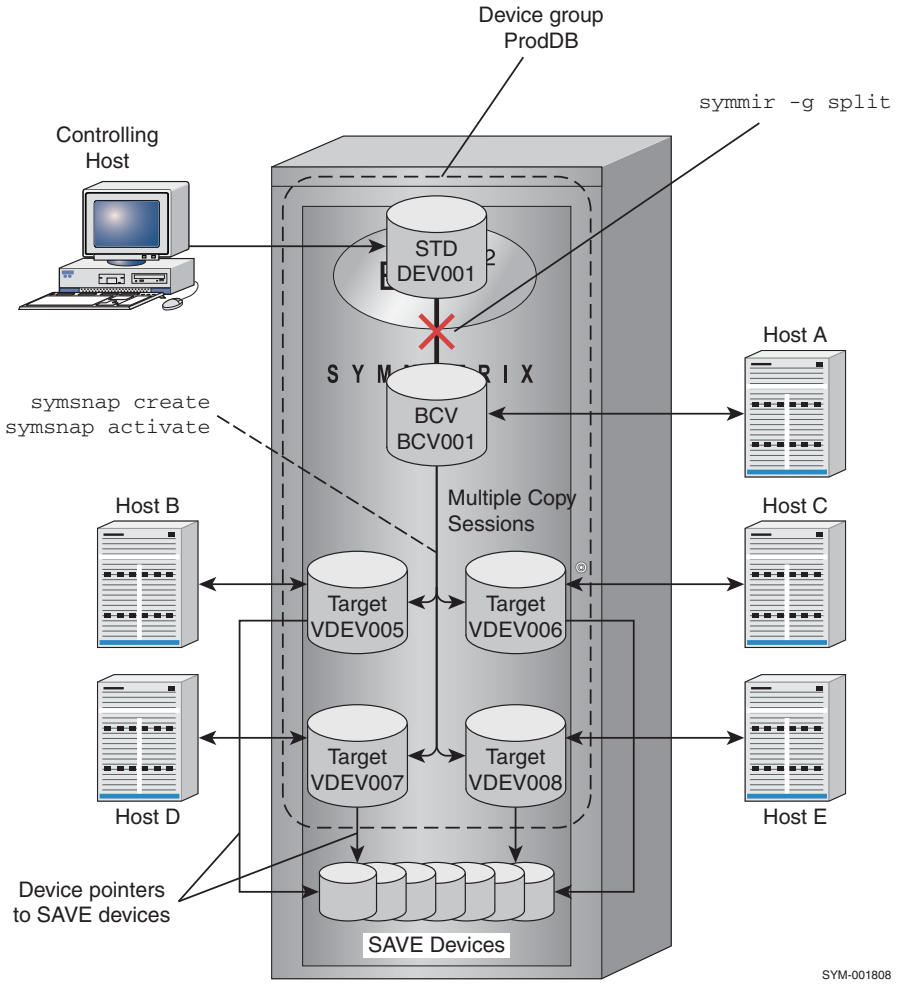

Figure 24 Creating multiple virtual copies from a BCV device

The following steps outline the example shown in Figure 24:

1. Perform an instant split on the BCV pair. Use the `-not_ready` option to prevent the BCV's host from writing to the device:

```
symmir -g ProdDB split DEV001 -not_ready -noprompt
```

Note: Chapter 6 provides additional information on using the `symmir` command and performing an instant split operation.

2. Verify that the background split is complete. The following command checks every five seconds:

```
symmir -g ProdDB verify DEV001 -split -bg -i 30
```

3. Create a copy session between the BCV source device `BCV001` and each of the four virtual target devices `VDEV005`, `VDEV006`, `VDEV007`, and `VDEV008`:

```
symsnap -g ProdDB create BCV001 vdev ld VDEV005
symsnap -g ProdDB create BCV001 vdev ld VDEV006
symsnap -g ProdDB create BCV001 vdev ld VDEV007
symsnap -g ProdDB create BCV001 vdev ld VDEV008
```

4. Activate the copy operation with one command to activate all sessions simultaneously:

```
symsnap -g ProdDB activateBCV001 vdev ld VDEV005
                          BCV001 vdev ld VDEV006
                          BCV001 vdev ld VDEV007
                          BCV001 vdev ld VDEV008
```

5. Using the `query` argument, you can display the state of all devices involved in the copy operation by including the `-multi` option:

```
symsnap -g ProdDB query -multi -BCV
```

6. When the host devices have finished accessing the data, the copy sessions and pair relationships can be terminated individually as needed:

```
symsnap -g ProdDB terminate BCV001 vdev ld VDEV005 -noprompt
symsnap -g ProdDB terminate BCV001 vdev ld VDEV006 -noprompt
symsnap -g ProdDB terminate BCV001 vdev ld VDEV007 -noprompt
symsnap -g ProdDB terminate BCV001 vdev ld VDEV008 -noprompt
```

7. Incrementally reestablish the BCV pair (`DEV001` and `BCV001`):

```
symmir -g ProdDB establish DEV001 -noprompt
```

Attaching source and target virtual devices

Use the `symsnap attach` and `detach` commands to set up preferred device pairs. Pre-determining attached device pairs eliminates the need to specify copy session target devices from within the device group for the `create` and `activate` arguments. The attached pairs will be used whenever a `symsnap` operation is requested for the specified device group. If a `symsnap create` or `activate` command does not specify a device pair from within the device group, the attached pair will automatically be used for the operation.

Note: The `attach` option can only be used when attaching a standard source device with a target VDEV device. You cannot use `symsnap attach` if the source device is a BCV device.

To set up a preferred attached device pair between source device `DEV001` in device group `ProdDB` and target device `VDEV002`, enter:

```
symsnap -g ProdDB attach DEV001 vdev ld VDEV002
```

To invoke a copy session operation from within a specified device group `ProdDB` using the pre-determined device pairs, enter:

```
symsnap -g ProdDB create
symsnap -g ProdDB activate
```

To detach (undo) the preferred device pair relationship, enter:

```
symsnap -g ProdDB detach DEV001 vdev ld VDEV002
```

Using composite groups to manage snap pairs across arrays

Figure 25 illustrates a production host that is locally connected to two arrays (A and B). A composite group is defined on the production host and includes source devices and target devices from these arrays. The target devices are virtual devices.

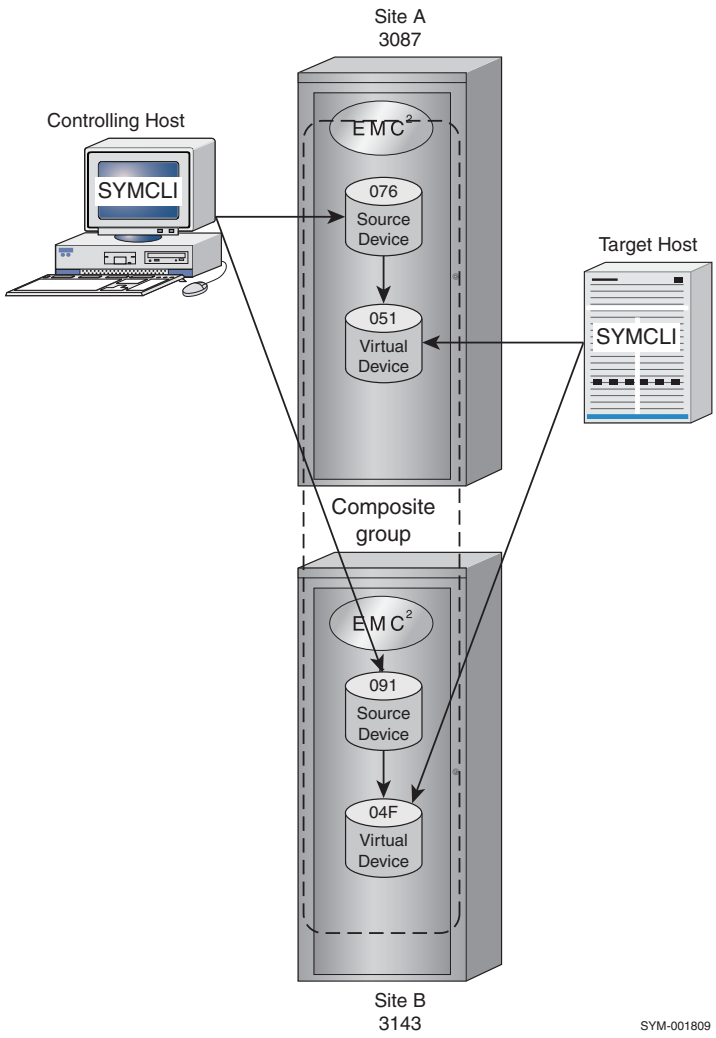

Figure 25 Using a composite group when a set of devices spans two arrays

Although snap operations might normally be performed from the production host because the composite group is defined there in its SYMAPI database, there are methods that would allow you to initiate copy sessions from another locally connected host like the target host. One way is to copy the composite group definition to another host.

A more efficient method is to enable Group Name Services (GNS), which automatically propagates the composite group definition to the arrays and other locally attached hosts that are running the GNS daemon.

The following steps outline the setup required for controlling a set of snap pairs that span two arrays, as shown in Figure 25 on page 97:

1. From the production host, create a Regular type composite group (for example, `MyGrp`):

   ```
   symcg create MyGrp -type regular
   ```

2. Add to the composite group those standard devices on array A (`3087`) and array B (`3143`) that are the source devices:

   ```
   symcg -cg MyGrp -sid 3087 add dev 0076
   symcg -cg MyGrp -sid 3143 add dev 0091
   ```

3. Add a virtual device from each array to the composite group:

   ```
   symcg -cg MyGrp -sid 3087 add dev 0051 -vdev
   symcg -cg MyGrp -sid 3143 add dev 004F -vdev
   ```

4. Create snap pair sessions from those devices in the composite group:

   ```
   symsnap -cg MyGrp create
   ```

5. Activate the snap pair sessions:

   ```
   symsnap -cg MyGrp activate
   ```

Composite group support allows the source of a snap operation to be a standard device (as shown in step 2 above) or a BCV device. If you add BCV devices to the composite group for the purpose of being snap sources, you must use the `-bcv` option with the `symsnap` commands. For example, the following command creates snap pairs in the composite group using the BCVs as source devices:

```
symsnap -cg MyGrp create -bcv
```

You can add nonsource BCV devices to the composite group for the purpose of restore operations. Composite group support allows you to restore to either the original source standard device or to a BCV that has been split from its paired standard device. The following commands split all BCV pairs in the composite group and perform a restore operation from the virtual devices to the BCVs:

```
symmir -cg MyGrp split
symsnap -cg MyGrp restore -bcv
```

For more information about performing an incremental restore to a BCV that is split from the original source device, refer to "Incrementally restoring to a BCV" on page 84.

You can control specific snap pairs within the composite group. To create and activate the DEV001/VDEV002 snap pair from devices in the group:

```
symsnap -cg MyGrp create DEV001 vdev ld VDEV002
symsnap -cg MyGrp activate DEV001 vdev ld VDEV002
```

Snapping a copy on a remote array

You can use a device group or composite group to snap devices on a remote array as shown in Figure 26. Performing SYMCLI commands from the controlling host allows the remote virtual device to receive a copy of the data from the R2 device. Remote Host A can access the snap data.

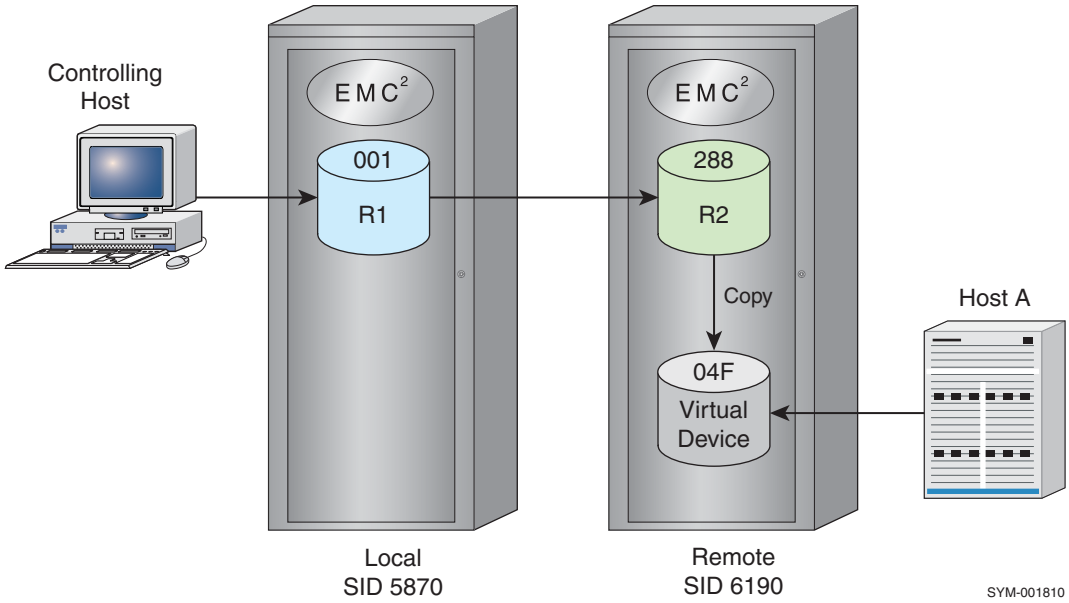

Figure 26 Snapping a copy on a remote array

The following steps outline an example of snapping local device data on a remote array:

1. Create an RDF1-type device group or composite group (for example, a device group named `remotesnaps`).

   ```
   symdg create remotesnaps -type rdf1
   ```

2. Add to the device group an R1 standard device (`001`) to hold the production data. Add a virtual device (`04F`) on the remote to hold the snap copy:

   ```
   symdg -g remotesnaps add dev 001 -sid 5870
   symdg -g remotesnaps add dev 04F -rdf -vdev
   ```

   If the local array has more than one RDF group (that is, concurrent RDF) linking the remote array, include the RDF group option (for example, `-rdfg 2`) when adding the remote virtual device.

3. Create a snap pair session from the devices in the device group:

   ```
   symsnap -g remotesnaps create -rdf DEV001 vdev ld RVDEV001
   ```

4. Activate the snap pair session:

```
symsnap -g remotesnaps activate -rdf DEV001 vdev ld RVDEV001
```

5. To query the progress of the remote snap operation:

```
symsnap -g remotesnaps query -rdf
```

Snapping a copy from a remote BCV

Using SRDF technology and the TimeFinder `-rdf` option, you can snap a copy from a BCV located on a remote array (Figure 27). By doing this, the remotely associated BCV is synchronized with the R2 device until you decide to split the BCV pair and snap a copy from the BCV to the virtual device.

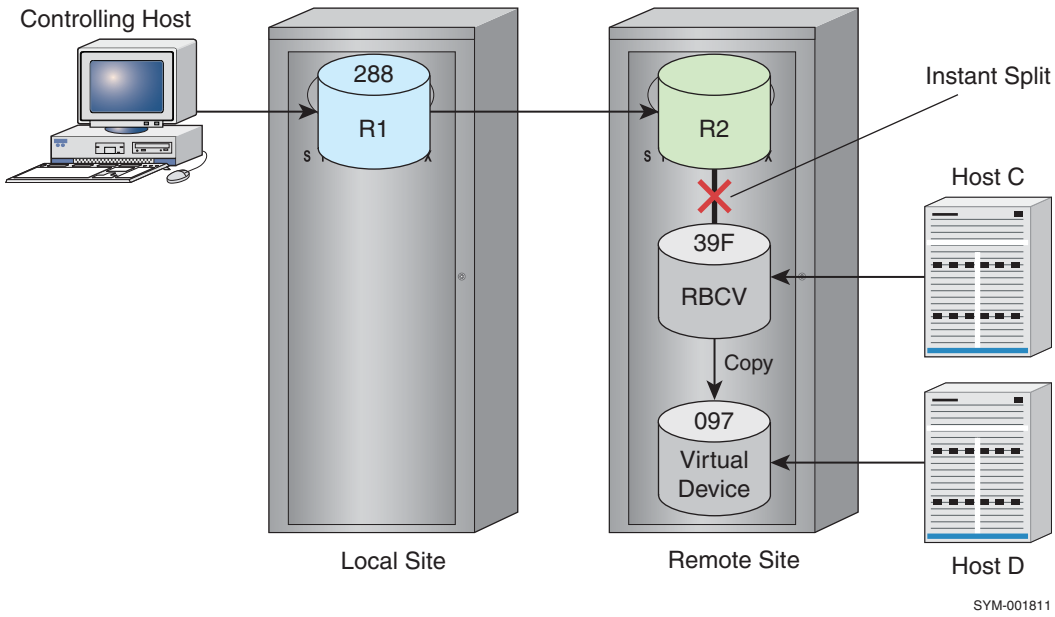

Figure 27 Snapping from a remote BCV source device

The following steps outline an example of creating a copy session between a remote BCV source device and a remote virtual device:

1. Create an RDF1-type device group or composite group (for example, a device group named `Rdf1Grp`):

```
symdg create Rdf1Grp -type rdf1
```

2. Add to the device group an R1 standard device on the local array to hold production data. Add a virtual device on the remote array to hold the snap copy:

```
symdg -g Rdf1Grp add dev 288
symdg -g Rdf1Grp add dev 097 -rdf -vdev
```

3. Associate with the device group an RBCV to be the snap source on the remote array:

```
symbcv -g Rdf1Grp associate dev 39F -rdf
```

4. Establish standard device 288 (`DEV001`) with the RBCV:

```
symmir -g Rdf1Grp establish -full DEV001 -rdf
```

5. When the remote BCV pair is fully synchronized, perform an instant split on the BCV pair:

```
symmir -g Rdf1Grp split -rdf
```

6. When the background split is complete, create and activate a copy session between the remote BCV source device and the remote virtual device:

```
symsnap -g Rdf1Grp snap create -rbcv RBCV001 vdev ld RVDEV001
symsnap -g Rdf1Grp snap activate -rbcv RBCV001 vdev ld RVDEV001
```

7. To query the progress of the remote snap operation or verify the copy session, you can issue the following commands that examine the snapped pair (the RBCV source device and the virtual device):

```
symsnap -g Rdf1Grp query -rbcv
symsnap -g Rdf1Grp verify -rbcv
```

Snapping copies of a source device's data locally and remotely

Figure 28 combines elements of Figures 26 and 27 to illustrate how to snap copies of a source device's data to devices on two arrays. By splitting the BCV pairs on both sides at the same time with a consistent split operation, the BCVs on both sides contain data that is consistent with the R1 source data up to the time of the split.

The controlling host can then perform a local snap and a remote snap, resulting in the virtual devices on both sides having copies of the source device's data (provided that the BCVs were not written to by their hosts prior to the snap operation).

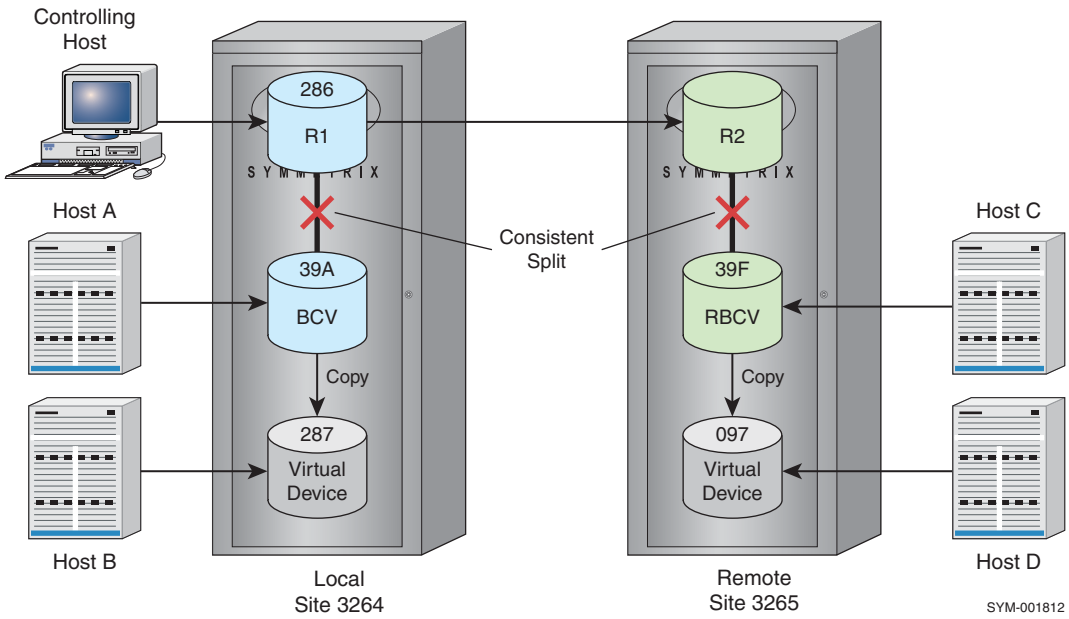

Figure 28 Snapping copies on local and remote arrays

The following steps outline the example shown in Figure 28:

1. Create an RDF1-type device group or composite group (for example, a device group named `Rdf1Grp`):

```
symdg create Rdf1Grp -type rdf1
```

2. Add to the device group an R1 standard device (`286`) on the local array (`sid 3264`) to hold production data. Add local virtual device `287` and remote virtual device `097` to hold the snap copies:

```
symdg -g Rdf1Grp –sid 3264 add dev 286
symdg -g Rdf1Grp –sid 3264 add dev 287 -vdev
symdg -g Rdf1Grp –sid 3264 add dev 097 –rdf -vdev
```

3. Associate with the device group local BCV `39A` and remote BCV `39F`:

```
symbcv -g Rdf1Grp –sid 3264 associate dev 39A
symbcv -g Rdf1Grp –sid 3264 associate dev 39F -rdf
```

4. Establish the BCV pairs on both arrays. DEV001 is the logical device name of the R1 source device (`286`):

```
symmir -g Rdf1Grp establish -full DEV001
symmir -g Rdf1Grp establish -full DEV001 –rdf
```

5. When the BCV pairs are fully synchronized, perform a consistent split on both BCV pairs. You can perform this operation with one command, using the `–both_sides` option. For example:

```
symmir -g Rdf1Grp split –consistent –both_sides
```

Note: To use the `-consistent` and `-both_sides` options, the SRDF pairs must be synchronized and in SRDF mode SYNCHRONOUS.

6. When the background split is complete, create the local and remote snap pairs:

```
symsnap -g Rdf1Grp create -bcv BCV001 vdev ld VDEV001
symsnap -g Rdf1Grp create -rbcv RBCV002 vdev ld RVDEV002
```

7. Activate the copy sessions for the local and remote snap pairs:

```
symsnap -g Rdf1Grp activate -bcv BCV001 vdev ld VDEV001
symsnap -g Rdf1Grp activate -rbcv RBCV002 vdev ld RVDEV002
```

8. To query the progress of the local snap operation or verify when the local copy is complete, you can issue the following commands that examine the local snap pair:

```
symsnap -g Rdf1Grp query -bcv
symsnap -g Rdf1Grp verify -bcv
```

9. To query the progress of the remote snap operation or verify when the remote copy is complete, you can issue the following commands that examine the remote snap pair:

```
symsnap -g Rdf1Grp query -rbcv
symsnap -g Rdf1Grp verify -rbcv
```

10. When subsequent snaps are no longer required, you can terminate the copy sessions by issuing commands that end copy sessions for the local and remote snap pairs:

```
symsnap -g Rdf1Grp terminate -bcv BCV001 vdev ld VDEV001 -bcv
symsnap -g Rdf1Grp terminate -rbcv RBCV002 vdev ld RVDEV002
```

Snapping multiple copies

The configuration in Figure 29 is basically the same as Figure 28 on page 101 except that this configuration snaps from a BCV source to four virtual devices on each array instead of to one virtual device on each array. This illustrates how a single `symsnap` command can create an image of a source device (a BCV in this case) on four virtual devices simultaneously.

As mentioned previously, by splitting the BCV pairs on both sides of an SRDF configuration at the same time, the BCVs on both sides contain data that is consistent with the R1 source data up to the time of the split. The controlling host can then perform a local snap and a remote snap, resulting in virtual devices on both sides having copies of the source device's data (provided that the BCVs were not written to by their hosts prior to the snap operation). In this configuration, 10 hosts can have access to copies of the source device's data (two copies on BCVs, and eight copies on virtual devices).

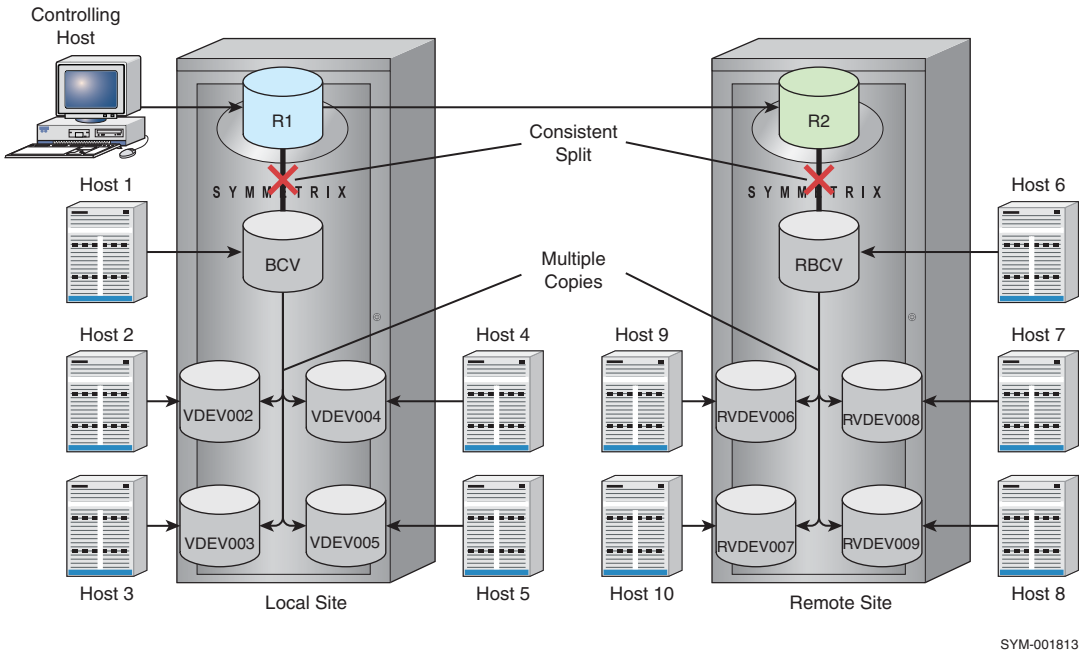

Figure 29 Snapping multiple copies on local and remote arrays

The process for building an RDF1-type device group is similar to the previous section. The difference is that a device group for this configuration contains the R1 standard device, two BCVs, and eight virtual devices. Establishing and splitting both BCV pairs is the same as the previous section.

What is different about this configuration is the ability to snap copies from a single source to multiple virtual devices simultaneously with a single command. Before initiating the snap operation, however, you must wait for completion of the BCV-pair background split involving the BCV that will be the source for the snap.

To snap copies to the four local virtual devices (DEV002 – DEV005) from the local BCV source device (BCV001), issue the symsnap create and the symsnap activate commands:

```
symsnap -g Rdf1Grp create -bcv   BCV001 vdev ld VDEV002
symsnap -g Rdf1Grp create -bcv   BCV001 vdev ld VDEV003
symsnap -g Rdf1Grp create -bcv   BCV001 vdev ld VDEV004
symsnap -g Rdf1Grp create -bcv   BCV001 vdev ld VDEV005

symsnap -g Rdf1Grp activate -bcv   BCV001 vdev ld VDEV002
                                   BCV001 vdev ld VDEV003
                                   BCV001 vdev ld VDEV004
                                   BCV001 vdev ld VDEV005
```

To snap copies to the four remote virtual devices (DEV006 – DEV009) from the remotely associated BCV source device (BCV002), issue the symsnap commands with the -rbcv option:

```
symsnap -g Rdf1Grp create -rbcv    RBCV002 vdev ld RVDEV006
symsnap -g Rdf1Grp create -rbcv    RBCV002 vdev ld RVDEV007
symsnap -g Rdf1Grp create -rbcv    RBCV002 vdev ld RVDEV008
symsnap -g Rdf1Grp create -rbcv    RBCV002 vdev ld RVDEV009

symsnap -g Rdf1Grp activate -rbcv   RBCV002 vdev ld RVDEV006
                                    RBCV002 vdev ld RVDEV007
                                    RBCV002 vdev ld RVDEV008
                                    RBCV002 vdev ld RVDEV009
```

Snapping a copy at the tertiary site of a cascaded SRDF configuration

Using SRDF technology and the TimeFinder-hop2 option, you can snap devices on an array located at the tertiary site of a cascaded SRDF configuration (Figure 30). Performing SYMCLI commands from the controlling host allows the remote virtual device to receive a copy of the data from the R1 device. Remote Host A can access the snap data.

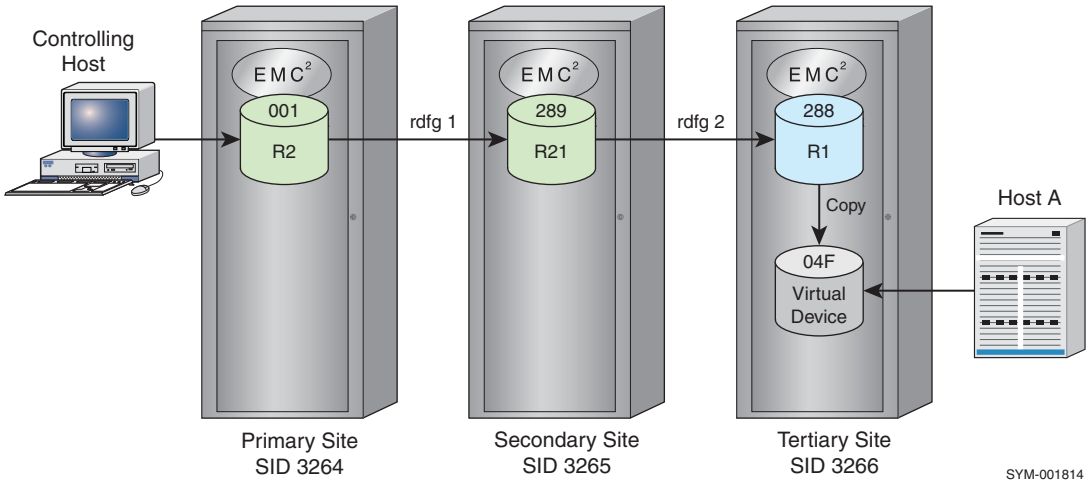

Figure 30 Snapping a copy at the tertiary site of a cascaded SRDF configuration

The following steps outline an example of snapping local device data on an array located at the tertiary site of a cascaded SRDF configuration:

1. Create an RDF2-type device group or composite group (for example, a group named `remotesnaps`):

   To create an RDF2-type device group:

   ```
   symdg create remotesnaps -type rdf2
   ```

   To create an RDF2-type composite group:

   ```
   symcg create remotesnaps -type rdf2
   ```

2. Add devices to the group. From the array at the primary site, add an R2 standard device. From the array at the tertiary site, add a virtual device to hold the snap copy.

   To add devices to a device group:

   ```
   symdg -g remotesnaps -sid 3264 add dev 001
   symdg -g remotesnaps -hop2 -rdfg 1 -remote_rdfg 3 -vdev add dev 04f
   ```

   To add devices to a composite group:

   ```
   symcg -cg remotesnaps -sid 3264 add dev 001
   ```

   ```
   symcg -cg remotesnaps -sid 3264 -hop2 -rdfg 1 -remote_rdfg 3 -vdev
   add dev 04f
   ```

3. Create a snap pair session from the devices in the group:

   ```
   symsnap -g remotesnaps -exact -hop2 create
   ```

4. Activate the snap pair session:

   ```
   symsnap -g remotesnaps -hop2 activate
   ```

5. To query the progress of the remote snap session:

   ```
   symsnap -g remotesnaps query -hop2
   ```

6. To terminate the remote snap session:

   ```
   symsnap -g remotesnaps -hop2 terminate
   ```

Snapping a copy from a clone target device

You can perform snap operations from clone target devices. This feature allows the target device of a clone session to be used as a source device for one or more snap sessions.

The following restrictions apply to snap from clone target devices:

- The original clone session (`symclone`) must be in the Copied or Split state.
- After one or more snap sessions begin using the original clone devices, the only action permitted on the original clone session is `terminate`. All other actions are blocked until all of the snaps are terminated.

  The only actions permitted on a snap session with a clone session target as their source are activate and terminate. All other actions are blocked until the original clone session is terminated.

For example:

```
symclone -g DgName create
symclone -g DgName activate
symsnap -g DgName -bcv create
symsnap -g DgName -bcv activate
symsnap -g DgName -bcv terminate
symclone -g DgName recreate
```

Command options with device groups or composite groups

Options to the `symsnap -g` and `-cg` command line arguments provide more action flexibility to control copy sessions when you are operating on device(s) of a specified device group or composite group. Table 10 lists the `symsnap` control operations and the possible options to use when targeting a specified device group or composite group.

Table 10 symsnap -g and -cg control arguments and possible options (page 1 of 2)

	Argument action									
Options	**create**	**activate**	**duplicate**	**recreate**	**terminate**	**restore**	**query**	**verify**	**attach**	**detach**
-bcv	Y	Y		Y	Y	Y	Y	Y		
-both_sides		Y	Y							
-c, -i	Y	Y		Y	Y	Y	Y	Y	Y	Y
-concurrent	Y	Y	Y	Y				Y		
-consistent		Y	Y							
-copied								Y		
-copyonwrite								Y		
-created								Y		
-duplicate	Y	Y			Y					
-exact	Y									
-force	Y	Y		Y	Y	Y				
-full						Y				
-hop2	Y	Y		Y	Y	Y	Y	Y	Y	Y
-multi							Y			
-noprompt	Y	Y		Y	Y	Y			Y	Y
-not_ready		Y	Y			Y				
-offline							Y	Y		
-preaction, -postaction		Y	Y							
-preservetgtlocks[a]-lockid	Y	Y		Y	Y	Y				
-rbcv	Y	Y		Y	Y	Y	Y	Y	Y	Y
-rdf	Y	Y		Y	Y	Y	Y	Y	Y	Y
-restinprog								Y		

Table 10 symsnap -g and -cg control arguments and possible options (page 2 of 2)

	Argument action									
Options	**create**	**activate**	**duplicate**	**recreate**	**terminate**	**restore**	**query**	**verify**	**attach**	**detach**
-restore							Y			
-restored					Y			Y		
-skip	Y	Y	Y	Y	Y					
-star	Y	Y		Y	Y	Y				
-summary							Y	Y		
-svp	Y									
-symforce					Y					
-v	Y	Y		Y	Y	Y			Y	Y

a. The `-preservetgtlocks` and `-lockid` option is not supported for `-cg` commands.

Note: For command syntax and descriptions of the `symsnap` options, refer to the *EMC Solutions Enabler CLI Command Reference*.

Command options with device files

With the `symsnap -file` command, you can perform snap control operations on device pairs defined in a device file. The device file includes a source device number, a target virtual device number, and the Symmetrix ID. Table 11 lists the `symsnap` control operations and the possible options to use when targeting device pairs specified in a device file.

Table 11 symsnap -file control arguments and possible options

	Argument action									
Options	**create**	**activate**	**duplicate**	**recreate**	**terminate**	**restore**	**query**	**verify**	**attach**	**detach**
-c, -i	Y	Y		Y	Y	Y	Y	Y	Y	Y
-concurrent				Y				Y		
-consistent		Y	Y							
-copied								Y		
-copyonwrite								Y		
-created								Y		
-duplicate	Y	Y			Y					
-force	Y	Y		Y	Y	Y				
-full						Y				
-multi							Y			
-noprompt	Y	Y		Y	Y	Y				

Table 11 symsnap -file control arguments and possible options (continued)

	Argument action									
Options	**create**	**activate**	**duplicate**	**recreate**	**terminate**	**restore**	**query**	**verify**	**attach**	**detach**
-not_ready		Y	Y		Y	Y				
-preaction, -postaction		Y	Y							
-preservetgtlocks, -lockid	Y	Y		Y	Y	Y				
-restinprog								Y		
-restore							Y			
-restored					Y			Y		
-sid	Y	Y			Y	Y	Y	Y	Y	Y
-skip	Y	Y		Y	Y					
-star	Y	Y		Y	Y	Y				
-symforce					Y					
-summary							Y	Y		
-v	Y	Y		Y	Y					

CHAPTER 5
Performing TimeFinder VP Snap Operations

This chapter describes how to perform TimeFinder VP Snap operations.

- TimeFinder VP Snap overview 110
- Creating a VP Snap copy session 110
- Activating a VP Snap session 111
- Restoring a VP Snap session 111
- Terminating a VP Snap copy session 112

TimeFinder VP Snap overview

For a high-level overview of TimeFinder VP Snap functionality, refer to the *EMC VMAX Family Product Guide*.

VP Snap leverages TimeFinder/Clone technology to create space-efficient snaps for thin devices by allowing multiple sessions to share capacity allocations within a thin pool. VP Snap provides the efficiency of Snap technology with improved cache utilization and simplified pool management. With VP Snap, tracks can be stored in the same thin pool as the source, or in another pool of your choice.

VP Snap sessions copy data from the source device to the target device only if triggered by a host I/O. Read I/Os to protected tracks on the target device do not result in data being copied.

Figure 31 shows several VP Snap sessions sharing allocations within a thin pool.

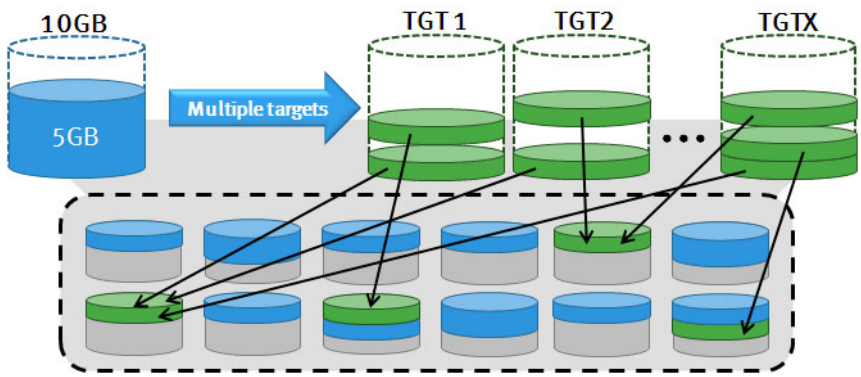

Figure 31 VP Snap sessions sharing allocations within a thin pool

Creating a VP Snap copy session

Example To begin a VP Snap session and define `DEV001` as the source device and `DEV005` as the target device, enter:

```
symclone -g ProdDB create DEV001 sym ld DEV005 -vse
```

The following restrictions apply to VP Snap:

- The -vse attribute may only be applied during the creation of the session.
- Both the source device and the target device must be thin devices.
- All of the VP Snap target devices for a particular source device must be bound to the same thin pool.
- A rebind operation is not allowed for devices that are the target of a VP Snap session.
- Once created, VP Snap sessions cannot be changed to any other mode.
- The source device and the target device must be the same size.
- VP Snap sessions cannot be combined with TimeFinder/Clone nocopy sessions on the same source device.

- If a FAST VP target extent is part of a VP Snap session, the shared tracks cannot be moved between tiers.
- TimeFinder/Snap sessions and VP Snap sessions cannot coexist on a source device.

Activating a VP Snap session

Example To activate the VP Snap session created in "Creating a VP Snap copy session" on page 110, enter:

```
symclone -g ProdDB activate DEV001 sym ld DEV005
```

This activates the copy operation from the source device to the target device. Activating the copy session places the target device in the Read/Write state. The target host can access the copied data and has access to data on the source host until the copy session is terminated.

Note: Copied data is made available as a point-in-time copy at the time of activation and not at the time that the session was created.

Using the establish command

To create and then immediately activate a copy session with a single command, you can use the `symclone establish` command.

Example To create and then activate the copy session shown in the example on page 110, enter:

```
symclone -g ProdDB establish DEV001 sym ld DEV005 -full -vse
```

Note: The `symclone establish` command sets the target device to Not Ready for a short time. If you are using a filesystem, unmount the target host before performing the establish command.

Restoring a VP Snap session

VP Snap supports incremental restore operations back to the original source device. Unlike TimeFinder/Clone restore operations, the source device does not have to be fully copied and the original session between the source device and the target device is maintained. The following restrictions apply:

- Only one restore session is allowed for the source device.
- Because the original session is maintained, a VP Snap restore operation uses two session slots on the source device (one for the original session and one for the restore session).
- If you have concurrent sessions off of a source device, all of the concurrent sessions must be in the Copied or CopyOnWrite state.
- Splitting a clone device pair is not supported for VP Snap restored sessions.

- Starting from the time of the initiation of the incremental restore command until the restore copy is complete, the following restrictions apply:
  - None of the existing sessions may be changed from one mode to another.
  - None of the existing sessions may perform a differential recreate or incremental establish operation.
  - No new sessions may be created with another create or full establish command.

In the case of an incremental restore, the original session copy direction is reversed and changed data is copied from the target device to the source device.

Example To incrementally restore data from the target device (`DEV005`) created in the example on page 110 to the original source device (DEV001), enter:

```
symclone -g ProdDB restore DEV001 sym ld DEV005
```

Note: When constructing a `symclone restore` command, the device receiving the data always appears first in the command, followed by the device from which the data is being copied. Therefore, in the above command, `DEV001` is actually the target of the data being copied from `DEV005`.

Terminating a VP Snap copy session

Terminating a copy session deletes the pairing information in the array and removes any hold on the target device.

VP Snap restore operations maintain the original session, so when a VP Snap session is restored, both the original CopyOnWrite session and the restore session exist. In this case, the restored session must be terminated before the original session. This is done by specifying the `-restored` switch on the terminate command. Once the restored session is terminated, the original session may be terminated with a normal terminate command without the `-restored` switch.

Example To terminate the restore session in the example on page 111 using the `symclone` command, enter:

```
symclone -g ProdDB terminate DEV001 sym ld DEV005 -restored
```

Example To terminate the original session in the example on page 110 using the `symclone` command, enter:

```
symclone -g ProdDB terminate DEV001 sym ld DEV005
```

CHAPTER 6
Performing TimeFinder/Mirror Operations

This chapter describes the business continuance model and explains how to manage and control TimeFinder/Mirror (BCV) devices.

Note: For configurations running Enginuity 5874 through 5876, TimeFinder/Mirror functions are performed through TimeFinder/Clone software using a process called Clone Emulation. Clone Emulation mode makes the use of RAID-protected BCVs transparent to the TimeFinder/Mirror user.

- Clone Emulation mode and TimeFinder/Mirror 114
- TimeFinder operations overview 119
- Listing BCV devices 120
- Associating BCV devices with a device group 121
- Disassociating BCV devices from a device group 125
- Moving BCV devices from one device group to another device group 126
- Managing BCV devices with composite groups 127
- Establishing BCV pairs 128
- Incrementally establishing BCV pairs 140
- Splitting BCV pairs 142
- Fully restoring BCV pairs 153
- Incrementally restoring BCV pairs 156
- Protecting BCV data during full or incremental restores 160
- Cancelling BCV pairs 160
- Querying BCV pairs 161
- Verifying BCV pair states 163
- Using composite groups to manage BCV pairs across arrays 165
- Preferred attachment of BCVs (optional operations) 166
- BCV pair states 172
- Command options with device groups 174
- Command options with composite groups 176
- Command options with device files 178
- Various remote multihop configurations 180

Clone Emulation mode and TimeFinder/Mirror

Clone Emulation mode is a mapping procedure that allows you to use RAID-protected BCV devices in a way that is transparent to TimeFinder/Mirror users. In Clone Emulation mode, Solutions Enabler CLI functions convert TimeFinder/Mirror commands to TimeFinder/Clone commands.

Note: For environments using Enginuity 5874 through 5876, TimeFinder/Mirror uses Clone Emulation mode for all operations. When running Enginuity 5874, any differences in operations that you need to keep in mind will be noted in their respective sections.

"TimeFinder/Clone Emulation" on page 117 contains greater detail.

Note: Data Domain devices are not supported as TimeFinder/Mirror source or target devices.

TimeFinder/Mirror

TimeFinder/Mirror is essentially a business continuance solution that allows the use of special *business continuance volume* (BCV) devices. Copies of data from a standard device (which are online for regular I/O operations from the host) are sent and stored on BCV devices to mirror the primary data. Uses for the BCV copies can include backup, restore, decision support, and applications testing. Each BCV device has its own host address, and is configured as a stand-alone device.

TimeFinder/Mirror involves *associating* and *establishing* the BCV device as a mirror of a specific standard device. As a result, the BCV device becomes inaccessible (*Not Ready* in Figure 32 on page 115) using its original device address while it is in an established pair. Once the BCV device is synchronized with its source, you can separate (*split*) it from the standard device with which it is paired, thereby making it available again to its host for backup or other host processes through its original device address.

After host processing on the BCV device is complete, the BCV may again be mirrored to a standard device (either the same device with which it was previously paired, or with a different device).

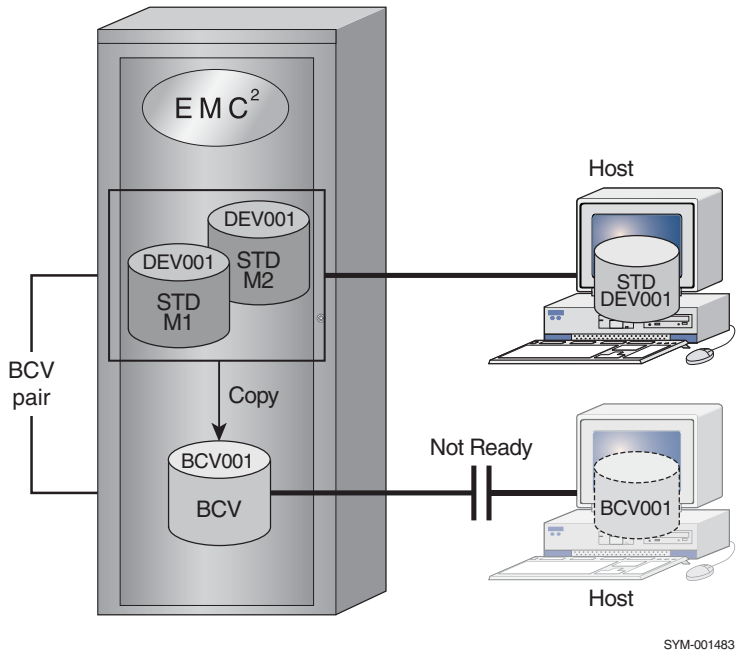

Figure 32 Establishing a BCV pair

A DMX [1] array allows up to four mirrors for each logical volume. The mirror positions are commonly designated M1, M2, M3, and M4. A single BCV can be the second, third, or fourth mirror of the standard device. In Figure 32, because standard device DEV001 is configured with two mirrors, BCV001 functions as the third mirror. A host logically views the M1/M2 mirrored devices as a single device.

SRDF-connected sites

SRDF is a Business Continuance solution that maintains a mirror image of data at the device level in arrays located in physically separate sites. In an SRDF configuration, the individual devices are designated as either a *source* or *target* to synchronize and coordinate SRDF activity.

1. Unless noted otherwise, all references to arrays discussed in the context of TimeFinder/Mirror indicate a DMX running Enginuity 5773 and lower.

Remotely mirroring the local standard

There are multiple types of SRDF-connected BCV devices. An SRDF-connected BCV can be paired with the R1 mirror or R2 mirror of the local RDF standard devices (RBCV) as shown in Figure 33.

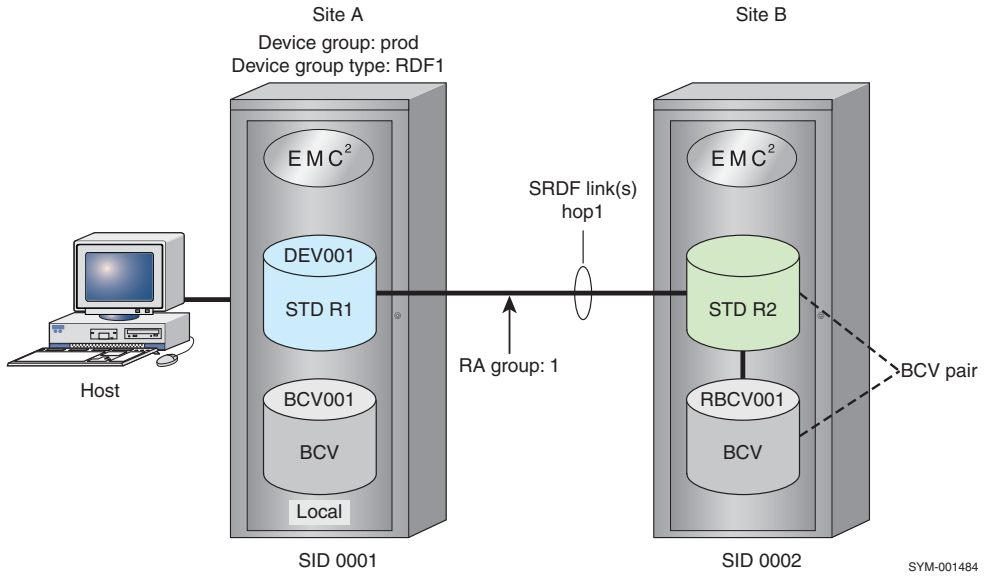

Figure 33 SRDF: Mirroring the local standard

Mirror types

Once a BCV device is established as a mirror of a standard device, those two devices together are referred to as a *BCV pair*. The pair consists of two types of mirrors: the standard device mirror(s) and the BCV mirror.

The standard device mirrors contain copies of the data contained in their associated standard devices. There can be up to three standard device mirrors (M1, M2, M3).

A *BCV mirror* is a standard device mirror. It can be a two-way mirror (M1, M2) that is assigned upon creation of the BCV pair.

Note: Mirroring tasks such as establish, split, and restore use the `symmir` command and are described later in this chapter.

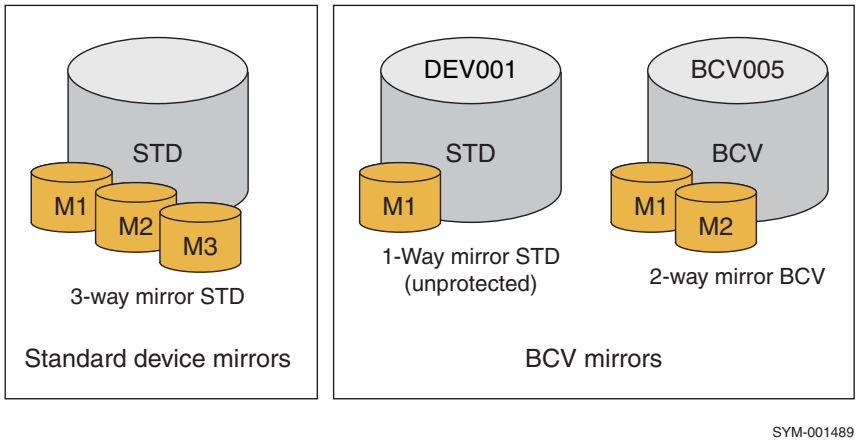

Figure 34 Mirror configuration types

TimeFinder/Clone Emulation

TimeFinder automatically maps a TimeFinder/Mirror command to the executable of the appropriate TimeFinder/Clone command when it encounters a BCV that is a RAID 5 or RAID 6 protected device. Under Clone Emulation mode, TimeFinder/Clone initiates the pre-copying of data.

Note: While Clone Emulation mode is primarily a RAID 5 BCV implementation, it can also be used with any other BCV protection. Clone Emulation is available with the TimeFinder/Clone license and can be used with existing TimeFinder/Mirror scripts.

When you establish a BCV pair under Clone Emulation, the `symmir establish -full` command maps to the `symclone create -precopy -differential` command. This action causes copying to begin while still checking for new writes. The `symmir split` command maps to the `symclone activate` command. This action causes the data to become available to the host as an instant point-in-time copy.

Table 12 details the mapping of TimeFinder/Mirror operations to their TimeFinder/Clone operational equivalents.

Table 12 TimeFinder commands mapped to clone operation

TimeFinder /Mirror symmir operations	TimeFinder/Clone symclone operations
FULL ESTABLISH	CREATE with pre-copy and differential
SPLIT	ACTIVATE or SPLIT
INCREMENTAL ESTABLISH	RECREATE
FULL RESTORE	FULL RESTORE
INCREMENTAL RESTORE	INCREMENTAL RESTORE
VERIFY	VERIFY
ATTACH	ATTACH
DETACH	DETACH
CANCEL	TERMINATE
PROTECT RESTORE	Default feature
PROT BCV ESTAB	Default feature[a]
QUERY	QUERY
List	List

a. Only after the completion of split is the target device fully synchronized as in a protected BCV Establish.

To operate in a mixed BCV set of RAID 5 BCVs and non-RAID 5 BCVs, you must set the Clone Emulation environment variable to ENABLED:

```
SYMCLI_CLONE_EMULATION=ENABLED
```

In a *mixed* BCV set, if Clone Emulation is *disabled* (the default), any control operation produces an error when a RAID 5 BCV is encountered.

When in Clone Emulation mode, a standard device can be paired with as many as eight concurrent BCVs (RAID 5 or any other BCV protection). Issue the `symmir establish -concurrent` command for the same standard device up to eight times, adding one additional BCV each time.

The following limitations apply to TimeFinder/Clone Emulation:

- The TimeFinder reverse split feature is not allowed.
- Restores will always be protected restores.
- Incremental Restore (Reverse Re-Snap) will only be accepted if all tracks were originally copied from the source prior to the restore taking place.

Note: With Clone Emulation mode, an incremental restore will only be accepted if the devices are in a Split state and there is no active split.

- The following option file settings will be ignored:

  SYMAPI_DEFAULT_BCV_SPLIT_TYPE, SYMAPI_DEFAULT_BCV_RESTORE_TYPE SYMAPI_DEFAULT_BCV_ESTABLISH_TYPE

- The maximum number of BCVs that can be incrementally established with a standard device will be eight instead of the 16 allowed by TimeFinder. SYMCLI_MAX_BCV_PAIRS can at most be eight.
- The BCV states Split-Before-Sync and Split-Before-Restore are not valid states for an emulation device. In both cases, a forced split will complete the synchronization operation.

TimeFinder operations overview

These management operations can safely be performed with SYMCLI on a copy of an actively changing set of the business data. Business Continuance management operations include backing up a static copy of a database, or preparing for application upgrades.

Device external locks

SYMCLI/SYMAPI uses *device external locks* in the array to lock device pairs during SRDF and TimeFinder control operations. The *EMC Solutions Enabler Array Management CLI User Guide* provides more information on external locks.

Skip locks

During TimeFinder/Mirror operations, you can choose to bypass the device external locks on standard devices by using the `-skip` option with the `symmir establish` and `split` commands. If the specified source devices are either all locked or all unlocked, this option will explicitly not lock those source devices.

Preserve target locks

Device external locks are used to lock BCV device pairs and devices participating in a copy session during TimeFinder BCV, Snap, and Clone operations. For target devices that have been previously locked with the same lock holder ID, you can preserve the original lock for use in performing additional TimeFinder control operations.

Use the `-preservetgtlocks` and `-lockid` options with the `symmir establish`, `restore`, and `split` commands to preserve the original device lock on target devices. You must specify the original lock holder ID number. This option causes the operation to not take out additional locks for the specified target devices.

Disallow synchronization actions

For some sites, it may be desirable to block users on a specific host from performing either an establish or restore operation on any of the devices. The SYMAPI_SYNC_DIRECTION parameter in the options file allows you to confine TimeFinder and SRDF operations to either just establish or restore actions. In this way, you can block a user on a host from executing a restore or an establish action using the following form:

```
SYMAPI_SYNC_DIRECTION=ESTABLISH | RESTORE | BOTH
```

where:

`ESTABLISH` confines the possible operations to just establish actions.

`RESTORE` confines the possible operations to just restore actions, which includes (allows) restore, failback, and R1 update actions.

`BOTH` (default) does not restrict any TimeFinder or SRDF actions.

Note: The *EMC Solutions Enabler CLI Command Reference* provides more details about the options file.

Wait for synchronization actions to complete

By default, the `symmir establish` command initiates a series of tasks that begins the synchronization of one or more BCV pairs. SYMCLI returns control to the caller while the establish operation is still in progress. The WAIT_FOR_BCV_SYNC parameter in the options file enables you to delay returning control to the caller until the establish operation (or a restore operation) is complete:

```
SYMAPI_WAIT_FOR_BCV_SYNC = TRUE | FALSE
```

where:

`TRUE` waits for the operation to complete before returning control to the caller.

`FALSE` (default) returns control to the caller before the operation completes.

Note: The *EMC Solutions Enabler CLI Command Reference* provides more details about the options file.

Listing BCV devices

Configuration and status information is stored in the configuration database file for each device on every array, including BCV devices.

You can find all BCV devices on an array and view their physical and device names. In addition, you can display details about the BCV devices, including the BCV device name, the device name of the paired standard device, the number of invalid tracks for both the BCV device and the standard device, and the BCV pair state.

Examples

To list all the BCV devices that are visible to the host, enter:

```
symbcv list pd
```

To list all the BCV devices, regardless of whether they are visible to the host, enter:

```
symbcv list dev
```

To list all the BCV devices that have SCSI reservations, regardless of whether they are visible to the host, enter:

```
symbcv -resv list
```

To list all the BCV sessions created on an array, enter:

```
symmir -sid SymmID list
```

To list all the BCV sessions created on array 3264:

```
symmir list -sid 3264

Symmetrix ID: 000000003264

  Standard Device       BCV Device              State
-------------------- ----------------------- --------------
           Invalid             Invalid   GBE
Sym        Tracks    Sym       Tracks          STD <=> BCV
-------------------- ----------------------- --------------
002B               0 0E0B            0   ... Synchronized
002E               0 0E00            0   ..X Synchronized
002E               0 0E0A            0   ... Synchronized
0032               0 0E0F            0   ... Split
00FF               0 00FD            0   ... Split
0DF5               0 0DA5            0   ..X Synchronized
0DF5               0 0DA4            0   ..X Synchronized
0F70               0 001B         3592   X.. SyncInProg
0F71               0 001C         4496   X.. SyncInProg
0F93               0 0DF9            0   ..X Split
1015               0 1069            0   X.. Synchronized

Total       --------          --------
  Tracks           0              8088
  MB(s)          0.0             505.5

Legend:

(G): X = The paired BCV device is associated with a group.
     . = The paired BCV device is not associated with a group.
(B): X = The paired BCV device is splitting in the background.
     . = The paired BCV device is not splitting in the background.
(E): X = The paired BCV device is emulation mode.
     . = The paired BCV device is not emulation mode.
```

Associating BCV devices with a device group

Various compound (even multihop) remote configurations can be managed by your host using various SYMCLI control components. To perform operations on a BCV device using the SYMCLI, the BCV device must be associated with an existing device group or composite group (this is not a requirement when using a device file).

Note: For information on associating BCV devices with a composite group, refer to "Associating BCV devices with a composite group" on page 128.

When you associate a BCV device with a device group, you can assign it a logical device name. If you do not assign the BCV device a logical device name, a logical device name will be assigned automatically.

Note: Mirroring tasks such as establish, split, and restore use the `symmir` command and are described later in this chapter.

Examples

You can associate BCV devices with a device group by using either the physical device name, or the device name. To associate a BCV device with a physical device name of `/dev/dsk/c0t2d0s2`, to a device group named `prod`, and naming the BCV device `BCV7`, enter:

```
symbcv -g prod associate pd c0t2d0 BCV7
```

To associate a BCV device, with a device name of `001F`, to a device group named `prod`, naming the BCV device `BCV5`, enter:

```
symbcv -g prod associate dev 001F BCV5
```

Moving device groups

By default, a BCV device cannot be associated with more than one device group at the same time when you are using one SYMCLI configuration database file. However, you can change this default behavior by enabling the SYMAPI_ALLOW_DEV_IN_MULT_GRPS parameter in the `options` file.

You can associate all BCV devices with a device group. Only BCV devices that are not already associated with a device group will be associated.

You can either associate all BCV devices that are visible to your host within a device group (the default), or are configured in an array.

To associate all BCV devices on array `123` with a device group `prod`, enter:

```
symbcv -g prod -sid 123 associateall dev
```

Host-visible BCV devices

To associate all BCV devices visible to your host with the device group `prod`, enter:

```
symbcv -g prod associateall
```

You can also associate a range of BCV devices that are visible to your host. For example, to associate devices `039A` through `039F` with the device group `prod`, enter:

```
symbcv -g prod associateall -devs 039A:039F
```

You can associate remote BCV devices with a device group. The following options allow all remote BCV devices of a specific type to be associated with a device group:

- `-rdf` specifies remote attached BCVs (RBCVs).
- `-bcv` specifies remote BCVs that mirror local BCVs (BRBCVs). This option must be used with the `-rdf` option.
- `-rrdf` specifies BCVs that are remotely associated with remote BCVs (RRBCVs).
- `-hop2` specifies BCV devices (2BCVs) that are remotely associated on the second hop of a cascaded SRDF configuration.

SRDF-connected BCV device

An SRDF-connected BCV device must be associated with a device group before it can be paired with a remote standard device. Figure 35 on page 124 shows a device group of type RDF1 (there are three other device group types, RDF2, RDF21, and REGULAR, which are not shown).

If the group is an RDF1 type, then the remote BCVs (RBCV) can only be paired with the R2 mirrors on the remote array. If the group is an RDF2 type, then the remote BCVs can only be paired with the R1 mirrors on the remote array. When dealing with concurrent RDF devices, you can only remotely associate with one RDF group.

To associate an SRDF-connected BCV device to a device group named `prod`, and assign it a logical device name of `RBCV001`, enter:

```
symbcv -sid 123 -g prod -rdf -RDFG 1 associate dev 000B
```

where:

`123` is the ID of the local array.

`-rdf` specifies a remote attachment.

`-RDFG 1` specifies the RDF group number (or RA group number) `1` at the local array through which the remote BCV is reached.

`dev 000B` specifies the device name of the BCV on the remote array.

In this example, the remote BCV pair is mirroring the local standard device as shown in Figure 35 on page 124.

SRDF-connected BCV mirror device

You can also associate a remote BCV pair that mirrors a local BCV device as shown in Figure 35 on page 124. For example:

```
symbcv -sid 123 -g prod -rdf -bcv -RDFG 2 associate dev 002A
```

where:

`123` is the ID of the local array.

`-rdf` specifies a remote attachment.

`-bcv` specifies that it is a remote BCV that mirrors the local BCV.

`-RDFG 2` specifies an RDF group number `2`, through which the remote BCV is reached.

Compounded remote configuration

Figure 35 shows how devices in a compounded remote configuration can be associated.

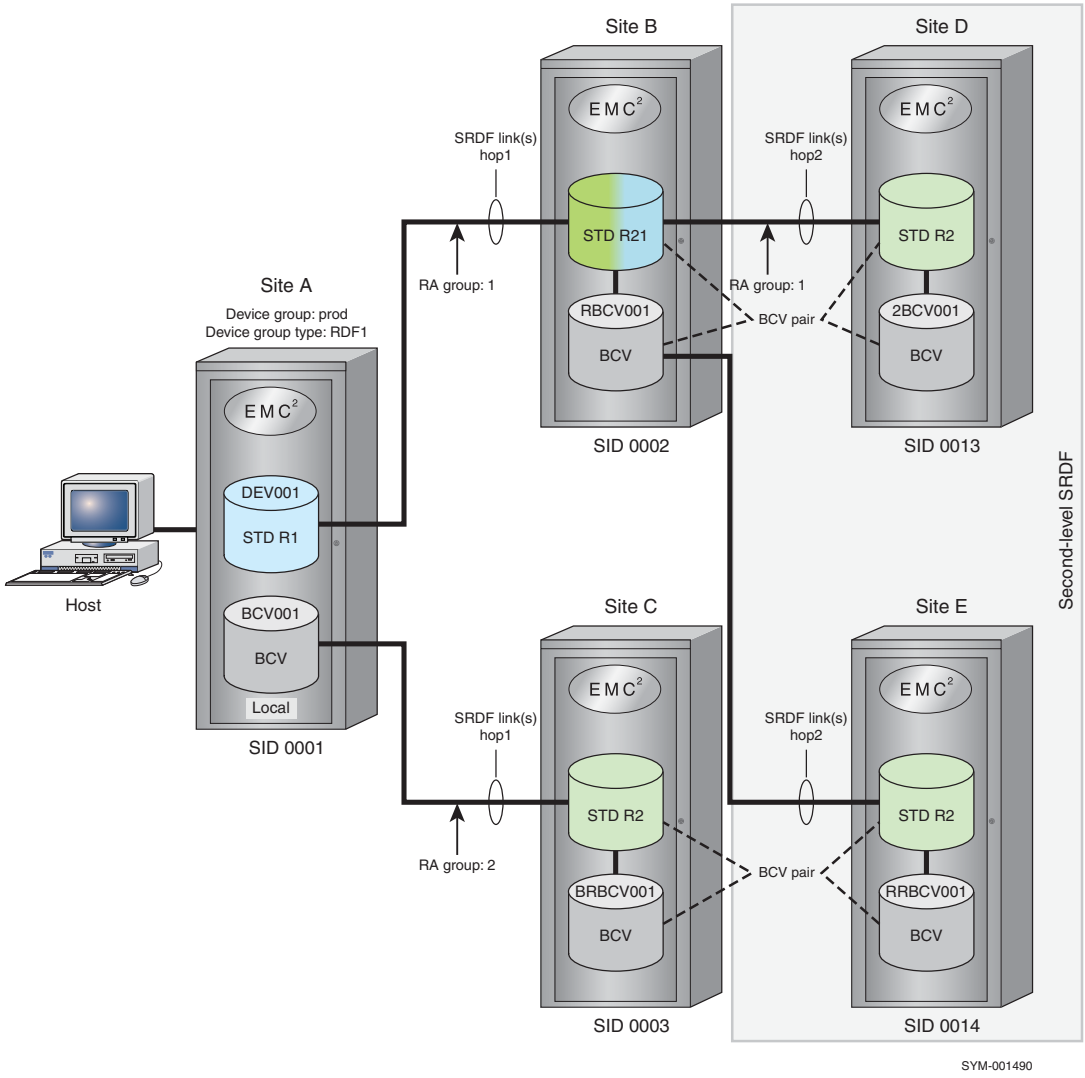

Figure 35 Compounded remote configuration example

- At site A, the SRDF links remotely mirror a local array.
- The remote site B functions as the remote mirror to the standard devices at site A.
- Remote site C (third site) uses an SRDF link to remotely mirror the BCV devices in the array at site A.
- The second-level SRDF shows how:
  - SRDF can be cascaded where remote site D (the Tertiary site) functions as a remote partner of the R21 device, which is the remote partner of the local RDF standard device.
  - SRDF can include multiple sites, for example where remote site E functions as a remote mirror to the standard devices of site A.

Disassociating BCV devices from a device group

Once a BCV device has been associated with a device group, you can disassociate it when the BCV device is in a state that allows it to be disassociated.

You can disassociate a BCV device by using either the physical device name, the device name, or the logical device name.

To disassociate a BCV device named `BCV7` from the `prod` device group, enter:

```
symbcv -g prod disassociate ld BCV7
```

To disassociate a BCV device on device `000F` from the `prod` device group, enter:

```
symbcv -g prod disassociate dev 000F
```

If the BCV device that you want to disassociate from a device group is currently paired via TimeFinder, TimeFinder/Snap, or TimeFinder/Clone, the `-force` option must be used.

Note: When you use the `-force` option, SYMCLI does not access the array (an offline operation). It disassociates the device from the device group without access to the array.

SRDF-connected BCV pair

An SRDF-connected BCV device can be disassociated from a device group.

You can disassociate a BCV device using either the device name or the logical device name.

To disassociate a remote BCV device that has a logical device name of `RBCV001` from a device group named `SAMPLE1`, enter:

```
symbcv -g SAMPLE1 -rdf disassociate ld RBCV001
```

To disassociate a remote BCV device that has a device name of `002B` from a device group named `SAMPLE1`, enter:

```
symbcv -g SAMPLE1 -rdf disassociate dev 002B
```

If you are attempting to disassociate a remote BCV device from a device group, and the BCV device is in the synchronized, restored, or transient BCV state, you must use the `-force` option.

Note: When you use the `-force` option, SYMCLI does not access the array (an offline operation). It disassociates the device from the device group without access to the array.

Remote SRDF-connected BCV pair

You can disassociate a remote BCV pair that mirrors the local BCV device, which is shown associated in Figure 35 on page 124.

For example:

```
symbcv -sid 123 -g SAMPLE1 -rdf -bcv disassociate dev 002A
```

where:

`123` is the ID of the local array.

`-rdf` specifies a remote attachment.

`-bcv` specifies that it is a remote BCV that mirrors the local BCV.

Remote BCV or a remote BCV pair

You can disassociate a remote BCV that mirrors the remote BCV device, which is shown associated in Figure 35 on page 124. This option can be used to disassociate a BCV device that is accessible via SRDF links two hops away. The group must be an RDF group.

For example:

```
symbcv -sid 123 -g SAMPLE1 -rrdf disassociate dev 002A
```

where:

`123` is the ID of the local array.

`-rrdf` specifies the BCV is being remotely disassociated with a remote BCV in the group.

Remote BCV on the second hop of a cascaded SRDF configuration

You can disassociate a remote BCV on the second hop of a cascaded SRDF configuration, which is shown associated in Figure 35 on page 124. This option can be used to disassociate a BCV device that is accessible via SRDF links two hops away. The group must be an RDF group.

For example:

```
symbcv -sid 123 -g SAMPLE1 -hop2 disassociate dev 002A
```

where:

`123` is the ID of the local array.

`-hop2` specifies the BCV is in the second hop of a cascaded SRDF.

Moving BCV devices from one device group to another device group

The `symbcv` command can be used to move one or all BCV devices from one existing device group to another existing device group. The source and destination groups can be of different types. When moving a BCV device from one group to another, you can choose to have the device's logical name renamed to the default naming convention of the destination group. This helps to avoid naming conflicts that may be encountered.

Other options are available to limit a move to the devices that meet a specified set of criteria, which can be specified along with the `moveall` action. For a full description of these options, refer to `symbcv` in the *EMC Solutions Enabler CLI Command Reference*.

Moving a specific device

To move BCV device `BCV001` from device group `prod` to the destination group `NewGroup` and rename the moved device, enter:

```
symbcv -g prod move ld BCV001 NewGroup -rename
```

Moving all BCV devices

You can move all local BCV devices from one device group to another device group. The source and destination groups can be of different types.

To move all BCV devices that are visible to your host from device group `prod` to device group `NewGroup`, enter:

```
symbcv -g prod moveall NewGroup
```

Moving remote BCV devices

Remote BCV devices can be moved from one device group to another device group. The following options allow all remote BCV devices of a specific type to be moved:

- `-rdf` specifies remote attached BCVs (RBCVs).
- `-bcv` specifies remote BCVs that mirror local BCVs (BRBCVs). This option must be used with the `-rdf` option.
- `-rrdf` specifies BCVs that are remotely associated with remote BCVs (RRBCVs).
- `-hop2` specifies BCV devices (2BCVs) that are remotely associated on the second hop of a cascaded SRDF configuration.

Removing devices

The `symbcv` command also contains a remove all action (`rmall`), which will remove all devices meeting the specified criteria from the specified device group.

Managing BCV devices with composite groups

When adding BCV devices to a composite group, consider the following:

- The user must specify a local array ID.
- If no devices have been added to the composite group yet, and if there is only one RDFG on the array, then that RDFG is assumed; otherwise, you must specify a RDFG when adding remote BCVs.
- The `-host`, `-SA`, and `-P` parameters are only valid when associating local BCV devices.
- If `-bcv` is specified, the BCV list affected is the BRBCV list (or the BCV of the remote BCV device).

  Note: `-rdf` is required when `-bcv` is specified.

- If `-rdf` is specified and `-bcv` is not specified, the BCV list affected is the RBCV list.
- If `-rrdf` is specified, the BCV list affected is the RRBCV list.

- If `-hop2` and `-remote_rdfg` are specified, the BCV list affected is the 2BCV list.
- If `-rdf` and `-bcv` and `-rrdf` are not specified, the BCV list affected is the BCV list.

Associating BCV devices with a composite group

Use the following syntax to associate a BCV using the given device name to the composite group:

```
symbcv -cg CgName -sid SymmID [[-rdf [-bcv]] | [-rrdf] | [-hop2]]
   [-rdfg GrpNum [-remote_rdfg RemoteGrpNum]]
   associate dev SymDevName [LdevName]
```

Use the following syntax to associate all BCV devices for the *SymmID* to the composite group:

```
symbcv -cg CgName -sid SymmID [[-rdf [-bcv]] | [-rrdf] | [-hop2]]
   [-rdfg GrpNum [-remote_rdfg RemoteGrpNum]]
   associateall [pd | -host HostName] [-sid SymmID]
          [-SA # | ALL] [-p #] [-N #]
          [-cap # [-captype mb | cyl]]
          [-R1 | -NOR1] [-R2 | -NOR2]
          [-sel_rdfg SelRdfGrpNum]
          [-devs <SymDevStart:SymDevEnd | SymDevName
              [,<SymDevStart:SymDevEnd | SymDevName>...]>]
```

Disassociating BCV devices from a composite group

Use the following syntax to disassociate a BCV using the device name:

```
symbcv -cg CgName -sid SymmID [[-rdf [-bcv]] | [-rrdf] | [-hop2]]
   [-rdfg GrpNum [-remote_rdfg RemoteGrpNum]]
   disassociate dev SymDevName [-force]
```

Moving BCV devices to a composite group

Use the following syntax to move the BCV that has the given device name to the destination composite group:

```
symbcv -cg CgName -sid SymmID [[-rdf [-bcv]] | [-rrdf] | [-hop2]]
   [-rdfg GrpNum [-remote_rdfg RemoteGrpNum]]
   move dev SymDevName DestCgName [-force]
```

Establishing BCV pairs

After configuration and initialization of an array, BCV devices contain no data. The BCV devices, like the standard devices, have unique host addresses and are online and ready to the hosts to which they are connected. The full establish must be used the first time the standard devices are paired with BCV devices.

Figure 36 on page 129 illustrates the initial configuration prior to performing any TimeFinder BCV operations. In this figure, the host views the M1/M2 mirrored pair as a single standard device (DEV001) and the BCV device as BCV001.

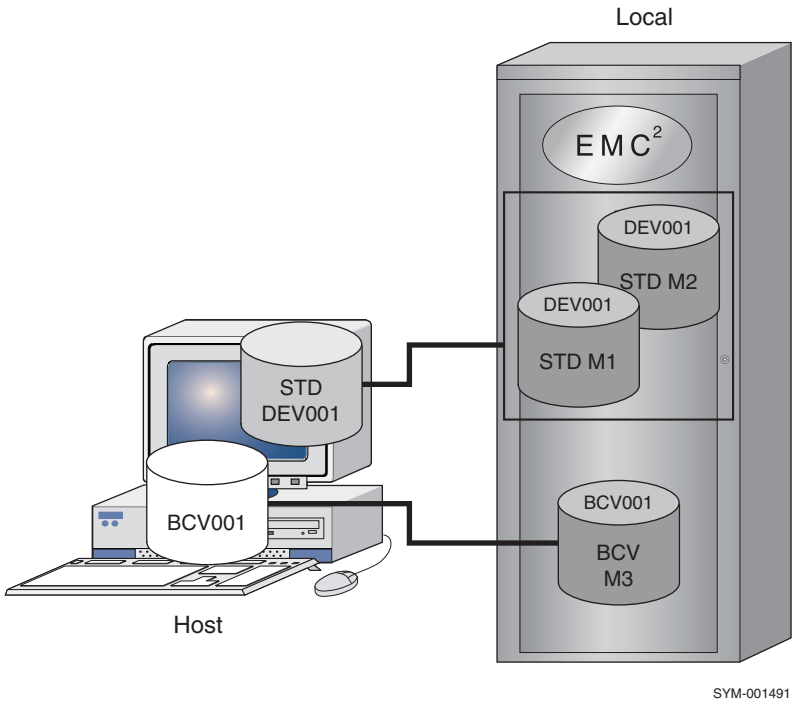

Figure 36 Initial BCV configuration

To obtain a copy of the data on a standard device, a BCV pair must be established. A BCV pair consists of a BCV device and a standard device. The standard device can have various mirror structures (unprotected, two-way or three-way mirrored, RAID, RAID with SRDF), as long as the number of mirrors does not exceed three. This constraint is in place because establishing a BCV pair requires assigning the BCV device as the next available mirror of the standard device.

Since there is a maximum of four mirrors allowed per device in the array, a device already having four mirrors is not able to accommodate another one.

Optionally, you can target devices in a device group, composite group, or device file:

```
symmir -g DgName -full establish
symmir -cg CgName -full establish
symmir -f[ile] FileName -full establish
```

To initiate a full establish on all the BCV pairs in the `prod` device group, enter:

```
symmir -g prod -full establish
```

To initiate a full establish on one BCV pair, `DEV001`, in the `prod` device group, enter:

```
symmir -g prod -full establish DEV001
```

To initiate a full establish on more than one BCV pair (list) in the `prod` device group with one command, enter:

```
symmir -g prod -full establish
          DEV001 BCV ld BCV005
          DEV002 BCV ld BCV007
          DEV003 BCV ld BCV008
```

Figure 37 illustrates the full establish of a BCV pair.

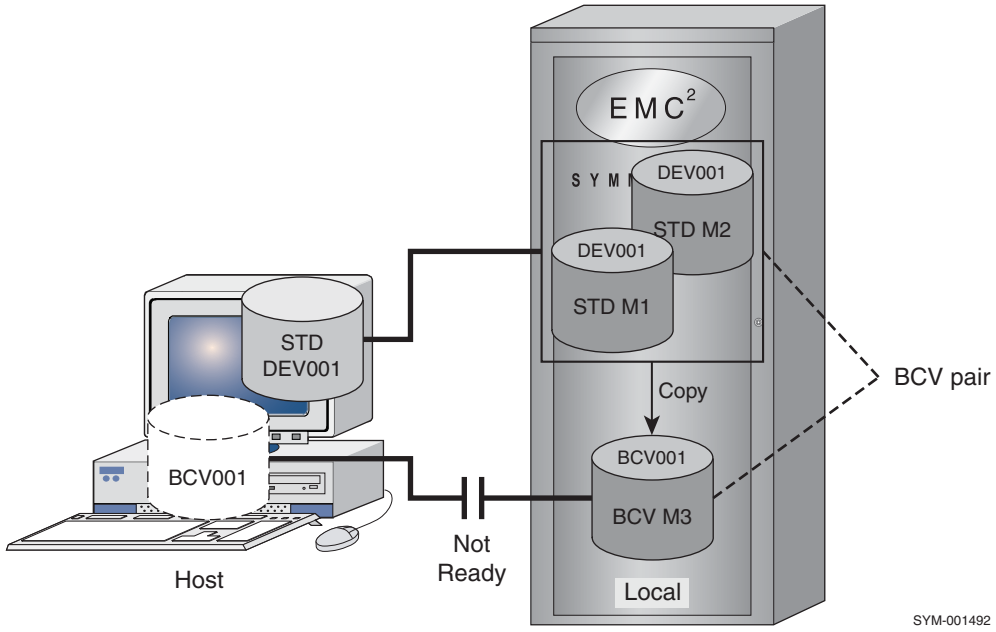

Figure 37 Fully establishing a BCV pair

When a full establish is initiated for each specified BCV pair in a device group:

- Command validity is checked. For example, the array makes sure that both the standard device and the BCV device are the same size, the device specified as the BCV has the BCV attribute, the standard device does not already have a BCV device assigned to it, and so on.

  If the standard device is a metahead device, the BCV must also share the same metadevice properties. All metamembers will be implicitly established along with the metahead device.

- The BCV device is set as Not Ready to the host.
- The BCV device is assigned as the next available mirror of the standard device. A BCV can be the second, third, or fourth mirror of the standard device. For example, in Figure 37, it is the third mirror (M3).
- The contents of the standard device are copied to the BCV. For example, in Figure 37, the BCV device receives its data from both the first fully valid mirror of the source.

The BCV pair is synchronized when the standard device mirrors and the BCV mirror contain identical data.

Note: The BCV device is not available for host use during the time that it is assigned as a BCV mirror on a standard device. However, any new data written to the standard device is copied to the BCV device while the BCV pair exists.

By default, Solutions Enabler rejects establish commands for CKD online devices. You can allow the establish command for CKD online devices by disabling the SYMAPI_TF_CHECK_ONLINE_CKD option. For instructions on disabling or enabling this option, refer to the *EMC Solutions Enabler Installation Guide*.

Specifying the default method for establishing BCV pairs

When specifying the default method for establishing BCV pairs, you can either set it at the system level by using an options parameter, or at the user level by using an environment variable.

Note: User level settings (environment variables) will override system level settings (options file parameters).

Note: Because of the load that establish operations place on an array, you should only change these settings under the direction of EMC. Please contact your EMC representative before changing these settings.

Specifying at the system level

To specify the default method for establishing BCV pairs at the system level, set the SYMAPI_DEFAULT_BCV_ESTABLISH_TYPE parameter in the `options` file. Possible values are:

- SINGULAR specifies to issue an establish to a single device, including a metamember, at a time. This method allows other tasks access to the array when doing a large number of establishes.
- PARALLEL (default) specifies to issue an establish to each servicing disk adapter (DA) in parallel, and then wait for a DA to finish before issuing another establish to that DA.
- SERIAL specifies to issue establishes as fast as the Global Special Task (GST) queue can handle them. However, all members of a meta must be established before continuing to the next meta or device. This is the default method when using metadevices.

Note: The *EMC Solutions Enabler CLI Command Reference* contains information on changing the `options` file parameters.

Specifying at the user level

To specify the default method for establishing BCV pairs at the user level, set the SYMCLI_BCV_EST_TYPE environment variable. Possible values for this variable are SINGULAR, PARALLEL, or SERIAL, as described earlier on this page.

When specifying the default method as SINGULAR or PARALLEL, you can also set the SYMCLI_BCV_EST_DELAY environment variable to specify how long to wait between sending commands to the array. Possible value for this variable range from 0 to 30, with 0 being the default setting.

Note: The *EMC Solutions Enabler CLI Command Reference* contains information on changing environment variables.

Instantly establishing multiple BCV pairs

The multi/instant establish option improves the performance of a typical establish operation by submitting multiple BCV pairs in a single system call to be established instantly.

You can enable (default)/disable this feature with the SYMAPI_TF_MULTI_ESTAB_REST environment variable. Setting the SYMAPI_DEFAULT_BCV_ESTABLISH_TYPE parameter to SERIAL or SINGULAR will cause this option to be ignored.

Establishing multiple BCVs with a single standard device

You can fully establish (at different times) up to 16 BCV devices (8 when using emulation mode) associated with a single standard device. The BCV devices must also be associated to the same device group.

Note: Using the environment variable SYMCLI_MAX_BCV_PAIRS, the maximum number of pairs (established or restored) can be adjusted between 1 to 16 BCV devices.

With this feature, standard devices retain a relationship with multiple BCVs as long as those BCVs do not become paired with another standard. Here, the information about changed tracks is saved for a split BCV device when another BCV device is subsequently established and split from the same standard device. By invoking a series of `split/-full establish` commands over time (as shown in Figure 38), a multi-BCV environment becomes established that retains progressive historical images of the specified standard.

Note: When the maximum number of multi-BCV pairs is reached, you can alter the BCV pair *cancel policy* that controls the round-robin device usage as you fully establish the next device beyond the set maximum pair count. Using environment variable SYMCLI_BCV_PAIR_POLICY, you can cancel the incremental relationship between the STD and the oldest BCV (default), cancel the newest, or don't cancel any BCV in the set.

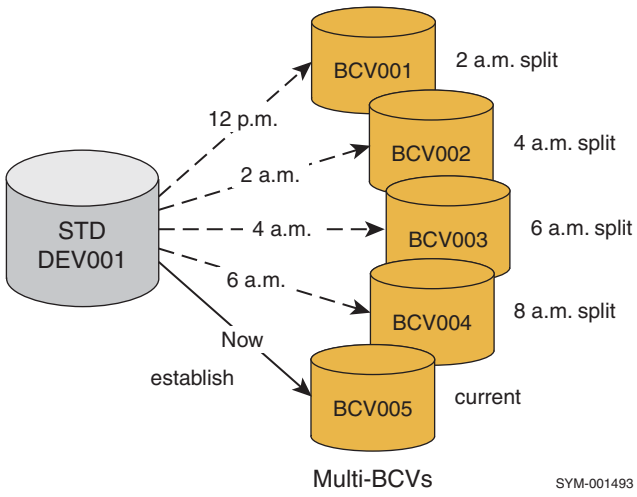

Figure 38 Establishing a multi-BCV environment

To fully establish the standard with `BCV005`, enter:

```
symmir -g prod split DEV001
symmir -g prod -full establish DEV001 BCV ld BCV005
```

In this environment, you can specify any one of these older BCVs to incrementally restore the standard back to a historical copy.

Canceling a multi-BCV relationship

You can completely cancel the incremental relationship between the STD and any one of the split multi-BCV devices from the set using the `cancel` operation. This operation will put the BCV in either of the following states, depending on whether you are running TimeFinder in native or emulation mode:

Table 13 BCV state

TimeFinder mode	BCV state
Native	SplitNoInc
Emulation	Never Establish

You cannot incrementally establish or incrementally restore that BCV pair again until you have performed another full establish operation or a full restore operation.

To remove `BCV001` from the set, enter:

```
symmir -g prod cancel DEV001 BCV ld BCV001
```

As shown in Figure 39, once you cancel the specified BCV, all records of track changes between the STD and the canceled BCV are destroyed.

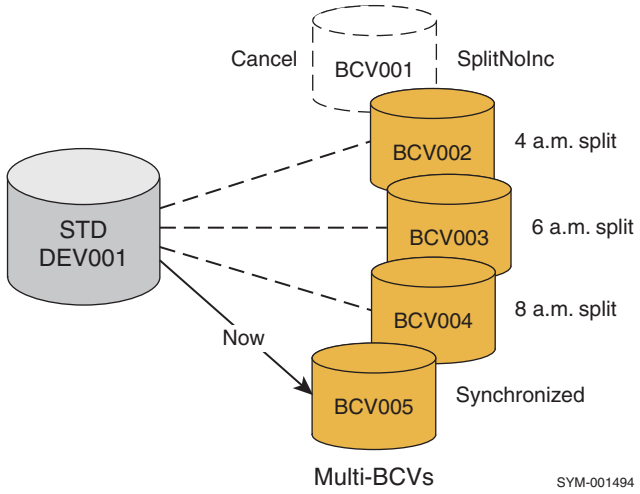

Figure 39 Canceling a multi-BCV

SRDF-connected BCV pairs

You can specify an `establish` action to a remote array using the RDF flag (`-rdf`), which fully establishes the remote BCV pairs.

To perform a full establish operation in the remote array at site B when the RDF flag is specified, use the following command:

```
symmir -g prod -rdf -full establish
```

To initiate a full establish on one remote BCV pair, `DEV001`, in the `prod` group, enter:

```
symmir -g prod -rdf -full establish DEV001 bcv ld RBCV001
```

In this case, the flag indicates that the BCV device being established is an SRDF-connected BCV device, which will be established with the remote mirror of the local RDF standard device.

To perform a full establish operation in the remote array at site B when the RDF and BCV flags (`-rdf` and `-bcv`) are specified, use the following command:

```
symmir -g prod -rdf -bcv -full establish
```

To initiate a full establish on one remote BCV pair, `BCV001`, in the `prod` group, enter:

```
symmir -g prod -rdf -bcv -full establish BCV001 BCV ld BRBCV001
```

In this case, the `-rdf` parameter indicates that the BCV device being established is an SRDF-connected BCV device, which will be established with the remote standard mirror of the local R1 BCV device.

Second-level remote BCV pairs

You can specify an `establish` action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which fully establishes second-level remote BCV pairs. To perform a full establish operation in the remote array at site C when the remotely attached remote BCV flag (`-rrbcv` option) is specified, use the following command:

```
symmir -g prod -rrbcv -full establish
```

To initiate a full establish on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv -full establish RBCV001 BCV ld RRBCV001
```

In this case, the flag indicates that the BCV device being established is a second hop SRDF-connected BCV device, which will be established with the remote standard mirror of the remote BCV device.

Hop 2 BCV pairs in a cascaded SRDF configuration

You can specify an `establish` action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform a full establish operation in the remote array at the tertiary site (C) when the hop 2 flag is specified, use the following command:

```
symmir -g prod -hop2 establish -full
```

In this case, the flag indicates that the SRDF-connected BCV device (2BCV001) is being established with the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Using cascaded Clone Emulation to cascaded clone

In environments running Enginuity 5874 through 5876, the target device of a Clone Emulation session can be used as the source for one or more clone sessions and vice versa:

- Clone session to a Clone Emulation session
- Clone Emulation session to a clone session

In environments running Enginuity 5875 and higher, performing an incremental restore to a cascaded clone target is supported. For example, devices in an A -› B -› C cascaded clone session can copy data from device C to device A.

Clone session to a Clone Emulation session

In this configuration, the source device A to target device B is a TimeFinder/Clone session, and source device B to target device C (BCV) is a TimeFinder/Clone Emulation session.

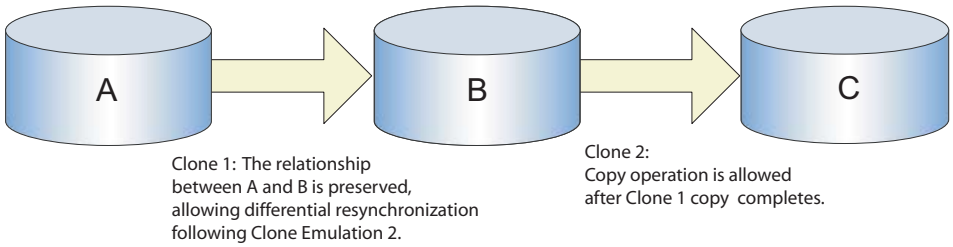

Figure 40 Cascaded Clone to a cascaded Clone Emulation session

Table 14 lists the Clone-to-Clone Emulation target session states.

Note: When A→B session is SyncInProgress or RestoreInProgress, the `-symforce` flag is required when performing a B→C split operation.

Table 14 Clone and B to C TimeFinder/Clone Emulation states

B→C session state	Clone A → Clone B → Clone C session state						
	A→B No session	A→B Created, Recreated	A→B Precopy	A→B CopyInProg CopyOnAccess CopyOnWrite	A→B Copied Split	A→B RestoreInProg	A→B Restored
B→C No session	Create A→B Full Establish A→B Full Restore A→B Full Establish B→C	Activate A→B Set Mode A→B	Activate A→B Set Mode A→B	Recreate A→B Establish A→B Set Mode A→B	Recreate A→B Establish A→B Restore A→B Set Mode A→B	Full Establish B→C	Split A→B Full Establish B→C
B→C Synchronized	Create A→B Full Establish A→B Full Restore A→B Split B→C	Activate A→B Set Mode A→B	Activate A→B Set Mode A→B	Recreate A→B Establish A→B Set Mode A→B	Recreate A→B Establish A→B Restore A→B Set Mode A→B Split B→C	Split B→C	Split A→B Split B→C
B→C SyncInProgress	Full Restore A→B	Not proper state	Not proper state	Not proper state	Not proper state	Not proper state	Split A→B
B→C Split (BG Split In Progress)	Full Restore A→B Establish B→C	Activate A→B Set Mode A→B (no precopy)	Not proper state	Set Mode A→B (no precopy)	Restore A→B Set Mode A→B (no precopy)	Establish B→C	Split A→B Establish B→C
B→C Split (BG split done)	Create A→B Full Establish A→B Full Restore A→B Establish B→C Restore B→C	Activate A→B Set Mode A→B	Activate A→B Set Mode A→B	Recreate A→B Establish A→B Set Mode A→B	Recreate A→B Establish A→B Restore A→B Set Mode A→B	Establish B→C	Split A→B Establish B→C
B→C RestoreInProg		Not proper state	Not proper state	Not proper state	Not proper state	Not proper state	Not proper state
B→C Restored	Full Restore A→B Split B→C	Not proper state	Not proper state	Not proper state	Not proper state	Split B→C	Split B→C

Clone Emulation session to a clone target session

In this configuration, the source device A to target device B (BCV) is a TimeFinder/Clone Emulation session, and source device B to target device C is a TimeFinder/Clone session. Table 15 lists the Clone Emulation to Clone target session states.

Note: When A→B session SyncInProgress or RestoreInProgress, `-symforce` flag is required on A→B split.

Table 15 Clone Emulation and clone target session states

B→C session state	Clone A → Clone B → Clone C session state					
	A→B No session	A→B Synchronized SyncInProgress	A→B Split (BG Split In Progress)	A→B Split (BG split done)	A→B RestoreInProg	A→B Restored
B→C No session	Full Establish A→B Create B→C Full Establish B→C Full Restore B→C	Split A→B	Establish A→B	Establish A→B Restore A→B Create B→C Full Establish B→C		Split A→B
B→C Created Recreated	Activate B→C Set Mode B→C	Not proper state	Not proper state	Activate B→C Set Mode B→C	Not proper state	Not proper state
B→C Precopy	Activate B→C Set Mode B→C	Not proper state	Not proper state	Activate B→C Set Mode B→C	Not proper state	Not proper state
B→C CopyInProg CopyOnAccess CopyOnWrite	Recreate B→C Establish B→C Set Mode B→C	Not proper state	Not proper state	Recreate B→C Establish B→C Set Mode B→C	Not proper state	Not proper state
B→C Copied Split	Recreate B→C Establish B→C Restore B→C Set Mode B→C	Not proper state	Not proper state	Recreate B→C Establish B→C Set Mode B→C Incr. Restore B→C	Not proper state	Not proper state
B→C RestoreInProg		Not proper state	Not proper state	Not proper state	Not proper state	Not proper state
B→C Restored	Split B→C	Not proper state	Not proper state	Split B→C	Not proper state	Not proper state

Establishing concurrent BCV pairs

When you establish a BCV device as a mirror of a standard device, that relationship is known as a BCV pair. When you sequentially establish/split/establish a number of BCV devices over time with a specified standard, that is known as a multi-BCV relationship.

You can establish two BCV devices (eight when using emulation mode) as concurrent mirrors of a single standard device all within the same `symmir` command line. This relationship is known as a *concurrent* BCV pair. This feature allows you or an application script to instantly generate two synchronized copies of the standard data. When the two BCVs are split from the standard, the BCV's hosts can access the data on either BCV.

When establishing concurrent BCV pairs, you can either specify the BCVs, or use the `-concurrent` option to allow the array to select suitable (in terms of size, emulation, etc.) BCV(s) from the available BCV list.

Example: Specifying the BCVs

To concurrently establish `BCV001` and `BCV002` with standard `DEV012` in device group `CncGrp`, enter:

```
symmir -g CncGrp establish -full DEV012 bcv ld BCV001
                                DEV012 bcv ld BCV002
```

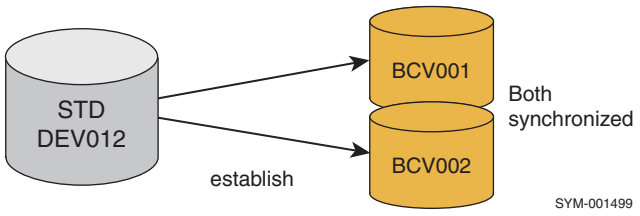

Figure 41 Establish concurrent BCV pairs

After the concurrent BCVs become split at some point, you can then concurrently resynchronize the BCVs with an incremental establish:

```
symmir -g CncGrp establish DEV012 bcv ld BCV001
                           DEV012 bcv ld BCV002
```

While these examples pair both BCVs at the same time, you can also establish the first BCV device now and the second one later. In either case, the concurrent BCVs become synchronized to the standard and remain that way until you split the BCVs from the standard.

The following is a valid concurrent BCV pair example, provided there is no split action between these commands:

```
symmir -g CncGrp establish -full DEV012 bcv ld BCV001
          .
          .
          .
symmir -g CncGrp establish -full DEV012 bcv ld BCV002
```

Example To instruct the array to select suitable BCV(s) to concurrently establish with standard `DEV012` in device group `CncGrp`, enter:

```
symmir -g CncGrp establish -full DEV012 -concurrent
```

In this case, if the standard is already synchronized with one BCV, the array will synchronize one other BCV with it. If the standard device is not yet synchronized with a BCV, the array will still only synchronize one BCV with it.

Performing a protected BCV establish (moving all mirrors)

The protected BCV establish (`-protbcvest`) option (also called the moveall establish) can be used with the establish action to move all mirrors of locally mirrored BCV devices to join the mirrors of the standard device. This protected BCV moving mirror feature improves the availability of the BCV mirrors, particularly after a split operation where all BCV mirrors are instantly synchronized. This feature, also known as instant BCV mirror sync, is available with full or incremental establish or restore operations. This feature is only useful in a native TimeFinder environment.

Note: This feature is not supported if the standard device is a dynamic concurrent SRDF device or if TimeFinder is in emulation mode.

Figure 42 compares the array behavior between a normal and a moving protected establish action.

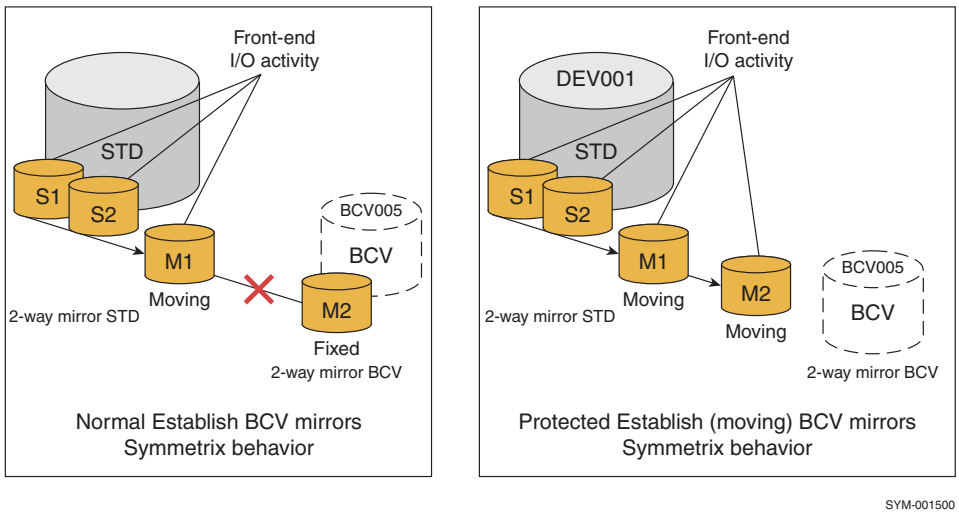

Figure 42 Establishing two-way BCV mirrors with protected establish

In a 2-way BCV mirror configuration for a normal establish, M2 is fixed and can only be updated from M1 after a split. For a 2-way BCV mirror device for a protected establish, both M1 and M2 move to the standard device mirror set and become instantly synchronized and available for updates from I/O activity on the standard device.

To initiate a full `-protbcvest` establish on a 2-way BCV pair with 2-way standard `DEV001` in device group `Prod`, enter:

```
symmir -g Prod establish -full -protbcvest DEV001 BCV ld BCV005
```

For more information about the affects of `-protbcvest` during a split operation, refer to "Splitting a protectively established BCV" on page 148.

Incrementally establishing BCV pairs

Incrementally establishing a BCV pair accomplishes the same thing as the establish process, with a major time-saving exception: the standard device (DEV001) copies to the BCV device (BCV001) only the new data that was updated on the standard device while the BCV pair was split. Any changed tracks on the BCV are also overwritten by the data on the corresponding tracks on the standard device.

This process is useful if the data yielded from running an application on the BCV device is not needed or if a fresh copy of current data is needed.

Optionally, you can target devices in a device group, composite group, or device file:

```
symmir -g DgName establish
symmir -cg CgName establish
symmir -f[ile] FileName establish
```

Examples

To initiate an incremental establish on all the BCV pairs in the `prod` device group, enter:

```
symmir -g prod establish
```

To initiate an incremental establish on a BCV pair, `DEV001`, in the `prod` device group, enter:

```
symmir -g prod establish DEV001
```

To initiate an incremental establish on a list of specific BCV pairs in the `prod` device group, enter:

```
symmir -g prod establish DEV001
                         DEV002
                         DEV005
```

The establish defaults to re-establishing the previous BCV pairing, unless you use either the `-full` option.

Automatically converting an incremental establish to a full establish

TimeFinder will automatically convert an incremental establish to a full establish when it determines that the requested devices (local or remote) have no prior relationship.

The SYMAPI_DEFAULT_BCV_ESTAB_INC_TO_FULL parameter in the options file allows you to enable this feature. Disabling this feature (default) will cause the SYMAPI to return the error SYMAPI_C_DEVICE_IS_NOT_PAIRED, as it did with previous versions.

Note: Enabling the SYMAPI_DEFAULT_BCV_ESTAB_INC_TO_FULL parameter does not eliminate the requirement to use the `-full` option with the `-opt` option.

Incrementally establishing multiple BCV pairs

You can incrementally establish (at different times) up to 16 (8 when using emulation) BCV devices associated with a single standard device. Note that initially, all the BCV devices must have been fully established before you perform any incremental establishes on them.

Note: Using the environment variable SYMCLI_MAX_BCV_PAIRS, the maximum number of pairs (established or restored) can be adjusted between 1 to 16 BCV devices.

With this feature, standard devices retain a relationship with multiple BCVs as long as those BCVs do not become paired with another standard. Here, the information about changed tracks is saved for a split BCV device if another BCV device is subsequently incrementally established and split from the same standard device. By invoking a series of `split/increment establish` commands over time (Figure 43), a multi-BCV environment becomes established that retains progressive historical images of the data on the specified standard.

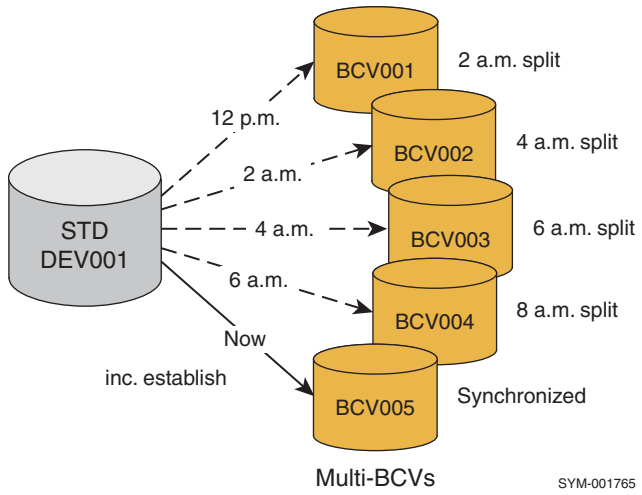

Figure 43 Establishing a multi-BCV environment

To query for existing multi-BCVs that were originally all previously (full) established, and then to incrementally establish `BCV005` with the standard, enter:

```
symmir -g DgName query -multi
symmir -g DgName split DEV001
symmir -g DgName establish DEV001 BCV ld BCV005
```

In this environment, you can specify any one of these older BCVs to incrementally restore or establish the standard back to a historical copy.

SRDF-connected BCV pairs

You can specify an incremental establish action to a remote site using the RDF flag (`-rdf` option), which incrementally establishes the remote BCV pairs.

To perform an incremental establish operation in the remote array at site B when the `-rdf` option is specified, use the following command:

```
symmir -g prod -rdf -establish
```

In this case, the `-rdf` option indicates that the BCV device being established is an SRDF-connected BCV device, which will be established with the remote standard mirror of the local RDF standard device.

Second-level remote BCV pairs

You can specify an incremental establish action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which incrementally establishes second-level remote BCV pairs. An incremental establish operation in the remote array at site C when the remotely attached remote BCV flag (`-rrbcv` option) is specified with the following command:

```
symmir -g prod -rrbcv establish
```

To initiate an incremental establish on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv establish RBCV001 BCV ld RRBCV001
```

In this case, the flag indicates that the BCV device being established is a second Hop SRDF-connected BCV device, which will be established with the remote standard mirror of the remote BCV device.

Hop 2 BCV pairs in a cascaded SRDF configuration

You can specify an incremental establish action to an array located at the tertiary site of a cascaded SRDF configuration.

Perform an incremental establish operation in the remote at the tertiary site (C) when the hop 2 flag is specified with the following command:

```
symmir -g prod -hop2 establish
```

In this case, the flag indicates that the SRDF-connected BCV device is being established with the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Protected BCV incremental establish

The protected BCV establish (`-protbcvest`) option (also known as the moveall establish) can be used with the incremental establish action to move all mirrors of locally mirrored BCV devices to join the mirrors of the standard device. This moving of the mirrors feature improves availability of the BCV mirrors, particularly after a split operation where all mirrors are instantly synchronized.

To initiate a protected (`-protbcvest`) incremental establish on a 2-way BCV pair with 2-way standard `DEV001` in device group `Prod`, enter:

```
symmir -g Prod establish -protbcvest DEV001 BCV ld BCV005
```

For more information about the protected BCV established environment, refer to "Performing a protected BCV establish (moving all mirrors)" on page 139.

Splitting BCV pairs

After an establish operation and the standard device and BCV mirrors are synchronized, the BCV device becomes a mirror copy of the standard device. You can split the paired devices to where each holds separate valid copies of the data, but will no longer remain synchronized to changes when they occur. SYMCLI processes can then be executed with the BCV device once the split completes.

Optionally, you can target devices in a device group, composite group, or device file:

```
symmir -g DgName split
symmir -cg CgName split
symmir -f[ile] FileName split
```

Examples To split all the BCV pairs in the `prod` device group, enter:

```
symmir -g prod split
```

To split a BCV pair, `DEV001`, in the `prod` device group, enter:

```
symmir -g prod split DEV001
```

To split a list of standard devices in the `prod` device group, enter:

```
symmir -g prod split DEV001 DEV002 DEV003
```

Figure 44 illustrates the splitting of a BCV pair.

Note: Relabeling devices is sometimes required in situations where devices are under an MS Windows type volume manager's control. If a BCV (TimeFinder) device holds an identical copy of its standard (paired) device, and when the BCV device becomes ready to the Windows operating system, the volume manager will detect two identical volumes with different mount points. This can cause the volume manager on Windows to exit and bring down a system.

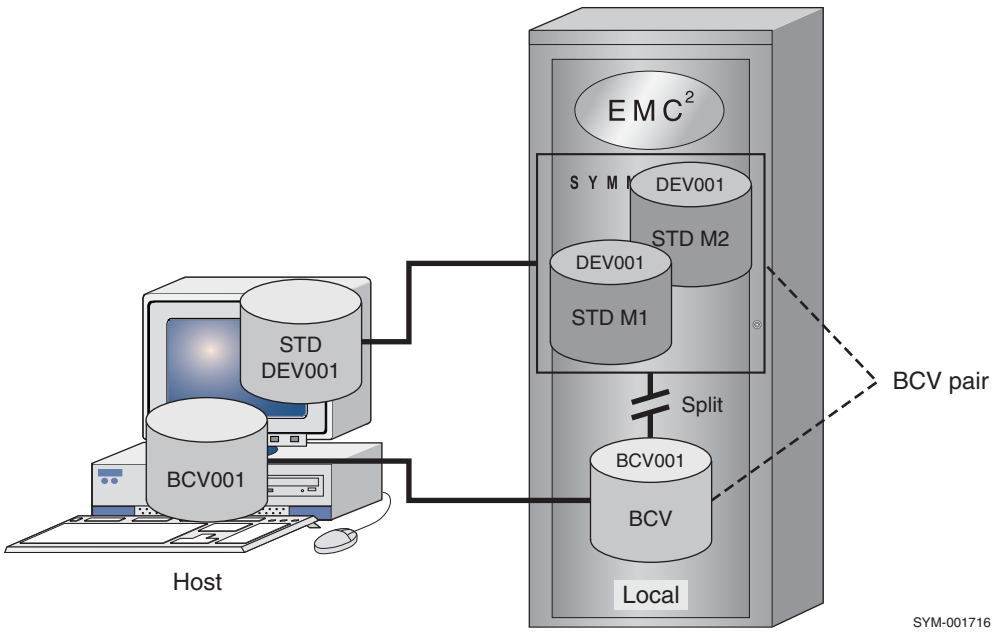

Figure 44 Split the BCV pair

When a *split* is initiated for each specified BCV pair in a device group, the following occurs:

- Command validity is checked. For example, the array makes sure that the standard device has an active BCV mirror and that the standard and BCV devices comprise a BCV pair.

- Any pending write transactions to the standard device and the BCV device are destaged.
- The BCV device is split from the BCV pair.
- If the device is a metadevice, all metamembers are implicitly split as well.
- The BCV device state is changed to Ready, enabling host access through its separate address (`BCV001`).
- Operation with the standard device is resumed and any tracks changed from write operations to the standard device are marked. (This is necessary for updating the BCV device if it is reestablished with the same standard device at a later time.)
- If the BCV device has any of its own mirrors, the mirrors are synchronized, unless protected establish or emulation is used.

Once you finish running any Business Continuance processes on the BCV device, the following options are available:

- Incremental establish or reestablish of the BCV pair.
- Establish new using the same BCV devices with a different standard device.
- Restore data to a standard device from the BCV device.
- Incrementally restore data to the standard device from the BCV device (if the devices were previously paired).

SRDF-connected BCV pairs

You can specify a `split` action to a remote site using the RDF flag (`-rdf` option), which splits the remote BCV pairs.

To perform the splitting of a remote BCV pair, use the following command:

```
symmir -g prod -rdf split
```

In this case, the flag indicates that the BCV device being split is an SRDF-connected device, which will be split from the remote standard mirror of the local RDF standard device.

If a BCV device has its own mirrors (local or remote), these mirrors will become synchronized with its first mirror after the BCV pair is split.

Second-level remote BCV pairs

You can specify a `split` action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which splits second-level remote BCV pairs. To perform a split operation in the remote array at site C when the remotely attached remote BCV flag (`-rrbcv` option) is specified, use the following command:

```
symmir -g prod -rrbcv split
```

To initiate a split on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv split RBCV001 BCV 1d RRBCV001
```

In this case, the flag indicates that the BCV device being split is a second HOP SRDF-connected BCV device, which will be split from the remote standard mirror of the remote BCV device.

Hop 2 BCV pairs in a cascaded SRDF configuration

You can specify a `split` action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform an establish operation in the remote array at the tertiary site (C) when the hop 2 flag is specified, use the following command:

```
symmir -g prod -hop2 split
```

In this case, the flag indicates that the SRDF-connected BCV device (2BCV001) is being split from the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Remote copy with BCV split

In addition to splitting a local BCV pair, you can further specify the remote (`-remote`) option which makes the R1 device ready on the link and propagates the R1 BCV copy across the SRDF link to the R2 that mirrors the BCV:

```
symmir -g prod split DEV001 -remote
```

Performing a reverse split

The reverse (`-reverse`) option initiates a reverse data copy from the fixed BCV mirror to the primary (moving) mirror of the BCV upon the completion of the split operation.

Note: A reverse split is not supported in TimeFinder/Clone Emulation mode.

Normal split behavior

Normally, when a BCV has two mirrors, only the primary mirror (M1) joins the standard device in establish or restore operations. As shown in Figure 45, the content of the primary BCV mirror is refreshed by data from the standard, when the BCV is established. The secondary BCV mirror (M2) is refreshed by data from the primary BCV mirror (M1), after the BCV mirror is split from the standard. The primary BCV mirror is referred to as the *moving mirror*, because it moves between the standard and the secondary (fixed) mirror.

Usually, after a split, the fixed BCV mirror is refreshed from the moving BCV mirror. This can either be a full copy operation or a differential copy. In a differential copy, only the tracks that have changed on the moving mirror during the time it was synchronized with the standard are refreshed.

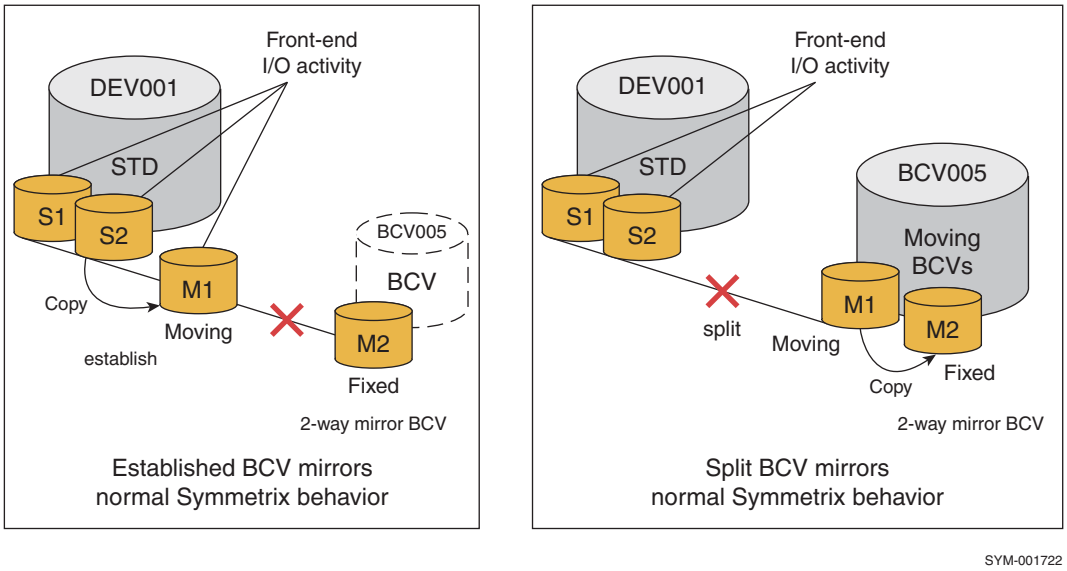

Figure 45 Two-way mirror BCV establish/split normal behavior

Reverse split behavior

In a reverse split operation, the direction of data flow between the BCV mirrors is reversed. As shown in Figure 46, during a reverse split, the fixed BCV mirror (M2) will refresh the moving mirror (M1) after the split operation. This behavior may be desirable when you need to revert to an older copy of the data that was on the BCV before it was established/restored with the standard.

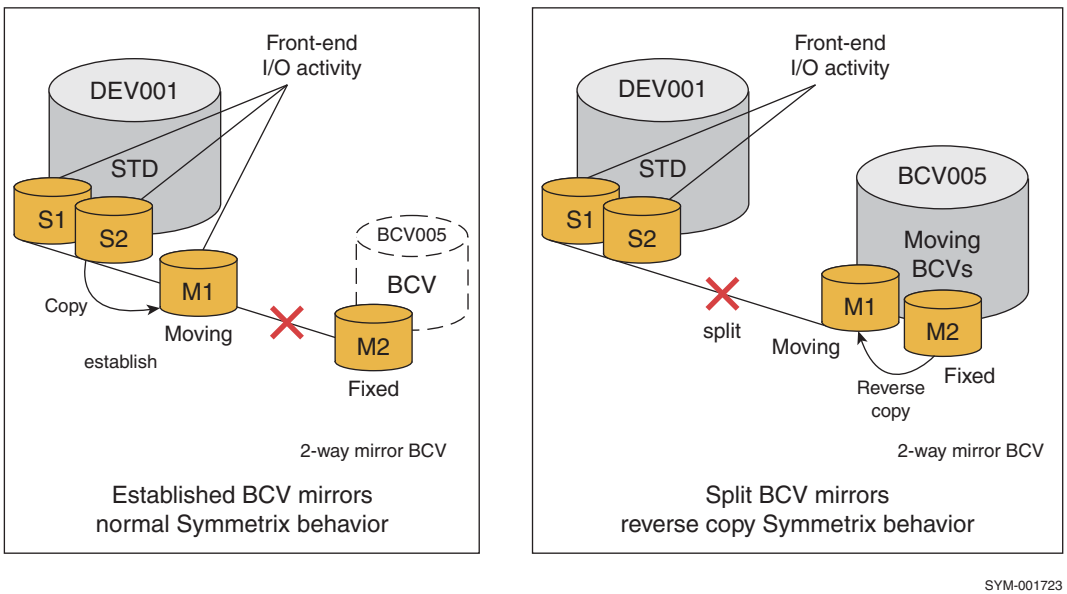

Figure 46 Two-way mirror reverse establish/split behavior

Note: Be sure this is the behavior you want before invoking the reverse split option since the primary BCV mirror data is refreshed with an older mirror of data.

A reverse split is permitted only if both BCV mirrors were completely synchronized before the moving BCV mirror was paired with the standard device. When you anticipate a need for future reverse split operations, the `-reverse` option is applied to an establish or restore operation. This option requests a verification check that the BCV's fixed mirror has valid data. You must verify that both mirrors are in the Ready state after the split.

To establish `DEV001`/`BCV005` and later perform a reverse split on `DEV001` in device group `Prod`, enter:

```
symmir -g Prod establish -reverse DEV001
symmir -g Prod split -reverse DEV001
symmir -g Prod verify -bcv_mirrors -ready
```

Note: Data may not be immediately available. Use the `verify` command to check that both mirrors are in the Ready state for the data to be available.

Figure 47 illustrates a practical use of a reverse split. At midnight a split results in a good point-in-time copy of a database. At 10 a.m. a corruption is discovered in the database, necessitating a database recovery. At 10:30 a.m. a restore operation is initiated from the BCV copy. Because the good data is immediately available to the BCV pair, the recovery begins shortly after initiating the restore process. At 11:00 a.m, during the recovery, one of the logs re-corrupts the database.

Though the data on the BCV's moving mirror has changed during the recovery process, a reverse split can be initiated. At 11:20 a.m. the BCV's fixed mirror refreshes its moving mirror, providing access to the good midnight copy of the data.

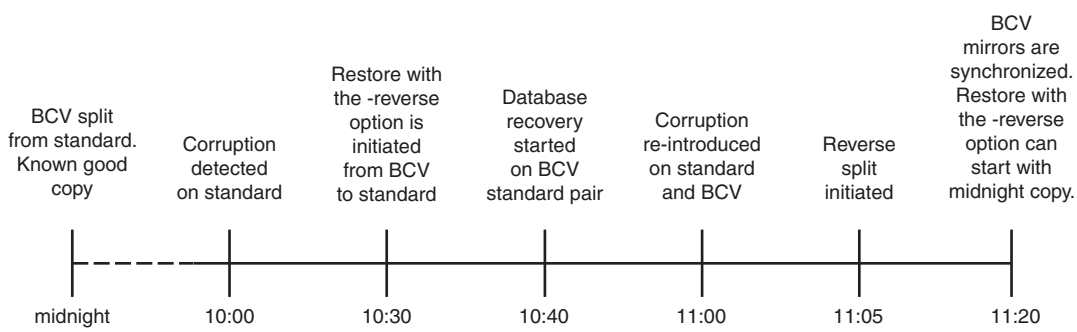

Figure 47 Practical use of a reverse split

Splitting concurrent BCV pairs

You can establish two BCV devices as concurrent mirrors of a single standard device. This relationship is known as a *concurrent* BCV pair. This feature allows you or an application to instantly generate two synchronized copies of the standard data (refer to "Establishing concurrent BCV pairs" on page 138).

When you apply a split action to a standard device that was concurrently established with two BCV mirrors, both BCVs become split from the standard.

To split concurrently established `BCV001` and `BCV002` with standard `DEV012` in device group `CncGrp`, enter:

```
symmir -g CncGrp split DEV012
```

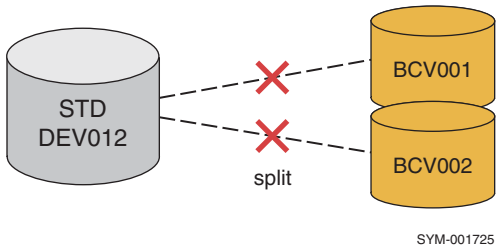

Figure 48 Splitting concurrent BCV pairs

Concurrent BCVs

If you do not want both of the concurrent BCVs to simultaneously split together, you can individually target the split action by explicitly specifying the BCV with the standard as follows:

```
symmir -g CncGrp split DEV012 bcv ld BCV001
```

or:

```
symmir -g CncGrp split DEV012 bcv ld BCV002
```

Concurrent splits

To display the status of the background concurrent split for both of these BCVs, enter:

```
symmir -g CncGrp query -multi -bg
```

Because invalid track tables are maintained, future concurrent incremental establish operations are possible on these split BCVs. After a concurrent split, it is possible to resynchronize just one BCV as follows:

```
symmir -g CncGrp establish DEV012
```

However, if you do not explicitly specify a BCV and you have not set an attachment preference, TimeFinder chooses the BCV to resynchronize, based on which of the two concurrent BCVs was split first.

You can also verify that the action has completed, as follows:

```
symmir -g CncGrp -split verify -bg DEV012 BCV ld BCV002
```

Splitting a protectively established BCV

A split action splits both of the BCV mirrors away from the standard device that were previously established with a `-protbcvest` option. Figure 49 illustrates the initial protected BCV established state and the resulting behavior of a split action on these BCV mirrors. For any split command, there is no need to apply the `-protbcvest` option to move all the mirrors away from the standard.

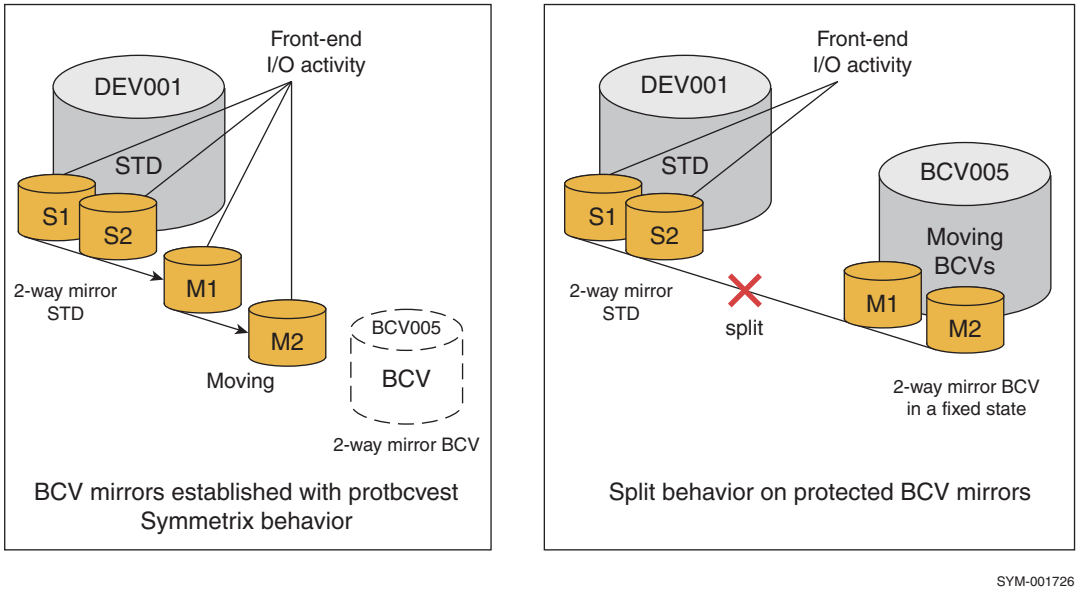

Figure 49 Split behavior on two-way BCV mirrors

Before you split, you may need to query the array to examine the protected STD/BCV mirrored environment to identify the established moved devices for the split action, as follows:

```
symmir -g prod query -protbcvest
```

To perform a protected establish and later split (for example) on standard `DEV001` with its BCV mirrors, in device group `prod`, enter:

```
symmir -g prod establish -protbcvest DEV001
   .
   .
   .
symmir -g prod split DEV001
```

For more information about a protected BCV establish, refer to "Performing a protected BCV establish (moving all mirrors)" on page 139.

TimeFinder consistent split

TimeFinder consistent split allows you to split off a consistent, restartable copy of a database management system within seconds with no interruption to online service. Consistency split helps to avoid inconsistencies and restart problems that can occur when splitting a database-related BCV without first quiescing the database. Consistent split can be implemented using Enginuity Consistency Assist (ECA) functionality or SRDF/A.

"Consistent split using Enginuity Consistency Assist" on page 150 contains greater detail. The *EMC Solutions Enabler SRDF Family CLI User Guide* provides complete details on SRDF/A.

Note: With Enginuity 5874 through 5876, TimeFinder includes the TimeFinder/Consistency Group (TimeFinder/CG) option. You do not need a separate license for TimeFinder/CG. For Enginuity 5875 and higher, the *EMC Solutions Enabler Version 8.0 Installation Guide* provides all of the licensing information.

Consistent split using Enginuity Consistency Assist

You can use Enginuity Consistency Assist (ECA) to perform consistent split operations across multiple, heterogeneous hosts.

Note: With Enginuity 5874 through 5876, TimeFinder includes the TimeFinder/CG option. You do not need a separate license for TimeFinder/CG.

Using ECA to consistently split BCV devices from the standards, you must have either a control host with no database or a database host with a dedicated channel to the gatekeeper devices. The dedicated channel cannot be used for servicing other devices to freeze I/O.

Figure 50 depicts how a control host can perform ECA consistent splits for three database hosts that access devices on an array.

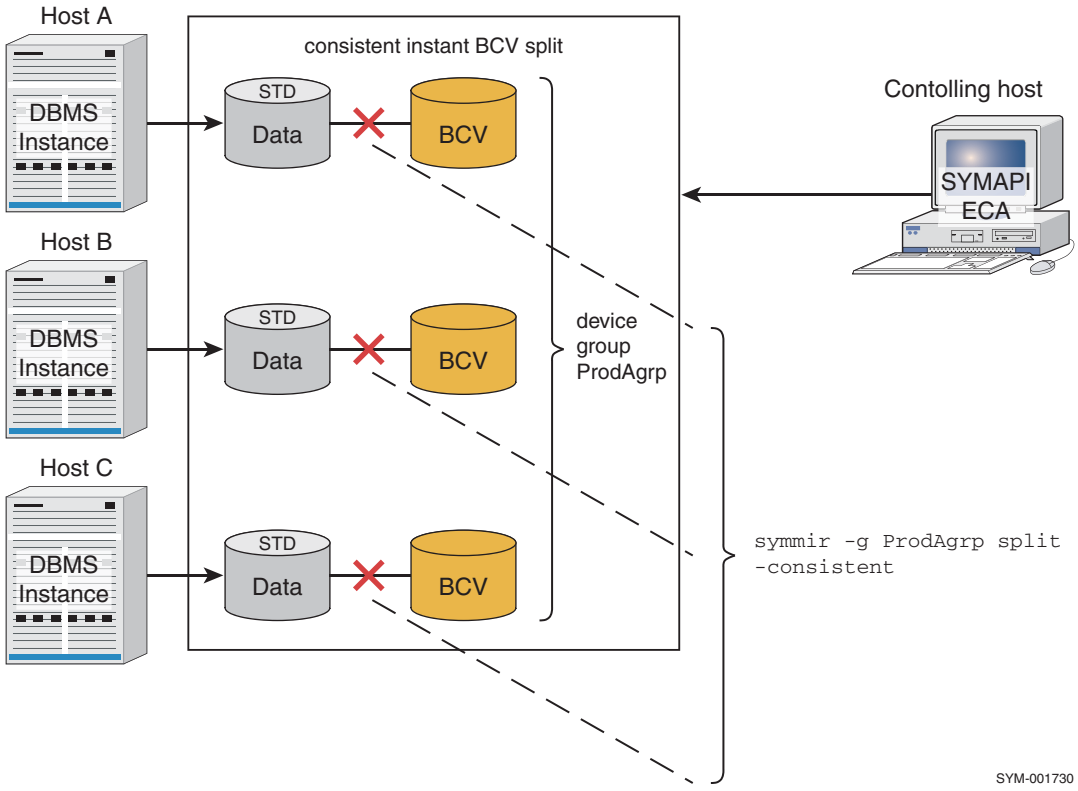

Figure 50 ECA consistent split

Device groups or composite groups must be created on the controlling host for the target database to be consistently split. Device groups can be created to include all of the devices being accessed or defined by database host access. For example, if you define a device group that includes all of the devices being accessed by Hosts A, B, and C (refer to Figure 50), then you can consistently split all of the BCV pairs related to those hosts with a single command. However, if you define a device group that includes only the devices accessed by Host A, then you can split those BCV pairs related to Host A without affecting the other hosts.

Note: For information on performing a consistent split to BCVs in both the local and remote arrays, refer to "Consistent split on both RDF sides using ECA" on page 152.

The following steps explain the example in Figure 50 of how to create one device group including all database host accessed devices and perform a consistent split operation on all of the BCV pairs accessed by those hosts.

1. Create a REGULAR type device group:

```
symdg create ProdAgrp -type REGULAR
```

2. Add all of the standard devices holding the database for each host (A, B, and C) to the device group:

```
symdg -g ProdAgrp addall -devs 0286:028B
symdg -g ProdAgrp addall -devs 0266:026B
symdg -g ProdAgrp addall -devs 0246:024B
```

3. Associate the BCV devices that will hold the restartable copy of the database with the device group:

```
symbcv -g ProdAgrp associateall -devs 039A:039F
symbcv -g ProdAgrp associateall -devs 037A:037F
symbcv -g ProdAgrp associateall -devs 035A:035F
```

4. Fully establish all BCV pairs in the device group:

```
symmir -g ProdAgrp establish -full -noprompt
```

   Note: When the BCV pairs in device group ProdAgrp (Host A, B, and C - BCV pairs) are synchronized, you can perform the consistent split using the `symmir split` command to split all of the BCV pairs associated with those hosts.

5. Use the `-consistent` option to perform a consistent (instant) split on all BCV pairs in the device groups:

```
symmir -g ProdAgrp split -consistent
```

Once the `symmir split -consistent` command is issued, I/O to the device group is frozen and a 30-second Enginuity protection timer begins. After the split completes (or 30 seconds passes, whichever comes first), the I/O channels are thawed, granting (pent up) operations access to the standard devices. Splits across all devices in a group are considered consistent, if the BCV split execution is performed within that window.

If for some unknown host or I/O channel reason, not all devices are split within the 30-second window, `symmir` returns the following reply at completion:

```
Consistency window was closed on some devices before the operation completed.
```

At this point, the final successful split outside the window is no longer considered to be consistent in execution across the device group. For consistency and reliability sake, it is recommended that you reestablish the device group, and then (later) attempt the consistent split again.

Consistent split for SRDF/A devices

TimeFinder consistent split allows you to consistently split BCVs from R2 devices operating in asynchronous mode (SRDF/A).

Although not required for SRDF/A mode, it is recommended that you use TimeFinder BCVs at the remote site to mirror R2 devices and preserve a consistent image of data before resynchronization operations. Also, R2 device BCVs can be split off of the R2 without having to drop the RDF links and without disruption to the SRDF/A operational cycles. R2 BCVs can be controlled from the R1-side or the R2-side host as long as the device groups have been defined on that host. Controlling the R2 BCVs from the R1-side host requires using the `symmir` command with the `-rdf` option. To consistently split BCVs off the R2 RDF/A device in group `prod` from the R1 host, enter:

```
symmir -g prod split -rdf -consistent
```

Consistent split on both RDF sides using ECA

In an RDF environment as shown in Figure 51, you can perform a consistent split to the BCVs in both the local and remote arrays.

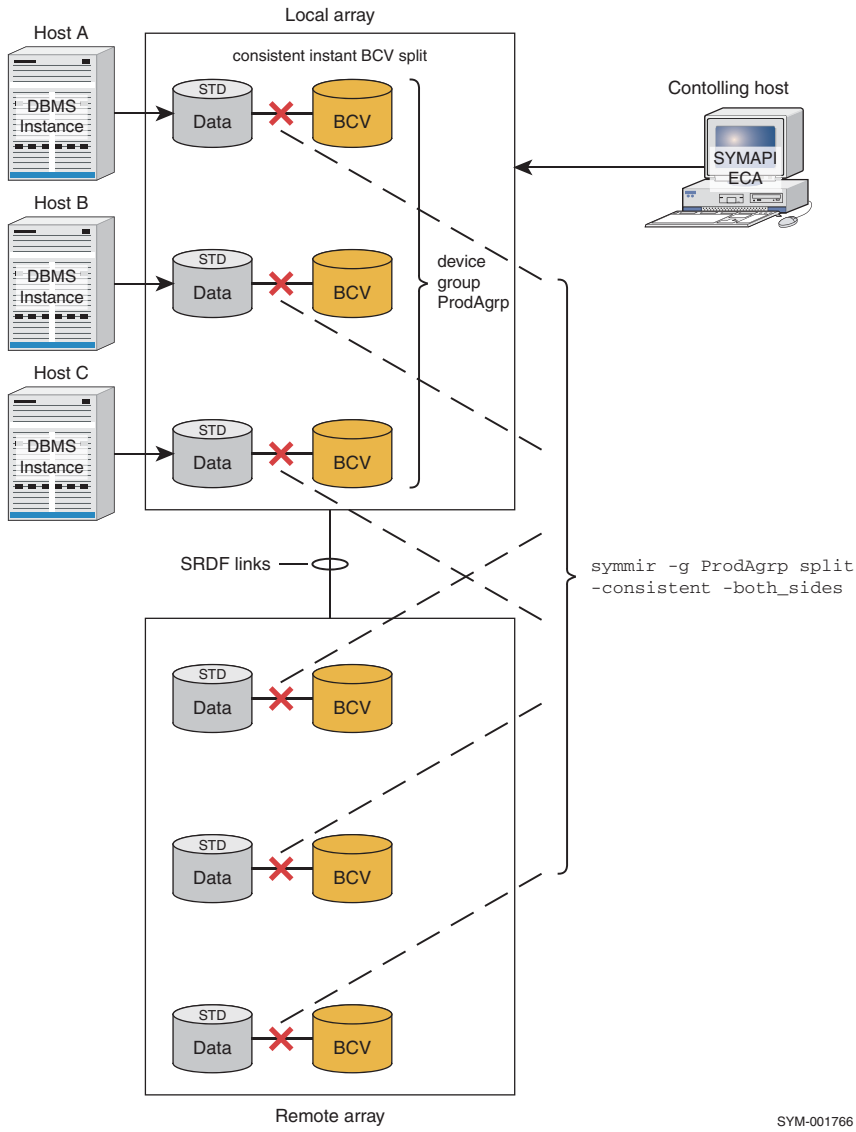

Figure 51 Consistent splits on both SRDF sides using ECA

In the above example, the controlling host issues a single consistent split command with the `-both_sides` option to split devices on both the local and remote arrays. For the host to perform this operation, the SRDF links must be up, the RDF mode must be synchronous, and the devices must have an RDF state of Synchronized.

Fully restoring BCV pairs

Like the full establish operation, a full restore operation copies the entire contents of the BCV devices to the standard device.

Optionally, you can target devices in a device group, composite group, or device file:

```
symmir -g DgName -full restore
symmir -cg CgName -full restore
symmir -f[ile] FileName -full restore
```

Examples To initiate a full restore on all the BCV pairs in the `prod` device group, enter:

```
symmir -g prod -full restore
```

To initiate a full restore on a BCV pair, `DEV001`, in the `prod` device group, enter:

```
symmir -g prod -full restore DEV001
```

To initiate a full restore on more than one (list) of BCV pairs in the `prod` device group, enter:

```
symmir -g prod -full restore  DEV001 BCV ld BCV001
                              DEV002 BCV ld BCV002
                              DEV005 BCV ld BCV003
```

The restoration process (Figure 52) is complete when the standard device and BCV device contain identical data.

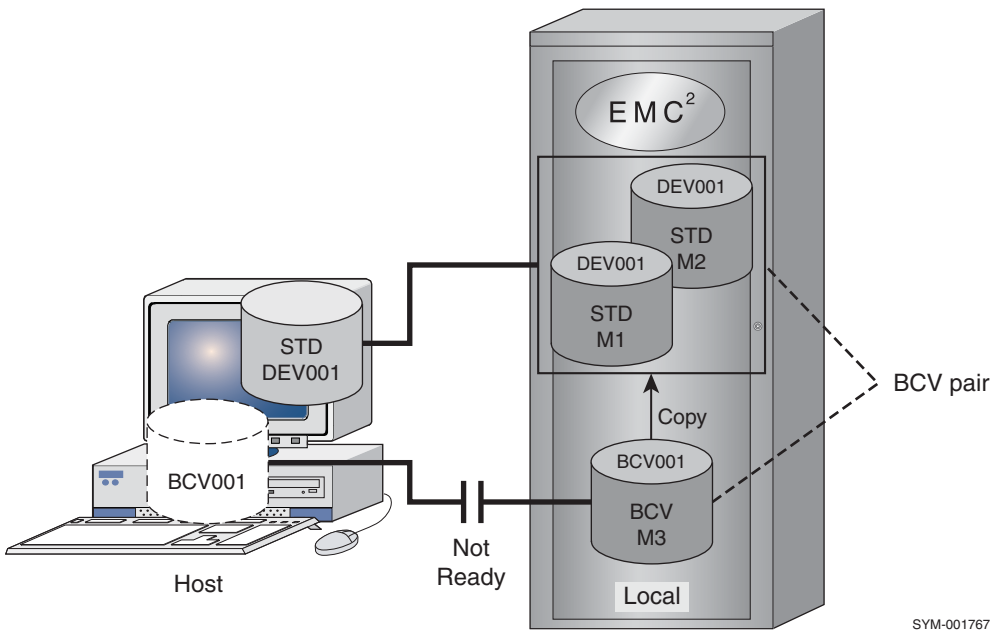

Figure 52 Full restore of the BCV pair

Note: The BCV device is not available for host use during the time that it is assigned as a BCV mirror on a standard device. However, unless the -protect option is used, any new data written to the standard device is copied to the BCV device while the BCV pair exists.

When a full restore is initiated for each specified BCV pair in a device group:

- Command validity is checked. For example, the command is rejected if the BCV device and the standard device are not the same size.
- The BCV device is set as Not Ready to the host.
- The BCV is assigned as the next available mirror of the standard device.
- The contents of the BCV device are copied to the standard device. For example, in Figure 52 on page 153, the array copies the contents of M3 to both M1 and M2, overwriting the data present on those devices.

To use a BCV device for Business Continuance procedures, you must again split the BCV pair to make the BCV device available to its host. If you want to use a fully synchronized copy of the data, suspend all applications that are using the standard device, and make sure that all host buffering and intermediate caching is flushed to the appropriate device on the array prior to performing the split operation. If you do not require a synchronized copy of the data for running a Business Continuance process, this step is unnecessary.

Note that the base tasks performed with `symbcv` such as list, associate, and disassociate locally or remotely connected BCV devices, are described at the beginning of this chapter.

By default, Solutions Enabler rejects restore commands for CKD online devices. You can allow the restore command for CKD online devices by disabling the SYMAPI_TF_CHECK_ONLINE_CKD option. For instructions on disabling or enabling this option, refer to the *EMC Solutions Enabler Installation Guide*.

Specifying the default method for restoring BCV pairs

The SYMAPI_DEFAULT_BCV_RESTORE_TYPE parameter in the options file enables you to specify the default method for restoring BCV pairs. Valid values are:

- SINGULAR specifies to issue the restore to one device at a time. This method allows other tasks access to the array when doing a large number of restores.
- PARALLEL (default) specifies to issue the restore to each servicing DA in parallel, and then wait for a DA to finish before issuing another restore to that DA.
- SERIAL specifies to issue restores as fast as the GST queue can handle them. However, all members of a meta must be restored before continuing to the next meta or device. This is the default method when using metadevices.

Note: The *EMC Solutions Enabler CLI Command Reference* contains information on changing the option file parameters.

Instantly restoring multiple BCV pairs

The multi/instant restore option improves the performance of a typical restore operation by submitting multiple BCV pairs in a single system call to be restored instantly.

You can enable (default)/disable this feature with the SYMAPI_TF_MULTI_ESTAB_REST environment variable. Setting the SYMAPI_DEFAULT_BCV_RESTORE_TYPE to SERIAL or SINGULAR will cause this option to be ignored.

SRDF-connected BCV pairs

You can also specify a full `restore` action to a remote site using the `-rdf` option, which fully restores the remote BCV pairs.

To perform a full restore operation in the remote array at site B when the RDF flag is specified with the following command:

```
symmir -g prod -rdf -full restore DEV001
```

In this case, the flag indicates that the BCV device being restored is an SRDF-connected BCV device, which will be established with the remote standard mirror of the local RDF standard device.

To perform a full restore operation in the remote array at site B when the `-rdf` and `-bcv` options are specified, use the following command:

```
symmir -g prod -rdf -bcv -full restore BCV001
```

In this case, the flags indicate that the BCV device being restored is an SRDF-connected BCV device, which will be established with the remote standard mirror of the local R1 BCV device.

Second-level remote BCV pairs

You can specify a full restore action to a second remote site using the remotely attached remote BCV`-rrbcv` option, which restores second-level remote BCV pairs.To perform a restore operation in the remote array at site C when the `-rrbcv` option is specified, use the following command:

```
symmir -g prod -rrbcv restore -full
```

To initiate a restore on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv restore RBCV001 BCV ld RRBCV001 -full
```

In this case, the flag indicates that the BCV device being restored is a second Hop SRDF-connected BCV device, which will be established with the remote standard mirror of the remote BCV device.

Hop 2 BCV pairs in a cascaded SRDF configuration

You can specify a full `restore` action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform a restore operation in the remote array at the tertiary site (C) with the `-hop2` option, use the following command:

```
symmir -g prod -hop2 restore -full
```

In this case, the `-hop2` option indicates that the SRDF-connected BCV device (2BCV001) is being established with the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Performing a remote copy with restore

In addition to restoring the specified BCV pair, you can further specify the remote (`-remote`) option, which will propagate the restored copy from the BCV pair across the SRDF link from the R2 standard to its R1 BCV in one command step:

```
symmir -g prod restore -rdf -bcv -remote -full
```

Note: The `-remote` option is not supported for restoring from an R2 standard to an R1 BCV. Instead you should use the two step method.

Note: Be sure you want the R1 BCV device to be updated with the restored copy before issuing the `-remote` option along with this restore operation.

It is good practice to perform restore operation at the remote site in two command steps (restore the BCV pair first, and then restore the R1 from the R2):

```
symmir -g DgName restore -bcv -rdf -full
symrdf -g DgName restore -bcv
```

Incrementally restoring BCV pairs

The incremental restore process (Figure 53 on page 157) accomplishes the same thing as the restore process with a major time-saving exception: the BCV (`BCV001`) copies to the standard device (`DEV001`) only the new data that was updated on the BCV device while the BCV pair was split. Any changed tracks on the standard device are also overwritten by the data on the corresponding tracks on the BCV device. This maximizes the efficiency of the synchronization process.

This process is useful if the results from running a new application on the BCV device were desirable, and the user wants to port the data and the new application to the standard device.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symmir -g DgName restore
symmir -cg CgName restore
symmir -f[ile] FileName restore
```

To initiate an incremental restore on all the BCV pairs in the `prod` device group, enter:

```
symmir -g prod restore
```

Note: It might be desirable for your site to set external device locks on all standard and BCV devices you are about to restore; refer to "Device external locks" on page 119.

To initiate an incremental restore on a BCV pair, `DEV001`, in the `prod` device group, enter:

```
symmir -g prod restore DEV001
```

To initiate an incremental restore on more than one (list) BCV pair in the `prod` device group, enter:

```
symmir -g prod restore DEV001 DEV002 DEV003
```

Figure 53 illustrates an incremental restore of a BCV pair.

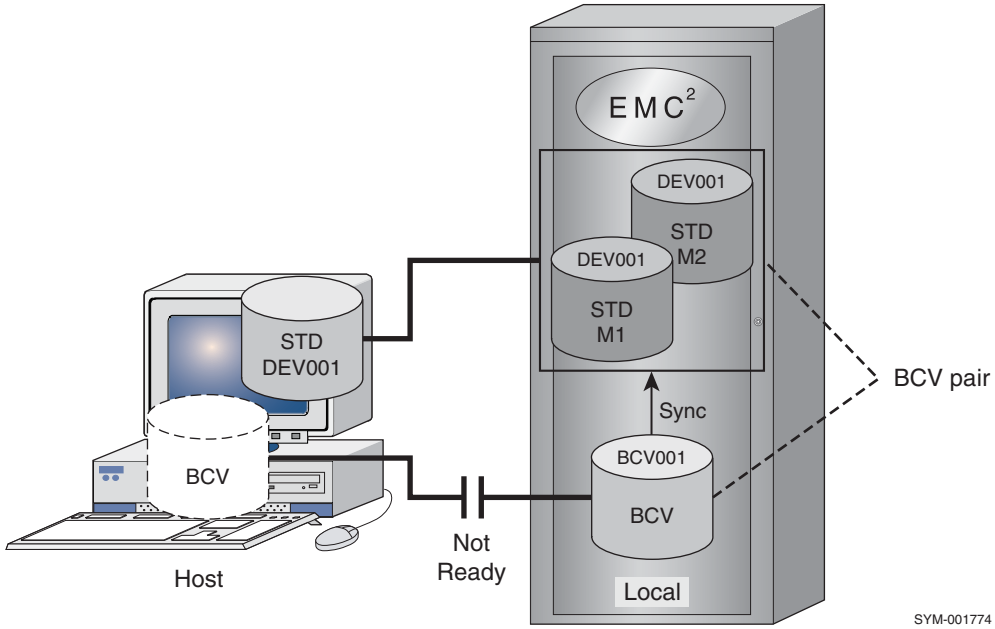

Figure 53 Incremental restore the STD

When an incremental restore is initiated for each specified BCV pair in a device group, the following occurs:

- Command validity is checked. For example, the command is rejected if the BCV device and the standard device were not previously paired.
- The BCV device is set as Not Ready to the host.
- The BCV device is assigned as the next available mirror of the standard device.
- The tracks are copied from the BCV device to the standard device. Any new data written to the BCV device while the BCV pair was split is written to the standard device. Any new data written to the standard device while the BCV pair was split is overwritten by the data on the corresponding track on the BCV device.

The BCV pair is synchronized when the standard device and the BCV device contain identical data.

Note: The BCV device is not available for host use while it is assigned as a BCV mirror on a standard device. However, any new data written to the standard device is copied to the BCV device while the BCV pair exists.

Multiple BCV pairs

You can incrementally establish or restore up to 16 BCV pairs (8 pairs when using emulation) associated with a single standard device. Using the environment variable SYMCLI_MAX_BCV_PAIRS, the maximum amount of pairs can be adjusted from 1 to 16

BCV devices. If a series of `split/increment establish` commands were invoked over time (refer to "Incrementally establishing multiple BCV pairs" on page 140), a multi-BCV environment becomes established that retains progressive historical images of the specified standard.

With the incremental restore command, you can specify any one of these older BCVs to incrementally restore the standard back to a specific historical copy. Figure 54 shows the standard being restored by the `BCV002` copy, which was split at 4 a.m.

Before performing a restore, you may need to query the array to see which BCVs can be incrementally restored:

```
symmir -g DgName query -multi
```

Note: Before invoking the restore command, be sure this is the data copy you want your standard restored to since this BCV is an older version of what is current.

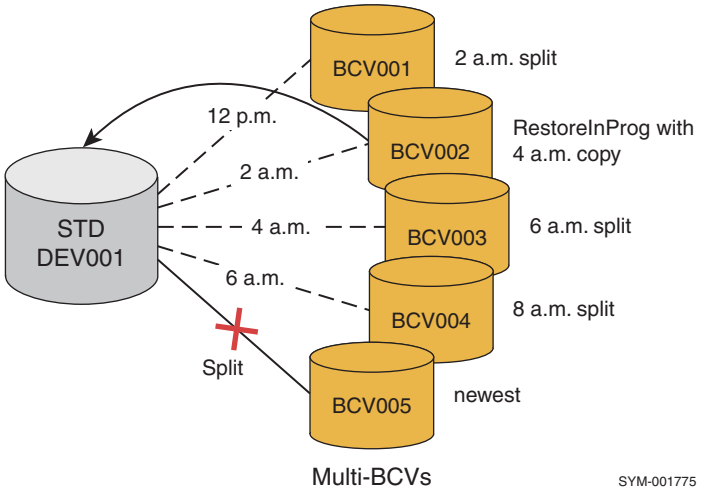

Figure 54 Restoring a BCV in a multi-BCV environment

To restore the current standard with old data from `BCV002`, enter:

```
symmir -g DgName split DEV001
symmir -g DgName restore DEV001 BCV ld BCV002
```

SRDF-connected BCV pairs

You can also specify an incremental `restore` action to a remote site using the RDF flag (`-rdf` option), which incrementally restores the remote BCV pairs.

Note: Be sure you want the standard R1 device to be updated with the restored copy before issuing the `-remote` option along with this restore operation.

To perform an incremental restore operation in the remote array at site B when the RDF flag is specified, use the following command:

```
symmir -g prod -rdf restore
```

In this case, the flag indicates that the BCV device being restored is an SRDF-connected BCV device, which will be established with the remote standard mirror of the local R1 standard device.

In addition to restoring the specified BCV pair, you can further specify the remote (`-remote`) option, which will propagate the restored copy from the BCV pair across the SRDF link to the R1 standard.

Note: The `-remote` option is required when restoring to an R1 device that is ready on the link.

It is good practice to perform restore operations in two command steps (restore the BCV pair first, then restore the R1 from the R2):

```
symmir -g DgName restore -rdf
symmrdf -g DgName restore
```

Second-level remote BCV pairs

You can specify an incremental `restore` action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which incrementally restores second-level remote BCV pairs.

To perform an incremental restore operation in the remote array at site C specifying the remotely attached remote BCV flag (`-rrbcv` option), use the following command:

```
symmir -g prod -rrbcv restore
```

To initiate a restore on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv restore RBCV001 BCV ld RRBCV001
```

In this case, the flag indicates that the BCV device being restored is a second HOP SRDF-connected BCV device, which will be established with the remote standard mirror of the remote BCV device.

Hop 2 BCV pairs in a cascaded SRDF configuration

You can specify a incremental `restore` action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform an incremental restore operation in the remote array at the tertiary site (C) specifying the hop 2 flag (`-hop2` option), use the following command:

```
symmir -g prod -hop2 restore
```

In this case, the flag indicates that the SRDF-connected BCV device (2BCV001) is being established with the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Protecting BCV data during full or incremental restores

Once you initiate a restore from the BCV to the standard device, data from the BCV is immediately available to a host accessing the standard device. During the time the pair are joined, writes are sent to the standard and the BCV, while reads are satisfied by the data on the BCV if the data on the standard has not yet been completely updated from the BCV. However, this process can alter the BCV data during the restore operation. If you want to retain the original BCV data, use the protected restore feature.

The protected restore feature allows the contents of a BCV to remain unchanged during and after a restore operation, even while the BCV and the standard are joined. Subsequently, any writes to the BCV pair are not propagated to the BCV while the standard and the BCV are joined in a RestInProg or Restored state. This protection offers the same advantage as a reverse split, but without the need for a mirrored BCV.

You can restore data from a BCV to a standard device without altering the contents of the BCV, by using the protect (`-protect`) option. It write-disables the BCV mirror(s) during and particularly after the restore operation.

Examples

To initiate a protected *full* restore (for example) on the STD mirrors (`DEV001`) in the `Prod` group, enter:

```
symmir -g Prod -full restore -protect DEV001
```

To initiate a protected *incremental* restore (for example) on the STD mirrors (`DEV001`) in the `Prod` group, enter:

```
symmir -g Prod restore -protect DEV001
```

Note: If you ever need to split a device again that was protected restored, you must use the `-protect` option on the split command:

```
symmir -g Prod split -protect DEV001
```

To view device information for the protected restore operation, enter:

```
symmir -g Prod query -protect
```

Note: The standard invalid track count displayed in the query operation does not reflect any new writes while the device is in the RestInProg state. When the device state changes to Restored, the invalid track count displays as zero.

Cancelling BCV pairs

The `symmir cancel` command allows you to cancel a BCV pair relationship on a device by device basis, or for all the devices in a device group or composite group.

When operating in native TimeFinder, cancelling a BCV pair cancels the existing relationship between the specified standard and BCV device(s). Once the relationship is cancelled, the corresponding BCV devices go into the SplitNoInc state, and the BCV pair can no longer be incrementally established or restored.

When operating in emulation mode, cancelling a BCV pair terminates the relationship between the specified standard and BCV device(s). Once the relationship is terminated, the corresponding BCV devices go into the Never Established state, and the BCV pair can no longer be incrementally established or restored.

When cancelling a muli-BCV relationship, only the primary BCV is cancelled. For information on cancelling a multi-BCV relationship, refer to "Canceling a multi-BCV relationship" on page 133.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symmir -g DgName cancel
symmir -cg CgName cancel
symmir -f[ile] FileName cancel
```

Examples To cancel the BCV relationship for all the devices in the `Prod` group, enter:

```
symmir -g Prod cancel
```

To cancel a specific standard/BCV pair relationship in the `Prod` group, enter:

```
symmir -g Prod cancel DEV001 BCV dev 009C
```

To cancel the relationship of SRDF-connected BCV pairs in the `Prod` device group, enter any of the following:

```
symmir -g prod cancel -rdf
```

Cancels the relationship between the remote mirror device(s) and the remote BCV device(s).

```
symmir -g prod cancel -rdf -bcv
```

Cancels the relationship between the SRDF-connected BCV pair remotely mirroring the local BCV device.

```
symmir -g prod cancel -rrbcv
```

Cancels the relationship between the remote mirror of the remotely attached BCV device (RBCV) and the remotely attached remote BCV (RRBCV).

```
symmir -g prod cancel -hop2
```

Cancels the relationship between the remote mirror and the BCV (2BCV) two hops away in a cascaded SRDF configuration.

Querying BCV pairs

You can perform a query to determine the state of a BCV pair or all BCV pairs in a device group, composite group, or device file. The query is sent via the gatekeeper device to the array, returning with information about the state of the BCV pair(s).

The following forms enable you to target devices in a device group, composite group, or device file:

```
symmir -g DgName query
symmir -cg CgName query
symmir -f[ile] FileName query
```

Examples To query the state of the BCV pairs in the `prod` device group, enter:

```
symmir -g prod query
```

To query the state of SRDF-connected BCV pairs in the `prod` device group, enter any of the following:

```
symmir -g prod query -rdf
symmir -g prod query -rdf -bcv
symmir -g prod query -rrbcv
symmir -g prod query -hop2
```

You can also obtain results using the `-offline` option, which looks at your configuration based on the host database.

The results of the query include the following information for each member of a BCV pair in a device group:

- Logical device name
- Device name
- Number of invalid tracks
- BCV pair state

To query the state of a split action on multi-BCVs or concurrent BCVs in a group `prod`, enter:

```
symmir -g prod query -multi
```

To query the state of any background split action on multi-BCVs or concurrent BCVs in a group `prod`, enter:

```
symmir -g prod query -multi -bg
```

To query the percent initiated on restore, establish, and split operations, enter:

```
symmir -g prod query -bg -percent
```

Using the -summary option

If you use the `-summary` option with the `query` argument, the results of the query will include the following information:

- Number of BCV pairs in each BCV pair state
- Number of invalid tracks
- Synchronization rate
- Estimated time to completion

The synchronization rate and estimated time to completion are shown only when `-i` or `-c` is specified and their has been a change in the number of invalid tracks since the previous iteration.

The `-summary` option also works with the `verify` argument.

Example To view the number of BCV pairs in the `prod` device group that are in each state, and to view the estimated time to completion, enter:

```
symmir -g prod query -summary -i 60
```

Verifying BCV pair states

You can use the `symmir verify` command to verify whether one or all BCV pair(s) in a device group, composite group, or device file are in a particular state. The command can be used in scripts to guarantee that the BCV device pair(s) are in a Synchronized, Restored, or Split state prior to executing subsequent SYMCLI commands. If you do not specify any qualifiers with the `symmir verify`, the default is to check for the Synchronized or Restored states.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symmir -g DgName verify
symmir -cg CgName verify
symmir -f[ile] FileName verify
```

The following options qualify the `symmir verify` command. If you need to verify a concurrent BCV pair, include `-concurrent` with the option (for example, `-synched -concurrent`):

- `-synched` option verifies the Synchronized state.
- `-syncinprog` option verifies the SyncInProg state.
- `-split` option verifies the Split state. With an instant split, the system verifies the Split state immediately even though the background split is still in progress. To verify completion of a background split after an instant split, use the `-split -bg` option. Until the background split is complete, you cannot perform BCV control operations. You can use the `-split -bg` option to verify that the instant split is 100 percent complete in the background. For example:

  symmir verify -g ProdBgrp -split -bg DEV001 bcv ld BCV002 –i 30

- `-restored` option verifies the Restored state. You can use the `-restored -protect` option to verify the Protected Restored state. In a concurrent BCV setup, you can use `-restored -concurrent` successfully only if the first BCV has already restored the standard and you are restoring now with the second BCV.
- `-restinprog` option verifies the RestInProg state.
- `-bcv_mirrors` option verifies that the mirrors of locally mirrored BCV devices are in the specified state. If you do not specify a state with this option, the default is to verify a Synchronized state.

Examples

For a multi-BCV or concurrent BCV device group, specifying the BCV on the command line ensures that the verify operation checks the status of the BCV. Otherwise, the verify operation checks the status of the standard device, which may no longer be established with the BCV that you want to verify. For example, the following command returns the status of standard device DEV002 with its last paired BCV:

symmir -g ProdBgrp verify DEV002

But the following command returns the status of a specific BCV pair (`DEV002` with `BCV001`):

symmir -g ProdBgrp verify DEV002 BCV ld BCV001

The following command checks status every 30 seconds until all BCV pairs in the device group (ProdBgrp) or composite group (MyConGrp) are in the Synchronized or Restored state (the default when no state is specified on the command line):

```
symmir -g ProdBgrp -i 30 verify
symmir –cg MyConGrp –i 30 verify
```

Possible outputs at 30-second intervals can be that none, not all, or all devices are synchronized or restored. The time to reach the Synchronized or Restored state varies with the number of devices being established or restored and the amount of data being copied.

The verify action returns a value of zero (code symbol CLI_C_SUCCESS) if the verify criteria are met, or one of the unique codes in Table 16 and Table 17 if the verify criteria are not met:

Table 16 Using options to verify a BCV mirror state

Options used with Verify	Code number	Code symbol
–bcv_mirrors	4	CLI_C_NOT_ALL_SYNCHRONIZED
–bcv_mirrors	5	CLI_C_NONE_SYNCHRONIZED
–bcv_mirrors -ready	62	CLI_C_NOT_ALL_READY
–bcv_mirrors -ready	63	CLI_C_NONE_READY
–bcv_mirrors -syncinprog	27	CLI_C_NOT_ALL_SYNCINPROG
–bcv_mirrors -syncinprog	28	CLI_C_NONE_SYNCINPROG
–bcv_mirrors -restinprog	29	CLI_C_NOT_ALL_RESTINPROG
–bcv_mirrors -restinprog	30	CLI_C_NONE_RESTINPROG

Table 17 lists the options to verify a BCV pair state.

Table 17 Using options to verify a BCV pair state

Options used with Verify	Code number	Code symbol
-synched	10	CLI_C_NOT_ALL_SYNCHED
-synched	11	CLI_C_NONE_SYNCHED
-restored	12	CLI_C_NOT_ALL_RESTORED
-restored	13	CLI_C_NONE_RESTORED
-split or -split -bg	25	CLI_C_NOT_ALL_SPLIT
-split or -split -bg	26	CLI_C_NONE_SPLIT
-syncinprog	27	CLI_C_NOT_ALL_SYNCINPROG
-syncinprog	28	CLI_C_NONE_SYNCINPROG
-restinprog	29	CLI_C_NOT_ALL_RESTINPROG
-restinprog	30	CLI_C_NONE_RESTINPROG

Using composite groups to manage BCV pairs across arrays

A composite group is a user-defined group of devices that can span multiple arrays. This feature provides greater flexibility than a device group, which can define devices only on a single array. You can control specific BCV pairs within the composite group instead of having to operate on the entire group as in previous versions.

Figure 55 illustrates a production host that is locally connected to two arrays (A and B). A composite group is defined on the production host and includes BCV pairs from each array. Another locally connected host allows you to access the BCVs once the BCV pairs are split.

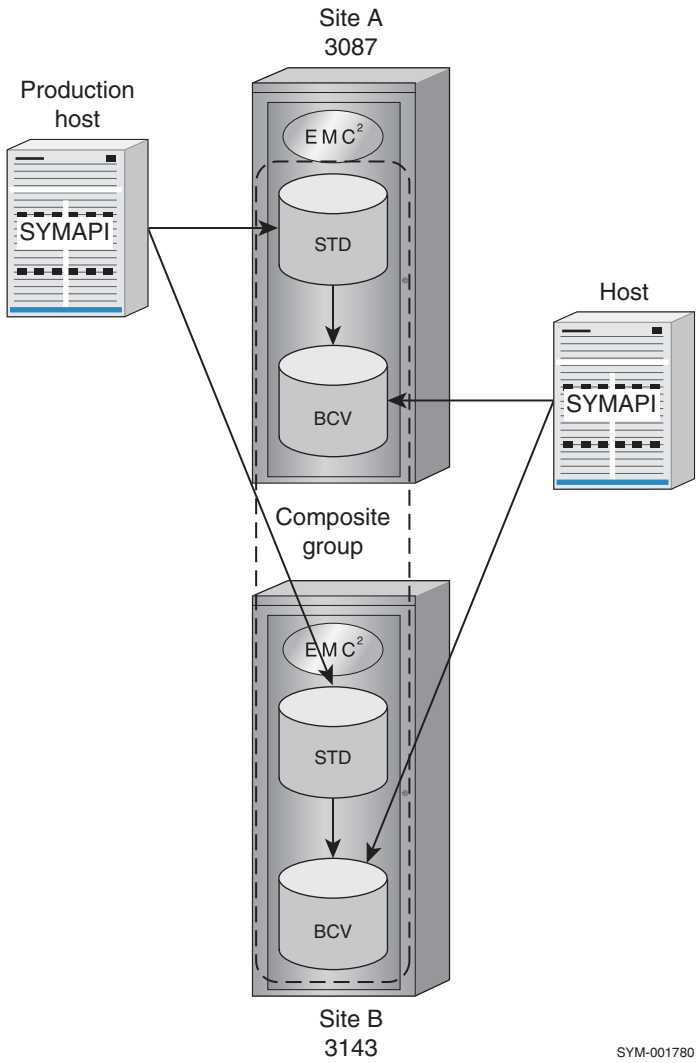

Figure 55 Using a composite group when a set of devices spans two arrays

Although TimeFinder control operations on BCV pairs might normally be performed from the production host (as shown in Figure 55) because the composite group is defined there in its SYMAPI database, there are methods that would allow you to initiate copy sessions from another locally connected host. One way is to copy the composite group definition to another host. A more efficient method is to enable Group Name Services (GNS), which

automatically propagates the composite group definition to the arrays and other locally attached hosts that are running the GNS daemon. For more information, refer to the *EMC Solutions Enabler Array Management CLI Product Guide*.

If you do not create each BCV pair explicitly, certain options such as `-opt`, `opt_rag`, and `-exact` allow you to control how multiple devices in a composite group are paired. Otherwise, a device-pairing algorithm checks if there were any previous pair assignments among the devices and, if not, pairs standards and BCVs of equal sizes.

Used only with a full establish operation, the optimize option that you choose depends on whether you are establishing local BCV pairs or remote BCV pairs. The `-opt` option is for local. It optimizes pairings across the local array without regard for whether the devices belong to different RDF (RA) groups. The `-opt_rag` option is for remote and requires the `-rdf` option. It uses optimization rules to create remote BCV pairs from devices within the same RDF (RA) group on an array.

The following steps outline the setup required for controlling a set of BCV pairs that spans two arrays as shown in Figure 55 on page 165:

1. From the production host, create a Regular type composite group (for example, `MyGrp`):

   ```
   symcg create MyGrp -type regular
   ```

2. Add to the composite group those standard devices on array A (3087) and array B (3143) that are the source devices:

   ```
   symcg -cg MyGrp -sid 3087 add dev 0076
   symcg -cg MyGrp -sid 3143 add dev 0091
   ```

3. Add a BCV device from each array to the composite group:

   ```
   symbcv -cg MyGrp -sid 3087 associate dev 0051
   symbcv -cg MyGrp -sid 3143 associate dev 004F
   ```

4. Create the BCV pairs and initiate full copying from the standards to the BCVs:

   ```
   symmir -cg MyGrp establish -full
   ```

5. When the BCV pairs are fully synchronized, you can split all BCV pairs in the composite group to access the BCVs:

   ```
   symmir -cg MyGrp split
   ```

You can control specific BCV pairs within the composite group instead of having to control the group as a whole. To establish only the `DEV001`/ `BCV001` pair from all devices in the composite group `MyGrp`:

```
symmir -cg MyGrp establish DEV001 bcv ld BCV001
```

Preferred attachment of BCVs (optional operations)

For advanced users, the preferred pair *attachment* (`attach` action) is an optional step in the management of BCV pairs that eliminates the need to specify a device for each subsequent full establish and full restore sequence in a script (for all Enginuity versions). It also applies to incremental establish and restore operations. It marks the specified BCV device as the preferred BCV to pair with the standard device.

After configuration and initialization of an array, BCV devices contain no data. The BCV devices, like the standard devices, have unique host addresses and are online and ready to the host(s) to which they are connected.

It is at this point, before any full establish or full restore operations are requested, you can validate your pairings as a preferred attachment before starting any data copy operations. The lists of individual standard devices and BCV devices can be examined, validated, and all devices sorted according to storage size, and subsequently, assigned as the preferred match (considering disk size) for attachment into BCV pairs.

Note: A full establish action with the optimize (`-opt`) or exact (`-exact`) option overrides the attach pairing scheme.

The following commands enable you to target devices in a device group, composite group, or a device file:

```
symmir -g DgName attach
symmir -cg CgName attach
symmir -f FileName attach
```

To initiate a preferred attachment on a BCV pair (`DEV001`) in the `prod` group, enter:

```
symmir -g prod attach DEV001 BCV ld BCV001
```

To initiate a preferred attachment on more than one BCV pair (list) in the `prod` group, enter:

```
symmir -g prod attach    DEV001 BCV ld BCV001
                         DEV002 BCV ld BCV002
                         DEV002 BCV ld BCV003
```

Note: The attach and detach preferred relationship are only known to the SYMAPI database on which you are operating.

The attach action checks command validity. For example, the array makes sure that both the standard device and the BCV device are the same size, the device specified as the BCV has the BCV attribute, the standard device does not already have a BCV device assigned to it, and so on.

If the standard device is a metahead device, then the BCV must also share the same metadevice properties. All metamembers are implicitly established along with the metahead device.

From this point forward, when you invoke the full establish or full restore control action with a BCV control operation, you will not need to specify the device names.

Detaching BCV preferences from devices

The `detach` action allows you to remove the preferred matched-pair association from the devices that was initially defined with the `attach` action.

The following forms enable you to target devices in a device group, composite group, or device file:

```
symmir -g DgName attach
symmir -cg CgName attach
symmir -f FileName attach
```

To detach the existing preferred attachment of various BCVs from their standard devices in the `prod` group, enter:

```
symmir -g prod detach
```

To detach the attached BCV preference on standard device (`DEV001`) in the `prod` group, enter:

```
symmir -g prod detach DEV001
```

Attaching remote devices as preferred pairs

For advanced usage, you can also specify an `attach` action to a remote site using the RDF flag (`-rdf` option), which attaches the remote mirror device(s) to the remote BCV device(s) as preferred pair(s).

To perform a preferred attachment operation in the remote array at site B with the RDF option (`-rdf`) specified, use the following command:

```
symmir -g prod -rdf attach DEV001 bcv ld RBCV001
```

In this case, the RDF flag indicates that the BCV device being attached is an SRDF-connected BCV pair, which will provide remote mirroring to the local standard device.

To perform an attach operation in the remote array at site B with the RDF and BCV options (`-rdf` and `-bcv`) specified, use the following command:

```
symmir -g prod -rdf -bcv attach BCV001 BCV ld BRBCV001
```

In this case, the flags indicate that the BCV pair being attached is an SRDF-connected BCV pair, which provides remote mirroring to the local BCV device.

Detaching BCV preferences for remote devices

You can specify a `detach` preference action to a remote site using the RDF flag (`-rdf` option), which detaches the remote BCV(s) from the remote standard device(s) as preferred pair(s).

To perform a detach operation in the remote array at site B with the RDF option specified, use the following command:

```
symmir -g prod -rdf detach DEV001
```

In this case, the flag indicates that the preferred BCV device being detached is an SRDF-connected BCV, which provides remote mirroring to the local standard device.

To perform a detach operation in the remote array at site B with the RDF and BCV options (`-rdf` and `-bcv`) specified, use the following command:

```
symmir -g prod -rdf -bcv detach BCV001 BCV ld BRBCV001
```

In this case, the flags indicate that the BCV pair being detached is an SRDF-connected BCV pair, which would provide remote mirroring to the local BCV device.

Attaching second-level remote devices as preferred pairs

You can specify an `attach` action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which attaches BCV preferences to second-level remote BCV pairs.

To perform an attach operation in the remote array at site C with the remotely attached remote BCV option (`-rrbcv`) specified, use the following command:

```
symmir -g prod -rrbcv attach
```

To initiate an attach on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv attach RBCV001 BCV ld RRBCV001
```

In this case, the flag indicates that the BCV device being attached is a second HOP SRDF-connected BCV device, which will be attached with the remote standard mirror of the remote BCV device.

Detaching BCV preferences from second-level remote devices

You can specify a `detach` action to a second remote site using the remotely attached remote BCV flag (`-rrbcv` option), which detaches BCV preferences from second-level remote BCV pairs.

To perform a detach operation in the remote array at site C with the remotely attached remote BCV option (`-rrbcv`) specified, use the following command:

```
symmir -g prod -rrbcv detach
```

To initiate a detach on one remote BCV pair, `RBCV001`, in the `prod` group, enter:

```
symmir -g prod -rrbcv detach RBCV001 BCV ld RRBCV001
```

In this case, the flag indicates that the BCV device being detached is a second HOP SRDF-connected BCV device, which will be detached from the remote standard mirror of the remote BCV device.

Attaching hop 2 devices as preferred pairs in a cascaded SRDF configuration

For advanced usage, you can specify an `attach` action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform an attach operation in the remote array at the tertiary site (C) when the hop2 (`-hop2` option) is specified with the following command:

```
symmir -g prod -hop2 attach DEV001
              bcv ld 2BCV001
```

In this case, the flag indicates that the SRDF-connected BCV device (2BCV001) is being attached with the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Detaching BCV preferences from hop 2 devices in a cascaded SRDF configuration

For advanced usage, you can also specify an `detach` preference action to an array located at the tertiary site of a cascaded SRDF configuration.

To perform a detach operation in the remote array at the tertiary site (C) with the hop 2 (`-hop2` option) specified, use the following command:

```
symmir -g prod -hop2 detach DEV001 BCV ld 2BCV001
```

In this case, the flag indicates that the SRDF-connected BCV device (2BCV001) is being detached from the remote partner of the R21 device, which is the remote partner of the local RDF standard device.

Script summary for typical TimeFinder operations

The following is an example script of a set of typical operations using SYMCLI commands to manage a BCV environment:

1. Create a device group:

   symdg create ProdBgrp

2. Add a standard device to a device group:

   symdg -g ProdBgrp add pd c0t2d4

   Repeat for all devices, or use RANGE, etc.

3. Associate a BCV device with a device group:

   symbcv -g ProdBgrp associate pd c4t2d4

   Repeat for all BCVs.

4. Either establish the entire group:

   symmir -g ProdBgrp -full establish -noprompt

   Or establish explicitly:

   symmir -g ProdBgrp -full establish DEV001 BCV ld BCV001 -noprompt

   Repeat this command for all pairs.

Transfer a different BCV device to the standard device:

1. Identify the established BCV pair.
2. Split the pair:

   symmir -g ProdBgrp split DEV001 -noprompt

3. Select a new BCV to establish with the standard device:

   symmir -g ProdBgrp -full establish DEV001 BCV ld BCV020 -noprompt

Script example for multi-BCV environment

The following is an example of a script for a multi-BCV environment:

You are tasked with testing business applications with incoming database data from certain anticipated peak periods in the day. Three copies of the database may be needed. To establish a multi-BCV environment, you must initially perform a full establish to each BCV device in the set.

1. For example, you plan to have `BCV001` through `BCV003` in the set to pair with `DEV001` that is the source of your test data:

   symmir -g MultigrpA -full establish DEV001 BCV ld BCV001

```
symmir -g MultigrpA split DEV001 #split at 3:10pm

symmir -g MultigrpA -full establish DEV001 BCV ld BCV002

symmir -g MultigrpA split DEV001 #split at 3:20pm

symmir -g MultigrpA -full establish DEV001 BCV ld BCV003
```

2. It is now 4:00 p.m. and `BCV003` is still currently established with `DEV001`. At this point, you are testing your business applications and want to reset your database back to the business activity that was current up till 3:10 p.m. To incrementally restore `DEV001` to the 3:10 p.m. business data:

```
symmir -g MultigrpA split DEV001 #split at 4:00pm
symmir -g MultigrpA restore DEV001 BCV ld BCV001
```

3. You are now working successfully with the 3:10 p.m. data and want to continue test operations with this data and remove the second split BCV that occurred at 3:20 p.m. as this data will not be needed.

```
symmir -g MultigrpA cancel DEV001 BCV ld BCV002
```

4. You decide to call it a day, keeping the remaining two multi-BCVs and need to reestablish (incrementally establish) `BCV003` to the current standard data:

```
symmir -g MultigrpA split DEV001 #split at 4:50pm
symmir -g MultigrpA establish DEV001 BCV ld BCV003
```

BCV pair states

When you invoke BCV control commands on a single BCV device, or on a group of BCV pair(s) using the `symmir` command, the BCV state is changed as illustrated in Table 18. You will see the abbreviated BCV pair state listed using the SYMCLI commands.

Table 18 BCV pair states

BCV pair state	BCV pair state (abbreviated for display)	Description
Never Established	NeverEstab	The BCV device is available for use, and was never established. Only the BCV device name is valid.
Sync In Progress	SyncInProg	When the Establish action is executed, data is copied from the standard device to the BCV device until both devices contain identical data.
Restore In Progress	RestInProg	When the restore action is executed, data is copied from the BCV to the standard device until both devices contain identical data.
Synchronized	Synchronized	The BCV and standard devices have identical data. Any changes to the standard device are also written to the BCV. The BCV is unavailable to the host for BC processing.
Restored	Restored	The BCV and standard devices have identical data, although the data was originally on the BCV before being synchronized. Any changes to the standard device are also written to the BCV. The BCV is unavailable to the host for BC processing.

Table 18 BCV pair states (continued)

BCV pair state	BCV pair state (abbreviated for display)	Description
Split in Progress	SplitInProg	The BCV devices are in the process of being separated, or split from the standard devices.
Split	Split	The BCV devices are completely separated, or split from the standard devices allowing each device to be accessed separately by the host.
Split No Incremental	SplitNoInc	The BCV devices are completely separated, or split from the standard devices but cannot be incrementally established or restored.
Split Before Sync	SplitBfrSync	The split occurred when a BCV device was synchronizing. The BCV device is separated from the standard device although the BCV device is not completely synchronized.
Split Before Restore	SplitBfrRest	The split occurred when a BCV device was being restored to a standard device. The BCV device is separated from the standard device although the standard device is not completely synchronized.
Invalid	Invalid	Not all metamembers are in the same BCV state.

Transient BCV pair states

When you initially invoke the `symmir` arguments for TimeFinder operations, BCV pairs enter a transient state and upon completion of the action, the BCV pairs enter a final BCV pair state (Table 19).

Table 19 Actions for BCV devices

Argument	Transient state	Final state
establish	SyncInProg	Synchronized
split	SplitInProg	Split
restore	RestInProg	Restored

BCV actions and applicable states

Table 20 describes which BCV control operations can be invoked for a given BCV state.

Invalid states can indicate that the devices in a BCV pair are in a different or *mixed* state.

The `-symforce` option must be used (where noted in the table as F) to force a pair to a specified BCV state.

Table 20 BCV control actions and applicable states

Control Operation	Never Estab	Sync In Prog	Synchronized	Split In Prog	Split	Split No Inc	Split Bfr Sync	Split Bfr Rest	Rest In Prog	Restored	Invalid
establish -full	Y				Y	Y	Y[1]	Y[2]			Y
establish					Y						
split		F[3]	Y						F[3]	Y	
restore -full	Y				Y	Y[1]	Y[2]	Y[1]			Y
restore					Y						
attach	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
detach	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
cancel					Y		Y	Y			

1. The BVC must be specified or you must use the `-exact` or `-force` option.
2. The BCV must be specified or you must use both the `-force` and `-symforce` options.
3. The F denotes that you must use the `-symforce` option.

Command options with device groups

Table 21 lists the `symmir` control operations and the possible options to use when targeting a specified device group.

Table 21 symmir -g control arguments and possible options

Options	Argument action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
`-bcv`	Y	Y	Y	Y	Y	Y	Y	Y
`-both_sides`	Y	Y						
`-bypass`			Y	Y	Y			
`-c`, `-i`	Y	Y	Y	Y	Y	Y	Y	Y
`-concurrent`	Y	Y						
`-consistent`			Y					
`-diff`			Y					
`-exact`	Y			Y				
`-force`, `-symforce`	Y	Y	Y	Y	Y			Y
`-hop2`	Y	Y	Y	Y	Y	Y	Y	Y
`-noprompt`	Y	Y	Y	Y	Y	Y	Y	Y
`-not_ready`			Y	Y	Y			

Table 21 symmir -g control arguments and possible options (continued)

Options	Argument action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
-opt	Y							
-preaction, -postaction	Y	Y	Y	Y	Y			
-preservetgtlocks, -lockid	Y	Y	Y	Y	Y			
-protbcvest	Y	Y						
-protect			Y	Y	Y			
-rdf	Y	Y	Y	Y	Y	Y	Y	Y
-remote	Y	Y	Y	Y	Y			
-reverse	Y	Y	Y	Y	Y			
-rrbcv	Y	Y	Y	Y	Y	Y	Y	Y
-star	Y	Y	Y	Y	Y			Y
-std_protect			Y					
-skip	Y	Y	Y					
-v	Y	Y	Y	Y	Y	Y	Y	Y

Table 22 lists the `symmir` view arguments and the possible options to use when targeting a specified device group.

Table 22 symmir -g view arguments and possible options

Options	Argument action	
	query	verify
-attach	Y	
-bcv, -rrbcv	Y	Y
-bcv_mirrors		Y
-bg	Y	Y
-c, -i	Y	Y
-concurrent		Y
-force		Y
-hop2	Y	Y
-multi	Y	
-offline	Y	Y
-percent	Y	Y
-protbcvest	Y	
-protect	Y	Y

Table 22 symmir -g view arguments and possible options (continued)

Options	Argument action	
	query	verify
-rdf	Y	Y
-ready		Y
-restinprog		Y
-restored		Y
-sid	Y	Y
-split		Y
-summary	Y	Y
-synched		Y
-syncinprog		Y

Note: The base tasks performed with `symbcv` such as `list`, `associate`, and `disassociate` locally or remotely attached BCV devices, are described at the beginning of this chapter.

Command options with composite groups

Options to the `symmir -cg` command line arguments provide more action flexibility to control BCV pairs when you are operating on device(s) of a specified composite group. Table 23 lists the `symmir` control operations and the possible options to use when targeting a specified composite group.

Table 23 symmir -cg control arguments and possible options (page 1 of 2)

Options	Argument action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
-bcv	Y	Y	Y	Y	Y	Y	Y	Y
-both_sides	Y	Y						
-bypass			Y	Y	Y			
-c, -i	Y	Y	Y	Y	Y	Y	Y	Y
-concurrent	Y	Y						
-consistent, -both_sides			Y					
-diff			Y					
-exact	Y			Y				
-force, -symforce	Y	Y	Y	Y	Y			
-hop2	Y	Y	Y	Y	Y	Y	Y	Y

Table 23 symmir -cg control arguments and possible options (page 2 of 2)

Options	Argument action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
-noprompt	Y	Y	Y	Y	Y	Y	Y	Y
-not_ready			Y	Y	Y			
-opt	Y							
-opt_rag	Y							
-preaction, -postaction	Y	Y	Y	Y	Y			
-protbcvest	Y	Y						
-protect			Y	Y	Y			
-rdf	Y	Y	Y	Y	Y	Y	Y	Y
-remote	Y	Y	Y	Y	Y			
-reverse	Y	Y	Y	Y	Y			
-rrbcv	Y	Y	Y	Y	Y	Y	Y	Y
-sid	Y	Y	Y	Y	Y	Y	Y	Y
-skip	Y	Y	Y					
-star	Y	Y	Y	Y	Y			
-std_protect			Y					
-v	Y	Y	Y	Y	Y	Y	Y	Y

Table 24 lists the `symmir` view arguments and the possible options to use when targeting a specified composite group.

Table 24 symmir -cg view arguments and possible options (page 1 of 2)

Options	Argument action	
	query	verify
-attach	Y	
-bcv, -rrbcv	Y	Y
-bcv_mirrors		Y
-bg	Y	Y
-c, -i	Y	Y
-concurrent		Y
-hop2	Y	Y
-force		Y
-multi	Y	
-offline	Y	Y

Table 24 symmir -cg view arguments and possible options (page 2 of 2)

Options	Argument action	
	query	verify
-percent	Y	Y
-protect	Y	
-rdf	Y	Y
-restored		Y
-sid	Y	Y
-split		Y
-synched		Y
-syncinprog		Y
-ready		Y
-restinprog		Y
-protbcvest	Y	
-summary	Y	

Command options with device files

With the `symmir -file` command, you can perform similar control operations on BCV device pairs defined in a device file of a specified array as you can when directing `symmir` to device groups (`-g`). These control operations (arguments) have similar options that allow flexibility in controlling STD/BCV pairs defined in a device file, as opposed to a device group. This command is particularly useful when operating on RDF BCV pairs in a remote array in the second-level multihop SRDF link.

Table 25 lists the `symmir` control operations and the possible options to use when targeting pairs specified in a device file of a given array.

Table 25 symmir -file control arguments and possible options (page 1 of 2)

Options	Argument Action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
-bypass			Y	Y	Y			
-c, -i	Y	Y	Y	Y	Y	Y	Y	Y
-consistent			Y					
-diff			Y					
-force, -symforce	Y	Y	Y	Y	Y			Y
-noprompt	Y	Y	Y	Y	Y	Y	Y	Y
-not_ready			Y	Y	Y			
-preaction, -postaction	Y	Y	Y	Y	Y			

Table 25 symmir -file control arguments and possible options (page 2 of 2)

Options	Argument Action							
	establish -full	establish	split	restore -full	restore	attach	detach	cancel
-preservetgtlocks, -lockid	Y	Y	Y	Y	Y			
-protbcvest	Y	Y						
-protect			Y	Y	Y			
-remote	Y	Y	Y	Y	Y			
-reverse	Y	Y	Y	Y	Y			
-skip	Y	Y	Y					
-star	Y	Y	Y	Y	Y			Y
-std_protect			Y					
-v	Y	Y	Y	Y	Y	Y	Y	Y

Table 26 lists the `symmir` view arguments and the possible options to use when targeting pairs specified in a device file of a given array.

Table 26 symmir -file view arguments and possible options

Options	Argument Action	
	query	verify
-attach	Y	
-bcv_mirrors		Y
-bg	Y	Y
-c, -i	Y	Y
-concurrent		Y
-force	Y	Y
-multi	Y	
-offline	Y	Y
-percent	Y	Y
-protbcvest	Y	
-protect	Y	
-ready		Y
-restinprog		Y
-restored		Y
-split		Y

Table 26 symmir -file view arguments and possible options (continued)

Options	Argument Action	
	query	verify
-summary	Y	
-synched		Y
-syncinprog		Y

Note: The Symmetrix ID option (-sid) is required for all symmir -file commands.

Various remote multihop configurations

Various compounded remote configurations can be managed by your host using both the TimeFinder and SRDF components of SYMCLI.

As Figure 56 on page 182 shows, you can have multiple sites (for example, remote sites C, E, F, and H) on SRDF links to remotely mirror a local array at site A. Remote site F, functioning as a remote mirror to the standard devices at site A, is most typical. You then can have a third site on an SRDF link (remote site H) to remotely mirror just the BCV devices in the array at site A.

You can also multihop to a second level SRDF where Remote site G functions as a remote mirror to the standard devices of site A and Remote site I remotely mirrors Site A's BCV.

In addition, you can also create a cascaded SRDF configuration, where tertiary site B functions as a remote partner to the R21 device at Site C, which is the remote partner of the local RDF standard device at Site A; and tertiary site D functions as a remote partner to the R21 device at Site E, which is the remote partner of the local BCV device at Site A.

Command symmir manages each of the BCV pairs at any site while symrdf manages the SRDF pairs in the SRDF link.

System-wide device groups

Before you begin applying any symmir operations, you must be working with an existing group of RDF devices. To create a device group containing STD and BCV RDF1 devices, enter:

```
symdg create prod -type RDF1
symdg -g prod add dev 0001 -sid 0001 DEV001
symbcv -g prod associate dev 000A BCV001
symbcv -g prod associate dev 000C -rdf RBCV001
symbcv -g prod associate dev 0009 -bcv -rdf BRBCV001
symbcv -g prod associate dev 0004 -rrdf RRBCV001
symbcv -g prod associate dev 0004 -hop2 2BCV001
```

At this point, all these devices must be established with the symmir and symrdf commands.

Commands to various multihop devices and links

This section describes the command application of targeting the various devices and links in complex multihop SRDF environments.

The following sequence of commands steps through some basic control operations that touch every device and RDF link in a complex multihop configuration. The following numbering of commands directly associates with the bubble numbers shown in the Figure 56 on page 182.

1	`symrdf -g <> establish`	Creates the standard-associated hop 1 copy.
2	`symmir -g <> split -rdf`	Splits the standard-associated hop 1 BCV device pair.
3	`symrdf -g <> establish -rbcv`	Creates the standard-associated hop 2 copy.
4	`symrdf -g <> restore -rbcv`	Restores the standard-associated hop 1 BCV with the hop 2 copy.
5	`symmir -g <> restore -rdf`	Restores the standard-associated hop 1 copy with the hop 1 BCV.
6	`symrdf -g <> restore`	Restores the standard device with the hop 1 copy.
7	`symmir -g <> split`	Splits the standard/BCV pair.
8	`symrdf -g <> establish -bcv`	Creates the BCV-associated hop 1 remote copy.
9	`symmir -g <> split -rdf -bcv`	Splits the BCV-associated hop 1 device pair.
10	`symrdf -g <> establish -brbcv`	Creates the BCV-associated hop 2 copy.
11	`symrdf -g <> restore -brbcv`	Restores the BCV-associated hop 1 BCV with the hop 2 copy.
12	`symmir -g <> restore -rdf -bcv`	Restores the BCV-associated hop 1 copy with the hop 1 BCV.
13	`symrdf -g <> restore -bcv`	Restores the BCV device with the hop 1 copy.
14	`symmir -g <> restore`	Restores the standard device with the BCV copy.
15	`symmir -file <> -sid 044 establish`	Creates the BCV-associated with the remote partner of the BRBCV (file only).
16	`symmir -file <> -sid 044 split`	Splits the BCV-associated with the remote partner of the BRBCV (file only).
17	`symrdf -g <> establish`	Creates the standard-associated hop 1 copy.
18	`symrdf -g <> establish -hop2`	Creates the standard-associated hop 2 copy.
19	`symmir -g <> establish -hop2`	Creates the BCV-associated hop 2 BCV copy.
20	`symmir -g <> split -hop2`	Splits the BCV-associated hop 2 device pair.
21	`symrdf -g <> restore -hop2`	Restores the standard with the hop 2 copy.
22	`symrdf -g <> restore`	Restores the standard device with the hop 1 copy.
23	`symrdf -g <> establish -bcv`	Creates the BCV-associated hop 1 remote copy.
24	`symrdf -g <> establish -bcv -hop2`	Creates the BCV-associated hop 2 copy.
25	`symrdf -g <> restore -bcv -hop2`	Restores the BCV-associated hop 2 copy.
26	`symrdf -g <> restore -bcv`	Restores the BCV device with the hop 1 copy.

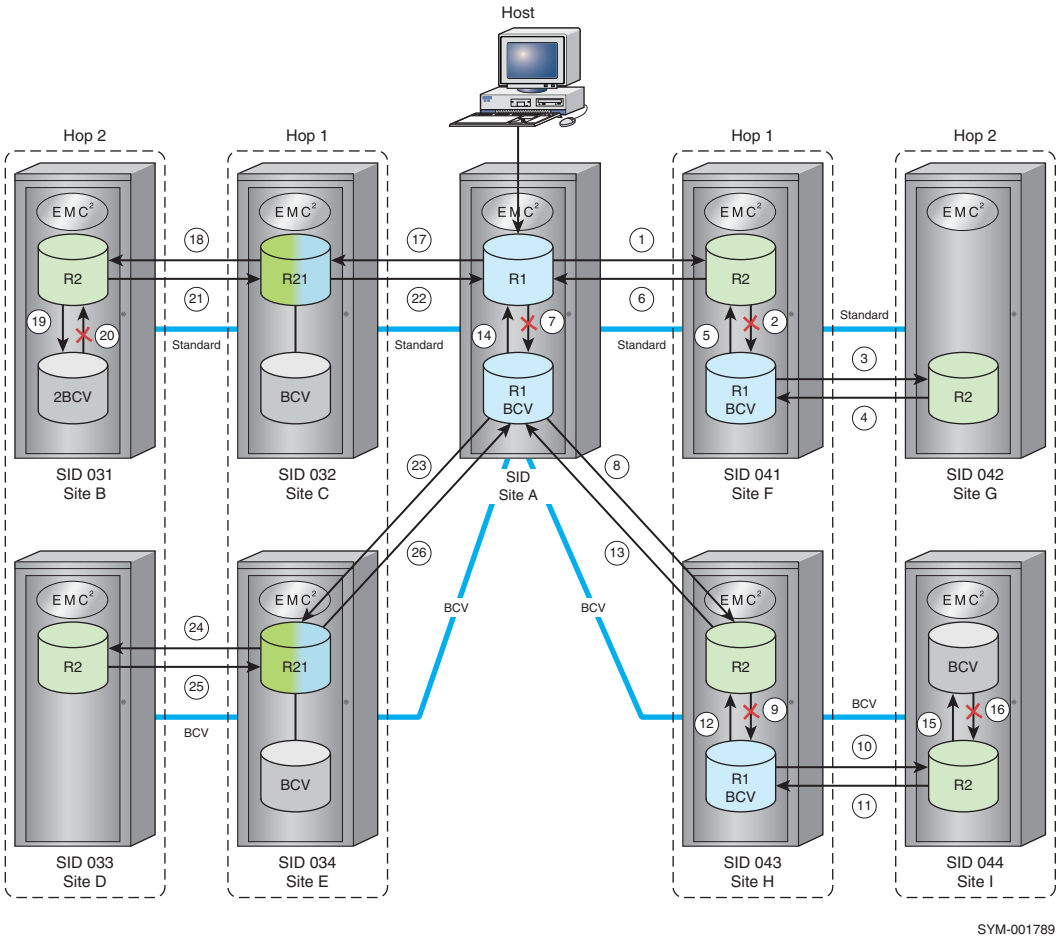

Figure 56 Control operations on multihop SRDF configurations

Second-level controls for multihop SRDF environments

As previously described, second-level multihop control operations were accomplished using the device file (`-file`) option for managing RDF BCV device pairs. You can use the `symmir` command for device groups (`-g`) for BCV control capability in second-level multihop SRDF environments.

The remote RDF BCV (RRBCV) devices must have been previously associated with the device group using the `symbcv -rrdf` command. "Compounded remote configuration" on page 124 contains specific information about how to associate second-level multihop BCVs with a device group.

Once the RRBCV devices have been associated with the device group, you can use the `symmir` command with the `-rrbcv` option to perform control operations on the remote mirror of the remote BCV to become established, split, or restored from its BCV. Other second-level multihop BCV control operations available with the `symmir` command include `query`, `verify`, `attach`, `detach`, and `cancel`.

Remote optimizing option

The remote optimize (`-opt_rag`) option only applies to the full Establish operation in a remote array that optimizes the disk I/O on the standard/BCV pair selection to achieve a high copy speed between them. (Basically, the device pair selection attempts to pair devices that are not on the same disk adapter to distribute I/O.) This option overrides all current pairing relationships.

Note: This option is only applicable for remote array optimization targeting composite groups (`-cg`).

The command line must include the `-rdf` option, as follows:

```
symmir -cg CgName -full establish -rdf -opt_rag
```

Using the -remote option on multihop split actions

This section describes the command application of targeting the various devices and links with the `-remote` option in complex multihop SRDF environments.

The following sequence of commands steps through some basic control operations that touch every device and RDF link in a complex multihop configuration. The following numbering of commands directly associates with the callouts shown in the Figure 57 on page 184.

1	`symrdf -g <> establish`	Creates the standard-associated hop 1 copy.
2	`symmir -g <> split -rdf -remote`	Splits the standard-associated hop 1 BCV device pair and creates a standard-associated hop 2 copy of the hop 1 BCV.
3	`symmir -g <> split -remote`	Splits the standard/BCV pair and creates a BCV-associated hop 1 copy of the local BCV.
4	`symmir -g <> split -rdf -bcv -remote`	Splits the BCV-associated hop 1 BCV device pair and creates a BCV-associated hop 2 copy of the hop 1 BCV.
5	`symmir -g <> split -rrbcv`	Splits the BCV-associated hop 2 BCV device pair. You cannot use the `-remote` option here.

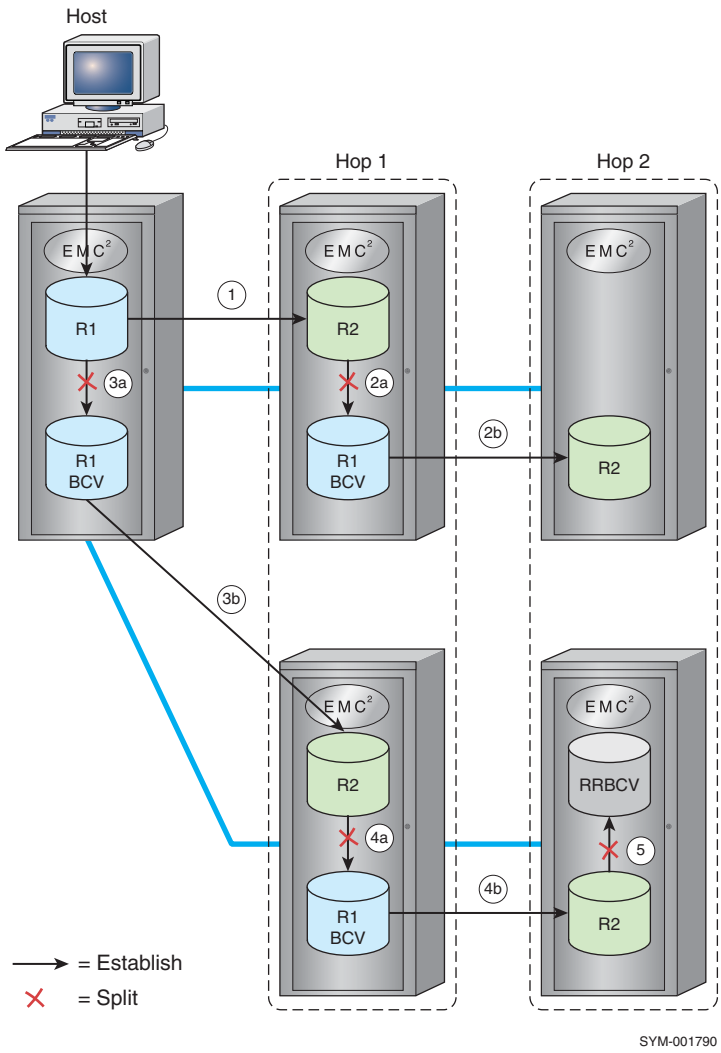

Figure 57 The -remote option on multihop configurations

APPENDIX A
TimeFinder State Rules Reference

This appendix helps you determine if the TimeFinder operation you want to complete is supported for devices in a particular pair state.

- TimeFinder SnapVX operations 186
- Using the tables for Clone, VP Snap, Snap, and Mirror 187
- TimeFinder/Clone operations 189
- VP Snap operations 201
- TimeFinder/Snap operations 207
- TimeFinder/Mirror operations 215

TimeFinder SnapVX operations

Table 27 describes the prerequisites, transient states, and final states for all SnapVX control operations.

Table 27 Prerequisites and states for SnapVX control operations.

Operation	Prerequisite	Transient State	Final State
Establish	None	Establish In Progress	Established
Restore	Established If the source device is a link target, it must be fully copied before initiating the restore operation.	Restore In Progress	Restored
Terminate	Established	Terminate In Progress	NA
Terminate with FLAG1_RESTORED	Restored	NA	NA
Setmode Copy	Linked NoCopy	Link Copy In Progress	Link Copied
Setmode NoCopy	Link Copy In Progress or Link Copied	NA	Linked
Set TTL	Established with no links or restores.	NA	NA
Link	Established	NA	Linked
Link with FLAG1_COPY	Established	Link Copy In Progress	Link Copied
Unlink	Linked or Link Copied If the target is the source of another snapshot, the link must be fully copied.	NA	NA
Relink	Linked or Link Copied	NA	Linked
Relink with FLAG1_COPY	Linked or Link Copied	Link Copy In Progress	Link Copied
Rename	Established	NA	NA

Using the tables for Clone, VP Snap, Snap, and Mirror

To determine which tables you should view for a particular configuration:

1. Go to the appropriate section for the operation that you want to perform and look at the basic operations table. For example, if you want to perform a TimeFinder/Clone operation, go to "TimeFinder/Clone operations" and look at Table 28, "Basic TimeFinder/Clone operations," If the operation is supported on the device in its current pair state, go to the next step.

2. If the device is part of a concurrent copy operation, look at the appropriate table in the concurrent operations section. A concurrent copy operation occurs when multiple copies are made from the same source. These copies can be any combination of TimeFinder/Clone, TimeFinder/Mirror, and TimeFinder/Snap.

   Figure 58 illustrates a concurrent copy operation.

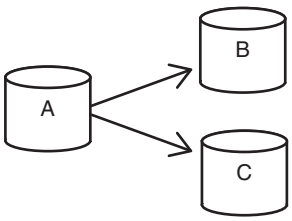

Figure 58 Concurrent copy operation

   For example, if you want to complete a TimeFinder/Clone operation on a device that is part of a concurrent copy operation in which both a clone and a snap are made from the same source device, look at Table 31, "Concurrent TimeFinder/Clone operations: Clone and Snap," If the operation is supported on the device in its current pair state, go to the next step.

3. If the device is part of a cascaded copy operation, look at the appropriate table in the cascaded operations section. A cascaded copy operation occurs when a device is both the source of a copy and the target of a copy, such that the devices are in the relationship of A -> B -> C. Examples of cascaded operations include Clone off Clone, Snap off Clone, and Snap off BCV.

   Figure 58 illustrates a cascaded copy operation.

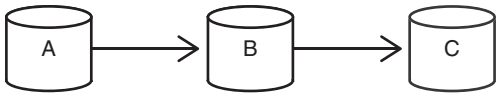

Figure 59 Cascaded copy operation

   For cascaded operations, you must consider which hop in the cascade you are interested in. For example, assume that your device is part of a Snap off Clone operation in which a device (B) is the target of a clone (A -> B) and the source of a snap (B -> C).

Figure 58 illustrates this example.

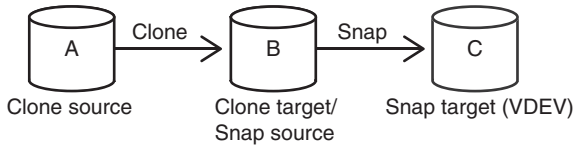

Figure 60 Cascaded copy example (Snap off Clone)

If you are interested in the TimeFinder/Clone hop of the cascade, look at Table 36, "Snap off Clone operations: Source -> Target," on page 200 of the TimeFinder/Clone Operations section. If you are interested in the TimeFinder/Snap hop of the cascade, look at Table 48, "Snap off Clone operations: B -> VDEV," on page 214 of the TimeFinder/Snap Operations section.

If the operation you want to complete is listed as supported in each of the appropriate tables, then that operation is supported on your device.

TimeFinder/Clone operations

This section describes the TimeFinder/Clone control operations that are allowed for devices in various pair states.

Basic TimeFinder/Clone operations

Figure 61 illustrates the relationship between devices for a basic TimeFinder/Clone operation.

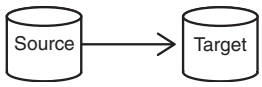

Figure 61 Basic TimeFinder/Clone device relationship

In Table 28 on page 189, "Pair state" refers to the current state of the source-target pair, and "Operation" refers to the intended control operation. Y indicates that the operation is allowable for that state.

Table 28 Basic TimeFinder/Clone operations (page 1 of 2)

TimeFinder/Clone Source -> Target Pair state: / TimeFinder/Clone Source -> Target Operation:	No Session	Create in Progress[1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress[1]	Invalid	Failed
Create	Y														
Recreate [2]						Y	Y			Y					Y [8]
Activate			Y	Y	Y										
Full Establish	Y														
Incremental Establish						Y	Y			Y					

Table 28 Basic TimeFinder/Clone operations (page 2 of 2)

TimeFinder/Clone Source -> Target Pair state: / TimeFinder/Clone Source -> Target Operation:	No Session	Create in Progress[1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress[1]	Invalid	Failed
Set Mode Copy [3]			Y	Y	Y			Y	Y						
Set Mode Nocopy [3, 7]			Y	Y	Y										
Set Mode Precopy [3]			Y	Y											
Full Restore [2]	Y						Y			Y					
Incremental Restore [2]							Y			Y					
Split					Y	Y			Y			Y			
Terminate		Y	Y	Y	Y	Y [4, 5]	Y	Y	Y	Y	Y [4, 6]	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.
3. If not already set to this mode.
4. Requires -symmforce (not recommended).
5. Target data will be incomplete.
6. Source data will be incomplete.
7. Cannot be differential.
8. Requires Enginuity 5876.163.105 and higher.

Concurrent TimeFinder/Clone operations

A concurrent copy occurs when a source device is copied to multiple different targets. These copy operations can be any combination of TimeFinder/Clone, TimeFinder Snap, and TimeFinder/Mirror copies.

Concurrent TimeFinder/Clone

Figure 62 illustrates the relationship between devices for a TimeFinder/Clone pairing of a source device with two targets.

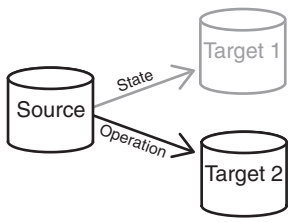

Figure 62 Concurrent TimeFinder/Clone device relationships

Note: The positions of Target 1 and Target 2 are interchangeable in the illustration above.

In Table 29, “Pair state” refers to the current state of the Source-Target 1 pair, and “Operation” refers to the intended control operation on the Source-Target 2 pair. Y indicates that the operation is allowable for that state.

Table 29 Concurrent TimeFinder/Clone operations

TimeFinder/Clone Source -> Target 1 Pair state: / TimeFinder/Clone Source -> Target 2 Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y		Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]			
Recreate [2]			Y	Y	Y	Y	Y	Y	Y	Y	Y	Y			Y [8]
Activate			Y	Y	Y	Y	Y	Y	Y	Y		Y			
Full Establish	Y		Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y [9]	Y[9]		Y [9]			
Incremental Establish			Y	Y	Y	Y	Y	Y	Y	Y		Y			
Set Mode Copy [3]			Y	Y	Y	Y	Y	Y	Y	Y		Y			
Set Mode Nocopy [3, 7]			Y	Y	Y	Y	Y	Y	Y	Y		Y			
Set Mode Precopy [3]			Y	Y	Y	Y	Y	Y	Y	Y		Y			
Full Restore [2]	Y						Y			Y					
Incremental Restore[2]										Y					
Split			Y	Y			Y			Y					
Terminate		Y	Y	Y	Y	Y [4, 5]	Y	Y	Y	Y	Y [4, 6]	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.
3. If not already set to this mode.
4. Requires -symmforce (not recommended).
5. Target data will be incomplete.
6. Source data will be incomplete.
7. Cannot be differential.
8. Requires Enginuity 5876.163.105 and higher.
9. Requires -concurrent flag only for dgleg operations.

Concurrent TimeFinder/Clone: Clone and Mirror

Figure 63 illustrates the relationship between devices for a concurrent pairing of a source device with a TimeFinder/Clone target and a TimeFinder/Mirror BCV.

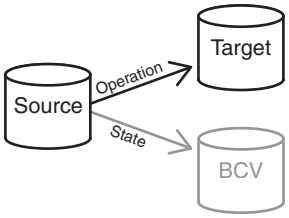

Figure 63 Concurrent TimeFinder/Clone device relationships: Clone and Mirror

Note: The positions of Target and BCV are interchangeable in the illustration above.

In Table 30, "Pair state" refers to the current state of the STD-BCV pair, and "Operation" refers to the intended control operation on the Source-Target pair. Y indicates that the operation is allowable for that state.

Table 30 Concurrent TimeFinder/Clone operations: Clone and Mirror

TimeFinder/Mirror STD -> BCV Pair state: / TimeFinder/Clone Source -> Target Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Create	Y	Y [5]	Y [5]		Y [4, 5]	Y [5]		Y [5]	Y [5]	
Recreate [1]	Y	Y	Y		Y [4]	Y		Y	Y	
Activate	Y	Y	Y		Y [4]	Y		Y	Y	
Full Establish	Y [5]	Y [5]	Y [5]		Y [4, 5]	Y [5]		Y [5]	Y [5]	
Incremental Establish	Y	Y	Y		Y [4]	Y		Y	Y	
Set Mode Copy [2]	Y	Y	Y		Y [4]	Y		Y	Y	
Set Mode Nocopy [2, 3]	Y	Y	Y		Y [4]	Y		Y	Y	
Set Mode Precopy [2,]	Y	Y	Y		Y [4]	Y		Y	Y	
Full Restore [1]	Y				Y [4]	Y		Y [4]	Y	Y
Incremental Restore [1]	Y				Y [4]	Y		Y [4]	Y	
Split	Y		Y [4]			Y		Y [4]	Y	
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Session must be differential.
2. If not already set to this mode.
3. Cannot be differential.

4. Native TimeFinder/Mirror only.
5. Requires -concurrent flag only for dgleg operations.

Concurrent TimeFinder/Clone: Clone and Snap

Figure 64 illustrates the relationship between devices for a concurrent pairing of a source device with a TimeFinder/Clone target and a TimeFinder/Snap VDEV.

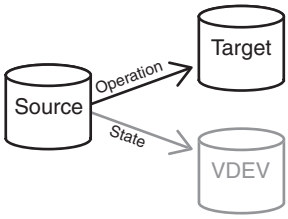

Figure 64 Concurrent TimeFinder/Clone device relationships: Clone and Snap

Note: The positions of Target and VDEV are interchangeable in the illustration above.

In Table 31, "Pair state" refers to the current state of the Source-VDEV pair, and "Operation" refers to the intended control operation on the Source-Target pair. Y indicates that the operation is allowable for that state.

Table 31 Concurrent TimeFinder/Clone operations: Clone and Snap

TimeFinder/Snap Source -> VDEV Pair state: / TimeFinder/Clone Source -> Target Operation:	No Session	Create in Progress [1]	Created	Recreated	COPIED	Copy on Write	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y		Y	Y	Y	Y					
Recreate [2]	Y		Y	Y	Y	Y					Y [5]
Activate	Y		Y	Y	Y	Y					
Full Establish	Y		Y	Y	Y	Y					
Incremental Establish	Y		Y	Y	Y	Y					
Set Mode Copy [3]	Y		Y	Y	Y	Y					
Set Mode Nocopy [3, 4]	Y		Y	Y	Y	Y					
Set Mode Precopy [3]	Y		Y	Y	Y	Y					
Full Restore [2]	Y										
Incremental Restore [2]	Y										
Split	Y										
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.

3. If not already set to this mode.
4. Cannot be differential.
5. Requires Enginuity 5876.163.105 and higher.

Cascaded TimeFinder/Clone operations

A cascaded copy operation occurs when a device is both the source of a copy and the target of a copy, such that the devices are in the relationship of A -> B -> C.

Cascaded TimeFinder/Clone (Clone off Clone): A -> B

Figure 65 illustrates the relationship between devices for a Clone off Clone operation in which source device A is paired with target device B and an additional session uses device B as a source paired with target device C.

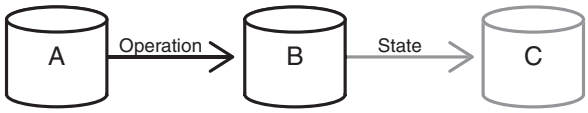

Figure 65 Clone off Clone device relationship: A -> B

Limitations

- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 32, “Pair state” refers to the current state of the B-C pair, and “Operation” refers to the intended control operation on the A-B pair. Y indicates that the operation is allowable for that state.

Table 32 Clone off Clone operations: A -> B (page 1 of 2)

TimeFinder/Clone B -> C Pair state: / TimeFinder/Clone A -> B Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y		Y [4]	Y [4]	Y		Y			Y					
Recreate [2]	Y		Y [4]	Y [4]	Y		Y			Y					Y [7]
Activate	Y		Y	Y	Y		Y			Y					
Full Establish	Y		Y	Y	Y		Y			Y					
Incremental Establish	Y		Y	Y	Y		Y			Y					
Set Mode Copy [3]	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Set Mode Nocopy [3, 6]	Y		Y	Y	Y	Y	Y	Y	Y	Y					

Table 32 Clone off Clone operations: A -> B (page 2 of 2)

TimeFinder/Clone B -> C Pair state: / TimeFinder/Clone A -> B Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Set Mode Precopy [2]	Y				Y		Y			Y					
Full Restore [2]	Y		Y	Y	Y	Y	Y	Y	Y	Y		Y			
Incremental Restore [2]	Y		Y	Y	Y	Y	Y	Y	Y			Y [5]			
Split	Y		Y	Y	Y	Y	Y	Y	Y	Y		Y			
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.
3. If not already set to this mode.
4. Not allowed with the PRECOPY flag.
5. Requires Enginuity 5875 and higher.
6. Cannot be differential.
7. Requires Enginuity 5876.163.105 and higher.

Cascaded TimeFinder/Clone (Clone off Clone): B -> C

Figure 66 illustrates the relationship between devices for a Clone off Clone operation in which source device A is paired with target device B and an additional session uses device B as a source paired with target device C.

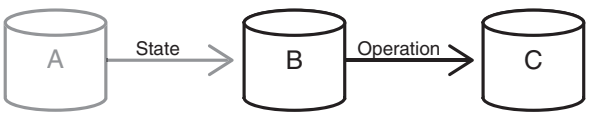

Figure 66 Clone off Clone device relationship: B -> C

Limitations:

- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 33 on page 196, “Pair state” refers to the current state of the A-B pair, and “Operation” refers to the intended control operation on the B-C pair. Y indicates that the operation is allowable for that state.

Table 33 Clone off Clone operations: B -> C

TimeFinder/Clone A -> B Pair state: / TimeFinder/Clone B -> C Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y		Y	Y	Y [4]	Y	Y	Y	Y	Y	Y	Y			
Recreate [2]	Y		Y	Y	Y [4]	Y	Y	Y	Y	Y	Y	Y			Y [7]
Activate	Y						Y			Y	Y	Y			
Full Establish	Y						Y			Y	Y	Y			
Incremental Establish	Y						Y			Y	Y	Y			
Set Mode Copy [3]	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y			
Set Mode Nocopy [3, 6]	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y			
Set Mode Precopy [3]	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y			
Full Restore [2]	Y											Y			
Incremental Restore [2]	Y						Y [5]			Y [5]		Y			
Split	Y						Y [5]			Y	Y	Y			
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.
3. If not already set to this mode.
4. Requires the PRECOPY flag.
5. Requires Enginuity 5875 and higher.
6. Cannot be differential.
7. Requires Enginuity 5876.163.105 and higher.

Mirror off Clone: Source -> STD

Figure 67 illustrates the relationship between devices for a Mirror off Clone operation in which the TimeFinder/Clone source device is paired with a target device and an additional session uses that device as the STD for TimeFinder/Mirror paired with a BCV.

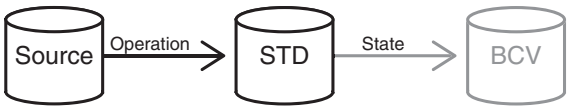

Figure 67 Mirror off Clone device relationship

Limitations:

- Thin BCVs are only supported for Enginuity 5875 and higher.
- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 34, "Pair state" refers to the current state of the STD-BCV pair, and "Operation" refers to the intended control operation on the Source-STD pair. Y indicates that the operation is allowable for that state.

Table 34 Mirror off Clone operations: Source -> STD

TimeFinder Mirror STD -> BCV Pair state: / TimeFinder/Clone Source -> STD Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Create	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Recreate [1]	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Activate	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Full Establish	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Incremental Establish	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Set Mode Copy [2]	Y		Y		Y [5]	Y	Y	Y	Y [5]	Y [5]
Set Mode Nocopy [2, 4]	Y		Y		Y [5]	Y	Y	Y	Y [5]	Y [5]
Set Mode Precopy [2]	Y		Y		Y [5]	Y		Y	Y [5]	Y [5]
Full Restore [1]	Y		Y		Y	Y	Y	Y	Y [5]	
Incremental Restore [1]	Y		Y		Y [3]	Y	Y	Y	Y [5]	
Split	Y		Y		Y [5]	Y		Y	Y [5]	
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Session must be differential.
2. If not already set to this mode.
3. Requires Enginuity 5875 and higher.
4. Cannot be differential.
5. Native TimeFinder/Mirror only.

Clone off Mirror: BCV -> Target

Figure 68 illustrates the relationship between devices for a Clone off Mirror operation in which the TimeFinder/Mirror STD device is paired with a BCV device and an additional session uses that device as the source for TimeFinder/Clone paired with a target device.

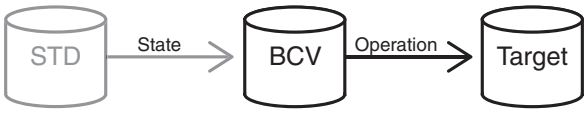

Figure 68 Clone off Mirror device relationship

Limitations:

- Thin BCVs are only supported for Enginuity 5875 and higher.
- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 35 on page 199, "Pair state" refers to the current state of the STD-BCV pair, and "Operation" refers to the intended control operation on the BCV-Target pair. Y indicates that the operation is allowable for that state.

Table 35 Clone off Mirror operations: BCV -> Target

TimeFinder/Mirror STD -> BCV Pair state: / TimeFinder/Clone BCV -> Target Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Create	Y					Y	Y	Y		
Recreate [1]	Y					Y	Y	Y		
Activate	Y					Y		Y		
Full Establish	Y					Y		Y		
Incremental Establish	Y					Y		Y		
Set Mode Copy [2]	Y					Y	Y	Y		
Set Mode Nocopy [2, 4]	Y					Y	Y	Y		
Set Mode Precopy [2]	Y					Y	Y	Y		
Full Restore [1]	Y					Y		Y		
Incremental Restore [1]	Y					Y		Y [3]		
Split	Y					Y		Y		
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Session must be differential.
2. If not already set to this mode.
3. Requires Enginuity 5875 and higher.
4. Cannot be differential.

Snap off Clone: Source -> Target

Figure 69 illustrates the relationship between devices for a Snap off Clone operation in which the TimeFinder/Clone source device is paired with a target device and an additional session uses the target device as the source for TimeFinder/Snap paired with a VDEV.

Figure 69 Snap off Clone device relationship

Note: The TimeFinder/Snap session is not considered to be a hop and does not contribute to the two hop maximum.

In Table 36, "Pair state" refers to the current state of the Target-VDEV pair, and "Operation" refers to the intended control operation on the Source-Target pair. Y indicates that the operation is allowable for that state.

Table 36 Snap off Clone operations: Source -> Target

TimeFinder/Snap Target -> VDEV Pair state: / TimeFinder/Clone Source -> Target Operation:	No Session	Create in Progress [1]	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y										
Recreate [2]	Y				Y [5]	Y [5]					Y [7]
Activate	Y				Y [6]	Y [6]					
Full Establish	Y										
Incremental Establish	Y										
Set Mode Copy [3]	Y										
Set Mode Nocopy [3, 4]	Y										
Set Mode Precopy [3]	Y										
Full Restore [2]	Y										
Incremental Restore [2]	Y							Y			
Split	Y				Y	Y					
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Session must be differential.
3. If not already set to this mode.
4. Cannot be differential.
5. If the TimeFinder/Clone target has a TimeFinder/Snap session in any state, recreate is allowed only with precopy mode.
6. If the TimeFinder/Clone target has a TimeFinder/Snap session in any state, activate is allowed only after the precopy operation has completed one cycle. This can be verified with a symclone query.
7. Requires Enginuity 5876.163.105 and higher.

VP Snap operations

VP Snap operations have some additional rules not covered in the previous tables.

Basic VP Snap operations

Figure 70 illustrates the relationship between devices for a basic VP Snap operation.

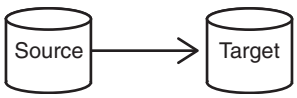

Figure 70 Basic VP Snap device relationship

In Table 37, “Pair state” refers to the current state of the Source-Target pair, and “Operation” refers to the intended control operation on the Source-Target pair. Y indicates that the operation is allowable for that state.

Table 37 Basic VP Snap operations (page 1 of 2)

VP Snap Source -> Target Pair state: / VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Create	Y								
Recreate				Y	Y				
Activate		Y	Y						
Full Establish	Y								
Incremental Establish				Y	Y				
Set Mode Copy									
Set Mode Nocopy									
Set Mode Precopy									
Full Restore									

Table 37 Basic VP Snap operations (page 2 of 2)

VP Snap Source -> Target Pair state: / VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Incremental Restore				Y	Y				
Split									
Terminate		Y	Y	Y	Y	Y	Y	Y	Y

Concurrent VP Snap: VP Snap with additional VP Snap

Figure 71 illustrates the relationship between devices for a VP Snap pairing of a source device with two targets.

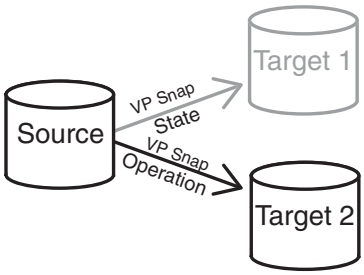

Figure 71 Concurrent VP Snap device relationships

In Table 38, “Pair state” refers to the current state of the Source-Target 1 pair, and “Operation” refers to the intended control operation on the Source-Target 2 pair. Y indicates that the operation is allowable for that state.

Table 38 Concurrent VP Snap: VP Snap with additional VP Snap

VP Snap Source -> Target Pair state: / Additional VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Create	Y	Y	Y	Y	Y			Y	Y
Recreate	Y	Y	Y	Y				Y	Y
Activate	Y	Y	Y	Y	Y			Y	Y
Full Establish	Y	Y	Y	Y	Y			Y	Y
Incremental Establish	Y	Y	Y	Y	Y			Y	Y
Set Mode Copy									
Set Mode Nocopy									
Set Mode Precopy									

Table 38 Concurrent VP Snap: VP Snap with additional VP Snap (continued)

VP Snap Source -> Target Pair state: / Additional VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Full Restore									
Incremental Restore	Y			Y	Y				
Split									
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y

Concurrent VP Snap: VP Snap with additional TimeFinder/Clone

Figure 72 illustrates the relationship between devices for a concurrent pairing of a source device with a VP Snap target and an additional TimeFinder/Clone target.

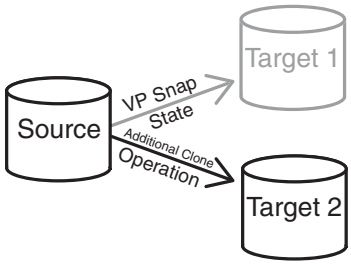

Figure 72 Concurrent VP Snap: VP Snap with additional TimeFinder/Clone

In Table 39, “Pair state” refers to the current state of the VP Snap Source-Target 1 pair, and “Operation” refers to the intended control operation on the TimeFinder/Clone Source-Target 2 pair. Y indicates that the operation is allowable for that state.

Table 39 Concurrent VP Snap: VP Snap with additional TimeFinder/Clone (page 1 of 2)

VP Snap Source -> Target Pair state: / Additional TimeFinder/Clone Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Create	Y	Y	Y	Y	Y			Y	Y
Recreate	Y	Y	Y	Y	Y				
Activate	Y	Y	Y	Y	Y			Y	Y
Full Establish	Y	Y	Y	Y	Y			Y	Y
Incremental Establish	Y	Y	Y	Y	Y			Y	Y
Set Mode Copy	Y	Y	Y	Y	Y			Y	Y

Table 39 Concurrent VP Snap: VP Snap with additional TimeFinder/Clone (page 2 of 2)

VP Snap Source -> Target Pair state: / Additional TimeFinder/Clone Source -> Target Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Set Mode Nocopy	Y							Y	Y
Set Mode Precopy	Y	Y	Y	Y	Y			Y	Y
Full Restore	Y							Y	Y
Incremental Restore	Y			Y	Y			Y	Y
Split	Y	Y	Y	Y	Y			Y	Y
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y

Concurrent VP Snap: TimeFinder/Clone with additional VP Snap

Figure 73 illustrates the relationship between devices for a concurrent pairing of a source device with a VP Snap target and an additional TimeFinder/Clone target.

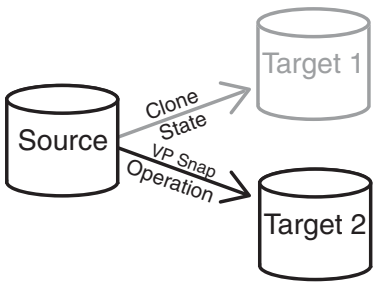

Figure 73 Concurrent VP Snap: TimeFinder/Clone with additional VP Snap

In Table 40, “Pair state” refers to the current state of the TimeFinder/Clone Source-Target 1 pair, and “Operation” refers to the intended control operation on the VP Snap Source-Target 2 pair. Y indicates that the operation is allowable for that state.

Table 40 Concurrent VP Snap: TimeFinder/Clone with additional VP Snap (page 1 of 2)

TimeFinder/Clone Source -> Target Pair state: / Additional VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Invalid	Failed
Create	Y	Y	Y	Y	Y			Y		Y	Y	Y
Recreate	Y	Y	Y	Y	Y			Y		Y	Y	Y
Activate	Y	Y	Y	Y	Y			Y		Y	Y	Y
Full Establish	Y	Y	Y	Y	Y			Y		Y	Y	Y

Table 40 Concurrent VP Snap: TimeFinder/Clone with additional VP Snap (page 2 of 2)

TimeFinder/Clone Source -> Target Pair state: / Additional VP Snap Source -> Target Operation:	No Session	Created	Recreated	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Invalid	Failed
Incremental Establish	Y	Y	Y	Y	Y			Y		Y	Y	Y
Set Mode Copy												
Set Mode Nocopy												
Set Mode Precopy												
Full Restore												
Incremental Restore	Y				Y			Y				
Split												
Terminate	Y	Y	Y	Y	Y			Y	Y	Y	Y	Y

Cascaded VP Snap (VP Snap off Clone): A -> B

Figure 74 illustrates the relationship between devices for a VP Snap off Clone operation in which source device A is paired with target device B and an additional VP Snap session uses device B as a source paired with target device C.

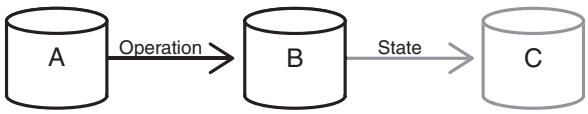

Figure 74 VP Snap off Clone device relationship: A -> B

In Table 41, “Pair state” refers to the current state of the B-C pair, and “Operation” refers to the intended control operation on the A-B pair. Y indicates that the operation is allowable for that state.

Table 41 VP Snap off Clone operations: A -> B

VP Snap B -> C Pair state: / TimeFinder/Clone A -> B Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Create	Y								
Recreate	Y			Y [1]	Y [1]				
Activate	Y			Y [2]	Y [2]				
Full Establish	Y								

Table 41 VP Snap off Clone operations: A -> B (continued)

VP Snap B -> C Pair state: TimeFinder/Clone A -> B Operation:	No Session	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Invalid	Failed
Incremental Establish	Y								
Set Mode Copy	Y								
Set Mode Nocopy	Y								
Set Mode Precopy	Y								
Full Restore	Y								
Incremental Restore	Y			Y	Y		Y		
Split	Y			Y	Y				
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. If device B has a VP Snap session in any state, recreate is allowed only with precopy mode.
2. If device B has a VP Snap session in any state, activate is allowed only after the precopy operation has completed one cycle. This can be verified with a symclone query.

Cascaded VP Snap (VP Snap off Clone): B -> C

Figure 75 illustrates the relationship between devices for a VP Snap off Clone operation in which source device A is paired with target device B and an additional VP Snap session uses device B as a source paired with target device C.

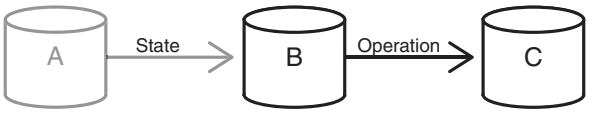

Figure 75 VP Snap off Clone device relationship: B -> C

In Table 42, “Pair state” refers to the current state of the A-B pair, and “Operation” refers to the intended control operation on the B-C pair. Y indicates that the operation is allowable for that state.

Table 42 VP Snap off Clone operations: B -> C

TimeFinder/Clone A -> B Pair state: VP Snap B -> C Operation:	No Session	Created	Recreated	Copy in Progress	Copied	Copy on Write	Copy on Access	Restore in Progress	Restored	Split	Invalid	Failed
Create	Y				Y					Y		
Recreate	Y				Y					Y		
Activate	Y				Y					Y		

Table 42 VP Snap off Clone operations: B -> C (continued)

TimeFinder/Clone A -> B Pair state: VP Snap B -> C Operation:	No Session	Created	Recreated	Copy in Progress	Copied	Copy on Write	Copy on Access	Restore in Progress	Restored	Split	Invalid	Failed
Full Establish	Y				Y					Y		
Incremental Establish	Y				Y					Y		
Set Mode Copy												
Set Mode Nocopy												
Set Mode Precopy												
Full Restore												
Incremental Restore	Y				Y					Y		
Split												
Terminate	Y	Y	Y	Y	Y			Y	Y	Y	Y	Y

TimeFinder/Snap operations

This section describes the TimeFinder/Snap control operations that are allowed for devices in various pair states.

Basic TimeFinder/Snap Operations

Figure 76 illustrates the relationship between devices for a basic TimeFinder/Snap operation.

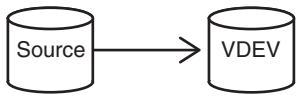

Figure 76 Basic snap device relationship

In Table 43, “Pair state” refers to the current state of the Source-VDEV pair, and “Operation” refers to the intended control operation. Y indicates that the operation is allowable for that state.

Table 43 Basic TimeFinder/Snap operations

TimeFinder/Snap Source -> VDEV Pair state: / TimeFinder/Snap Source -> VDEV Operation:	No Session	Create in Progress [1]	Created	Recreated	Copied	Copy on Write	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y										
Create - duplicate					Y	Y					
Duplicate					Y	Y					
Recreate [2]					Y	Y					
Activate			Y	Y							
Activate -duplicate			Y								
Full Establish [5]											
Incremental Establish [5]											
Full Restore [3]					Y	Y					
Incremental Restore					Y	Y					
Terminate		Y	Y	Y	Y	Y	Y [4]	Y	Y	Y	Y

1. Transient state.
2. Requires Enginuity 5874 through 5876.
3. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
4. Requires -symforce (not recommended).
5. Requires Solutions Enabler version 7.4 and higher.

Concurrent TimeFinder/Snap operations

A concurrent copy occurs when a source device is copied to multiple different targets. These copy operations can be any combination of TimeFinder/Clone, TimeFinder Snap, and TimeFinder/Mirror copies.

Concurrent TimeFinder/Snap

Figure 77 illustrates the relationship between devices for a TimeFinder/Clone pairing of a source device with two targets.

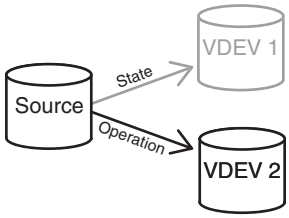

Figure 77 Concurrent TimeFinder/Snap device relationships

Note: The positions of VDEV 1 and VDEV 2 are interchangeable in the illustration above.

In Table 44, "Pair state" refers to the current state of the Source-VDEV 1pair, and "Operation" refers to the intended control operation on the Source-VDEV 2 pair. Y indicates that the operation is allowable for that state.

Table 44 Concurrent TimeFinder/Snap operations

TimeFinder/Snap Source -> VDEV 1 Pair state: TimeFinder/Snap Source -> VDEV 2 Operation:	No Session	Create in Prog[1]	Created	Recreated	Copied	Copy on Write	Restore in Prog[1]	Restored	Terminate in Prog	Invalid	Failed
Create	Y		Y	Y	Y	Y					
Create -duplicate	Y		Y	Y	Y	Y		Y			
Duplicate	Y		Y	Y	Y	Y		Y			
Recreate [2]	Y		Y	Y	Y	Y					
Activate	Y		Y	Y	Y	Y					
Activate -duplicate	Y		Y	Y	Y	Y		Y			
Full Establish [4]	Y		Y	Y	Y	Y					
Incremental Establish [4]	Y		Y	Y	Y	Y					
Full Restore [3]	Y		Y	Y	Y	Y					
Incremental Restore	Y				Y	Y					
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Requires Enginuity 5874 through 5876.
3. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
4. Requires Solutions Enabler version 7.4 and higher.

Concurrent TimeFinder/Snap: Snap and Mirror

Figure 78 illustrates the relationship between devices for a concurrent pairing of a source device with a TimeFinder/Snap VDEV and a TimeFinder/Mirror BCV.

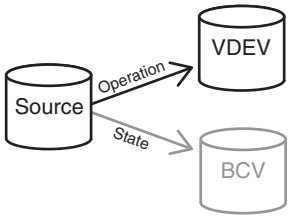

Figure 78 Concurrent TimeFinder/Snap device relationships: Snap and Mirror

Note: The positions of VDEV and BCV are interchangeable in the illustration above.

In Table 45, "Pair state" refers to the current state of the STD-BCV pair, and "Operation" refers to the intended control operation on the Source-VDEV pair. Y indicates that the operation is allowable for that state.

Table 45 Concurrent TimeFinder/Snap operations: Snap and Mirror

TimeFinder/Mirror STD -> BCV Pair state: / TimeFinder/Snap Source -> VDEV Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Create	Y	Y	Y			Y		Y	Y	
Create - duplicate	Y	Y	Y			Y		Y	Y	
Duplicate	Y	Y	Y			Y		Y	Y	
Recreate [1]	Y	Y	Y			Y		Y	Y	
Activate	Y	Y	Y			Y		Y	Y	
Activate -duplicate	Y	Y	Y			Y		Y	Y	
Full Establish [3]	Y	Y	Y			Y		Y	Y	
Incremental Establish [3]	Y	Y	Y			Y		Y	Y	
Full Restore [2]	Y	Y	Y			Y		Y	Y	
Incremental Restore	Y					Y		Y	Y	
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Requires Enginuity 5874 through 5876.
2. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
3. Requires Solutions Enabler version 7.4 and higher.

Concurrent TimeFinder/Snap: Snap and Clone

Figure 79 illustrates the relationship between devices for a concurrent pairing of a source device with a TimeFinder/Snap VDEV and a TimeFinder/Clone Target.

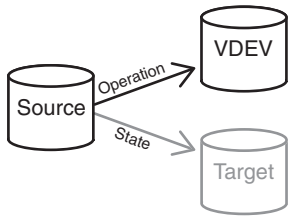

Figure 79 Concurrent TimeFinder/Snap device relationships: Snap and Clone

Note: The positions of Target and VDEV are interchangeable in the illustration above.

In Table 46, "Pair state" refers to the current state of the Source-Target pair, and "Operation" refers to the intended control operation on the Source-VDEV pair. Y indicates that the operation is allowable for that state.

Table 46 Concurrent TimeFinder/Snap operations: Snap and Clone

TimeFinder/Clone Source -> Target Pair state: / TimeFinder Snap Source -> VDEV Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Create - duplicate	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Duplicate	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Recreate [2]	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Activate	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Activate -duplicate	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Full Establish [4]	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Incremental Establish [4]	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Full Restore [3]	Y		Y	Y	Y	Y	Y	Y	Y	Y					
Incremental Restore	Y						Y			Y					
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Requires Enginuity 5874 through 5876.
3. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
4. Requires Solutions Enabler version 7.4 and higher.

Cascaded TimeFinder/Snap operations

A cascaded copy operation occurs when a device is both the source of a copy and the target of a copy, such that the devices are in the relationship of A -> B -> C.

Snap off Mirror: BCV -> VDEV

Figure 80 illustrates the relationship between devices for a Snap off Mirror operation in which the TimeFinder/Mirror STD device is paired with a BCV device and an additional session uses that device as the source for TimeFinder/Snap paired with a VDEV.

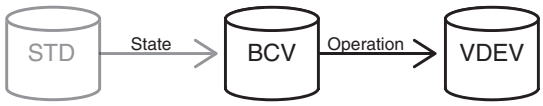

Figure 80 Snap off Mirror device relationship

Limitations:

- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.
- The TimeFinder/Snap session is not considered to be a hop and does not contribute to the two hop maximum. Therefore, the STD -> BCV pairing could be preceded by a clone session such that the final cascade is Clone Source -> STD -> BCV -> VDEV.

In Table 47 on page 213, “Pair state” refers to the current state of the STD-BCV pair, and “Operation” refers to the intended control operation on the BCV-VDEV pair. Y indicates that the operation is allowable for that state.

Table 47 Snap off Mirror operations: BCV -> VDEV

TimeFinder Mirror STD -> BCV Pair State: / TimeFinder/Snap BCV -> VDEV Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Create	Y					Y		Y		
Create - duplicate	Y					Y		Y		
Duplicate	Y					Y		Y		
Recreate [1]	Y					Y		Y		
Activate	Y					Y		Y		
Activate -duplicate	Y					Y		Y		
Full Establish [4]	Y					Y		Y		
Incremental Establish [4]	Y					Y		Y		
Full Restore [2]	Y					Y		Y		
Incremental Restore [3]	Y					Y		Y		
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Requires Enginuity 5874 through 5876.
2. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
3. Requires Enginuity 5874 through 5876.
4. Requires Solutions Enabler version 7.4 and higher.

Snap off Clone: B -> VDEV

Figure 81 illustrates the relationship between devices for a Snap off Clone operation in which the TimeFinder/Clone source device is paired with a target device and an additional session uses the target device as the source for TimeFinder/Snap paired with a VDEV.

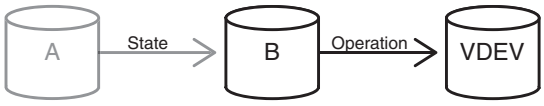

Figure 81 Snap off Clone device relationship

Limitations:

- Cascading with thin devices is supported on Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.
- The TimeFinder/Snap session is not considered to be a hop and does not contribute to the two hop maximum. Therefore, the STD -> BCV pairing could be preceded by a clone session such that the final cascade is Clone Source -> STD -> BCV -> VDEV.

In Table 48, “Pair state” refers to the current state of the A-B pair, and “Operation” refers to the intended control operation on the B-VDEVpair. Y indicates that the operation is allowable for that state.

Table 48 Snap off Clone operations: B -> VDEV

TimeFinder/Clone A -> B Pair state: / TimeFinder/Snap B -> VDEV Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Create	Y						Y			Y					
Create - duplicate	Y						Y			Y					
Duplicate	Y						Y			Y					
Recreate	Y						Y[2]			Y[5]					
Activate	Y						Y			Y					
Activate -duplicate	Y									Y					
Full Establish [4]	Y						Y			Y					
Incremental Establish [4]	Y						Y			Y[5]					
Full Restore [3]	Y			Y		Y	Y			Y				Y	Y
Incremental Restore	Y						Y			Y					
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient State
2. Requires Enginuity 5874 through 5876.
3. The restore to a third device must not be related to any session associated with the TimeFinder/Snap source device. The third device must be the same thick or thin configuration as the TimeFinder/Snap source device.
4. Requires Solutions Enabler version 7.4 and higher.
5. Requires Solutions Enabler version 7.5 and higher and Enginuity 5874 through 5876.

TimeFinder/Mirror operations

This section describes the TimeFinder/Mirror control operations that are allowed for devices in various pair states.

Basic TimeFinder/Mirror operations

Figure 82 illustrates the relationship between devices for a basic TimeFinder/Mirror operation.

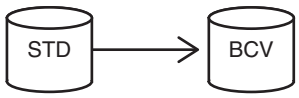

Figure 82 Basic TimeFinder/Mirror device relationship

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.

In Table 49, “Pair state” refers to the current state of the STD-BCV pair, and “Operation” refers to the intended control operation. Y indicates that the operation is allowable for that state.

Table 49 Basic TimeFinder/Mirror operations

TimeFinder/Mirror STD -> BCV Pair state: / TimeFinder Mirror STD -> BCV Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Full Establish	Y					Y		Y	Y	
Incremental Establish								Y		
Split		Y [1, 2]	Y	Y [1, 3, 4]	Y					
Full Restore	Y					Y		Y		Y
Incremental Restore								Y		
Cancel								Y		

1. Requires -symmforce (not recommended)
2. With native TimeFinder/Mirror, this results in the Split Before Sync state, which indicates that the BCV data is not complete. With TimeFinder/Clone Emulation mode, this results in the Split state, which indicates that the BCV data is complete.
3. Results in the Split Before Restore state, which indicates that the STD data is not complete.
4. Native TimeFinder/Mirror only.

Concurrent TimeFinder/Mirror operations

A concurrent copy occurs when a source device is copied to multiple different targets. These copy operations can be any combination of TimeFinder/Clone, TimeFinder/Snap, and TimeFinder/Mirror copies.

Concurrent TimeFinder/Mirror

Figure 83 illustrates the relationship between devices for a TimeFinder/Mirror pairing of a STD device with two BCVs.

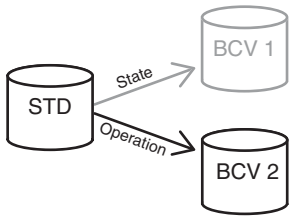

Figure 83 Concurrent TimeFinder/Mirror device relationships

Note: The positions of BCV 1 and BCV 2 are interchangeable in the illustration above.

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.

In Table 50 on page 217, “Pair state” refers to the current state of the STD-BCV 1 pair, and “Operation” refers to the intended control operation on the STD-BCV 2 pair. Y indicates that the operation is allowable for that state.

Table 50 Concurrent TimeFinder/Mirror operations

TimeFinder/Mirror STD -> BCV 1 Pair state: / TimeFinder/Mirror STD -> BCV 2 Operation:	Never Established	Synch In Progress	Synchronized	Restore in Progress	Restored	Split No Incremental	Split In Progress	Split	Split Before Sync	Split Before Restore
Full Establish	Y	Y	Y			Y		Y	Y	
Incremental Establish	Y	Y	Y			Y		Y	Y	
Split	Y	Y	Y			Y		Y	Y	
Full Restore	Y					Y		Y	Y	Y
Incremental Restore	Y					Y		Y	Y	Y
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Concurrent TimeFinder/Mirror: Mirror and Clone

Figure 84 illustrates the relationship between devices for a concurrent pairing of a STD device with a TimeFinder/Mirror BCV and a TimeFinder/Clone target.

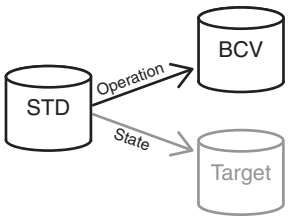

Figure 84 Concurrent TimeFinder/Mirror device relationships: Mirror and Clone

Note: The positions of BCV and Target are interchangeable in the illustration above.

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.

In Table 51, “Pair state” refers to the current state of the STD-Target pair, and “Operation” refers to the intended control operation on the STD-BCV pair. Y indicates that the operation is allowable for that state.

Table 51 Concurrent TimeFinder/Mirror operations: Mirror and Clone

TimeFinder/Clone STD -> Target Pair state: / TimeFinder/Mirror STD -> BCV Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Full Establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y					
Incremental Establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y					
Split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y [2]			
Full Restore	Y														
Incremental Restore	Y														
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Native TimeFinder/Mirror only.

Concurrent TimeFinder/Mirror: Mirror and Snap

Figure 85 illustrates the relationship between devices for a concurrent pairing of a STD device with a TimeFinder/Mirror BCV and a TimeFinder/Snap VDEV.

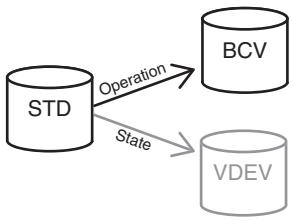

Figure 85 Concurrent TimeFinder/Mirror device relationships: Mirror and Snap

Note: The positions of BCV and VDEV are interchangeable in the illustration above.

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.

In Table 52, “Pair state” refers to the current state of the STD-VDEV pair, and “Operation” refers to the intended control operation on the STD-BCV pair. Y indicates that the operation is allowable for that state.

Table 52 Concurrent TimeFinder/Mirror operations: Mirror and Snap

TimeFinder/Snap STD -> VDEV Pair state: / TimeFinder/Mirror STD -> BCV Operation:	No Session	Create in Progress [1]	Created	Recreated	COPIED	Copy on Write	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Full Establish	Y	Y	Y	Y	Y	Y					
Incremental Establish	Y	Y	Y	Y	Y	Y					
Split	Y	Y	Y	Y	Y	Y		Y[2]			
Full Restore	Y										
Incremental Restore	Y										
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.
2. Native TimeFinder/Mirror only.

Cascaded TimeFinder/Clone operations

A cascaded copy operation occurs when a device is both the source of a copy and the target of a copy, such that the devices are in the relationship of A -> B -> C.

Clone off Mirror: STD -> BCV

Figure 86 illustrates the relationship between devices for a Clone off Mirror operation in which the TimeFinder/Mirror STD device is paired with a BCV and an additional session uses that device as the source for TimeFinder/Clone paired with a target device.

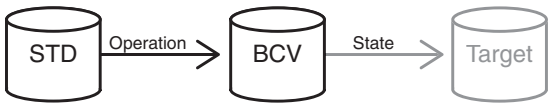

Figure 86 Clone off Mirror device relationship

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 53, “Pair state” refers to the current state of the BCV-Target pair, and “Operation” refers to the intended control operation on the STD-BCV pair. Y indicates that the operation is allowable for that state.

Table 53 Clone off Mirror operations: STD -> BCV

TimeFinder/Clone BCV -> Target Pair state: / TimeFinder/Mirror STD -> BCV Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Full Establish	Y														
Incremental Establish	Y														
Split	Y														
Full Restore	Y														
Incremental Restore	Y														
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.

Mirror off Clone: STD -> BCV

Figure 87 illustrates the relationship between devices for a Mirror off Clone operation in which the TimeFinder/Clone source device is paired with a target device and an additional session uses that device as the STD for TimeFinder/Mirror paired with a BCV.

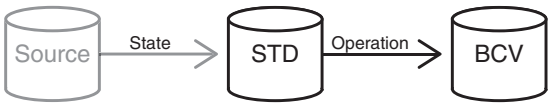

Figure 87 Mirror off Clone device relationship

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 54, "Pair state" refers to the current state of the Source-STD pair, and "Operation" refers to the intended control operation on the STD-BCV pair. Y indicates that the operation is allowable for that state.

Table 54 Mirror off Clone operations STD -> BCV

TimeFinder/Clone Source -> STD Pair state: / TimeFinder/Mirror STD -> BCV Operation:	No Session	Create in Progress [1]	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Full Establish	Y										Y	Y			
Incremental Establish	Y										Y	Y			
Split	Y						Y			Y	Y	Y			
Full Restore	Y														
Incremental Restore	Y														
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.

Snap off Mirror: STD -> BCV

Figure 88 illustrates the relationship between devices for a Snap off Mirror operation in which the TimeFinder/Mirror STD device is paired with a BCV device and an additional session uses the BCV device as the source for TimeFinder/Snap paired with a VDEV.

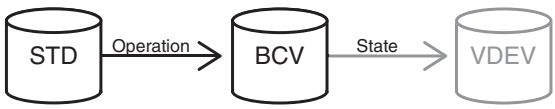

Figure 88 Snap off Mirror device relationship

Limitations:

- With Enginuity 5874 through 5876, all TimeFinder/Mirror operations are done with TimeFinder/Clone Emulation. With Enginuity 5874 and lower, the default mode is native TimeFinder/Mirror, and TimeFinder/Clone Emulation is a selectable option.
- RAID5 and RAID6 BCVs automatically use TimeFinder/Clone Emulation.
- Thin BCVs are supported with Enginuity 5875 and higher.
- Mixed thick and thin devices are not supported. All devices in a cascaded session must be thick or thin.
- Cascaded sessions cannot exceed two hops.

In Table 55, “Pair state” refers to the current state of the BCV-VDEV pair, and “Operation” refers to the intended control operation on the STD-BCV pair. Y indicates that the operation is allowable for that state.

Table 55 Snap off Mirror operations: STD -> BCV

TimeFinder/Snap BCV -> VDEV Pair state: / TimeFinder/Mirror STD -> BCV Operation:	No Session	Create in Progress [1]	Created	Recreated	COPIED	Copy on Write	Restore in Progress	Restored	Terminate in Progress [1]	Invalid	Failed
Full Establish	Y										
Incremental Establish	Y										
Split	Y										
Full Restore	Y										
Incremental Restore	Y										
Cancel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Transient state.

APPENDIX B
SRDF State Rules Reference

This appendix describes the applicable SRDF pair states that rule the TimeFinder SnapVX, TimeFinder/Clone, and TimeFinder/Snap copy session operations.

- SnapVX-SRDF Interaction Rules State Determination 224
- SRDF pair states 226
- State rules for TimeFinder SnapVX operations 228
- State rules for TimeFinder/Clone operations 231
- State rules for TimeFinder/Snap operations 235

SnapVX-SRDF Interaction Rules State Determination

Table 56 describes how SnapVX states are identified for SRDF operations allowed with SnapVX source devices. Display examples for each active state are shown in "SnapVX State Determination Examples" on page 224.

Table 56 SnapVX State Determination

Snapvx State	Snapvx CLI	Determination of State
Established	`symsnapvx list`	The flags field indicates: `(F)ailed: = . for No Failure`
Restore In Progress	`symsnapvx list -restored -detail`	The Done(%) field is not at 100%
Restored	`syms'napvx list -restored -detail`	The Done(%) field is at 100%
Terminate in Progress	`symsnapvx list`	The snapshot is still seen on the display.
Terminated	`symsnapvx list`	The snapshot is not seen on the display.
Failed	`symsnapvx list`	The flags field indicates: `(F)ailed = X for Failed`
Link Copy In Progress	`symsnapvx list -link -tgt -detail`	The flags field indicates: `(C)opy = I for CopyInProg` The Done (%) field is not at 100%
Link Copied	`symsnapvx list -link -tgt -detail`	The flags field indicates: `(C)opy = C for Copied or D for Copied/Destaged` The Done (%) field is at 100%
Linked	`symsnapvx list -linked`	The flags field indicates : `(C)opy = . for NoCopy Link`

SnapVX State Determination Examples

Established

```
#> symsnapvx -cg Sparrow -snapshot_name Bailey list

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

----------------------------------------------------------------------------
Sym                                     Num  Flgs
Dev   Snapshot Name                     Gens FLRG Last Snapshot Timestamp
----- ------------------------------- ---- ---- -------------------------
000A0 Bailey                               1 .... Fri Oct 24 15:51:19 2014
000A1 Bailey                               1 .... Fri Oct 24 15:51:19 2014

Flgs:
(F)ailed  : X = Failed, . = No Failure
  (L)ink    : X = Link Exists, . = No Link Exists
  (R)estore : X = Restore Active, . = No Restore Active
  (G)CM     : X = GCM, . = Non-GCM
```

Restore in Progress

```
#> symsnapvx -cg Sparrow -sna Bailey list -restored -detail

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

----------------------------------------------------------------------------
Sym                             Flgs                          Remaining  Done
Dev   Snapshot Name      Gen    F   Snapshot Timestamp       (Tracks)   (%)
----- ------------------ ---- ---- ------------------------ ---------- -----
000A0 Bailey              0    .   Fri Oct 24 15:51:18 2014 72206       47
000A1 Bailey              0    .   Fri Oct 24 15:51:18 2014 0           100


Flgs:
(F)ailed   : X = Failed, . = No Failure
```

Restored

```
#> symsnapvx -cg Sparrow -sna Bailey list -restored -detail

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

-----------------------------------------------------------------------------
Sym                              Flgs                           Remaining Done
Dev   Snapshot Name        Gen    F     Snapshot Timestamp       (Tracks)  (%)
----- ----------------- ----- ------ ------------------------ -------- -----
000A0 Bailey               0     .     Fri Oct 24 15:51:18 2014   0       100
000A1 Bailey               0     .     Fri Oct 24 15:51:18 2014   0       100

Flgs:
(F)ailed   : X = Failed, . = No Failure
```

Link Copy in Progress

```
#> symsnapvx -cg Sparrow -sna Bailey list -linked -detail

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

-----------------------------------------------------------------------------
Sym                           Link  Flgs                          Remaining Done
Dev   Snapshot Name    Gen    Dev   FCMD  Snapshot Timestamp      (Tracks)   (%)
----- --------------- ----- ----- ----- ------------------------ ---------- ----
000A0 Bailey            0    000B0 .I..   Fri Oct 24 15:51:19 2014  118579   14
000A1 Bailey            0    000B1 .I..   Fri Oct 24 15:51:19 2014  128275    7

Flgs:
(F)ailed   : F = Force Failed, X = Failed, . = No Failure
  (C)opy     : I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odified : X = Modified Target Data, . = Not Modified
  (D)efined  : X = All Tracks Defined, . = Define in progress
```

Link Copied

```
#> symsnapvx -cg Sparrow -sna Bailey list -linked -detail

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

-----------------------------------------------------------------------------
Sym                          Link  Flgs                          Remaining  Done
Dev   Snapshot Name    Gen   Dev   FCMD Snapshot Timestamp       (Tracks)   (%)
----- --------------- ----- ----- ---- ------------------------ -------- ------
000A0 Bailey           0    000B0 .D.X  Fri Oct 24 15:51:18 2014       0  100
000A1 Bailey           0    000B1 .D.X  Fri Oct 24 15:51:18 2014       0  100

Flgs:
(F)ailed   : F = Force Failed, X = Failed, . = No Failure
```

```
  (C)opy     : I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odified : X = Modified Target Data, . = Not Modified
  (D)efined  : X = All Tracks Defined, . = Define in progress
#> symsnapvx -cg Sparrow -sna Bailey list -linked

Composite Group (CG) Name : Sparrow
CG's Type                 : ANY
Number of Symmetrix Units : 1

CG's Symmetrix ID         : 000197300076    (Microcode Version: 5977)

-------------------------------------------------------------------------------
Sym                                          Link  Flgs
Dev   Snapshot Name                     Gen  Dev   FCMD Snapshot Timestamp
----- ------------------------------- ---- ----- ---- ------------------------
000A0 Bailey                               0 000B0 .... Fri Oct 24 15:51:18 2014
000A1 Bailey                               0 000B1 .... Fri Oct 24 15:51:18 2014

Flgs:

  (F)ailed   : F = Force Failed, X = Failed, . = No Failure
  (C)opy     : I = CopyInProg, C = Copied, D = Copied/Destaged, . = NoCopy Link
  (M)odified : X = Modified Target Data, . = Not Modified
  (D)efined  : X = All Tracks Defined, . = Define in progress
```

Linked

SRDF pair states

Certain TimeFinder copy operations are not allowed within arrays employing SRDF for remote mirroring as these operations can conflict with one another. The availability of some actions depends on the current state of SRDF pairs. Refer to your product guide for details about supported features.

Table 57 provides a description the various SRDF pair states.

Table 57 SRDF pair states

State	Description
Consistent	The R2 mirrors of SRDF/A devices are in a Consistent state. Consistent state signifies the normal state of operation for device pairs operating in asynchronous mode.
Failed Over	The R1 is currently Not Ready or write disabled and operations have been failed over to the R2.
Invalid	The default state when no other SRDF state applies. The combination of R1, R2, and SRDF link states and statuses do not match any other pair state. This state may occur if there is a problem at the disk director level.
Mixed	A composite device group SRDF pair state. There exists different SRDF pair states within a device group.
Partitioned	Solutions Enabler is currently unable to communicate through the corresponding SRDF path to the remote array. Partitioned may apply to devices within an RA group. For example, if Solutions Enabler is unable to communicate to a remote array via an RA group, devices in that RA group will be marked as being in the Partitioned state.
R1 Updated	The R1 is currently Not Ready or write disabled to the host, there are no local invalid tracks on the R1 side, and the link is Ready or write disabled.
R1 UpdInProg	The R1 is currently Not Ready or write disabled to the host, there are invalid local (R1) tracks on the source side, and the link is Ready or write disabled.
Split	The R1 and the R2 are currently Ready to their hosts, but the link is Not Ready or write disabled.

Table 57 SRDF pair states (continued)

State	Description
Suspended	The SRDF links have been suspended and are Not Ready or write disabled. If the R1 is Ready while the links are suspended, any I/O will accumulate as invalid tracks owed to the R2.
Synchronized	The R1 and the R2 are currently in a Synchronized state. The same content exists on the R2 as the R1. There are no invalid tracks between the two pairs.
SyncInProg	A synchronization is currently in progress between the R1 and the R2. There are existing invalid tracks between the two pairs and the logical link between both sides of an SRDF pair is up.
Transmit Idle	The SRDF/A session cannot push data in the transmit cycle across the link because the link is down.

State rules for TimeFinder SnapVX operations

This section identifies the `symsnapvx` control operations that are available for use within each of the SRDF pair states.

TimeFinder SnapVX R1 source

Table 58 identifies the `symsnapvx` control operations that are allowed on R1 source devices while SRDF is in the various pair states.

Table 58 SnapVX control operations allowed on R1 source device by SRDF state

SRDF State: / SnapVX Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Establish	Y[1, 3]	Y	Y[3]	Y[3]		Y[3]	Y[3]				Y[1, 3]	
Restore	Y[1, 2]	Y[2]	Y	Y		Y					Y[1, 2]	Y[1, 2]
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Action is not allowed if there are local R1 invalids on the R1 side or remote invalids on the R2 side.
2. The remote option (-remote) must be applied.
3. Not allowed with both sides operation.

Note: The Partitoned1 pair state indicates that the remote Symmetrix is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote Symmetrix is not in the SYMAPI database and was not discovered, or was removed from this database.

TimeFinder SnapVX R2 source

Table 59 identifies the `symsnapvx` control operations that are allowed on R2 source devices while SRDF is in the various pair states.

Table 59 SnapVX control operations allowed on R2 source device by SRDF state

SRDF State: / SnapVX Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Establish		Y[1]	Y[3]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	
Restore			Y[3]	Y[3]	Y[3]	Y[3]						
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if Adaptive Copy.
2. Not allowed with both sides operation.
3. Not allowed if target device is an R2 larger than the R1.

Note: The Partitoned1 pair state indicates that the remote Symmetrix is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote Symmetrix is not in the SYMAPI database and was not discovered, or was removed from this database.

TimeFinder SnapVX R1 target

Table 60 identifies the `symsnapvx` control operations that are allowed on R1 target devices while SRDF is in the various pair states

Table 60 SnapVX control operations allowed on R1 target device by SRDF state

SRDF State: / SnapVX Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Link/Relink												
Link/Relink Copy	Y[1, 2]	Y[2]	Y	Y		Y					Y[1, 2]	Y[1, 2]
Unlink	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Action is not allowed if there are local R1 invalids on the R1 side or remote invalids on the R2 side.
2. The remote option (-remote) must be applied.

Note: The Partitoned1 pair state indicates that the remote Symmetrix is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote Symmetrix is not in the SYMAPI database and was not discovered, or was removed from this database.

TimeFinder SnapVX R2 target

Table 61 identifies the `symsnapvx` control operations that are allowed on R2 target devices while SRDF is in the various pair states.

Table 61 SnapVX control operations allowed on R2 target device by SRDF state

SRDF State: / SnapVX Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Link/Relink												
Link/Relink Copy			Y[2]	Y[2]	Y[1, 2]	Y[2]						
Unlink	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if Async.
2. Not allowed if target device is an R2 larger than the R1.

Note: The Partitoned1 pair state indicates that the remote Symmetrix is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote Symmetrix is not in the SYMAPI database and was not discovered, or was removed from this database.

State rules for TimeFinder/Clone operations

This section identifies the `symclone` copy actions that are available for use within each of the SRDF pair states.

This section refers to the copy source and the copy target. It is important to note that “source” and “target” refer to the direction of the data flow. In this context, these terms do not refer to the TimeFinder/Clone device pair relationship.

Figure 89 shows an establish operation for which the R1 device is the source of the data for the clone copy operation. In other words, the TimeFinder/Clone source device is the source for the clone copy.

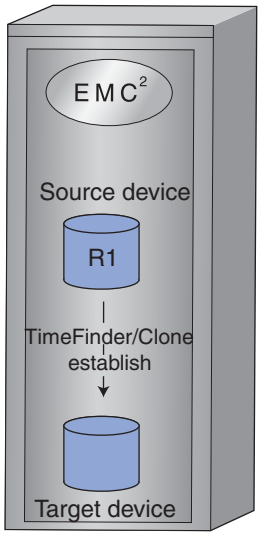

Figure 89 R1 device as TimeFinder/Clone copy source.

Figure 89 shows a restore operation for which the R1 device is the target of the data for the clone copy operation. In other words, for a restore, the TimeFinder/Clone source device is the target of the clone copy.

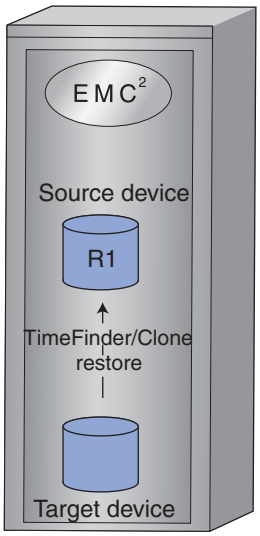

Figure 90 R1 device as TimeFinder/Clone copy target.

TF/Clone operations R1 source

Table 62 TF/Clone operations allowed when R1 is source of clone copy data

SRDF State: / Clone Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate	Y	Y	Y	Y	Y	Y		Y				Y
Activate	Y [1]	Y	Y	Y	Y	Y		Y [2]	Y [2]		Y [1]	Y
Establish	Y [1]	Y	Y	Y	Y	Y		Y [2]	Y [2]		Y [1]	Y
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Restore	Y [1]	Y	Y	Y		Y		Y	Y		Y [1]	Y
Split	Y	Y	Y	Y	Y	Y	Y	Y			Y	Y

1. Action is not allowed if there are local R1 invalids or remote R2 invalids.
2. Action is not allowed if the -consistent option is specified.

TF/Clone operations R1 target

Table 63 TF/Clone operations allowed when R1 is target of clone copy data

SRDF State: / Clone Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate	Y [1, 2, 3]	Y [2, 3]	Y	Y		Y					Y [1, 2, 3]	Y
Activate	Y [1, 2, 3]	Y [2, 3]	Y	Y		Y					Y [1, 2, 3]	Y
Establish	Y [1, 2, 3]	Y [2, 3]	Y	Y		Y					Y [1, 2, 3]	Y
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Restore	Y [1, 2, 3]	Y [2, 3]	Y	Y		Y					Y [1, 2, 3]	Y
Split	Y [1, 2, 3]	Y [2, 3]	Y	Y		Y	Y				Y [1, 2, 3]	Y

1. Action is not allowed if there are local R1 invalids or remote R2 invalids.
2. The -force option must be applied.
3. Action is not allowed with CopyOnAccess.

TF/Clone operations R2 source

Table 64 TF/Clone operations allowed when R2 is source of clone copy data

SRDF State: / Clone Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate	Y [3]	Y	Y [3]	Y [3]	Y [2, 3]	Y		Y [2, 3]	Y [2, 3]		Y [2, 3]	Y [5]
Activate	Y [4]	Y	Y	Y	Y	Y		Y	Y		Y [1]	Y [1]
Establish	Y [3, 4]	Y	Y [3]	Y [3]	Y [2, 3]	Y		Y [2, 3]	Y [2, 3]		Y [1, 2, 3]	Y [1, 5]
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Restore		Y	Y	Y	Y	Y						
Split		Y	Y	Y	Y	Y	Y					

1. Action is not allowed if there are local R1 invalids or remote R2 invalids.
2. With Enginuity 5874 and lower, the precopy option (-precopy) must be applied.
3. If the precopy option is not applied, the action is not allowed if either of the following applies:
   a. SRDF/A device-level write pacing is not activated and supported on the SRDF/A session, or
   b. The SRDF pair is the R21-> R2 of a cascaded configuration and any of the following apply:
      i. The R21 Symmetrix array is running an Enginuity level lower than 5876.159.102, or
      ii. The R2 Symmetrix array is running an Enginuity level lower than 5875, or
      iii. The R21 device is not pace-capable.
4. Action is not allowed if there are remote R1 invalids or local R2 invalids.
5. The precopy option (-precopy) must be applied for SRDF/A.

TF/Clone operations R2 target

Table 65 TF/Clone operations allowed when R2 is target of clone copy data

SRDF State: / Clone Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate			Y [2]	Y [2]	Y [1, 2]	Y [2]						
Activate			Y [2]	Y [2]	Y [1, 2]	Y [2]						
Establish			Y [2]	Y [2]	Y [1, 2]	Y [2]						
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Restore			Y [2]	Y [2]	Y [1, 2]	Y						
Split			Y [2]	Y [2]	Y [1, 2]	Y [2]						

1. Action is not allowed if the R2 target device is in asynchronous mode.
2. Action is not allowed if the target device is an R2 larger than the R1.

State rules for TimeFinder/Snap operations

This section identifies what `symsnap` copy actions are available for use within each of the SRDF pair states.

TimeFinder/Snap operations R1 source

Table 66 TimeFinder/Snap operations allowed when R1 is source of the snap

SRDF State: / Snap Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate	Y	Y	Y	Y	Y	Y		Y	Y		Y	Y
Activate	Y [4]	Y	Y	Y	Y	Y		Y [1]	Y [1]		Y [1]	Y
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Incremental restore to source	Y [3, 4]	Y [3]	Y	Y		Y					Y [1, 2]	Y
Incremental restore to a split BCV or full restore to any device	Y	Y	Y	Y	Y	Y		Y [1]	Y [1]		Y	Y

1. Action is not allowed if there are local R1 invalids or remote R2 invalids.
2. The force option (-force) must be applied.
3. The force option (-force) must be applied.
4. Action is not allowed if there are local invalids on the R1 side or remote invalids owed to the R1 on the R2 side.

TimeFinder/Snap operations R1 target

Table 67 TimeFinder/Snap operations allowed when R1 is target of the snap

SRDF State: / Snap Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Incremental restore to a split BCV or full restore to any device	Y	Y	Y	Y		Y				Y	Y	Y

TimeFinder/Snap operations R2 source

Table 68 TimeFinder/Snap operations allowed when R2 is source of snap

SRDF State: / Snap Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Create/recreate	Y [2]	Y	Y	Y [3]	Y [3]	Y		Y [3]	Y [3]		Y	
Activate	Y [5]	Y	Y	Y	Y	Y		Y	Y		Y	
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	
Incremental restore to source		Y	Y	Y	Y	Y		Y [4]	Y [4]			
Incremental restore to a split BCV or full restore to any device	Y	Y	Y	Y	Y	Y		Y [1]	Y [4]			

1. The force option (-force) must be applied.
2. Action is not allowed if either of the following is true:
   a. SRDF/A device-level write pacing is not activated and supported on the SRDF/A session, or
   b. The SRDF pair is the R21-> R2 of a cascaded configuration, and any of the following apply:
      i. The R21 Symmetrix array is running an Enginuity level lower than 5876.159.102, or
      ii. The R2 Symmetrix array is running an Enginuity level lower than 5875, or
      iii. The R21 device is not pace-capable.
3. Action is not allowed if the SRDF pair is operating in asynchronous mode and any of the following are true:
   a. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the SRDF/A session, or
   b. The SRDF pair is the R21-> R2 of a cascaded configuration and either of the following applies:
      i. The R21 Symmetrix array is running an Enginuity level lower than 5876 Q2012 SR, or
      ii. The R2 Symmetrix array is running an Enginuity level lower than 5875.

c. The SRDF pair is not the R21-› R2 of a cascaded configuration and either the R1 or the R2 Symmetrix array is running an Enginuity level lower than 5875.

4. Action is not allowed if the R2 target is in asynchronous mode.
5. Action is not allowed if there are remote invalids owed to the R2 on the R1 side or local invalids on the R2 side.

TimeFinder/Snap operations R2 target

Table 69 TimeFinder/Snap operations allowed when R2 is target of the snap

SRDF State: / Snap Control Operation:	SYNCINPROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILEDOVER	PARTITIONED1	PARTITIONED2	R1UPDATED	R1UPDINPROG	INVALID	CONSISTENT	TRANSIDLE
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Incremental restore to a split BCV or full restore to any device			Y	Y	Y	Y		Y	Y	Y		

APPENDIX C
Rcopy State Rules Reference

This appendix describes the applicable Rcopy pair states that rule the TimeFinder SnapVX copy operations. It also describes which TimeFinder/Snap and TimeFinder/Clone operations are permissible outside of any current replicator session and whether an Rcopy operation is permissible for certain specified device types.

- TimeFinder SnapVX operations 240
- TimeFinder/Snap and TimeFinder/Clone operations 245

TimeFinder SnapVX operations

This section identifies the `symsnapvx` control operations that are available for use within each of the Rcopy pair states.

SnapVX source with Rcopy push

Table 70 identifies the `symsnapvx` control operations that are allowed on SnapVX source devices with Rcopy push sessions while Rcopy is in the various pair states.

Table 70 SnapVX control operations allowed on source device with Rcopy push

Rcopy state: / SnapVX Control Operation:	NONE	CREATE IN PROG	CREATED	COPY IN PROG	COPY ON WRITE	COPIED	RECREATE IN PROG	RECREATED	TERM IN PROG	FAILED	INVALID	VERIFY IN PROG	RESTORE IN PROG	RESTORED	PRECOPY	SYNC IN PROG	SYNCHRONIZED	STOPPED
Establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Restore						Y				Y								
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

SnapVX source with Rcopy pull

Table 71 identifies the `symsnapvx` control operations that are allowed on SnapVX source devices with Rcopy pull sessions while Rcopy is in the various pair states.

Table 71 SnapVX control operations allowed on source device with Rcopy pull

Rcopy state: / SnapVX Control Operation:	NONE	CREATE IN PROG	CREATED	COPY IN PROG	COPY ON WRITE	COPIED	RECREATE IN PROG	RECREATED	TERM IN PROG	FAILED	INVALID	VERIFY IN PROG	RESTORE IN PROG	RESTORED	PRECOPY	SYNC IN PROG	SYNCHRONIZED	STOPPED
Establish														Y[1]				
Restore														Y[1]				
Terminate	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed with Donor Update.

TimeFinder SnapVX target with Rcopy push

Table 72 identifies the `symsnapvx` control operations that are allowed on SnapVX target devices with Rcopy push sessions while Rcopy is in the various pair states.

Table 72 SnapVX control operations allowed on target device with Rcopy push

RCOPY state: / SnapVX Control Operation:	NONE	CREATE IN PROG	CREATED	COPY IN PROG	COPY ON WRITE	COPIED	RECREATE IN PROG	RECREATED	TERM IN PROG	FAILED	INVALID	VERIFY IN PROG	RESTORE IN PROG	RESTORED	PRECOPY	SYNC IN PROG	SYNCHRONIZED	STOPPED
Link/Relink																		
Link/Relink Copy						Y				Y								
Unlink	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

TimeFinder SnapVX target with Rcopy pull

Table 73 identifies the `symsnapvx` control operations that are allowed on SnapVX target devices with Rcopy pull sessions while Rcopy is in the various pair states.

Table 73 SnapVX control operations allowed on target device with Rcopy pull

RCOPY state: / SnapVX Control Operation:	NONE	CREATE IN PROG	CREATED	COPY IN PROG	COPY ON WRITE	COPIED	RECREATE IN PROG	RECREATED	TERM IN PROG	FAILED	INVALID	VERIFY IN PROG	RESTORE IN PROG	RESTORED	PRECOPY	SYNC IN PROG	SYNCHRONIZED	STOPPED
Link/Relink																		
Link/Relink Copy														Y[1]				
Unlink	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed with Donor Update.

Rcopy push with TimeFinder SnapVX source

Table 74 identifies RCOPY operations that are allowed when a SnapVX source is the control device for an RCOPY PUSH session.

Table 74 Rcopy push operations allowed on SnapVX source control devices

SnapVX state: / RCOPY Operation:	No Snapshot	Establish In Progress	Established	Restore in Progress	Restored	Term In Progress	Failed	Link Copy in Progress	Link Copied	Linked
Create	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Activate	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Recreate	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Set Precopy	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Incremental Restore	Y							NA	NA	NA
Set Nocopy	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Set Copy	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Set Pace	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Terminate	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Remove	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Rename	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA

Rcopy push with TimeFinder SnapVX target

Table 75 identifies RCOPY operations that are allowed when a SnapVX target is the control device for an RCOPY PUSH session.

Table 75 Rcopy push operations allowed on SnapVX target control devices

SnapVX state: / RCOPY Operation:	No Snapshot	Establish In Progress	Established	Restore in Progress	Restored	Term In Progress	Failed	Link Copy in Progress	Link Copied	Linked
Create	Y	NA	NA	NA	NA	NA			Y	
Activate	Y	NA	NA	NA	NA	NA			Y	
Recreate	Y	NA	NA	NA	NA	NA			Y	
Set Precopy	Y	NA	NA	NA	NA	NA			Y	
Incremental Restore	Y	NA	NA	NA	NA	NA				

Table 75 Rcopy push operations allowed on SnapVX target control devices (continued)

SnapVX state: / RCOPY Operation:	No Snapshot	Establish In Progress	Established	Restore in Progress	Restored	Term In Progress	Failed	Link Copy in Progress	Link Copied	Linked
Set Nocopy	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Set Copy	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Set Pace	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Terminate	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Remove	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Rename	Y	NA	NA	NA	NA	NA	Y	Y	Y	

Rcopy pull with TimeFinder SnapVX source

Table 76 identifies RCOPY operations that are allowed when a SnapVX source is the control device for an RCOPY PULL session.

Table 76 Rcopy pull operations allowed on SnapVX source control devices

SnapVX state: / RCOPY Operation:	No Snapshot	Establish In Progress	Established	Restore in Progress	Restored	Term In Progress	Failed	Link Copy in Progress	Link Copied	Linked
Create	Y	Y	Y		Y			NA	NA	NA
Activate	Y	Y	Y		Y			NA	NA	NA
Set Nocopy	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Set Copy	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Set Pace	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Terminate	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Remove	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Rename	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Donor Update Off	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Failback	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA
Frontend Zero Off	Y	Y	Y	Y	Y	Y	Y	NA	NA	NA

Rcopy pull with TimeFinder SnapVX target

Table 77 identifies RCOPY operations that are allowed when a SnapVX target is the control device for an RCOPY PULL session.

Table 77 Rcopy pull operations allowed on SnapVX target control devices

SnapVX state: / RCOPY Operation:	No Snapshot	Establish In Progress	Established	Restore in Progress	Restored	Term In Progress	Failed	Link Copy in Progress	Link Copied	Linked
Create	Y	NA	NA	NA	NA	NA			Y	
Activate	Y	NA	NA	NA	NA	NA			Y	
Set Nocopy	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Set Copy	Y	NA	NA	NA		NA	Y	Y	Y	
Set Pace	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Terminate	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Remove	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Rename	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Donor Update Off	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Failback	Y	NA	NA	NA	NA	NA	Y	Y	Y	
Frontend Zero Off	Y	NA	NA	NA	NA	NA	Y	Y	Y	

TimeFinder/Snap and TimeFinder/Clone operations

Table 78 describes the TimeFinder/Snap and TimeFinder/Clone operations that are allowed outside of any current replicator session.

Table 78 TimeFinder/Snap and TimeFinder/Clone operations

Action	Allowed
Snap — Create/Recreate	
To RCopy control device pushing out	No
To RCopy control device pulling in	No
From RCopy control device pushing out	Yes
From RCopy control device pulling in	No
Snap — Restore	
To RCopy control device pushing out	No
To RCopy control device pulling in	No
VDEV	
Can a VDEV be an RCopy control device pushing out	Yes (cold push only)
Can a VDEV be an RCopy control device pulling in	No
Clone — Create/Recreate	
To RCopy control device pushing out	Yes, only if in the copied or restored state without donor_update
To RCopy control device pulling in	No
From RCopy control device pushing out	Yes
From RCopy control device pulling in	Yes
Clone — Restore	
To RCopy control device pushing out	No
To RCopy control device pulling in	No