EMC® Solutions Enabler SRDF Family

Version 8.0.1

CLI User Guide

VMAX® 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.

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 SRDF CLI

Introduction 18
SRDF documentation 19
What's new in Solutions Enabler 8.0.1 19
Restrictions 21
SRDF backward compatibility to Enginuity 5876 22
SYMCLI for SRDF 24
SYMCLI SRDF commands 25
symrdf command options 27
Commands to display, query and verify SRDF configurations 29
symrdf list command options 32
ping command 34
Using the verify command 34
Understanding SRDF pair states and links 37
About SRDF pair states 38
Before you begin 40
Array access rights 41
Device external locks 41
SRDF operations and copy sessions 41
Migrating R1 to a larger R2 device 41
Restricting synchronization 42
Enabling SRDF software and hardware compression 42
SRDF/A and the consistency exempt option 43
Enabling mixed-mode workloads on an SRDF director 43
FAST VP SRDF coordination 44

Chapter 2 Basic SRDF Control Operations

Basic SRDF control operations summary 46
SRDF basic control operations 49
Set mode 49
Establish (full) 52
Establish (incremental) 54
Failback 56
Failover 58
Invalidate R1 tracks 59
Invalidate R2 tracks 60
Make R1 ready 61
Make R1 not ready 61
Make R2 ready 62
Make R2 not ready 62
Merge track tables 63
Move one-half of an SRDF pair 63
Move both sides of SRDF device pairs 64
Read/write disable target device 64
Refresh R1 65
Refresh R2 66

Restore (full) 66
Restore (incremental) 68
Resume I/O on links 69
Split 70
Suspend I/O on links 72
Swap one-half of an SRDF pair 72
Swap SRDF pairs 73
Update R1 mirror 74
Write disable R1 76
Write disable R2 76
Write enable R1 76
Write enable R2 77

Chapter 3 Dynamic Operations

Overview 80
SRDF dynamic group attributes 81
Managing dynamic SRDF groups 82
Adding dynamic SRDF groups 82
Setting SRDF group attributes 85
Adding/removing supporting directors for an SRDF group 86
Removing dynamic SRDF groups 87
Dynamic device pair operations 88
Creating a device file 88
Creating dynamic SRDF device pairs 89
Creating dynamic concurrent pairs 94
Deleting dynamic SRDF device pairs 95
Grouping, moving and swapping dynamic devices 98
Grouping dynamic pairs with a device file 98
Moving dynamic SRDF device pairs 98
Swapping SRDF devices 100
Dynamic failover operations 103

Chapter 4 SRDF/Asynchronous Operations

Overview 108
SRDF/A benefits 111
SRDF /A features 111
Restrictions 112
Using BCVs to protect the R2 side 113
Immediate option 113
SRDF/Asynchronous operations 114
Transitioning to asynchronous mode 115
Setting SRDF/A group cycle time, priority, and transmit idle 116
Checking for R1 invalid tracks 117
Enabling/disabling consistency for SRDF/A devices 118
Adding/removing devices using consistency exempt 119
Displaying checkpoint complete status 122
Managing Delta Set Extension 123
Managing the DSE SRP capacity (HYPERMAX OS) 124
Managing DSE pools (Enginuity 5773 - 5876) 125
Managing transmit idle 131
Managing SRDF/A write pacing 131
Displaying SRDF/A 138

Chapter 5 Consistency Group Operations

Overview 142
Consistency protection using the SRDF daemon 142
Redundant consistency protection 143
SRDF consistency group operations 145
Creating an SRDF consistency group 145
Creating composite groups from various sources 146
Enabling and disabling SRDF consistency protection 148
Restrictions 148
Enabling consistency: composite group vs. SRDF group name 149
Enabling SRDF consistency protection for concurrent SRDF devices 151
Checking if device pairs are enabled for consistency protection 153
Blocking symcg enable on R2 side 154
Deleting an SRDF consistency group 154
Suspending SRDF consistency protection 155
Using the msc_cleanup command 156
Modifying consistency groups 158
Restrictions 158
Preparing the staging area 159
Restrictions - add operations 162
Restrictions - remove operations 163
Adding devices to an RDF1 consistency group 164
Adding devices to a concurrent RDF1 consistency group 164
Adding devices to a cascaded RDF1 consistency group 166
Removing devices from an RDF1 consistency group 167
Removing devices from a concurrent RDF1 consistency group 168
Removing devices from a cascaded RDF1 consistency group 168
Recovering from a failed dynamic modify operation 168
Consistency groups with a parallel database 170
Consistency groups with BCV access at the target site 171

Chapter 6 Concurrent Operations

Overview 174
Restrictions 174
Other documentation 176
Concurrent SRDF operations 176
Creating/establishing concurrent SRDF devices 176
Splitting concurrent SRDF devices 177
Restoring concurrent devices 177
Viewing concurrent SRDF devices 179

Chapter 7 Device Migration Operations

Overview 182
Requirements 182
R1 device migration 183
R2 device migration 186
R1 and R2 migration procedures 189
Before you begin 189
Restrictions and limitations 190
Sample procedure: migrating R1 devices 191
Sample procedure: migrating R2 devices 199
SRDF pair states for migration 201
Pair states for migrate -setup 202

Pair states for migrate -replace for first leg of concurrent SRDF 203
Pair states for migrate -replace for second leg of concurrent SRDF 205

Chapter 8 SRDF/Automated Replication

Overview 210
Restrictions 210
SRDF/Automated Replication operations 211
Configuring single-hop sessions 211
Configuring multi-hop sessions 215
Concurrent BCVs with SRDF/AR 217
Setting replication cycle parameters 218
Logging symreplicate steps 220
Clustered SRDF/AR 220
Writing log files to a specified SFS 220
Restarting from another host 221
Displaying log files written to the SFS 222
Deleting a log file written to SFS 222
Setting symreplicate parameters in the options file 222
Option file format 223
Setting replication retry and sleep times 223
Setting the symreplicate control parameters 224
Managing locked devices 228
Recovering locks 228
Releasing locks 228
Acquiring persistent locks 229

Chapter 9 TimeFinder and SRDF operations

TimeFinder consistent splits in SRDF configurations 232
Enginuity Consistency Assist 232
Multi-hop operations 233
Before you begin: system-wide device groups 233
System-wide splits 235
Controlling basic operations in a multi-hop configuration 236

Chapter 10 Automated Recovery Operations

Overview 240
Requirements and strategies 240
symrecover command 241
Starting SRDF Automated Recovery 242
Stopping SRDF Automated Recovery 244
symrecover options file parameters 244

Appendix A SRDF operations and pair states

SRDF operations and applicable pair states 250
Control operations for R1 -> R2 pair states 250
Concurrent SRDF operations and applicable pair states 254
Concurrent SRDF: 1st leg R1-> 2nd leg R1 pair states 254
Concurrent SRDF: 1st leg R2 -> 2nd leg R2 pair states 257
Consistency group operations and applicable pair states 260

Appendix B SRDF operations and TimeFinder sessions

SRDF operations during TimeFinder/Snap/VP Snap sessions 262
SRDF operations when R1 is source of TimeFinder Snap/VP Snap..... 262
SRDF operations when R1 is target of TimeFinder Snap/VP Snap 264
SRDF operations when R2 is source of TimeFinder Snap/VP Snap..... 266
SRDF operations when R2 is target of TimeFinder Snap/VP Snap 269
SRDF operations for TimeFinder/Clone sessions..... 272
SRDF operations when R1 is source of TimeFinder Clone 272
SRDF operations when R1 is target of TimeFinder Clone..... 274
SRDF operations when R2 is source of TimeFinder Clone 277
SRDF operations when R2 is target of TimeFinder Clone..... 280
SRDF operations for Extent-level TimeFinder/Clone sessions 282
SRDF operations when R1 is source of Extent-level Clone 282
SRDF operations when R1 is target of Extent-level Clone..... 284
SRDF operations when R2 is source of Extent-level Clone 287
SRDF operations when R2 is target of Extent-level Clone..... 290
SRDF set operations for TimeFinder/Snap sessions 292
SRDF set operations when R1 is source of TimeFinder/Snap 292
SRDF set operations when R1 is target of TimeFinder/Snap 292
SRDF set operations when R2 is source of TimeFinder/Snap 293
SRDF set operations when R2 is target of TimeFinder/Snap 293
SRDF set operations for TimeFinder/Clone sessions 295
SRDF set operations when R1 is source of TimeFinder/Clone 295
SRDF set operations when R1 is target of TimeFinder/Clone 295
SRDF set operations when R2 is source of TimeFinder/Clone 296
SRDF set operations when R2 is target of TimeFinder/Clone 296
SRDF set operations for Extent-level TimeFinder/Clone sessions..... 298
SRDF set operations when R1 is source of Extent-level Clone..... 298
SRDF set operations when R1 is target of Extent-level Clone 298
SRDF set operations when R2 is source of Extent-level Clone..... 299
SRDF set operations when R2 is target of Extent-level Clone 299
SRDF operations for TimeFinder Snapvx 300
SRDF operations when R1 is source of Snapvx..... 300
SRDF operations when R1 is target for Snapvx..... 302
SRDF operations when R2 is source of Snapvx..... 303
SRDF operations when R2 is target for Snapvx..... 306
SRDF set operations for TimeFinder Snapvx sessions..... 307
SRDF set operations when R1 is source of TimeFinder Snapvx 308
SRDF set operations when R1 is target of TimeFinder Snapvx..... 308
SRDF set operations when R2 is source of TimeFinder Snapvx 309
SRDF set operations when R2 is target of TimeFinder Snapvx..... 309

FIGURES

	Title	Page
1	Typical 2-site SRDF configuration in open systems host environment	18
2	SRDF device and link states	37
3	Full establish	53
4	Incremental establish	55
5	Failback of an SRDF device	57
6	Failover of an SRDF device	59
7	Restore (full) an SRDF device	67
8	Incremental restore an SRDF device	68
9	Split an SRDF pair	70
10	Update SRDF device track tables	74
11	SRDF/A legacy mode	109
12	SRDF/A multi-cycle mode	110
13	Running redundant hosts to ensure consistency protection	144
14	Staging area for adding devices to the R1CG consistency group	161
15	R1CG consistency group after a dynamic modify add operation	161
16	Preparing the staging area for removing devices from the MyR1 CG	162
17	MyR1 CG after a dynamic modify remove operation	162
18	Adding a device to independently-enabled SRDF groups of a concurrent CG	165
19	Adding devices to independently-enabled SRDF groups of a cascaded CG	167
20	Using an SRDF consistency group with a parallel database configuration	170
21	Using an SRDF consistency group with BCVs at the target site	171
22	Concurrent SRDF	174
23	Concurrent SRDF/S to both R2 devices	175
24	Concurrent SRDF/A to both R2 devices	175
25	Restoring the R1 a concurrent configuration	178
26	Restoring the source device and mirror in a concurrent SRDF configuration	179
27	R1 migration: configuration setup	183
28	R1 migration: establishing a concurrent relationship	184
29	R1 migration: replacing the source device	185
30	Migrating R2 devices	186
31	R2 migration: configuration setup	187
32	R2 migration: establishing a concurrent relationship	188
33	R2 migration: replacing the target device	189
34	R1 migration example: Initial configuration	191
35	Concurrent SRDF relationship	194
36	Migrated R1 devices	197
37	R2 migration example: Initial configuration	199
38	Concurrent SRDF relationship	200
39	Migrated R2 devices	201
40	R1 migration: applicable R1/R2 pair states for migrate -setup	202
41	R2 migration: applicable R1/R2 pair states for migrate -setup	203
42	R1 migration: R11/R2 applicable pair states for migrate -replace (first leg)	204
43	R2 migration:R11/R2 applicable pair states for migrate -replace (first leg)	205
44	R1 migration: applicable R11/R2 pair states for migrate -replace (second leg)	206
45	R2 migration: applicable R11/R2 pair states for migrate -replace (second leg)	207
46	Automated data copy path in single-hop SRDF systems	211
47	Automated data copy path in multi-hop SRDF	215
48	Concurrent BCV in a multi-hop configuration	218
49	ECA consistent split	233
50	Commands used to perform splits in a complex configuration	234

51 Basic operations in multi-hop SRDF configurations..... 237
52 SRDF Recovery environment 240

TABLES

	Title	Page
1	SRDF documentation	19
2	SYMCLI SRDF commands	25
3	symrdf command options	27
4	Commands to display and verify SRDF, devices, and groups	29
5	Options for symrdf list command	32
6	SRDF pair states	38
7	Possible SRDF device and link state combinations	40
8	Access rights required by an array	41
9	SRDF control operations summary	46
10	Device type combinations for creating SRDF pairs	90
11	SRDF device states before swap operation	101
12	SRDF/A control operations	114
13	Consistency modes for concurrent mirrors	153
14	Allowable device types for adding devices to an RDF1 CG	164
15	Allowable device types for adding devices to a concurrent RDF1 CG	164
16	Supported consistency modes for concurrent SRDF groups	164
17	Allowable device types for adding devices to a cascaded RDF1 CG	166
18	Supported consistency modes for cascaded hops	166
19	Allowable device types for removing devices from an RDF1 CG	167
20	Allowable device types for removing devices from a concurrent RDF1 CG	168
21	Allowable device types for removing devices from a cascaded RDF1 CG	168
22	SRDF migrate -setup control operation and applicable pair states	201
23	SRDF migrate -replace control operation and applicable pair states	203
24	SRDF migrate -replace control operation and applicable pair states	205
25	Initial setups for cycle timing parameters	219
26	Basic operations in a multi-hop configuration	236
27	symrecover options file parameters	244
28	SRDF control operations and applicable pair states	250
29	SRDF control operations and applicable states for concurrent R1 pairs	254
30	SRDF control operations and applicable states for concurrent R2 pairs	257
31	SRDF control operations and applicable pair states for devices in an SRDF/CG	260
32	Allowable SRDF operations when R1 is the source of a TimeFinder/Snap or VP Snap	262
33	Allowable SRDF operations when R1 is the target of a TimeFinder/Snap or VP Snap	264
34	Allowable SRDF operations when R2 is the source of a TimeFinder/Snap or VP Snap	266
35	Allowable SRDF operations when R2 is the target of a TimeFinder/Snap or VP Snap	269
36	Allowable SRDF operations when R1 is the source of a TimeFinder/Clone	272
37	Allowable SRDF operations when R1 is the target of a TimeFinder/Clone	274
38	Allowable SRDF operations when R2 is the source of a TimeFinder/Clone	277
39	Allowable SRDF operations when R2 is the target of a TimeFinder/Clone	280
40	Allowable SRDF operations when R1 is the source of an Extent-level Clone	282
41	Allowable SRDF operations when the R1 is the target of an Extent-level Clone	284
42	Allowable SRDF operations when R2 is the source of an Extent-level Clone	287
43	Allowable SRDF operations when the R2 is the target of an Extent-level Clone	290
44	Allowable SRDF set operations when R1 is the source of a TimeFinder/Snap	292
45	Allowable SRDF set operations when R1 is the target of a TimeFinder/Snap	292
46	Allowable SRDF set operations when R2 is the source of a TimeFinder/Snap	293
47	Allowable SRDF set operations when R2 is the target of a TimeFinder/Snap	293
48	Allowable SRDF set operations when R1 is the source of a TimeFinder/Clone	295
49	Allowable SRDF set operations when R1 is the target of a TimeFinder/Clone	295
50	Allowable SRDF set operations when R2 is the source of a TimeFinder/Clone	296

51 Allowable SRDF set operations when R2 is the target of a TimeFinder/Clone 296
52 Allowable SRDF set operations when R1 is the source of an Extent-level Clone 298
53 Allowable SRDF set operations when R1 is the target of an Extent-level Clone..... 298
54 Allowable SRDF set operations when the R2 is the source of an Extent-level Clone 299
55 Allowable SRDF set operations when R2 is the target of an Extent-level Clone..... 299
56 Allowable SRDF operations when R1 is the source of a TimeFinder Snapvx 300
57 Allowable SRDF operations when R1 is the target of a TimeFinder Snapvx..... 302
58 Allowable SRDF operations when R2 is the source of a TimeFinder Snapvx 304
59 Allowable SRDF operations when R2 is the target of a TimeFinder Snapvx..... 306
60 Allowable SRDF set operations when R1 is the source of a TimeFinder Snapvx..... 308
61 Allowable SRDF set operations when R1 is the target of a TimeFinder Snapvx 308
62 Allowable SRDF set operations when R2 is the source of a TimeFinder Snapvx..... 309
63 Allowable SRDF set operations when R2 is the target of a TimeFinder Snapvx 309

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 EMC Online Support https://support.EMC.com. Check to ensure that you are using the latest version of this document.

Purpose

This document describes how to use Solutions Enabler SYMCLI to manage SRDF®.

Audience

This document is for advanced command-line users and script programmers to manage various types of control operations on arrays and devices using Solutions Enabler's SYMCLI commands.

Related documentation

The following documents provide additional information about Solutions Enabler:

- *EMC Solutions Enabler V8.0.1 Release Notes* - Describes new features and any known limitations.
- *EMC Solutions Enabler Installation Guide* - Provides host-specific installation instructions.
- *EMC Solutions Enabler CLI Command Reference* - Documents the SYMCLI commands, daemons, error codes and option file parameters provided with the Solutions Enabler man pages.
- *EMC VMAX3 Family Product Guide* - Describes the VMAX3 platform and software products available.
- *EMC Solutions Enabler Array Management CLI User Guide* - Describes how to configure array control, management, and migration operations using SYMCLI commands.
- *EMC Solutions Enabler TimeFinder Family CLI User Guide* - Describes how to configure and manage TimeFinder environments using SYMCLI commands.
- *EMC Solutions Enabler SRM CLI User Guide* - Provides Storage Resource Management (SRM) information related to various data objects and data handling facilities.
- *EMC VMAX Family Security Configuration Guide* - Describes how to configure VMAX Family security settings.

The following provide additional information:

- *EMC VMAX V3 Family Documentation Set* - Contains documentation related to the VMAX 100K, 200K, and 400K arrays.
- *EMC VMAX Family (10K, 20K, 40K) Documentation Set* - Contains documentation related to the VMAX 10K, 20K, and 40K arrays.
- *EMC VMAX3 Family with HYPERMAX OS Release Notes* - Detail new features and any known limitations.
- *EMC VMAX Family Viewer for Desktop and iPad®* - Illustrates system hardware, incrementally scalable system configurations, and available host connectivity offered for VMAX arrays.
- E-Lab™ Interoperability Navigator (ELN) - Provides a web-based interoperability and solution search portal. You can find the ELN at https://elabnavigator.EMC.com.
- SolVe Desktop - — Provides links to documentation, procedures for common tasks, and connectivity information for 2-site and 3-site SRDF configurations. To download the SolVe Desktop tool, go to EMC Online Support at https://support.EMC.com 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 it is installed, please familiarize yourself with the information under Help tab.

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 arrays, and notifications
Bold	Used in running (nonprocedural) text for names of commands, daemons, options, programs, processes, services, applications, utilities, kernels, notifications, array 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, 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 SRDF CLI

This chapter describes the following topics:

- Introduction 18
- SRDF documentation 19
- What's new in Solutions Enabler 8.0.1 19
- SYMCLI for SRDF 24
  - "SYMCLI SRDF commands" on page 25
  - "symrdf command options" on page 27
  - "Commands to display, query and verify SRDF configurations" on page 29
  - "symrdf list command options" on page 32
  - "ping command" on page 34
  - "Using the verify command" on page 34
- Understanding SRDF pair states and links 37
- Before you begin 40

Introduction

SRDF® is a family of disaster recovery, parallel processing, and data migration solutions.

SRDF configurations require at least two arrays; a *primary* array and at least one *secondary* array. The arrays can be located in the same room, in different buildings within the same campus, or hundreds, even thousands of kilometers apart.

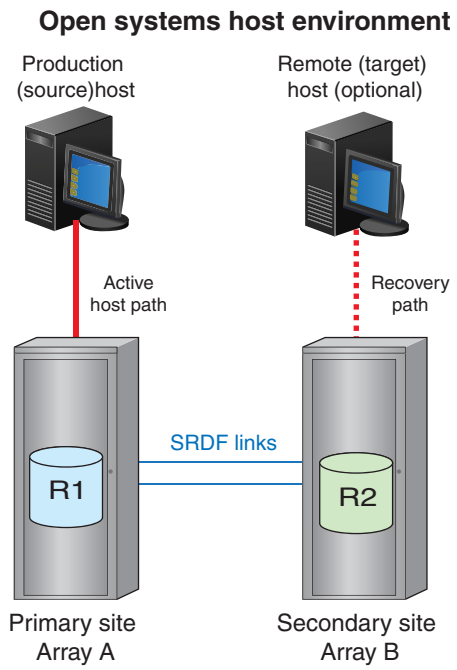


Figure 1 Typical 2-site SRDF configuration in open systems host environment

In the **open systems host** environment:

- A control host at the production site is connected to the primary array.
- Primary and secondary sites are connected other over SRDF links.
- The production host writes I/O to devices (R1) at the primary site.
- SRDF mirrors the production I/O to the devices (R2) at the secondary site(s).

HYPERMAX OS

VMAX 100K/200K/400K arrays (referred to as VMAX3™ arrays) running HYPERMAX OS can use SRDF to replicate to:

- VMAX3 arrays running HYPERMAX OS.
- VMAX 10K/20K/40K arrays running Enginuity™ version 5876 with applicable ePack.

Enginuity versions 5773 - 5876

Refer to the *SRDF Two-site Interfamily Connectivity* tool for information about SRDF features supported between arrays running Enginuity versions 5773 through 5876.

SRDF documentation

Table 1 SRDF documentation

For information on	See
Technical concepts and operations of the SRDF product family. Topics include: • SRDF Solutions • SRDF interfamily connectivity • SRDF concepts and terminology • SRDF/DM, SRDF/AR, SRDF/Concurrent • SRDF integration with other products	*EMC VMAX3 Family Product Guide*
Configure and manage arrays using the SYMCLI.	*EMC Solutions Enabler Array Management CLI User Guide*
Determine which SRDF replication features are supported between two or three arrays running Enginuity 5773-5876, or HYPERMAX OS.	Download the SolVe Desktop and load the VMAX Family and DMX procedure generator. Select *SRDF 2-site Interfamily connectivity* or *SRDF 3-site Interfamily connectivity.*
Securing your configuration	*EMC VMAX Family Security Configuration Guide*
Host connectivity	EMC Host Connectivity Guides for your operating system.
Managing legacy versions of SRDF using SYMCLI	Download the SolVe Desktop and load the VMAX Family and DMX procedure generator. Select VMAX 10K, 20K, 40K, DMX -> Customer procedures -> *Managing SRDF using SYMCLI*.

What's new in Solutions Enabler 8.0.1

This section lists changes introduced by HYPERMAX OS/Solutions Enabler 8.0.1.

Multi-cores, multi-ports per director

In 5876 and earlier, all front-end emulations supported a fixed number (up to two) of ports. Multiple front-end emulations could exist on the same director board, providing additional host connectivity, but all such front-end directors were limited to one or two physical ports.

VMAX 100K/200K/400K arrays (referred to as VMAX3 arrays) running HYPERMAX OS/Solutions Enabler 8.0.1 support a single front-end emulation of each type (FA, EF, etc.) per director, but each of these emulations supports variable number of physical ports. Both the SRDF Gigabit Ethernet (RE) and SRDF Fibre Channel (RF) emulations can use any port on the director. The relationship between the SRDF emulation and resources on a director is configurable:

1 director/1 or multiple CPU cores/1 or multiple ports

Connectivity is not bound to a fixed number of CPU cores. You can change the amount of connectivity without changing CPU power.

The SRDF emulation supports up to 16 front-end ports per director (4 front-end modules per director), any or all of which can be used by SRDF. Both the SRDF Gigabit Ethernet and SRDF Fibre Channel emulations can use any port.

Note: If hardware compression is enabled, the maximum number of ports per director is 12.

When you create an SRDF group on VMAX 100/200/400 arrays, you must select both the director AND the ports for the SRDF emulation to use on each side.

"symrdf addgrp command syntax" and "symrdf modifygrp command syntax" have been modified to specify multiple ports.

A new option **-port ‹# | ALL›** for the **symsan list -sanrdf** command displays either the specified port number (0 - 31) or all ports. Output of the command displays fields for local and remote ports and status for local port.

Improved SRDF/A resilience and performance

SRDF/A's new multi-cycle mode creates multiple SRDF/A cycles on the R1 side at regular intervals, to provide smaller incremental updates to the R2.

Creation of a new cycle is de-coupled from the data being committed to the R2 side. If the link speed decreases or the apply rate on the R2 side increases, more SRDF/A cycles can be added to the R1 side. Smaller cycles on the R1 side provide smaller incremental updates to the R2. Smaller cycles transmitted to the R2 side removes the need for spillover on the R2 side.

2 fields in the output of the **symrdf query** command have new meaning in multi-cycle mode:

- Avg Transmit Cycle Time - Average time to transfer a Transmit Cycle from the R1 side to the R2 side.

  R1 Transmit cycles are transferred to the R2 side independent of the creation of a new Capture cycle on the R1 side.

- Transmit Queue Depth on R1 Side - Total number of Transmit cycles on the R1 side.

  The total of R1 cycles is retrieved from the R1 side, regardless of which side (R1 or R2) the CLI command is executed on.

EMC VMAX3 Family Product Guide provides a detailed description of the changes to SRDF/A.

Improved DSE

Delta Set Extension (DSE) is enabled by default, and uses a preconfigured storage resource pool (SRP), so you do not need to configure and manage a DSE pool, or associate an SRDF group with a DSE pool.

When DSE is activated, SRDF/A group-level write pacing paces host I/O to match the DSE spillover rate, increasing the resilience of SRDF/A sessions.

You can modify the amount of capacity in the SRP.

EMC VMAX3 Family Product Guide provides a detailed description of the changes to DSE.

Dynamic SRDF only

All SRDF groups and devices are created as dynamic.

In configurations where one array is running HYPERMAX OS, and one or more other arrays are running Enginuity 5876, all SRDF groups and devices must be dynamic.

No adaptive copy write pending

Adaptive copy write pending mode is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS.

The following operations are impacted:

- **createpair** - If adaptive copy write pending mode is specified and the R1 side is on an array running HYPERMAX OS, the operation fails.
- **set mode acp_wp** - If the R1 mirror of the SRDF pair is on an array running HYPERMAX OS, the operation fails.
- **swap, failover, failback, -establish|-restore** - If the pair being operated on is in adaptive copy write pending mode, and the R2 side is on an array running HYPERMAX OS, after the operation the RDF pair is set to adaptive copy disk mode.
- **migrate - replace R1**- When replacing an existing R1 device in an SRDF pair with a new device on a different array, the mode of the new device pair is set to the RDF mode of the R1 device being replaced.

No meta devices

HYPERMAX OS does not support meta devices.

Device pairs between non-meta devices on an array running Solutions Enabler 8.0.1/HYPERMAX OS and a meta device on an array running Solutions Enabler 7.6/Enginutiy 5876 are supported, but require the applicable Enginuity ePack on the array running Enginuity 5876.

The meta device in device pairs between a non-meta device (Solutions Enabler 8.0.1/HYPERMAX OS) and a meta device (Solutions Enabler 7.6/Enginutiy 5876) can be a CKD striped device, or an FBA concatenated or striped device.

Increased track size

Track size for FBA devices increased from 64K in Enginuity 5876 to 128K in HYPERMAX OS. Geometry Compatibility Mode supports full SRDF functionality for devices on arrays running Enginuity 5876 with an odd number of cylinders paired with devices on arrays running HYPERMAX OS.

See "Geometry Compatible Mode" on page 23.

Display remote SID

symdev show and **symcfg -ra** have been updated to report the remote Symmetrix ID when the RDF pair state is Partitioned.

Restrictions

- The following configurations are not supported in Solutions Enabler 8.0.1/HYPERMAX OS:

- Cascaded SRDF
- Cascaded SRDF/Star
- Cascaded SRDF/EDP
- Concurrent SRDF/Star

- Solutions Enabler 7.6/Enginuity 5876 cannot discover arrays running Solutions Enabler 8.0.1/HYPERMAX OS.
- When one side of an SRDF link is running HYPERMAX OS, and the other side is running Enginuity 5876, the RDF pair state is Partitioned.

SRDF backward compatibility to Enginuity 5876

You can use SRDF features in Solutions Enabler 8.0.1/HYPERMAX OS to replicate to/from:

- VMAX 3 arrays also running HYPERMAX OS.
- VMAX 10K/20K/40K arrays running Enginuity 5876 with the applicable ePack.

When one array in an SRDF configuration is running HYPERMAX OS, and one or more other arrays are running Enginuity 5876, the following rules and restrictions apply:

- All SRDF groups and devices must be dynamic.
- SRDF/A sessions use legacy mode. See "SRDF/A cycle modes" on page 108.
- Directors on arrays running HYPERMAX OS support up to 16 ports and 250 SRDF groups. If a port on the array running HYPERMAX OS is connected to an array running Enginuity 5876:
  - The port supports a maximum of 64 RDF groups.
  - The director associated with the port supports a maximum of 186 RDF groups.
- SRDF device pairs with meta-devices on one side are allowed if the meta-devices are on the array running Enginuity 5876.

  Output of the **symrdf query**, **symrdf list**, and **symdev show** commands has been enhanced to display RDF mode as MIXED when a meta head device on an array running Enginuity 5876 has different RDF modes than its members.

  When you see a device in MIXED mode, you can then use the **set mode** command to choose the appropriate mode for the device pair.
- The **symcfg list -ra** command has been modified to report the remote SID when the RDF Pair State is Partitioned.
- Adaptive copy write pending is not supported in HYPERMAX OS.
  - For **swap** and **failover** operations - If the R2 device is on an array running HYPERMAX OS, and the mode of the R1 is adaptive copy write pending, SRDF sets the mode to adaptive copy disk.
  - For **migrate -replace R1** operations - If the R1 (after the replacement) is on an array running HYPERMAX OS, and the mode of the R1 is adaptive copy write pending mode, SRDF sets the mode of the migrated pair to adaptive copy disk.

Deprecated commands and options

- The **symrdf monitor** command, including all options and arguments has been removed from the Solutions Enabler SYMCLI. The **symcfg monitor** command provides identical functionality.
- The **-port** option has been removed from the **symcfg -ra all list** command.
- The **-vxfs** and **-rdb** options have been removed from the **symreplicate start** command.
- The SYMAPI_SYNC_DIRECTION option has been removed from the options file.

Geometry Compatible Mode

An array running HYPERMAX OS cannot create a device that is exactly the same size as a device with an odd number of cylinders on an array running Enginuity 5876. In order to support the full suite of features:

- SRDF requires that R1 and R2 devices in a device pair be same size.
- TimeFinder requires that source and target devices are the same size.

HYPERMAX OS manages the size difference without user intervention, using a new device attribute, Geometry Compatible Mode (GCM). A device with GCM set is treated as half a cylinder smaller than its true configured size, enabling full functionality between HYPERMAX OS and Enginuity 5876 for SRDF, TimeFinder SnapVX, and TimeFinder emulations (TimeFinder/Clone, TimeFinder VP Snap, TimeFinder/Mirror), and ORS.

You can use the **set** command to manually set the GCM attribute for devices on V3 arrays. However, for most operations, Solutions Enabler sets it automatically when required. For example, Solutions Enabler automatically sets the GCM attribute when restoring from a physically larger R2.

- The **symdev set** /**unset**, **symdg set**/**unset**, **symcg set**/**unset**, and **symsg set**/**unset** commands have been enhanced with a new option **-gcm** to set and unset GCM for a device or group.
- The **symrdf createpair** command has been enhanced to transparently set/unset the GCM attribute as part of the create pair operation, as follows:
  - Set the GCM attribute for a target device that is configured ½ a cylinder larger. The source of the copy can be:
    - A device on an array running Enginuity 5876 with an odd number of cylinders and capacity that matches the GCM size of the target device.
    - A GCM device on an array running HYPERMAX OS.
  - Unset the GCM attribute for a target device that is configured the exact same size as the source of the copy. The source of the copy can be:
    - A source device on an array running Enginuity 5876 with even number of the cylinders and capacity that matches the size of the target device on the array running HYPERMAX OS
    - A source device on the array running HYPERMAX OS without the GCM attribute.

IMPORTANT

The GCM setting for a device cannot be changed if the target of the data device is already part of another replication session.

- The **symdev show**, **symdev list –v**, **symdg show ld**, **symdg list ld –v**, **sympd show**, and **sympd list –v** commands have been enhanced to report the GCM attribute.

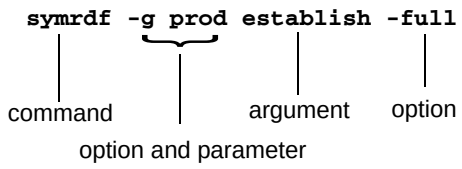
SYMCLI for SRDF

This section describes:

- "SYMCLI command syntax"
- "Getting command help"
- "Setting environmental variables"
- "Presetting names and IDs"
- Table 2 on page 25 lists the four main SRDF SYMCLI commands to establish, maintain and monitor SRDF configurations.
- Table 3 on page 27 lists options for the symrdf command.
- Table 4 on page 29 lists a variety of commands to display, query and verify your SRDF configuration.
- Table 5 on page 32 lists options for the symrdf list command

SYMCLI command syntax

The following example shows the command syntax for initiating a full establish for the SRDF pairs in the `prod` device group.



Getting command help

Type *`command`* `-h` to display command line help for the specified command.

To display help for the symrdf command:

```
symrdf - h
```

On UNIX arrays, type `man` *`command`* to display the man page for the specified command.

To display the man page for the symrdf command:

```
man symsrdf
```

- **On UNIX hosts**: specify the SYMCLI man page directory (/usr/symcli/man/) in the SYMCLI_MANPATH environment variable.
- **On Windows hosts**: the default directory for man pages is C:\Program Files\EMC\symcli\man

Setting environmental variables

SYMCLI includes variables to streamline command line sessions.

To display a list of variables that can be set for your SYMCLI session:

```
symcli -env
```

To view the variables that are set:

```
symcli - def
```

To set a variable, type `setenv VARIABLE_NAME value`:

```
setenv SYMCLI_VERBOSE 1
```

To turn off a variable, type `unsetenv VARIABLE_NAME`:

```
unsetenv SYMCLI_VERBOSE
```

Presetting names and IDs

Use the SYMCLI environmental variables to preset the identity of objects, such as SID. Once the object's identity is defined, you do not need to type them in the command line.

To set the SID for all **-sid** arguments:

```
set env SYMCLI_SID 000192601365
```

To view a list of environment variables that can be set for a given SYMCLI session:

```
symcli -env
```

To view the current setting for all environment variables:

```
symcli -def
```

SYMCLI SRDF commands

Table 2 SYMCLI SRDF commands (page 1 of 2)

Command	Description	For more information
`symrdf`	Control operations on SRDF devices, including: • Establishes (mirrors) an SRDF pair by initiating a data copy from the source (R1) side to the target (R2) side. This operation can be a full or incremental establish. • Restores remote mirroring. Initiates a data copy from the target (R2) side to the source (R1) side. This operation can be a full or incremental restore. • Splits an SRDF pair, which stops mirroring for the SRDF pairs in a device group. • Fails over and back from the source (R1) side to the target (R2) side, switching data processing to the target (R2) side. • Updates the source (R1) side after a failover, while the target (R2) side may still be operational to its local host(s). • Swaps the source (R1) and target (R2) destinations between the target and the source. • Creates, deletes, or swaps dynamic SRDF device pairs. • Performs dynamic SRDF group controls to add, modify, and remove dynamic groups. • Enables link domino locally or remotely when creating dynamic groups. • Enables auto link recovery locally or remotely when creating dynamic groups. • Enables/disables consistency for SRDF/A capable devices operating in asynchronous mode that are managed by a device group or file.	See: • "Basic SRDF control operations summary" on page 46 • "SRDF basic control operations" on page 49 • `symrdf` man page.

Table 2 SYMCLI SRDF commands (page 2 of 2)

Command	Description	For more information
`symreplicate`	Invokes a replicate session that generates automated recurrent, background copies of the standard data following a path across SRDF links and cascading BCVs. Start, stop, and restart a replicate session. Used for SRDF/Automated Replication.	See: • "Automated Recovery Operations" on page 239 • `symreplicate` man page.
`symstar`	Uses concurrent SRDF/Synchronous and SRDF/Asynchronous links to replicate source data synchronously to a nearby regional site and asynchronously to a distant remote site.	See: • "SRDF/Star Operations" on page 211 • `symstar` man page.
`symrecover`	Monitor the session state during attempts to restart a group session if it enters the suspended or partitioned state.	See: • "symrecover command" on page 241 • `symrecover` man page.

symrdf command options

Note: Refer to the `symrdf` man page for more detailed descriptions of the options used with the symrdf command.

Table 3 symrdf command options (page 1 of 3)

Command option	Description
`-all`	Targets the SRDF action at all devices in the device group, which includes standard SRDF devices and any BCV SRDF devices that are locally associated with the device. When used with `list`, the `-all` option shows all SRDF mirrors of the selected devices. **Note:** The -all flag is not supported for SRDF control operations on device groups or composite groups with type ANY.
`-autostart`	Specifies whether SRDF/A DSE is automatically activated when an SRDF/A session is on (Enabled) or off (Disabled) for the SRDF group. Valid values are on (Enabled) or off (Disabled). **Note:** AutoStart for DSE is enabled by default in HYPERMAX OS.
`-bcv`	Targets the specified BCV devices associated with a device or composite group and are configured as SRDF BCV devices. By default, only the SRDF standard devices are affected by the SRDF control operations.
`-brbcv`	Targets the SRDF action at the specified remotely associated SRDF (Hop 2) BCV devices that can be paired with the remote mirrors of the local BCV devices.
`-both_sides`	Targets the SRDF control operation at both sides of an SRDF link.
`-bypass`	Causes the SRDF control operation to bypass existing exclusive locks. Use this option ONLY if no other SRDF operation is in progress at either the local and/or remote arrays.
`-c`	Counts the number of times to display or to attempt acquiring exclusive locks on the host database, the local array, and the remote arrays. If the (`-c`) option is not specified and an interval (`-i`) is specified, the program loops continuously to produce infinite redisplays, or until the SRDF control or `set` operation starts.
`-cg`	Specifies the composite group for SRDF operations.
`-cons_exempt`	For an SRDF group supporting an active SRDF/A session, allows devices to be added, removed, or suspended without affecting the state of the SRDF/A session or requiring that other devices in the session be suspended to perform the control operation. When used with `list`, this option shows devices that are consistency exempt or that are paired with devices that are consistency exempt.
`-fibre`	Uses the Fibre Channel communication protocol.
`-file`	Specifies the device file for SRDF operations.
`-force`	Performs the control operations on SRDF devices that are not in the expected state for a control operation. By using this option, the control operation is attempted, regardless of the pair state of the SRDF devices, and according to the rules in Table 28 on page 250.
`-format`	Used with `createpair` to clear all tracks on the R1 and R2 sides, ensuring no data exists on either side, and makes the R1 read write to the host.
`-full`	Requests a full establish or restore operation.
`-g`	Specifies the device group for SRDF operations.
`-h`	Provides brief, online help.

Table 3 symrdf command options (page 2 of 3)

Command option	Description
-hwcomp	Enables or disables hardware compression, which minimizes the amount of data to transmit over an SRDF link.
-i	Executes a command at repeat intervals to display information or to attempt to acquire an exclusive lock on the host database, the local array, and the remote arrays. The default interval is 10 seconds. The minimum interval is 5 seconds.
-immediate	Applies only to SRDF/A-backed devices. Causes failover, split, and suspend actions to drop the SRDF/A session immediately.
-label	Specifies a label for a dynamic SRDF group.
-noecho	Suppresses the display of progress status information.
-noprompt	Suppresses the message asking you to confirm an SRDF control operation.
-nowd	Bypasses the check to ensure the target of the operation is not writable by the host.
-offline	Obtains the data strictly from the configuration database. No connections are made to any arrays. The symrdf command uses information previously gathered from the array and held in the host database as opposed to interrogating the array directly. The offline option can alternatively be set by assigning the environment variable SYMCLI_OFFLINE to 1.
-rdfa_devpace	Indicates the operation affects the SRDF/A device-level write pacing feature.
-rdfa_dse	Indicates the operation affects the SRDF/A Delta Set Extension (DSE) feature.
-rdfa_pace	Indicates the operation affects both the group-level and the device-level components of the SRDF/A write pacing feature.
-rdfa_wpace	Indicates the operation affects the SRDF/A group-level write pacing feature.
-rdfa_wpace_exempt	Excludes the specified devices from SRDF/A group-level write pacing.
-rdfg	Targets a specific SRDF group number.
-rdf_mode	Used in createpair to set the SRDF mode of device pairs to one of the following: synchronous (sync), semi-synchronous (semi), asynchronous (async), adaptive copy disk mode (acp_disk), or adaptive copy write pending mode (acp_wp). **Note:** Adaptive copy write pending mode (acp_wp) is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS.
-refresh	Marks the source (R1) devices or the target (R2) devices to refresh from the remote mirror.
-remote	Requests a remote data copy with the failback, restore, resume, createpair and update actions. When the concurrent links are ready, data is also copied to the concurrent SRDF mirror. For these actions to execute, use this option or suspend the concurrent links.
-remote_rdfg	Specifies the SRDF group number for the remote array.
-remote_sid	Specifies the remote array ID.
-restore	Used with failover to swap the R1 and R2 and restore the invalid tracks on the new R2 side (formerly R1) to the new R1 side (formerly R2). For more information, refer to "Dynamic failover restore" on page 104.
-rp	Used with -establish|-restore, createpair, failback, merge, restore, resume, update, and refresh, to allow the operation even when one or more devices are tagged for RecoverPoint. When used with refresh, only allowed for refresh R1.
-rrbcv	Targets the SRDF action at the specified remotely associated SRDF (Hop 2) BCV devices, which can be paired with the remote mirrors of the local standard devices.

Table 3 symrdf command options (page 3 of 3)

Command option	Description
`-sid`	Specifies the local array ID.
`-swcomp`	Enables or disables software compression, which minimizes the amount of data to transmit over an SRDF link.
`-symforce`	Requests that the array force an operation by overriding all instances causing the array to reject an operation. The SYMAPI_ALLOW_RDF_SYMFORCE setting in the options file must be set to TRUE to use -symforce. With -symforce, a split command executes on an SRDF pair, even during a sync in progress state. **Note:** Use caution when applying this option as data can become lost or corrupted.
`-until`	Checks the number of invalid tracks that are allowed to build up from the active R2 local I/O before another update (R2 to R1) copy is retriggered. The update sequence loops until the invalid track count is less than the number specified for the `-until` value. Refer to "Continuous R1 updates" on page 75 for more information.
`-v`	Provides more detailed, verbose command output.

Commands to display, query and verify SRDF configurations

Table 4 lists SYMCLI commands to display, query, and verify your SRDF configuration.

Table 4 Commands to display and verify SRDF, devices, and groups

SYMCLI command	Description of command output
symcg list	
`symcg list`	Display the arrays attached to your hosts and SRDF information, such as: • SRDF devices on the arrays • SRDF device states • Number of invalid tracks for the source and the target devices
symcg show	
`symcg show` *`CompositeGroupName`*	Displays detailed information about the specified composite group.
symcfg list	
`symcfg list`	Displays the connectivity (Local or Remote) of each array. Useful for verifying that only one array is connected to the host in a SRDF/Star configuration.
`symcfg list -v`	Displays a more detailed (verbose) listing, including: • Concurrent SRDF Configuration State • Dynamic SRDF Configuration State • Concurrent Dynamic SRDF Configuration • RDF Data Mobility Configuration State
`symcfg list -sid` *`SID`* `-rdfg {all|`*`RDF_group_number`*`}`	Displays SRDF group-level settings for a specific group or all groups on a array, such as: • Group type • Director configuration • Group flags, including auto link recovery, link domino, SRDF/Star mode, SRDF software and hardware compression, and SRDF single round trip • SRDF flags, including consistency and SRDF status and mode

Table 4 Commands to display and verify SRDF, devices, and groups

SYMCLI command	Description of command output

`symcfg list -RA {all|Director}`
 	
`symcfg list -RA {all|Director} -rdfg RDF_group_number`
	Display all RDF directors, or a a specified RDF director.
	Display RDF directors associated with a specified SRDF group.

`symcfg list -RA {all|Director} -p {all|Port}`	HYPERMAX OS only. Display all ports or specified port for SRDF groups configured on all or the specified director.: • Port ID • Negotiated speed (Gb/second) • Maximum speed (Gb/second) • Port status (online or offline)
symdev list	
`symdev list -r1`	Displays only the R1 side of the SRDF configuration. R1 devices not in a device group are displayed as N/Grp'd.
`symdev list -sid SID -r1 -bcv`	Displays the RDF1 BCV devices for the specified array.
`symdev list -sid SID -devs Device:Device -lock`	Display devices with a device external lock. Displays a specified range of devices that have a device external lock.
symdev show	
`symdev show Device_number -sid SID`	Displays information about the specified SRDF devices, including: • SRDF device type and its group number • Whether it is paired with a diskless or concurrent device • Whether it has a standard/thin relationship • If the R2 device is larger than its R1 • Whether SRDF/A group-level and/or device-level write pacing is currently activated and supported for the SRDF/A session • Whether it is pace-capable
symdg show	
`symdg show RDF_group_name`	Displays detailed information about the specified RDF group.
symmir query	
`symmir -g RDF_group_name query`	Displays the BCV pairs in the specified device group and their state of mirroring.
symrdf list	
`symrdf list`	Displays the SRDF configuration, including source devices, remote target devices, and whether a device is an R1 or R2, SRDF group, replication method, pair state, invalid tracks, and the state of each device and the SRDF links that connect them. See Table 5, "Options for symrdf list command," for a list of symrdf list command options.

Table 4 Commands to display and verify SRDF, devices, and groups

SYMCLI command	Description of command output
symrdf query	
`symrdf -g RDF_group_name query`	Displays the state of the SRDF devices and their SRDF links in the specified device group. During normal operations, the SRDF pair is Synchronized: • The R1 devices and SRDF links are read-writable. • The R2 devices are write disabled. • The link is in synchronous replication. During failed over operations: • The R1 devices are write disabled. • The R2 devices are read/write. • The SRDF links are suspended.
`symrdf -g RDF_group_name query -all`	Displays the SRDF pair state of all devices in the specified device group, regardless of the device type.
`symrdf -g RDF_group_name query -bcv`	Displays the SRDF pair state of the SRDF BCV devices in the specified device group.
`symrdf -g RDF_group_name query -summary`	Displays summarized information about the state of the SRDF devices and their SRDF links in the specified device group, including: • Pair state • Number of invalid tracks on the source and target • Synchronization rate • Estimated time remaining for SRDF pair synchronization.
`symrdf -cg composite_group_name query`	Displays the state of the SRDF devices and their SRDF links in the specified composite group.
symrdf verify (group)	
`symrdf -g RDF_group_name verify`	Verifies/displays the state of devices in the specified device group.
`symrdf -g RDF_group_name verify -failedover`	Verifies/displays whether any devices in the specified device group are in the 'Failed Over' state.
`symrdf -g RDF_group_name verify -synchronized`	Verifies/displays whether any devices in the specified device group are in the 'Synchronized' state.
`symrdf -g RDF_group_name verify –i 30 -synchronized`	Verifies/displays a message every 30 seconds as to whether any devices in the specified device group are in the 'Synchronized' state.
`symrdf -g <RDF group name> verify -all -i 5 -synchronized`	Verifies/displays a message every 5 seconds as to whether any devices in the specified device group are in the 'Synchronized' state until all SRDF pairs are synchronized.
`symrdf -g RDF_group_name verify -split`	Verifies/displays whether any devices in the specified device group are in the 'Split' state.
`symrdf -g RDF_group_name verify -syncinprog`	Verifies/displays whether any devices in the specified device group are in the 'SyncInProg' state.
symrdf verify (composite group)	
`symrdf -cg composite_group_name verify`	Displays the state of devices in the specified composite group.
`symrdf -cg composite_group_name verify -consistent`	Verifies/displays whether devices in the specified composite group are in the 'Consistent' state.

Table 4 Commands to display and verify SRDF, devices, and groups

SYMCLI command	Description of command output
`symrdf -cg` *`composite_group_name`* `verify -consistent -noinvalids -i 60`	Monitors and reports (one line message) the clearing of invalid tracks. Verifies/displays a one-line message every 60 minutes as to whether any devices in the specified composite group are in the 'Consistent with no invalid tracks' state until all SRDF pairs in the group are the "Consistent with no invalid tracks" state.
`symrdf verify -summary -consistent -noinvalids -cg` *`composite_group_name`* `-i 45`	Monitors and reports (detailed message) the clearing of invalid tracks. Verifies/displays a detailed message every 45 minutes as to whether any devices in the specified composite group are in the 'Consistent with no invalid tracks' state until all SRDF pairs in the group are the "Consistent with no invalid tracks" state.
symstar list	
`symstar list`	Displays all the SRDF/Star composite groups visible to the host.
`symstar list -local`	Displays all the SRDF/Star composite groups local to your host.
symstat command options	
`-rdfg #|ALL`	Collect/display statistics for SRDF/A sessions by specified RA group number or all groups.
`-RepType rdf|rdfa`	Collect/display statistics for the specified replication type.
`-type REQUESTS|CACHE|CYCLE|RDF`	Collect/display specified statistic type. REQUEST (default) -I/O requests and throughput for device(s) and director(s). CACHE - Cache activity for selected front-end or remote link director(s). CYCLE - Active SRDF/A sessions or sessions that have non-zero cache usage. RDF - SRDF/A sessions.
`-RE|-RA|ALL`	Collect/display statistics for the specified SRDF director (-RA), GigE SRDF director (-RE) or both (ALL).
`-rdflink`	Collect/display SRDF link-level statistics.
`-rdf_nw_comp`	Collect/display SRDF network compression (iSCSI statistics).
`-rdf_spdlmt`	Collect/display SRDF speed limit information (iSCSI statistics).

symrdf list command options

Table 5 lists options for the `symrdf list` command, and describes the resulting output.

Table 5 Options for symrdf list command (page 1 of 3)

symrdf list option	Description of output
`-all`	Lists all mirrors of the selected SRDF devices.
`-bcv`	Lists only BCV devices.
`-both`	Lists all SRDF devices that are RDF1 or RDF2 capable, when used with -dynamic.
`-c`	Specifies the number (count) of times to repeat the operation, displaying results appropriate to the operation at each iteration.

Table 5 Options for symrdf list command (page 2 of 3)

symrdf list option	Description of output
`-concurrent`	Lists concurrent SRDF (RDF11, RDF22, and RDF21) devices and the SRDF devices paired with a concurrent SRDF device. When used with `-R1`, lists RDF11 devices and RDF1 devices that are paired with a concurrent SRDF device. When used with `-R2`, lists RDF22 devices and RDF2 devices that are paired with a concurrent device.
`-consistency`	Displays the SRDF consistency state when listing SRDF devices. To show the consistency state in the list of all the SRDF devices in array 333: `symrdf -sid 333 -consistency list`
`-cons_exempt`	Lists devices that are consistency exempt or are paired with devices that are consistency exempt.
`-dir`	Lists the local directors (separated by commas), such as, 1a, 1b, and so on.
`-diskless_rdf`	Lists diskless SRDF devices and the devices paired with diskless SRDF devices. When used with `-R1`, lists RDF1 devices that are either diskless or that are paired with a diskless device. When used with `-R2`, lists RDF2 devices that are either diskless or are paired with a diskless device. When used with `-R21`, lists RDF21 devices that are either diskless or that are paired with a diskless device.
`-dup_pair`	Lists SRDF devices that are paired with the same SRDF type. To list all of the duplicate pair devices in array 333: `symrdf -sid 333 -dup_pair list` **Note:** Duplicate pair devices can result from an SRDF/Star failover scenario or a configuration change.
`-dynamic`	Lists devices configured as dynamic SRDF. Use the qualifiers of `-R1`, `-R2`, or `BOTH` to restrict the display to the specified device type.
`-half_pair`	Lists devices whose partner is not an SRDF device. To list all of the half pair devices in array 333: `symrdf -sid 333 -halfpair list` **Note:** Half pair devices can result from an SRDF/Star failover scenario, a `half_deletepair` operation, or a configuration change.
`-nobcv`	Lists standard SRDF devices only (excludes SRDF BCV devices).

`-R1`
`-R2`
`-R21`
	Lists devices of RDF1 types (`-R1`), RDF2 types (`-R2`), or RDF21 types (`-R21`), respectively.

`-rdfa`	Lists devices that are SRDF/A-capable.
`-rdfa_not_pace_capable`	Lists devices participating in the SRDF/A session that are not pace-capable
`-rdfa_wpace_exempt`	Lists devices that are exempt from group-level write pacing

Table 5 Options for symrdf list command (page 3 of 3)

symrdf list option	Description of output
`-rdfg`	Lists all devices within a specified SRDF group.
`-resv`	Lists SRDF devices with SCSI reservations. To list all the SRDF devices in array 333 that have SCSI reservations: `symrdf -sid 333 -resv list`
`-star_mode`	Lists device that are SRDF/Star protected. For more information, refer to the *EMC VMAX3 Family Product Guide.*

ping command

Use the `symrdf -rdf ping` command to determine if a array using SRDF links is up and running.

To ping SID 123:

```
symrdf -rdf -sid 123 ping
```

The return codes tell you whether some or all of the arrays were successfully pinged.

For more information on return codes, refer to the *EMC Solutions Enabler CLI Command Reference*.

Using the verify command

Use the `symrdf verify` command to verify the SRDF mode and pair states of device groups, composite groups, and device files.

Use the `symrdf verify -enabled` command to verify that device pairs are enabled for consistency protection.

Verify SRDF mode

When verifying two or more SRDF modes using one command, Solutions Enabler logically ORs each mode to determine the result.

Example In the following example, a device group named STAGING contains devices in synchronous (-sync), and adaptive copy disk (-acp_disk) modes, but no devices in asynchronous (-async) mode.

- If the verify command specifies only asynchronous mode:

  ```
  symrdf -g STAGING -rdfg 129 verify -async
  ```

  None of the device pairs in STAGING are in asynchronous mode, and the following message is displayed:

  ```
  None of the devices in the group 'STAGING' are in 'Asynchronous'
  mode.
  ```

- If the verify command specifies asynchronous, synchronous mode, OR adaptive copy disk mode:

  ```
  symrdf -g STAGING -rdfg 129 verify -async -sync -acp_disk
  ```

  All device pairs in STAGING are using synchronous OR adaptive copy disk mode. The following message is displayed, even though NO devices are in asynchronous mode:

```
All devices in the group 'STAGING' are in 'Asynchronous, Synchronous,
Adaptive Copy Disk' modes.
```

Verify SRDF pair states

When verifying two or more SRDF pair states using one command, Solutions Enabler logically ORs each pair state to determine the result.

Example In the following example, a device group named STAGING contains devices in -split, -suspended, and -synchronized states, but no devices in -consistent state.

- If the verify command specifies only Consistent state:

  symrdf -g STAGING -rdfg 129 verify -consistent

  None of the device pairs in STAGING are in the Consistent state, and the following message is displayed:

  ```
  None of the devices in the group 'STAGING' are in 'Consistent' state.
  ```

- If the verify command specifies Consistent OR Split state:

  symrdf -g STAGING -rdfg 129 verify -consistent -split

  Some of the device pairs are in the Split state, none are in the Consistent state, and the message is:

  ```
  Not All devices in the group 'STAGING' are in 'Consistent, Split'
  states.
  ```

- If the verify command specifies Consistent, Split, Suspended, OR Synchronized states:

  symrdf -g STAGING -rdfg 129 verify -consistent -split -suspended -synchronized

  All device pairs in STAGING are in the Split, Suspended, OR Synchronized state. The following message is displayed, even though NO devices are in the Consistent state:

  ```
  All devices in the group 'STAGING' are in 'Consistent, Split,
  Suspended, Synchronized' states.
  ```

Verify both SRDF mode and pair state in one command line

When verifying both SRDF states and modes in the same command line, Solutions Enabler logically ORs the states, logically ORs the modes, and then logically ANDs the two results.

Example In the following example, a device group named STAGING has devices in:

- Synchronous, and adaptive copy disk modes
- Synchronized, suspended and split states, but NOT consistent state

- If the verify command specifies synchronous, OR adaptive copy disk mode, AND Synchronized, Suspended, OR Split states:

  symrdf -g STAGING -rdfg 129 verify -sync -acp_disk -synchronized -suspended -split

  All device pairs in STAGING are using synchronous OR adaptive copy disk mode AND are in the Synchronized, Suspended, OR Split state, and the following message is displayed:

```
All devices in the group 'STAGING' are in 'Synchronized, Suspended,
Split' states and 'Synchronous, Adaptive Copy Disk' modes.
```

- If the verify command specifies adaptive copy disk mode AND the Synchronized, Suspended, OR Split state:

  symrdf -g STAGING -rdfg 129 verify -acp_disk -synchronized -suspended -split

  Some device pairs in the STAGING group are using synchronous mode, and the following message is displayed:

  ```
  Not All devices in the group 'STAGING' are in 'Synchronized,
  Suspended, Split' states and 'Adaptive Copy Disk' modes.
  ```

- If the verify command specifies synchronous, adaptive copy disk mode AND the Consistent state:

  symrdf -g STAGING -rdfg 129 verify -sync -acp_disk -consistent

  None of the device pairs in the STAGING group are in the Consistent state, and the following message is displayed:

  ```
  None of the devices in the group 'STAGING' are in 'Consistent' state
  and 'Synchronous, Adaptive Copy Disk' modes.
  ```

Understanding SRDF pair states and links

IMPORTANT

Before you begin SRDF control operations, you must understand how SRDF devices and links work together to secure data within SRDF configurations.

Note: The following content assumes you understand SRDF devices, including R1, R11, R2, and R21. For a detailed description of SRDF devices, refer to *EMC VMAX3 Family Product Guide.*

An SRDF pair state encompasses:

- SRDF device state on the source (R1) device
- SRDF device state on the target (R2) device
- The number of tracks owed between the R1 and R2 devices (invalid tracks), and
- The SRDF link state between the R1 and R2 devices

Note: See "Invalid tracks in SRDF pairs" on page 39.

Figure 2 shows states SRDF devices and links can report.

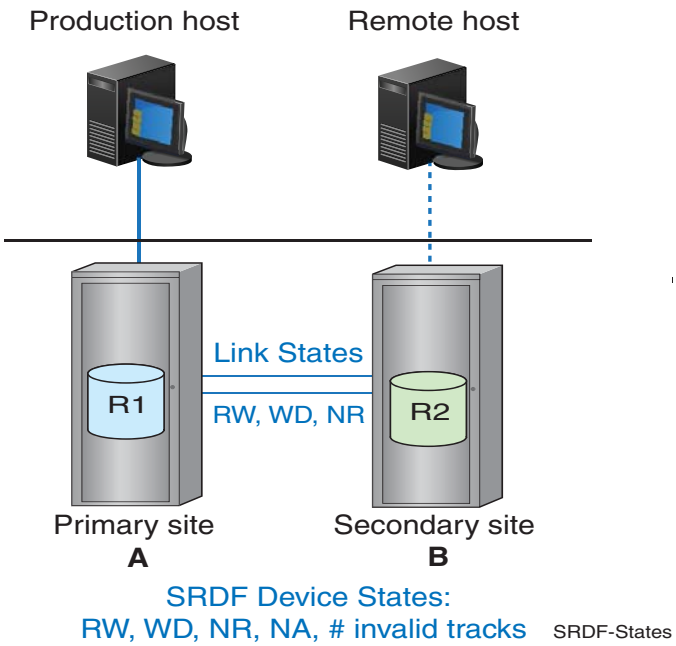


Figure 2 SRDF device and link states

NR	Not Ready. Reads and writes are both disabled.
RW	Ready. Enabled for both reads and writes.
WD	Write Disabled. Enabled for reads but not writes.
NA	Not Available. Unable to report on correct state.

About SRDF pair states

SRDF operation requires that the SRDF pair(s) on which an operation acts must be in a pair state that is valid for that operation.

"SRDF operations and pair states" on page 249 lists control actions and the prerequisite SRDF pair state for each action, including:

- "Concurrent SRDF operations and applicable pair states" on page 254
- "Concurrent SRDF operations and applicable pair states" on page 254
- "Consistency group operations and applicable pair states" on page 260

"Commands to display, query and verify SRDF configurations" on page 29 describes the SYMCLI commands to verify pair states.

Table 6 lists the name and description of SRDF pair states.

Table 6 SRDF pair states (page 1 of 2)

Pair State	Description
SyncInProg	A synchronization is currently in progress between the R1 and the R2 devices: • There are existing invalid tracks between the two pairs, and • The logical links between both sides of an SRDF pair are up.
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.
Split	The R1 and the R2 are currently ready to their hosts, but: • The links are not ready or, • Write disabled.
Failed Over	The R1 is currently not ready or write disabled. Operations have been failed over to the R2.
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 links are 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, • Data is being copied from the R2 to the R1 device, and the links are ready.
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.
Partitioned	The SYMAPI is currently unable to communicate through the corresponding SRDF path to the remote array. The Partitioned state may apply to devices within an RA group. For example, if SYMAPI is unable to communicate to a remote array from an RA group, devices in that RA group will be marked as being in the Partitioned state. A half pair and a duplicate pair are also reported as Partitioned.
Mixed	A composite SYMAPI device group SRDF pair state. There are different SRDF pair states within a device group.

Table 6 SRDF pair states (page 2 of 2)

Pair State	Description
Invalid	The default state when no other SRDF state applies. • The combination of the R1 device, the R2 device, and the SRDF link states do not match any other pair state. • This state may occur if there is a problem at the disk director level.
Consistent	The R2 SRDF/A capable devices are in a consistent state. Consistent state signifies the normal state of operation for device pairs operating in asynchronous mode.
Transmit Idle	The SRDF/A session cannot push data in the transmit cycle across the link because the link is down.

Invalid tracks in SRDF pairs

On both sides of an SRDF configuration, the array keeps an account of the tracks that are "owed" to the other side. *Invalid tracks* are tracks that are not synchronized between the two devices in an SRDF pair. *Remote invalids* are tracks owed the remote member of the device pair.

For example:

- The logical connection between an R1 device and its R2 is suspended.
- If both devices are made write-accessible, hosts on both sides of the SRDF links write to their respective devices, without the writes being mirrored.
- This creates invalid tracks on the R1 side, and remote invalid tracks on the R2 side.
- Each invalid track represents a track of data that has changed since the two sides were split. To re-establish the logical links between the R1 and R2, the invalid tracks must first be resolved.

How you resolve invalid tracks depends on which control operation you perform. For example if you have remote invalids on both the R1 and R2 sides:

- An `establish` operation copies the modified R1 tracks to the R2 side.

  Any tracks that were modified on the R2 side are overwritten with data from corresponding tracks on the R1 side.
- A `restore` operation copies the modified R2 tracks to the R1 side.

  Any tracks that were modified on the R1 side are overwritten with data from corresponding tracks on the R2 side.

SRDF device and link state combinations

Control actions on an SRDF pair may change the SRDF pair state.

Additionally, the state of a device can change if its front-end or back-end directors change in the SRDF links.

Table 7 shows:

- SRDF pair states resulting from the combination of the states of the source and target devices and the SRDF links.

- The possible R1 or R2 invalid tracks for each SRDF pair state.

Table 7 Possible SRDF device and link state combinations

SRDF pair state	Source (R1) SRDF state	SRDF link state	Target (R2) SRDF state	R1 or R2 invalid tracks
Synchronized	Ready (RW)	Ready (RW)	Not Ready or WD	0
Failed Over	Not Ready or WD	Not Ready	Ready (RW)	—
R1 Updated	Not Ready or WD	Ready (RW) or WD	Ready (RW)	0[1]
R1 UpdInProg	Not Ready or WD	Ready (RW) or WD	Ready (RW)	>0[1]
Split	Ready (RW)	Not Ready or WD	Ready (RW)	—
SyncInProg	Ready (RW)	Ready (RW)	Not Ready or WD	>0
Suspended	Any status[2]	Not Ready or WD	Not Ready or Write Disabled	—
Partitioned[3]	Any status	Not Ready	Not Available	—
Partitioned[4]	Not Available	Not Ready	Any status	—
Mixed	[5]	[5]	[5]	—
Invalid[5]	Any status[6]	Any status	Any status	—
Consistent	Ready (RW)[6]	Ready (RW)	Not Ready or WD	0
Transmit Idle	Ready (RW)[6]	Ready (RW)	Not Ready or WD	—

1. Refers to invalid local (R1) tracks on source.
2. Any status value is possible (Ready, Not Ready, Write Disabled, or Not Available).
3. Viewed from the host locally connected to the source (R1) device
4. Viewed from the host locally connected to the target (R2) device.
5. When no other SRDF states apply, the state defaults to Invalid.
6. The combination of source SRDF, SRDF links, and target SRDF statuses do not match any other SRDF state; therefore, the SRDF state is considered Invalid.

Before you begin

This section includes the following topics:

- "Array access rights"
- "Device external locks"
- "SRDF operations and copy sessions"
- "Migrating R1 to a larger R2 device"
- "Restricting synchronization"
- "Enabling SRDF software and hardware compression"
- "SRDF/A and the consistency exempt option"
- "Enabling mixed-mode workloads on an SRDF director"

- "FAST VP SRDF coordination"

Array access rights

Hosts must have specific access rights to a array to perform certain control operations. Table 8 lists common control operations and the required array access rights.

Table 8 Access rights required by an array

Operations	Required access rights
`symrdf set -rdf`	CFGSYM or SRDF
`symrdf set -rdfa`	CFGSYM or SRDF
`symrdf set rdfa_dse`	CFGSYM or SRDF
`symrdf set rdfa_pace`	CFGSYM or SRDF
`symrdf addgrp`	CFGSYM
`symrdf modifygrp`	CFGSYM
`symrdf removegrp`	CFGSYM
`symqos set IO`	CFGSYM
`symqos reset IO`	CFGSYM

Device external locks

SYMAPI and SYMCLI use device external locks to lock BCV pairs during TimeFinder control operations and to lock SRDF device pairs during SRDF control operations.

When a `symrdf` control command is initiated, device external locks are set on all SRDF devices. Device external locks are automatically released when the control operation completes.

"Managing locked devices" on page 228 describes how to acquire, recover, and release external locks.

SRDF operations and copy sessions

Certain SRDF operations are not allowed for arrays employing either TimeFinder/Snap or TimeFinder/Clone operations, which use copy session pairs. The availability of some SRDF actions depends on the current pair state of the TimeFinder/Snap or TimeFinder/Clone copy session devices.

"SRDF operations and TimeFinder sessions" on page 261 describes the TimeFinder/Snap and TimeFinder/Clone pair states, and which SRDF operations are available in each state.

Migrating R1 to a larger R2 device

You can copy data from an R1 device to a larger R2 device but the following restrictions apply:

- All swap and SRDF/Star operations are blocked.

- The `SYMAPI_RDF_CREATEPAIR_LARGER_R2` option in the options file must be set to `ENABLE`.

  If `SYMAPI_RDF_CREATEPAIR_LARGER_R2` is set to `DISABLE`, all `createpair` operations are blocked.
- Data mirrored to a larger R2 device cannot be restored back to its R1 device.

Note: For some types of file arrays and attached hosts, host-dependent operations may be required to access data migrated to a larger R2 device.

Restricting synchronization

The `SYMAPI_SYNC_DIRECTION` option in the options file allows you to confine SRDF and TimeFinder operations to only establish or restore actions.

HYPERMAX OS

`SYMAPI_SYNC_DIRECTION` option is not supported on VMAX 3 arrays running HYPERMAX OS.

Enginuity 5876 or less

For some sites running Enginuity 5876 or less, it may be desirable to block users on a host from performing either establish or restore operations.

To block a user on a host from executing a restore or an establish action, edit the following entry in the options file:

```
SYMAPI_SYNC_DIRECTION=ESTABLISH | RESTORE | BOTH
```

- `ESTABLISH` restricts operations to only establish actions.
- `RESTORE` restricts operations to restore, failback, and R1 update actions.
- `BOTH` (default) does not restrict any SRDF or TimeFinder actions.

Enabling SRDF software and hardware compression

Compression minimizes the amount of data transmitted over an SRDF link.

You can activate both software and hardware compression simultaneously for SRDF traffic over GigE and Fibre Channel.

Data is first compressed by software and then further compressed by hardware.

Hardware compression is available on Fibre Channel directors.

You can enable software and hardware compression on both the R1 and R2 sides, but the actual compression happens from the side initiating the I/O, make sure you enable compression on the R1 side.

symrdf set rdfg command syntax

To set hardware and software compression for an SRDF group:

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
   [-noprompt] [-i Interval] [-c Count]

.............

set rdfg
    [-hwcomp {on|off}]
    [-swcomp {on|off}]>
```

```
[-both_sides]
```

"Setting SRDF group attributes" on page 85 provides more information about SRDF group attributes.

Examples To turn on software compression on both sides of SRDF group 12:

```
symrdf -sid 134 -rdfg 12 set rdfg -swcomp on -both_sides
```

To turn off hardware compression on both sides of SRDF group 12:

```
symrdf -sid 134 -rdfg 12 set rdfg -hwcomp off -both_sides
```

To view SRDF software and hardware compression status for an SRDF group:

```
symcfg list -rdfg
```

To view software or hardware compression status for a specified group (12) and specified SID (432):

```
symcfg list -sid 432 -rdfg 12
```

SRDF/A and the consistency exempt option

By default, control operations for an active SRDF/A session are targeted at all device pairs in the session.

The `-cons_exempt` option marks devices targeted by the command as consistency exempt. Devices marked consistency exempt can be controlled independently of other devices in the active SRDF/A session.

Enginuity automatically clears the consistency exempt status when:

- The affected device pairs have become consistent, and
- When the data on the R1 gets applied to the R2.

Enabling mixed-mode workloads on an SRDF director

For arrays running Enginuity 5876.x, you can use the `symqos` command to set the percentage of the SRDF director (RA) CPU resources assigned to each workload type.

Workload percentages must add up to 100%, and can include:

- Synchronous I/Os
- Asynchronous I/Os
- Copy I/Os

Workload settings for the director are used until you explicitly reset them. After reset, the array-level distributions are used.

For information on the `symqos` command syntax, see the *EMC Solutions Enabler Array Management CLI User Guide*.

symqos command syntax

The syntax for the command is:

```
symqos -RA -sid SymmID
```

```
   enable -io
   disable -io

symqos -RA -sid SymmID

   set IO -default
       -sync SyncPercent -async AsyncPercent -copy CopyPercent

   set IO -dir <# | ALL>
       -sync SyncPercent -async AsyncPercent -copy CopyPercent

   reset IO -dir <# | ALL>

symqos -RA [-sid SymmID]

      list -io
```

Examples

To enable the workload percentage settings for synchronous, asynchronous, and copy I/Os on SID 1234:

symqos -RA -sid 1234 enable -io

To set the default settings of the workload percentages for all directors on SID 1234 to 60% for Synchronous I/Os, 30% for asynchronous I/Os and 10% for copy I/Os:

symqos -RA -sid 1234 set IO -default -sync 60 -async 30 -copy 10

To set the settings of the workload percentages on director 8G of SID 1234 to 50% for synchronous I/Os, 30% for asynchronous I/Os, and 20% for copy I/Os:

symqos -RA -sid 1234 -dir 8G set IO -sync 50 -async 30 -copy 20

To reset the customized settings of the workload percentages to the default settings on director 8G of SID 1234:

symqos -RA -sid 1234 -dir 8G reset IO

FAST VP SRDF coordination

If *both* arrays on an SRDF link are running Enginuity 5876 or later, you can enable SRDF coordination to instruct FAST VP to factor the R1 device statistics into move decisions on the R2 device.

For information on FAST and FAST VP, see the *EMC Solutions Enabler Array Management CLI User Guide*.

CHAPTER 2
Basic SRDF Control Operations

This chapter describes the following topics:

- Basic SRDF control operations summary 46
  Table 9 on page 46 lists the SRDF control operations described in this chapter, the arguments associated with the symrdf command used to perform the operation, and a summary of the operation.
- SRDF basic control operations 49
  The remainder of this chapter consists of detailed descriptions and command examples for the operations listed in Table 9.

Basic SRDF control operations summary

Table 9 SRDF control operations summary (page 1 of 3)

Control operation	symrdf argument	Description
Set SRDF mode of replication page 49	set mode [sync|asynch|acp_disk|acp_wp|acp|off]	Set the replication mode for a device, device group, composite group, or list of devices in a device file.
Enable or disable consistency protection page 52	enable disable	Enables or disables consistency protection for SRDF/A capable devices.
Establish an SRDF pair (full) page 52	establish -full	Establishes remote mirroring and initiates a full data copy from the source (R1) device to the target (R2) device. Use this for: • Initial synchronization of SRDF mirrors. • Replacement of a failed drive on the R2 side.
Establish an SRDF pair (incremental) page 54	establish	Establishes remote mirroring and initiates an incremental data copy from the source (R1) device to the target (R2) device. Use this to resynchronize after a split if you can discard the target data.
Failback page 56	failback	Switches data processing from the target side (R2) back to the source (R1) side. Use this to return the source site from the target site after resolving the cause of a failure.
Failover page 58	failover	Switches data processing from the source (R1) side to the target (R2) side. Use this when a failure occurs on the source side.
Invalidate R1 page 59	invalidate r1	Invalidates all tracks on the source (R1) side so that they can be copied over from the target (R2) side.
Invalidate R2 page 59	invalidate r2	Invalidates all tracks on the target (R2) side so that they can be copied over from the source (R1) side.
Make ready the R1 page 61	ready r1	Sets the source (R1) device to be SRDF ready to its local host.
Make ready the R2 page 62	ready r2	Sets the target (R2) device to be SRDF ready to its local host.
Make the R1 not ready page 61	not_ready r1	Sets the source (R1) device to be SRDF not ready to its local host.
Make the R2 not ready page 62	not_ready r2	Sets the target (R2) device to be SRDF not ready to its local host.
Merge the track tables of the R1 and R2 devices page 63	merge	Merges the track tables between the source (R1) and the target (R2) side.
Move one-half of an SRDF pair page 63	half_movepair	Moves one-half of the SRDF device pair to a different SRDF group. **Note:** If the RA ends up supporting more than 64K devices in the new SRDF group, this operation fails.

Table 9 SRDF control operations summary (page 2 of 3)

Control operation	symrdf argument	Description
Move SRDF device pairs page 64	movepair	Moves the SRDF device pair to a different SRDF group. **Note:** If the RA ends up supporting more than 64K devices in the new SRDF group, this operation fails.
Read/write disable target device page 64	rw_disable r2	Read/write disables the target (R2) device to its local host.
Refresh R1 page 65	refresh r1	Marks any changed tracks on the source (R1) side to be refreshed from the R2 side.
Refresh R2 page 65	refresh r2	Marks any changed tracks on the target (R2) side to be refreshed from the R1 side.
Restore from a target device page 66	restore -full	Resumes remote mirroring and initiates a full data copy from the target (R2) device to the source (R1) device. Use this for: • Initial (reverse) synchronization of SRDF mirrors. • Replacement of a failed drive on the R1 side.
Restore from a target device (incremental) page 68	restore	Resumes remote mirroring and initiates an incremental data copy from the target (R2) device to the source (R1) device. Use this for resynchronizing SRDF mirrors after a split if you can discard the source data.
Resume SRDF links page 69	resume	Resumes I/O traffic on the SRDF links for the remotely mirrored SRDF pairs in the group.
Split an SRDF pair page 70	split	Stops remote mirroring between the source (R1) device and the target (R2) device. The target device is made available for local host operations. Use this when both sides require independent access, such as for testing purposes.
Suspend SRDF links page 72	suspend	Suspends I/O traffic on the SRDF links for the remotely mirrored SRDF pairs in the group.
Swap SRDF pairs page 73	swap	Swaps the SRDF personality of the designated dynamic SRDF pair. Source R1 devices become target R2 devices and target R2 devices become source R1 devices.
Swap one-half of an SRDF pair page 72	half_swap	Swaps the SRDF personality of one half of the designated dynamic SRDF pair. Source R1 devices become target R2 devices or target R2 devices become source R1 devices.
Update R1 mirror page 74	update	Updates the source (R1) side with the changes from the target (R2) side while the target (R2) side is still operational to its local hosts. Use this to synchronize the R1 side with the R2 side as much as possible before performing a failback, while the R2 side is still online to the host.
Write disable source device page 76	write_disable r1	Write disables the source (R1) device to its local host.
Write disable target device page 76	write_disable r2	Write disables the target (R2) device to its local host.
Write enable source device page 76	rw_enable r1	Write enables the source (R1) device to its local host.

Table 9 SRDF control operations summary (page 3 of 3)

Control operation	symrdf argument	Description
Write enable target device page 76	rw_enable r2	Write enables the target (R2) device to its local host.

SRDF basic control operations

The remainder of this chapter describes the steps to perform typical SRDF operations.

Set mode

SRDF modes of operation determine the following:

- How R1 devices are remotely mirrored to R2 devices across the SRDF links
- How I/Os are processed in an SRDF solution
- When the production host's write I/O command is acknowledged.

This section describes the commands to set SRDF mode, including:

- "Setting the default mode"
- "Setting the mode"
- "Setting mode: synchronous"
- "Setting mode: adaptive copy"
  - "Setting mode: adaptive copy write pending"
  - "Setting mode: adaptive copy disk"
- "Setting mode: asynchronous"
- "Establish (full)"

Setting the default mode

The default mode of operation is adaptive copy disk. If you create device pairs without setting a mode, the devices are created in adaptive copy disk mode.

Use the SYMAPI_DEFAULT_RDF_MODE parameter in the options file to modify the default mode.

Setting the mode

Use the `createpair` command to set the SRDF replication mode when you create SRDF device pairs.

See "symrdf createpair command syntax" on page 90.

Use `symrdf set` command to set or modify the SRDF replication mode for a device group, a composite group, or for devices listed in a device file.

The syntax to set mode on a device group, composite group, and device file:

```
symrdf -g DgName set mode
symrdf -cg CgName set mode
symrdf -f[ile] FileName set mode -sid SymmID -rdfg GroupNumber
```

Setting mode: synchronous

Synchronous mode maintains a real-time image of data between the R1 and R2 devices.

Data must be successfully stored in cache at both the primary and the secondary site before an acknowledgment is sent to the production host at the primary site.

Synchronous mode ensures that the source (R1) and target (R2) devices contain identical data.

To set the replication mode in group `prod` to synchronous:

```
symrdf -g prod set mode sync
```

Setting mode: adaptive copy

Adaptive copy mode is designed to transfer large amounts of data without loss of performance.

Adaptive copy mode allows the R1 and R2 devices to be more than one I/O out of synchronization. Unlike the asynchronous mode, adaptive copy mode does not guarantee a dependent-write consistent copy of data on R2 devices.

The amount of data (number of tracks) out of synchronization between the R1 and the R2 devices at any given time is determined by the maximum skew value. "Setting adaptive copy disk skew" describes the steps to set the maximum skew value.

Adaptive copy modes revert to the specified mode of operation (synchronous mode or semi-synchronous mode) when certain conditions are met.

There are two types of adaptive copy mode:

- "Setting mode: adaptive copy write pending"
- "Setting mode: adaptive copy disk"

Setting mode: adaptive copy write pending

In adaptive copy write pending (acp_wp) mode, the array acknowledges all writes to the source (R1) device as if it is a local device.

The amount of data (number of tracks) out of synchronization between the R1 and the R2 devices at any given time is determined by the maximum skew value. You can set the maximum skew value using SRDF software.

New data accumulates in cache until it is successfully written to the source (R1) device and the remote director has transferred the write to the target (R2) device.

This SRDF mode is designed to have little or no impact on performance between the host and the array containing the source (R1) device.

HYPERMAX OS

Adaptive copy write pending mode is not supported if the R1 side of the SRDF device pair is on an array running HYPERMAX OS.

You cannot set the mode to adaptive copy write pending mode if the R1 side of an SRDF pair is on an array running HYPERMAX OS.

HYPERMAX OS/Enginuity 5876 backward compatibility

In SRDF configurations where R1 devices are on an array running HYPERMAX OS, connected to one or more arrays are running Enginuity 5876, the following restrictions apply:

- For swap and failover operations - If the R2 is on an array running HYPERMAX OS, and the mode of the R1 is adaptive copy write pending mode, SRDF sets the mode to adaptive copy disk.

- For migrate -replace R1 operations - If the R1 being replaced is on an array running HYPERMAX OS, and the mode of the R1 is adaptive copy write pending mode, SRDF sets the mode of the migrated pair to adaptive copy disk.

Examples To set the replication mode in group `prod to` adaptive copy write pending:

```
symrdf -g prod set mode acp_wp
```

To disable adaptive copy write pending and set the replication mode in group `prod` to synchronous:

```
symrdf -g prod set mode acp_off
```

Setting mode: adaptive copy disk

Adaptive copy disk (acp_disk) mode is designed to transfer large amounts of data without loss of performance.

Because the array cannot fully guard against data loss should a failure occur, EMC recommends:

- Use the adaptive copy disk mode to transfer the bulk of your data to target (R2) devices, and then,
- Switch to synchronous mode to ensure full data protection.

When you set the SRDF mode to adaptive copy disk, the array acknowledges all writes to source (R1) devices as if they were local devices. New data accumulates on the source (R1) device and is marked by the source (R1) side as *invalid tracks* until it is subsequently transferred to the target (R2) device. The remote director transfers each write to the target (R2) device whenever link paths become available.

Examples To set the replication mode in group `prod` to adaptive copy disk:

```
symrdf -g prod set mode acp_disk
```

To disable adaptive copy disk mode and set the replication mode in group `prod` to synchronous:

```
symrdf -g prod set mode acp_off
```

Setting adaptive copy disk skew

Skew is an attribute that defines the maximum number of invalid tracks supported by adaptive copy disk mode.

If the number of invalid tracks defined by the skew attribute is exceeded, the remotely-mirrored device switches to synchronous mode.

As soon as the number of invalid tracks drops below the skew threshold, the remotely-mirrored pair reverts back to adaptive copy mode.

Skew is configured at the device level and may be set to a value between 0 and 65,534 tracks. For devices with more than a 2 GB capacity drive, you can specify a value of 65,535 to indicate all tracks of any given drive.

Examples To:

- Change the adaptive copy skew value to the number of tracks on device `BCV023` of group `prod`, and

- Lock device `BCV023` into adaptive copy disk mode (since the number of invalid tracks cannot exceed the maximum threshold of 65, 535):

```
symrdf -g prod set acp_skew 65535 -bcv BCV023
```

To change the adaptive copy skew value to 30,000 tracks for device `BCV023` of group `prod`:

```
symrdf -g prod set acp_skew 30000 -bcv BCV023
```

Setting mode: asynchronous

In asynchronous mode (SRDF/A), data is transferred from the source (R1) site in predefined timed cycles or *delta sets* to ensure that data at the remote (R2) site is *dependent write consistent*.

The array acknowledges all writes to the source (R1) devices as if they were local devices. Host writes accumulate on the source (R1) side until the cycle time is reached and are then transferred to the target (R2) device in one delta set. Write operations to the target device are confirmed when the current SRDF/A cycle commits the data to disk by successfully de-staging it to the R2 storage devices.

Because the writes are transferred in cycles, any duplicate tracks written to can be eliminated through *ordered write processing*, which transfers only the changed tracks within any single cycle.

The point-in-time copy of the data at the secondary site is slightly behind that on the primary site.

SRDF/A has little or no impact on performance at the primary site as long as the SRDF links contain sufficient bandwidth and the secondary array is capable of accepting the data as quickly as it is being sent across the SRDF links.

When you set the mode as asynchronous for an SRDF group, all devices in the group must operate in that mode.

To set the replication mode in group `prod` to asynchronous:

```
symrdf -g prod set mode async
```

Note: A device status check is performed on all TimeFinder Snap and Clone device pairs in the group before the `set mode async` operation is allowed. Depending on the device pair state, asynchronous mode may not be allowed for devices employing either TimeFinder/Snap or TimeFinder/Clone operations. "SRDF operations and TimeFinder sessions" on page 261 explains the applicable device pair states for TimeFinder/Snap or TimeFinder/Clone operations.

"SRDF/Asynchronous operations" on page 114 provides additional information on operating in asynchronous mode.

Establish (full)

A full establish initiates the following activities for each specified SRDF pair in a device group, consistency group, or list of devices in a device file:

- The target (R2) device is write disabled to its local host I/O.

- Traffic is suspended on the SRDF links.
- All the tracks on the target (R2) device are marked invalid.
- All tracks on the R2 side are refreshed by the R1 source side. The track tables are merged between the R1 and R2 side.
- Traffic is resumed on the SRDF links.

When the establish control operation has successfully completed and the device pair is in the Synchronized state, the source (R1) device and the target (R2) device contain identical data.

A full establish on SRDF pairs is required only:

- At initial set up of SRDF pairs,
- When an R2 member of an SRDF pair is either fully invalid, or has been replaced.

Figure 3 illustrates establishing an SRDF pair.

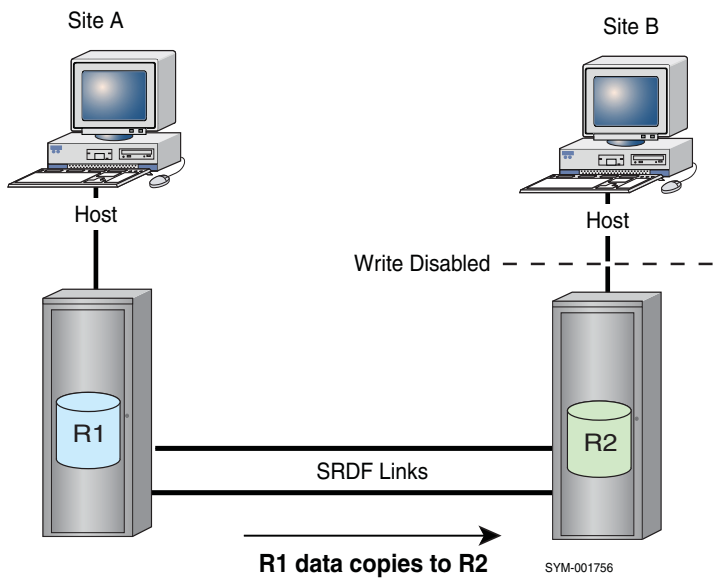


Figure 3 Full establish

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

"establish -full" on page 251 lists the applicable SRDF pair states for this operation.

symrdf establish -full command syntax

You can issue the `establish -full` operation on a device group, composite group, and device file:

```
symrdf -g DgName establish -full
symrdf -cg CgName establish -full
symrdf -f[ile] FileName establish -full
```

"Creating a device file" on page 88 describes the steps to create a device file.

Use `verify` to confirm that the establish is complete:

- SRDF pairs are in the Synchronized state, and
- Remote mirroring is resumed.

Examples To establish all the SRDF pairs in the device group `prod`:

```
symrdf -g prod establish -full
```

To establish one SRDF pair with logical device `DEV001` in the device group `prod`:

```
symrdf -g prod establish -full DEV001
```

To establish a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod establish -full DEV001 DEV002 DEV003
```

Note: The R2 may be set to read/write disabled (not ready) if `SYMAPI_RDF_RW_DISABLE_R2=ENABLE` is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

Establish (incremental)

An incremental establish re-synchronizes data on the source (R1) and target (R2) device when a split RDF pair is rejoined. Only the new data that was updated on the source (R1) device while the SRDF pair was split is copied, greatly reducing the amount of data that is copied.

An incremental establish initiates the following activities for each specified SRDF pair in a device group:

- The target (R2) device is write disabled to its local host I/O.
- Traffic is suspended on the SRDF links.
- The invalid tracks on the target (R2) device are refreshed from the changed tracks of the source (R1) device.
- The track tables are merged between the source (R1) device and the target (R2) device.
- Traffic is resumed on the SRDF links.

The SRDF pair is in the Synchronized state when the source (R1) device and the target (R2) device contain identical data.

Figure 4 illustrates the incremental establishing of an SRDF pair.

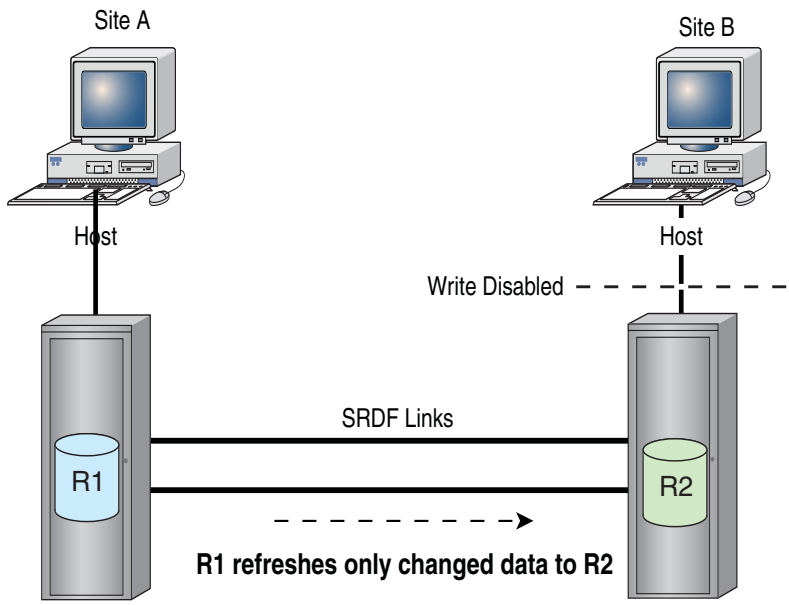


Figure 4 Incremental establish

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

"establish" on page 251 lists the applicable SRDF pair states for this operation.

symrdf establish command syntax

You can issue the `incremental -establish` operation on a device group, composite group, and device file:

```
symrdf -g DgName establish
symrdf -cg CgName establish
symrdf -f[ile] FileName establish
```

"Creating a device file" on page 88 describes the steps to create a device file.

Examples To initiate an incremental establish on all SRDF pairs in the `prod` device group:

```
symrdf -g prod establish
```

To initiate an incremental establish on one SRDF pair with logical device `DEV001` in the `prod` device group:

```
symrdf -g prod establish DEV001
```

To initiate an incremental establish for a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod establish DEV001 DEV002 DEV003
```

Note: R2 may be set to read/write disabled (not ready) if SYMAPI_RDF_RW_DISABLE_R2=ENABLE is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

Failback

After a failover (planned or unplanned), use the failback command to resume normal SRDF operations by initiating read/write operations on the source (R1) devices, and stop read/write operations on the target (R2) devices.

Failback initiates the following activities for each specified SRDF pair in a device group:

- The target (R2) device is write disabled to its local hosts.
- Traffic is suspended on the SRDF links.
- If the target side is operational, and there are invalid remote (R2) tracks on the source side (and the force option is specified), the invalid R1 source tracks are marked to refresh from the target side.
- The invalid tracks on the source (R1) side are refreshed from the target R2 side. The track tables are merged between the R1 and R2 sides.
- Traffic is resumed on the SRDF links.
- The source (R1) device is read/write enabled to its local hosts.

The target (R2) devices become read-only to their local hosts while the source (R1) devices are read/write enabled to their local hosts.

Failback includes the following general steps:

1. Stop I/Os on the failover host at site B.
2. Make all R2 devices in array at site B Not Ready or Read Only (Write Disabled) to the host.
3. If the array at site A was powered off, ensure that SRDF links between array A and array B are disabled before powering on the array at site A.
4. If the array at site A running Enginuity 5874 or earlier was powered off and you do not want to discard its changed data, disconnect or disable the SRDF links between array A and array B before powering on array A. In this way, you will prevent changed data from array B (secondary) from moving automatically to array A (primary).
5. Power on the array at site A and make R1 devices Read/Write enabled to the production host.
6. Enable the SRDF links between the array at site A and the array at site B.
7. Bring the SRDF links online and restart the local host. The R1 devices automatically receive data from the R2 devices which accumulated invalid tracks on their R2 SRDF mirrors during production processing.
8. Once all SRDF pairs are synchronized, enable consistency groups on the SRDF links between the array at site A and the array at site B.
9. Restart the site A host and applications.

Figure 5 shows the failback of an SRDF pair.

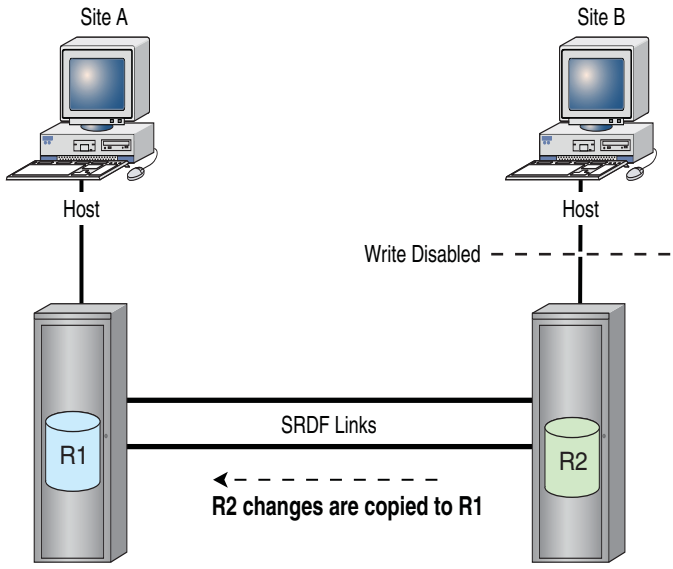


Figure 5 Failback of an SRDF device

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

"failback" on page 251 lists the applicable SRDF pair states for this operation.

symrdf failback command syntax

You can issue the `failback` control operation a device group, composite group, or device file:

```
symrdf -g DgName failback
symrdf -cg CgName failback
symrdf -f[ile] FileName failback
```

Note: The R2 may be set to read/write disabled (not ready) if `SYMAPI_RDF_RW_DISABLE_R2=ENABLE` is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

"Creating a device file" on page 88 describes the steps to create a device file.

Examples To initiate a failback on all the SRDF pairs in the `prod` device group:

```
symrdf -g prod failback
```

To initiate a failback on one SRDF pair, `DEV001`, in the `prod` device group:

```
symrdf -g prod failback DEV001
```

To initiate a failback on a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod failback DEV001 DEV002 DEV003
```

Failover

Failovers are used to move processing to the R2 devices during scheduled maintenance (planned failover) or when an outage makes the R1 devices unreachable (unplanned failover).

A failover transfers processing to the target (R2) devices and makes them read/write enabled to their local hosts.

Failover initiates the following activities for each specified SRDF pair in a device group:

- If the source (R1) device is operational, the SRDF links are suspended.
- If the source side is operational, the source (R1) device is write disabled to its local hosts.
- The target (R2) device is read/write enabled to its local hosts.

A *planned* failover is a controlled failover operation to test the robustness of the disaster restart solution, or to perform maintenance at the primary site. The secondary site temporarily becomes the primary/production site.

A planned failover includes the following general steps:

1. Shut down all applications on the production host.
2. Take all SRDF links between array A and array B offline to suspend remote mirroring.
3. When SRDF/CG is enabled, disable consistency groups between array A and array B.
4. Swap personalities between R1 and R2 devices.

   SRDF devices at array B are now R1 devices.

   SRDF devices at array A are now R2 devices.

   In SRDF/S configurations, devices are ready to resume production operations at array B.
5. When SRDF/CG is used, enable consistency between array B and array A.
6. Bring all SRDF links between array B and array A online to resume remote mirroring.
7. Start production applications from the host attached to array B.

An *unplanned* failover moves production applications from the primary site to the secondary site after an unanticipated outage at the primary site, and the primary site is not available.

A unplanned failover includes the following general steps:

1. Take all SRDF links between array A and array B offline to suspend remote mirroring.
2. Change the R2 device states to Read/Write to the secondary host connected to array B.
3. Start applications on the secondary host and resume production to write-enabled R2 devices in array B.

Figure 6 shows failover of an SRDF pair.

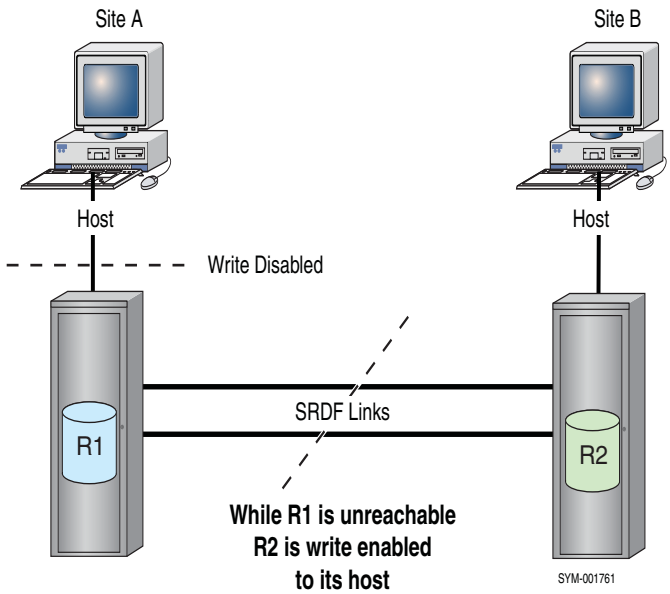


Figure 6 Failover of an SRDF device

"failover" on page 251 lists the applicable SRDF pair states for this operation.

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

---

symrdf failover command syntax

You can issue the `failover` operation on a device group, composite group, and device file:

```
symrdf -g DgName failover
symrdf -cg CgName failover
symrdf -f[ile] FileName failover
```

"Creating a device file" on page 88 describes the steps to create a device file.

Examples To perform a failover on all the pairs in the `prod` device group:

```
symrdf -g prod failover
```

To perform a failover on one SRDF pair with device DEV001 in the prod device group:

```
symrdf -g prod failover DEV001
```

To perform a failover on a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod failover DEV001 DEV002 DEV003
```

Invalidate R1 tracks

The `invalidate r1` action invalidates all tracks on the source (R1) side, so they can be copied over from the target (R2) side.

symrdf invalidate r1 command syntax

You can issue the `invalidate r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName invalidate r1
symrdf -cg CgName invalidate r1
symrdf -f[ile] FileName invalidate r1
```

Note: Use the `-nowd` option with the `invalidate r1` command to bypass the validation check to ensure that the target of operation is write disabled to the host.

"Creating a device file" on page 88 provides details about defining a device file.

Examples

To invalidate the source (R1) devices in all the SRDF pairs in device group `prod`:

```
symrdf -g prod invalidate r1
```

To invalidate the source (R1) device in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod invalidate r1 DEV007
```

To invalidate the source (R1) device for a list of SRDF pairs in device group `prod`:

```
symrdf -g prod invalidate r1 DEV002 DEV003 DEV007
```

Note: To invoke this operation, the SRDF pairs at the source must already be Suspended and write disabled (not ready).

Invalidate R2 tracks

The `invalidate r2` action invalidates all tracks on the target (R2) side so that they can be copied over from the source (R1) side.

symrdf invalidate r2 command syntax

You can issue the `invalidate r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName invalidate r2
symrdf -cg CgName invalidate r2
symrdf -f[ile] FileName invalidate r2
```

Note: Use the `-nowd` option with the `invalidate r1` or `invalidate r2` commands to bypass the validation check to ensure that the target of operation is write disabled to the host.

"Creating a device file" on page 88 provides details about defining a device file.

Examples

Note: To invoke this operation, the SRDF pairs at the source must already be Suspended and write disabled (not ready).

To invalidate the target (R2) devices in all the SRDF pairs in device group `prod`:

```
symrdf -g prod invalidate r2
```

To invalidate the target (R2) device in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod invalidate r2 DEV007
```

Make R1 ready

The ready state means the specified mirror is ready to the host. The mirror is enabled for both reads and writes.

`ready r1` sets the source (R1) devices to ready for their local hosts.

This operation is particularly helpful when all SRDF links are lost and the devices are operating in domino mode.

"ready R1" on page 251 lists the applicable SRDF pair states for this operation.

symrdf ready r1 command syntax

You can issue the `ready r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName ready r1
symrdf -cg CgName ready r1
symrdf -f[ile] FileName ready r1
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples To make the source (R1) device ready in all the pairs in device group `prod`:

```
symrdf -g prod ready r1
```

To make the source (R1) device ready in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod ready r1 DEV007
```

To make the source (R1) device ready in a list of SRDF pairs in device group `prod`:

```
symrdf -g prod ready r1 DEV002 DEV003 DEV007
```

Make R1 not ready

The not ready state means the specified mirror is not ready to the host. Both reads and writes are disabled.

`not_ready r1` sets the source (R1) devices to not ready for their local hosts.

"not_ready R1" on page 251 lists the applicable SRDF pair states for this operation.

symrdf not-ready r1 command syntax

You can perform the `not_ready r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName not_ready r1
symrdf -cg CgName not_ready r1
symrdf -f[ile] FileName not_ready r1
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples To make the source (R1) devices not ready in all the SRDF pairs in device group `prod`:

```
symrdf -g prod not_ready r1
```

To make the source (R1) device not ready in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod not_ready r1 DEV007
```

To make the source (R1) device not ready in a list of SRDF pairs, `DEV007`, in device group `prod`:

```
symrdf -g prod not_ready r1 DEV002 DEV003 DEV007
```

Make R2 ready

The ready state means the specified mirror is ready to the host. The mirror is enabled for both reads and writes.

`ready r2` sets the target (R2) devices to ready for their local hosts.

"ready R2" on page 252 lists the applicable SRDF pair states for this operation.

symrdf ready r2 command syntax

You can issue the `ready r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName ready r2
symrdf -cg CgName ready r2
symrdf -f[ile] FileName ready r2
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples

To make the target (R2) devices ready in all the SRDF pairs in device group `prod`:

```
symrdf -g prod ready r2
```

To make the target (R2) device ready in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod ready r2 DEV007
```

To make the source (R2) device ready in a list of SRDF pairs in device group `prod`:

```
symrdf -g prod ready r2 DEV002 DEV003 DEV007
```

Make R2 not ready

The not ready state means the specified mirror is not ready to the host. Both reads and writes are disabled.

`not_ready r2` sets the target (R2) devices to not ready for their local hosts.

"not_ready R2" on page 251 lists the applicable SRDF pair states for these operations.

symrdf not-ready r2 command syntax

You can issue the `not_ready r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName not_ready r2
symrdf -cg CgName not_ready r2
symrdf -f[ile] FileName not_ready r2
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples To make the target (R2) devices not ready in all pairs in device group `prod`:

`symrdf -g prod not_ready r2`

To make the target (R2) device in one pair not ready, `DEV007`, in device group `prod`:

`symrdf -g prod not_ready r2 DEV007`

To make the target (R2) device not ready in a list of pairs in device group `prod`:

`symrdf -g prod not_ready r2 DEV002 DEV003 DEV007`

Merge track tables

The `merge` operation merges the track tables between the source (R1) and the target (R2) devices.

Merge compares track tables on SRDF device pairs in a device group. Use the `merge` operation to compare the track tables between SRDF device pairs that have been split and re-established.

symrdf merge command syntax

"merge" on page 251 lists the applicable SRDF pair states for this operation.

You can issue the `merge` operation on a device group, composite group, and device file:

```
symrdf -g DgName merge
symrdf-cg CgName merge
symrdf -f[ile] FileName merge
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples To merge the track tables of all the SRDF pairs in device group `prod`:

`symrdf -g prod merge`

To merge the track table of one SRDF pair, `DEV007`, in device group `prod`:

`symrdf -g prod merge DEV007`

To merge the track table of a list SRDF pairs in device group `prod`:

`symrdf -g prod merge DEV002 DEV003 DEV007`

Move one-half of an SRDF pair

The `half_movepair` command moves one side of a dynamic SRDF pair from one SRDF group to another. The current invalid track counters on both R1 and R2 stay intact, so resynchronization is required.

Example To move one-half of the SRDF pairing of SRDF group 10 to a new SRDF group 15:

`symrdf half_movepair -sid 123 -file devicefile -rdfg 10 -new_rdfg 15`

This command moves the first device listed in each line of the device file to the new SRDF group.

"Moving dynamic SRDF device pairs" on page 98 also provides information on the `symrdf movepair` command for device files.

Move both sides of SRDF device pairs

The `movepair` command moves both the R1 and R2 sides of devices from one SRDF group to another. The current invalid track counters on both R1 and R2 stay intact, so resynchronization is required.

Note: You can only move pairs from like devices: from R1 to R1 or from R2 to R2.

The `movepair` operation has the following restrictions:

- A device cannot move when it is enabled for SRDF consistency.
- A device cannot move if it is in asynchronous mode when an SRDF/A cleanup or restore process is running.
- When moving one mirror of a concurrent R1 or an R21 device to a new SRDF group, the destination SRDF group must not be the same as the one supporting the other SRDF mirror.
- When issuing a full `movepair` operation, the destination SRDF group must connect the same two arrays as the original SRDF group.
- If the destination SRDF group is in asynchronous mode, the SRDF group type of the source and destination group must match. In other words, in asynchronous mode, devices can only be moved from R1 to R1, or from R2 to R2.
- If the destination SRDF group is supporting an active SRDF/A session, the `-cons_exempt` option must be specified.
- If the original SRDF group is supporting an active SRDF/A session, the device pairs being moved must have been suspended using the `-cons_exempt` option.

symrdf movepair command syntax

You can move SRDF pairs using a device file or device group:

```
symrdf movepair -sid SID -g DgName -rdfg RDFgroup -new_rdfg NewRDFgroup
symrdf movepair -sid SID -f FileName -rdfg RDFgroup -new_rdfg NewRDFgroup
```

Usage notes

- `-new_rdfg` *`NewRDFgroup`* is required.
- `-cons_exempt` allows devices to be moved into an active SRDF/A session without affecting the state of the session or requiring that other devices in the session be suspended.

Example To move pairs in a file from SRDF group 10 to SRDF group 15:

```
symrdf movepair -sid 123 -file devicefile -rdfg 10 -new_rdfg 15
```

This moves the first line of the device file to the new SRDF group, and moves the second device listed in each line of the remote SRDF group that is paired with the new SRDF group.

Read/write disable target device

The `rw_disable r2` action blocks reads from and writes to the target (R2) devices from their local host.

Use this command to set the specified device to the not ready state on the R2 side by making the device not ready on the RA.

"rw_disable R2" on page 252 lists the applicable SRDF pair states for this operation.

symrdf rw_disable r2 command syntax

You can issue the `rw_disable r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName rw_disable r2
symrdf -cg CgName rw_disable r2
symrdf -f[ile] FileName rw_disable r2 -rdfg2
```

"symrdf movepair command syntax" on page 99 provides details on the symrdf movepair command for device files.

Examples To read/write disable all the target (R2) mirrors in the SRDF pairs in a device group `prod`:

`symrdf -g prod rw_disable r2`

To read/write disable the target (R2) mirror in the SRDF pair, `DEV007`, in device group `prod`:

`symrdf -g prod rw_disable r2 DEV007`

To read/write disable the target (R2) mirror in a list of SRDF pairs in device group `prod`:

`symrdf -g prod rw_disable r2 DEV002 DEV003 DEV007`

Refresh R1

The refresh R1 mirror action marks any changed tracks on the source (R1) side to refresh from the R2 side.

Use the refresh R1 mirror action when the R2 device holds the valid copy and the R1 device's invalid tracks will be refreshed using the R2 data.

"refresh R1" on page 252 lists the applicable SRDF pair states for this operation.

symrdf refresh r1 command syntax

You can issue the `refresh r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName refresh r1
symrdf -cg CgName refresh r1
symrdf -f[ile] FileName refresh r1
```

Examples To refresh all the source (R1) devices in all the SRDF pairs in the device group `prod`:

`symrdf -g prod refresh r1`

To refresh the source (R1) device in the SRDF pair, `DEV007`, in the device group `prod`:

`symrdf -g prod refresh r1 DEV007`

To refresh the source (R1) device in the list of SRDF pairs in the device group `prod`:

`symrdf -g prod refresh r1 DEV002 DEV003 DEV007`

Refresh R2

The refresh R2 mirror action marks any changed tracks on the target (R2) side to refresh from the R1 side.

Use the refresh R2 mirror action when the R1 device holds the valid copy and the R2 device's invalid tracks will be refreshed using the R1 data.

"refresh R2" on page 252 lists the applicable SRDF pair states for this operation.

symrdf refresh r2 command syntax

You can issue the `refresh r1` and `refresh r2` operations on a device group, composite group, and device file:

```
symrdf -g DgName refresh r2
symrdf -cg CgName refresh r2
symrdf -f[ile] FileName refresh r2
```

Examples To refresh the target (R2) devices in all the SRDF pairs in device group `prod`:

```
symrdf -g prod refresh r2
```

To refresh the target (R2) device in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod refresh r2 DEV007
```

To refresh the target (R2) device for a list of SRDF pairs in device group `prod`:

```
symrdf -g prod refresh r2 DEV002 DEV003 DEV007
```

Restore (full)

Full restore copies the entire contents of the target (R2) device to the source (R1) device. After the restore operation completes, the pairs synchronize.

When a restore is initiated for each specified SRDF pair in a device group, the following occurs:

- The source (R1) device is write disabled to its local hosts.
- The target (R2) device is write disabled to its local hosts.
- Traffic is suspended on the SRDF links.
- All tracks on the source (R1) device are marked as invalid.
- All R1 tracks are refreshed from the R2 side. The track tables are merged between the R1 and R2 side.
- Traffic is resumed on the SRDF links.
- The source (R1) device is read/write enabled to its local hosts.

The restoration process is complete when the source (R1) and target (R2) device contain identical data. After the restore is complete, the SRDF pair is in the Synchronized state.

Note: R2 may be set to read/write disabled (not ready) if `SYMAPI_RDF_RW_DISABLE_R2=ENABLE` is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

Figure 7 illustrates restoring an SRDF pair.

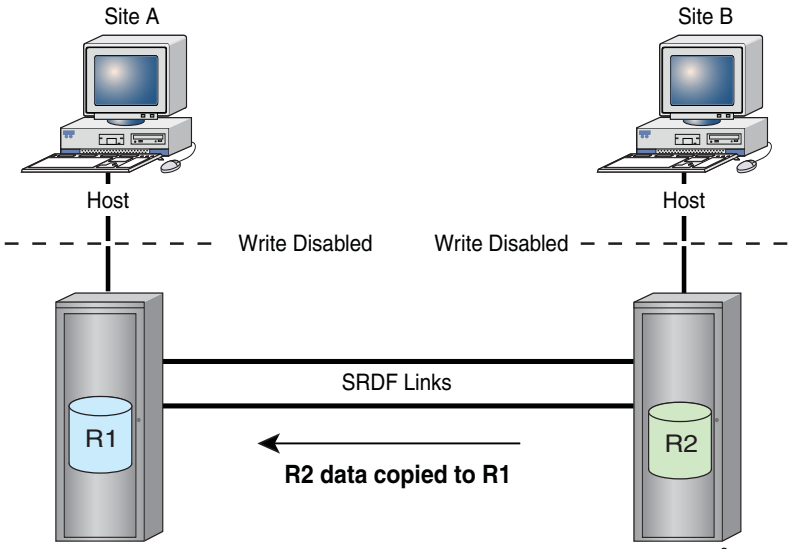


Figure 7 Restore (full) an SRDF device

"restore -full" on page 251 lists the applicable SRDF pair states for this operation.

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

---

symrdf restore -full command syntax

You can issue the `restore -full` operation on a device group, composite group, or device file:

```
symrdf -g DgName restore -full
symrdf -cg CgName restore -full
symrdf -f[ile] FileName restore -full
```

"Creating a device file" on page 88 provides details about defining a device file.

Use `verify` to confirm that the SRDF pairs are in the Synchronized state.

Examples

To initiate a full restore on all SRDF pairs in the `prod` device group:

```
symrdf -g prod restore -full
```

To initiate a full restore on one SRDF pair with logical device `DEV001` in the `prod` device group:

```
symrdf -g prod restore -full DEV001
```

To initiate a full restore on a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod restore -full DEV001 DEV002 DEV003
```

Restore (incremental)

An incremental restore re-synchronizes data from the target (R2) to the source (R1) device when a split RDF pair is rejoined. Only those tracks on the target (R2) device that changed while the SRDF pair was split are copied, greatly reducing the amount of data that is copied.

For example, when results from running a new application on the target (R2) device are desirable, and you want to move the changed data and the new application to the source (R1) device.

An incremental restore initiates the following activities for each specified SRDF pair in a device group:

- The source (R1) device is write disabled to its local hosts.
- The target (R2) device is write disabled to its local hosts.
- Traffic is suspended on the SRDF links.
- The invalid tracks on the source (R1) device are refreshed from the changed tracks on the target (R2) side. The track tables are merged between the R1 and R2 side.
- Traffic is resumed on the SRDF links.
- The source (R1) device is read/write enabled to its local hosts.

After the restore control operation has successfully completed, the SRDF pairs will synchronize.

Note: R2 may be set to read/write disabled (not ready) if `SYMAPI_RDF_RW_DISABLE_R2=ENABLE` is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix CLI Command Reference*.

Figure 8 illustrates the incremental restore of an SRDF pair.

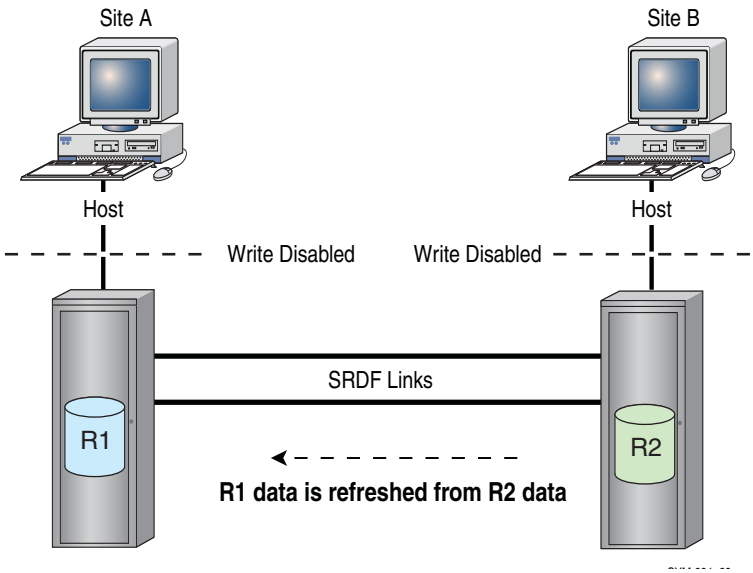


Figure 8 Incremental restore an SRDF device

"restore" on page 251 lists the applicable SRDF pair states for this operation.

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

symrdf restore command syntax

You can issue the incremental `restore` operation con a device group, composite group, or device file:

```
symrdf -g DgName restore
symrdf -cg CgName restore
symrdf -f[ile] FileName restore
```

"Creating a device file" on page 88 provides details about defining a device file.

Use `verify` to confirm that the SRDF pairs are in the Synchronized state.

Examples To initiate an incremental restore on all SRDF pairs in the `prod` device group:

```
symrdf -g prod restore
```

To initiate an incremental restore on one SRDF pair with logical device `DEV001` in the `prod` device group:

```
symrdf -g prod restore DEV001
```

To initiate an incremental restore for a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod restore DEV001 DEV002 DEV003
```

Resume I/O on links

The `resume` action argument resumes I/O traffic on the SRDF links for all remotely mirrored SRDF pairs in a group or device file.

"resume" on page 252 lists the applicable SRDF pair states for this operation.

symrdf resume command syntax

You can issue the `resume` control operation on device groups, composite groups, and device files:

```
symrdf -g DgName resume
symrdf -cg CgName resume
symrdf -f[ile] FileName resume
```

Note: The `resume` operation is rejected if you do not specify `-force` when the merge track table is required,

Examples To resume the SRDF links between all the SRDF pairs in device group `prod`:

```
symrdf -g prod resume
```

To resume the SRDF links between one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod resume DEV007
```

To resume the SRDF links (between the pairs) on a list of SRDF pairs in device group `prod`:

```
symrdf -g prod resume DEV002 DEV003 DEV007
```

Split

Split SRDF pairs when you require read and write access to the target (R2) side of one or more devices in a device group, composite group, or device file.

A split performs the following activities for each specified SRDF pair in a device group:

- Traffic is suspended on the SRDF links.
- The target (R2) device is read/write enabled to its local hosts.
- After the target (R2) device is split from the source (R1) device, the SRDF pair is in the Split state.

Figure 9 illustrates splitting an SRDF pair.

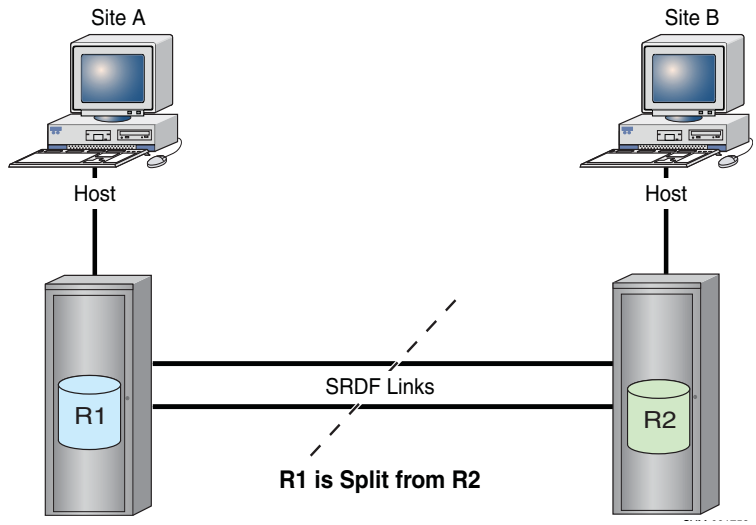


Figure 9 Split an SRDF pair

"split" on page 251 lists the applicable SRDF pair states for this operation.

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to establish. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

---

symrdf split command syntax

You can issue the `split` operation on a device group, composite group, and device file:

```
symrdf -g DgName split
symrdf -cg CgName split
symrdf -f[ile] FileName split
```

Note: The `-force` option is required when the device pairs are in domino mode or adaptive copy mode.

"Creating a device file" on page 88 provides details about defining a device file.

Examples To perform a split on all the SRDF pairs in the `prod` device group:

```
symrdf -g prod split
```

To perform a split on one SRDF pair with logical device `DEV001` in the `prod` group:

```
symrdf -g prod split DEV001
```

To initiate a split to a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod split DEV001 DEV002 DEV003
```

Splits that impact databases

If the `split` action impacts the access integrity of a database, additional actions such as freezing may be necessary. The freeze action suspends writing database updates to disk.

Use the `freeze` action in conjunction with the TimeFinder or `split` operation.

Use the `symioctl` command to invoke I/O control operations to freeze access to a specified relational database or database objects.

Note: For access to the specified database, set SYMCLI_RDB_CONNECT to your username and password.

Freeze access to a database

To freeze all I/O access to a specified relational database:

`symioctl freeze -type` *`DbType Object Object`*

SQL Server allows some or all databases to be specified. Oracle and Informix allow you to freeze or thaw an entire DB array.

If you have set the connection environment variables, the syntax is:

`symioctl freeze` *`Object Object`*

To freeze databases HR and Payroll:

```
symioctl freeze HR Payroll
```

Thaw access to a database

Once the freeze action is completed, the split may proceed.

When the split operation completes, use the `symioctl thaw` command to resume full I/O access to the database instance.

To resume I/O access:

```
symioctl thaw
```

Oracle databases: Hot backup control

For Oracle only, you can perform hot backup control on a list of tablespace objects. Hot backup control must be performed before and after a freeze/thaw command.

The steps required to split a group of SRDF pairs follows:

1. Issue the `symioctl begin backup` command.
2. Issue the `symioctl freeze` command.
3. Split the SRDF pairs. This may involve several steps depending on your environment.
4. Issue the `symioctl thaw` command.

5. Issue the `symioctl end backup` command.

Consistency groups

"SRDF consistency group operations" on page 145 describes consistency group split operations.

Suspend I/O on links

The `suspend` action suspends I/O traffic on the SRDF links for all remotely mirrored SRDF pairs in the group or device file.

When the suspend has completed successfully, the devices are suspended on the SRDF links and their link status is set to not ready (NR).

Note: The `suspend` operation is rejected if the specified device is in domino mode.

"suspend" on page 252 lists the applicable SRDF pair states for this operation.

symrdf suspend command syntax

You can issue the `suspend` operation on a device group, composite group, and device file:

```
symrdf -g DgName suspend [-immediate | -cons_exempt]
symrdf -cg CgName suspend [-immediate | -cons_exempt]
symrdf -f[ile] FileName suspend [-immediate | -cons_exempt]
```

Note: Use the consistency exempt (`-cons_exempt`) option to allow devices to be suspended without affecting the state of the SRDF/A session or requiring that other devices in the session be suspended.

Examples

To suspend the SRDF links between all the pairs in device group `prod`:

```
symrdf -g prod suspend
```

To suspend the SRDF links between one pair, `DEV007`, in device group `prod`:

```
symrdf -g prod suspend DEV007
```

To suspend the SRDF links (between the pairs) on a list of pairs in device group `prod`:

```
symrdf -g prod suspend DEV002 DEV003 DEV007
```

Suspend/resume timestamp

Suspend/resume causes SRDF link status to change from read/write to not ready and not ready to read/write. This status information is displayed in the output of the `symdev`, `sympd`, and `symdg show` commands.

Note: The timestamp in the displays is relative to the clock on the host where the command was issued and is reported for each SRDF mirror on both the R1 and R2 mirrors. This timestamp is not associated with the R2 data for SRDF/A.

Swap one-half of an SRDF pair

The `half_swap` command swaps the personality of one half of an SRDF relationship. It changes an R1 mirror to an R2 mirror *or* an R2 mirror to an R1 mirror.

You can swap one half of a designated SRDF pair as specified in a device file, device group, or composite group.

The `half_swap` operation has the following restrictions:

- The R2 device cannot be larger than the R1 device.
- A swap cannot be performed during an active SRDF/A session or when cleanup or restore is running.
- Adaptive copy write pending is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS. If the R2 side is on an array running HYPERMAX OS and the mode of the R1 is adaptive copy write pending, SRDF sets the mode to adaptive copy disk.

Example To swap the R1 designation of the associated BCV RDF1 pairs in device group `prod`, and refresh the data on the current R1 side:

```
symrdf -g Prod -bcv half_swap -refresh R1
```

"Half-swapping dynamic R1/R2 devices" on page 103 describes half-swapping dynamic pairs.

Swap SRDF pairs

The `swap` command swaps the personality of both halves in an SRDF relationship. The source (R1) device becomes the target (R2) device *and* the target (R2) device becomes the source (R1) device.

The `swap` operation has the following restrictions:

- A swap is not allowed if the R1 device (which becomes the R2) is currently a target for a TimeFinder/Snap or TimeFinder/Clone emulation. A device may not have two sources for data (in this case, the R1 and the emulation source). This is not allowed even if the emulation session has already completed copying the data.
- Adaptive copy write pending is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS. If the R2 side is on an array running HYPERMAX OS, and the mode of the R1 is adaptive copy write pending, SRDF sets the mode to adaptive copy disk.

IMPORTANT

The current states of the various devices involved in the SRDF swap must be considered before executing a swap action. Table 11 on page 101 lists which states are legal for this operation.

Example To swap the R1 designation of the associated BCV RDF1 pairs in device group `prod`, and refresh the data on the current R1 side:

```
symrdf -g Prod -bcv swap -refresh R1
```

"Swapping dynamic R1/R2 devices" on page 102 describes swap operations for device files, device groups, or composite groups.

Update R1 mirror

The `update` operation starts an update of the source (R1) side after a failover while the target (R2) side may still be operational to its local hosts.

Use `update` to perform an incremental data copy of only the changed tracks from the target (R2) device to the source (R1) device while the target (R2) device is still Write Enabled to its local host.

An update is initiated for each specified SRDF pair in a device group as follows:

- The SRDF (R1 to R2) links are suspended when the SRDF links are up.
- If there are invalid remote (R2) tracks on the source side and the force option was specified, tracks that were changed on the source devices are marked to refresh from the target side.
- The invalid tracks on the source (R1) side are refreshed from the target R2 side. The track tables are merged between the R1 and R2 sides.
- Traffic is resumed on the SRDF links.

IMPORTANT

If you perform an update while the SRDF pair is Suspended and not ready at the source, the SRDF pair types are in an Invalid state as the update completes. To resolve this condition, use the `rw_enable r1` command to make the SRDF pairs become Synchronized.

When the update has completed successfully, the pairs are in the R1 Updated state.

Figure 10 illustrates the update of an SRDF pair.

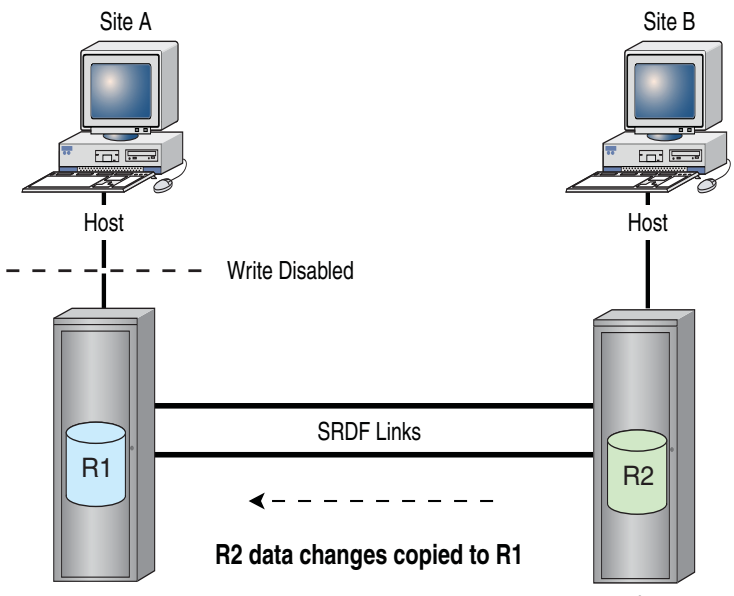


Figure 10 Update SRDF device track tables

"update" on page 251 lists the applicable SRDF pair states for this operation.

Note: When the `symrdf` command is initiated, device external locks are set on all SRDF devices you are about to control. See "Device external locks" on page 41 and "Display devices with a device external lock." on page 30.

symrdf update command syntax

You can issue the `update` operation on a device group, composite group, and device file:

```
symrdf -g DgName update
symrdf -cg CgName update
symrdf -f[ile] FileName update
```

"Creating a device file" on page 88 provides details about defining a device file.

Examples

To initiate an update of all the source (R1) devices in the SRDF pairs, for device group `prod`:

```
symrdf -g prod update
```

To initiate an update of the source (R1) device in the SRDF pair with logical device `DEV001` in device group `prod`:

```
symrdf -g prod update DEV001
```

To initiate an update on a list of SRDF pairs in the device group `prod`:

```
symrdf -g prod update DEV001 DEV002 DEV003
```

Continuous R1 updates

Use the `update -until #` command for scenarios where you want I/O to continue from the remote host and periodically update an inactive R1 device over an extended period of time.

The `-until` option checks the number of invalid tracks that are allowed to build up from the active R2 local I/O before another update (R2 to R1 copy) is triggered. The update sequence loops until the invalid track count is less than the number specified by the `#` value.

Note that these update sequences start with an immediate update once this command is started as follows:

1. Update R1 mirror.
2. Changed tracks are built on R2.
3. Check invalid track count.

Note: If the invalid track count is less than the number of tracks specified by the `-until #` value, the command exits, otherwise, the above sequence of operations for update R1 mirror is retriggered until the threshold is reached.

To update the R1 mirror of device group `prod` continuously until track the number of tracks to be copied is below 1000:

```
symrdf -g prod update -until 1000
```

Write disable R1

The write disable R1 action write disables the source (R1) devices for their local hosts.

"write_disable R1" on page 252 lists the applicable SRDF pair states for these operations.

symrdf write_disable r1 command syntax

You can issue the `write_ disable r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName write_disable r1
symrdf -cg CgName write_disable r1
symrdf -f[ile] FileName write_disable r1
```

Examples To write disable all the source (R1) mirrors in the SRDF pairs in device group `prod`:

```
symrdf -g prod write_disable r1
```

To write disable the source (R1) mirror in the SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod write_disable r1 DEV007
```

To write disable the source (R1) mirror in a list of SRDF pairs, (`DEV002`, `DEV003`, `DEV007`) in device group `prod`:

```
symrdf -g prod write_disable r1 DEV002 DEV003 DEV007
```

Write disable R2

The write disable R2 action write disables the source (R2) devices for their local hosts.

"write_disable R2" on page 252 lists the applicable SRDF pair states for these operations.

symrdf write_disable r2 command syntax

You can issue the `write_ disable r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName write_disable r2
symrdf -cg CgName write_disable r2
symrdf -f[ile] FileName write_disable r2
```

Examples To write disable all the target (R2) mirrors in the SRDF pairs in device group prod:

```
symrdf -g prod write_disable r2
```

To write disable the target (R2) mirror in the SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod write_disable r2 DEV007
```

To write disable the target (R2) mirror in a list of SRDF pairs in device group `prod`:

```
symrdf -g prod write_disable r2 DEV002 DEV003 DEV007
```

Write enable R1

The read/write enable R1 action enables the source (R1) devices for their local hosts.

"rw_enable R1" on page 252 lists the applicable SRDF pair states for these operations.

You can issue the `rw_enable r1` operation on a device group, composite group, and device file:

```
symrdf -g DgName rw_enable r1
symrdf -cg CgName rw_enable r1
symrdf -f[ile] FileName rw_enable r1
```

Examples To read/write enable all the source (R1) mirrors in all the SRDF pairs in device group `prod`:

```
symrdf -g prod rw_enable r1
```

To read/write enable the source (R1) mirrors in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod rw_enable r1 DEV007
```

Write enable R2

The read/write enable R2 action write enables the target (R2) devices for their local hosts.

"rw_enable R2" on page 252 lists the applicable SRDF pair states for these operations.

You can issue the `rw_enable r2` operation on a device group, composite group, and device file:

```
symrdf -g DgName rw_enable r2
symrdf -cg CgName rw_enable r2
symrdf -f[ile] FileName rw_enable r2
```

Examples To read/write enable all the target (R2) mirrors in the SRDF pairs in device group `prod`:

```
symrdf -g prod rw_enable r2
```

To read/write enable the target (R2) mirror in one SRDF pair, `DEV007`, in device group `prod`:

```
symrdf -g prod rw_enable r2 DEV007
```

To read/write enable the target (R2) mirror of a list of SRDF pairs in device group `prod`:

```
symrdf -g prod rw_enable r2 DEV002 DEV003 DEV007
```

CHAPTER 3
Dynamic Operations

This chapter describes the following topics:

- Overview 80
- SRDF dynamic group attributes 81
- Adding dynamic SRDF groups 82
- Removing dynamic SRDF groups 87
- Creating dynamic SRDF device pairs 89
- Deleting dynamic SRDF device pairs 95
- Moving dynamic SRDF device pairs 98
- Swapping SRDF devices 100
- Dynamic failover operations 103

Overview

An SRDF group is comprised of SRDF devices and SRDF directors that reside on a given array. The SRDF mirrors that belong to these SRDF devices point to the SRDF partner devices that reside on another array and are configured to the partner SRDF group.

SRDF groups communicate with their partner SRDF groups in another array across the SRDF links. SRDF group configuration parameters include the partner array identification and the set of SRDF directors that belong to the partner SRDF group.

You must create SRDF groups on both ends of the SRDF links.

SRDF groups can be created, modified, and deleted on demand while the array is in operation.

As soon as an empty SRDF group is created on one array, a partner SRDF group must be created on the other array. The SRDF directors assigned to each group share CPU processing power, SRDF ports, and serve all SRDF devices that are added to an SRDF group associated with that director. SRDF directors on each side of the SRDF links cooperate to support regular SRDF I/O operations.

The maximum number of SRDF groups and SRDF groups per SRDF director varies by operating system version:

- Enginuity 5773:
  - 128 SRDF groups,
  - 32 SRDF groups per SRDF director
- Enginuity 5874 and higher:
  - 250 SRDF groups
  - 64 SRDF groups per SRDF director
- HYPERMAX OS
  - 250 SRDF groups per SRDF director

HYPERMAX OS

All SRDF devices and SRDF groups on arrays running HYPERMAX OS are created as dynamic.

For configurations where one array is running HYPERMAX OS, and the second array is running Enginuity 5876, SRDF groups on the 5876 array must be created as dynamic. You cannot pair static SRDF groups or devices on one array with dynamic SRDF groups or devices on a second array.

HYPERMAX OS supports multiple ports per director.

When both arrays connected by an SRDF group are running HYPERMAX OS:

- Up to 250 SRDF groups can be defined across all of the ports per SRDF director or
- Up to 250 SRDF groups can be defined on 1 port on a specific RDF director.

When one array is running HYPERMAX OS and the second array is running Enginuity 5876:

- The port on the array running HYPERMAX OS which is connected to a port on an array running Enginuity 5876 can support up to 64 SRDF groups.

Thus, the maximum number of SRDF groups supported on the HYPERMAX OS director is effectively 186 (250-64).

SRDF dynamic group attributes

All SRDF groups have configurable attributes that are applied to the devices in the group, including:

- "Link limbo"
- "Domino mode"
- "Autolink recovery"
- "Hardware compression"
- "Software compression"

Note: SRDF/A device groups have additional configurable attributes. See "Setting SRDF/A group cycle time, priority, and transmit idle" on page 116.

Link limbo

Note: This option is for advanced users only.

Link limbo is an advanced user feature that allows you to set a specific length of time for Enginuity to wait when a down link is detected before updating the link status.

You can specify a link limbo value on the local side or both the local and remote sides of a dynamic SRDF group. If the link status is still not ready after the link limbo time expires, devices are marked not ready to the link.

Valid values for the link limbo timer are 0 through 120 seconds.

The default is 10 seconds.

To protect from session drops after the maximum link limbo time, enable the Transmit Idle feature. "Managing transmit idle" on page 131 provides more information.

IMPORTANT

Because the setting of the link limbo timer affects the application timeout period, it is not recommended to set while running in synchronous mode.
Switching to SRDF/S mode with the link limbo parameter configured for more than 10 seconds could result in an application, database, or host failure if SRDF is restarted in synchronous or semi-synchronous mode.

Domino mode

Under certain conditions, the SRDF devices can be forced into the Not Ready state to the host. For example, if the host I/Os cannot be delivered across the SRDF link.

Use the domino attribute to stop all subsequent write operations to both R1 and R2 devices to avoid data corruption.

While such a shutdown temporarily halts production processing, domino modes can protect data integrity in case of a rolling disaster.

Autolink recovery

If all SRDF links fail, the array remembers the SRDF states of the affected SRDF devices and can automatically restore the devices to these states once the SRDF links become operational.

Enable the Autolink recovery attribute (`-autolink_recovery`) to allow SRDF to automatically restore the SRDF links.

Valid values for `-autolink_recovery` are on (enabled) and off (disabled).

The default is off.

Hardware compression

SRDF hardware compression (`-hwcomp`)is available over Fibre Channel and GigE links. Compression minimizes the amount of data transmitted over an SRDF link.

Valid values for `-hwcomp` are on (enabled) and off (disabled).

The default is off.

Software compression

Software compression (`-swcomp`) can be applied to SRDF traffic over Fibre Channel and GigE SRDF links. If software compression is enabled, Enginuity compresses data before sending it across the SRDF links.

Arrays at both sides of the SRDF links must support software compression and must have the software compression feature enabled in the configuration file.

Valid values for `-swcomp` are on (enabled) and off (disabled).

The default is off.

Managing dynamic SRDF groups

This section includes procedures to create, manage, and delete SRDF groups. Topics include:

- "Adding dynamic SRDF groups"
- "Setting SRDF group attributes"
- "Adding/removing supporting directors for an SRDF group"
- "Removing dynamic SRDF groups"

Adding dynamic SRDF groups

The following are requirements for adding a dynamic SRDF group:

- The `dynamic_rdf` parameter must be enabled.
- The local or remote array must not be in the symavoid file.

  If either array is in the symavoid file, an error message is generated.
- You can perform multiple operations (`addgrp`, `modifygrp`, `removegrp`), but each operation must complete before the next can be started.

- A group label must be specified when adding a dynamic group.

Create a dynamic SRDF group and add pairs

Use the `symrdf addgrp` command to create a new SRDF group.

symrdf addgrp command syntax

```
symrdf addgrp -sid SID -label GrpLabel -rdfg GrpNum [-noprompt] [-i
   Interval] [-c Count]

   ...........

   -dir Dir:Port,Dir:Port,…
   -remote_rdfg GrpNum
   -remote_sid SID
   -remote_dir Dir:Port,Dir:Port,…
   -fibre | -gige | -farpoint
   -link_domino {on|off}
   -remote_link_domino
   -auto_link_recovery {on|off}
   -remote_auto_link_recovery
   -link_limbo Secs
   -rem_link_limbo Secs
```

Where:

- `-dir Dir:Port, Dir:Port` specifies a comma-separated list one or more ports on a local director to be added to the group.
- `-remote_dir Dir:Port, Dir:Port` specifies a comma-separated list one or more ports on a remote director to be added to the group.

HYPERMAX OS

Arrays running HYPERMAX OS support multiple ports per director. You must specify both the director ID and the port number when specifying the local and remote ports to add to the new SRDF group. To specify 3 ports on each array:

```
symrdf addgrp -label new_group -rdfg 39 -remote_rdfg 49
-dir 2f:11,1f:12,2h:3 -remote_dir 1h:2,2e:3,2f:12 -sid 000197100001
-remote_sid 000197100228 -nop
```

Enginuity 5773 - 5876

Arrays running Enginuity 5773 - 5876 support a single port per director. Specify only the director ID when specifying the local and remote ports to add to the new SRDF group. For example:

```
symrdf addgrp -label new_group -rdfg 39 -remote_rdfg 49
-dir 2f -remote_dir 1h -sid 000195700001
-remote_sid 000195700228 -nop
```

Mixed configurations

When one array in an SRDF configuration is running HYPERMAX OS, and one array is running Enginuity 5876, specify only the director ID on the array running 5876, and specify both the director ID and port number on the array running HYPERMAX OS. For example:

```
symrdf addgrp -label new_group -rdfg 39 -remote_rdfg 49
-dir 3h:12 -remote_dir 5f -sid 000197100001
-remote_sid 000195700228 -nop
```

- `-remote_rdfg GrpNum` specifies the SRDF group number on the remote array.
- `-remote_sid SID` specifies the ID of the remote array.
- `-fibre | -gige | -farpoint` species the communication protocol for the group: Fibre Channel, Gigabit Ethernet, or FarPoint.

- `-link_domino {on|off}` enables/disables "Establish (full)".
- `-remote_link_domino {on|off}` enables "Establish (full)" remotely.
- `-auto_link_recovery {on|off}` enables/disables local "Autolink recovery".
- `-remote_auto_link_recovery` enables "Autolink recovery" remotely.
- `-link_limbo` *`0 - 120`* enables "Link limbo".
- `-rem_link_limbo` *`0 - 120`* enables "Link limbo" remotely.

Procedure To create a dynamic SRDF group, do the following:

1. Use the `symcfg list` command to display the arrays visible to the host.
2. Use the `symsan list -sanrdf` command to display the SRDF topology from the local array, including available director pairs on the two arrays.

   For example, to determine which remote directors are visible from array 6180:

   `symsan -sanrdf -sid 6180 -dir all list`

   In this example, the output shows that director 13a on array 6240 is visible from director 12a on array 6180

```
Symmetrix ID: 00019490 6180

    Flags                Remote
--- ------- ---------------------------------
    Dir Lnk
Dir CT  S   Symmetrix ID Dir WWN
--- --- --- ------------ --- ----------------
12A SO  C   00019260 6240 13A C465090872090050
14A SO  C   000192602586 15A C465090872016879


Legend:
  Director:
    (C)onfig : S = Fibre-Switched, H = Fibre-Hub
               G = GIGE, - = N/A
    S(T)atus : O = Online, F = Offline, D = Dead, - = N/A

  Link:
    (S)tatus : C = Connected, P = ConnectInProg
               D = Disconnected, I = Incomplete, - = N/A
```

3. Use the `symrdf addgrp` command to create an empty dynamic SRDF group.

   In the following example, the `symrdf addgrp` command:

   - Creates a new dynamic SRDF group, specifying the local array (`-sid 6180)` and remote array (`-remote_sid 6240`).
   - Assigns an SRDF group number for the local array (`-rdfg 4`), and for the remote array (`-remote_rdfg 4`) to the new group.
   - Assigns a group label (`-label dyngrp4`) to the new group.

     This label can be up to 10 characters long, and provides a user-friendly ID to modify or delete the new group.

     The group label is required to add/remove directors from the SRDF group. See "Adding/removing supporting directors for an SRDF group" on page 86.

- Adds directors on the local array (`-dir 12a`) and the remote array (`-remote_dir 13a`) to the new group:

```
symrdf addgrp -sid 6180 -rdfg 4 -label dyngrp4 -dir 12a
-remote_rdfg 4 -remote_sid 6240 -remote_dir 13a
```

IMPORTANT

Network topology is important when choosing director endpoints. If using Fibre Channel protocol, the director endpoints chosen must be able to see each other through the Fibre Channel fabric in order to create the dynamic SRDF links. Ensure that the physical connections between the local RA and remote RA are valid and operational.

4. Use the `symcfg -sid SID list -rdfg GroupNumber` command to confirm that the group was added to both arrays.
5. Use the `symrdf createpair` command to add SRDF pairs to the new group.

Note: When creating an RDF pair between HYPERMAX OS and Enginuity 5876, the maximum symdev number that can be used on the array running HYPERMAX OS is FFBF (65471).

In the following example, the `symrdf createpair` command:

- Adds the dynamic SRDF pairs listed in the device file (`-file dynpairsfile`) to the new dynamic SRDF group 4 (`-rdfg 4`)
- Specifies the local array (`-sid 6180`) as the R1 side for the group (`-type R1`)
- The -invalidate option (`-invalidate R2`) indicates that the R2 devices are the targets that will be refreshed from the R1 source devices.

```
symrdf createpair -sid 6180 -rdfg 4 -file dynpairsfile -type R1
-invalidate R2
```

"Dynamic device pair operations" on page 88 provides more information.

Use the `symcfg list -ra all -switched` command to display all SRDF groups on the local array and its remotely connected arrays.

Setting SRDF group attributes

Use the `symrdf set rdfg` command to set the attributes for an SRDF group.

Note: The remote side must be reachable in order to set the SRDF group attributes.

symrdf set rdfg command syntax

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
   [-noprompt] [-i Interval] [-c Count]

.............

set rdfg
    [-limbo {0 - 120}]
    [-domino {on|off}]
    [-autolink_recovery {on|off}]
```

```
[-hwcomp {on|off}]
[-swcomp {on|off}]
[-both_sides]
```

Where:

- `-both_sides` applies the group attribute to both the source and target sides of an SRDF session. If this option is not specified, attributes are only applied to the source side.
- `-limbo {0 - 120}` enables/disables "Link limbo".
- `-domino {on|off}` enables/disables "Domino mode"
- `-autolink_recovery {on|off}` enables/disables "Autolink recovery"
- `-hwcomp {on|off}]` enables/disables "Hardware compression"
- `-swcomp {on|off}` enables/disables "Software compression"

Note: For arrays running Enginuity 5876 or earlier, you can also use the `symconfigure` command to set SRDF group attributes. For more information, see the *EMC Solutions Enabler Symmetrix Array Controls CLI Product Guide.*

Examples

To set the link limbo value to one minute (60 seconds) for both sides of SRDF group 4 on array 6180:

```
symrdf -sid 6180 -rdfg 4 set rdfg -limbo 60 -both_sides
```

To set the Link Domino mode on both sides of group 4 on array 6180:

```
symrdf -sid 6180 -rdfg 4 set rdfg -domino on -both_sides
```

To set the Autolink Recovery mode on both sides of group 4 on array 6180:

```
symrdf -sid 6180 -rdfg 4 set rdfg -autolink_recovery on -both_sides
```

To set limbo to thirty seconds and turn off Link Domino and Autolink Recovery modes for SRDF group 12:

```
symrdf -sid 134 -rdfg 12 set rdfg -limbo 30 -domino off -autolink_recovery off
```

To turn on software compression and turn off hardware compression on both sides of the SRDF group 12:

```
symrdf -sid 134 -rdfg 12 set rdfg -swcomp on -hwc off -both_sides
```

Adding/removing supporting directors for an SRDF group

Use the `symrdf modifygrp` command to modify a dynamic SRDF group.

symrdf modifygrp command syntax

```
symrdf modifygrp {-add | -remove} -rdfg GrpNum|-label GrpLabel-sid SID
   -i Interval -c Count -star
   .........
   -dir Dir:Port,Dir:Port,…
   -remote_dir Dir:Port,Dir:Port,…
```

Where:

- `-dir Dir:Port, Dir:Port` specifies a comma-separated list of one or more ports on a local director to be added to the group.

- `-remote_dir Dir:Port, Dir:Port` specifies a comma-separated list of one or more ports on a remote director to be added to the group.

HYPERMAX OS

Arrays running HYPERMAX OS support multiple ports per director. You must specify both the director ID and the port number when modifying the local and remote ports. To add port 12 on local director 3h to SRDF group 38:

```
symrdf modifygrp -add -rdfg 38 -dir 3h:12 -sid 000197100001 -nop
```

Enginuity 5773 - 5876

Arrays running Enginuity 5773 - 5876 support a single port per director. Specify only the director ID when specifying the ports to add/remove to/from the SRDF group. For example:

```
symrdf modifygrp -add -rdfg 38 -dir 3h -sid 000195700001 -nop
```

Mixed configurations

When one array in an SRDF configuration is running HYPERMAX OS, and one array is running Enginuity 5876, specify only the director ID on the array running 5876, and specify both the director ID and port number on the array running HYPERMAX OS. For example:

```
symrdf modifygrp -add -rdfg 38 -dir 3h:12 -remote_dir 5f -sid
000197100001 -remote_sid 000195700228 -nop
```

When adding a director to a dynamic group, the specified director for the local array must be online and a physical link to one online director in the remote array must exist.

IMPORTANT

Making physical cable changes within the SRDF environment may disable the ability to modify and delete dynamic group configurations.

Note: Reassigning directors for SRDF dynamic groups requires that you understand the network fabric topology when choosing director endpoints.

The group label or group number is required for modify operations.

Examples

To remove director 13a from the group `dyngrp4` on the local array 6180:

```
symrdf modifygrp -sid 6180 -label dyngrp4 -remove -dir 13a
```

To add director 12a group `dyngrp4` on the local array 6180:

```
symrdf modifygrp -sid 6180 -label dyngrp4 -add -dir 12a
```

Removing dynamic SRDF groups

Use the `symrdf removegrp` command to remove an SRDF group

The following must be true before you can remove an SRDF group:

- Both sides of the SRDF configuration must be defined and reachable
- The group must be empty.

  Use the `symrdf deletepair` command to remove all devices from the group.
- At least one physical connection between arrays must exist.

Note: Deleting the group removes all local and remote director support.

Example In the following example:

- The `symrdf deletepair` command deletes SRDF dynamic pairs defined in a device file `dynpairsfile`,
- The `symrdf removegrp` command removes the local and remote dynamic SRDF groups:

```
symrdf deletepair -sid 80 -rdfg 4 -file dynpairsfile

symrdf removegrp -sid 80 -label dyngrp4
```

Removing an SRDF group from one side of an SRDF configuration

Use the `-symforce` option with `symrdf removegrp` command to remove a dynamic SRDF group from one side of an SRDF configuration.

The following must be true before you can remove one side of an SRDF group:

- The other side is not defined or reachable.

  If the other side of the SRDF configuration is reachable, you cannot execute this command.
- The group is empty.

Example The following example removes `dyngrp4` from array 180 on the local side:

```
symrdf removegrp -sid 180 -label dyngrp4 -symforce
```

Dynamic device pair operations

SRDF enables the creation and deletion of SRDF pairs while the array is in operation. This section describes the steps to add and delete dynamic SRDF pairs, including:

- "Creating a device file"
- "Creating dynamic SRDF device pairs"
- "Creating dynamic concurrent pairs"
- "Deleting dynamic SRDF device pairs"

Creating a device file

To create a device file to define SRDF device pairs:

- Create a text file containing two columns.
- Add a separate line in the file for each device pair.

  All devices for one side of the SRDF pair must be in the first column, and all devices for the other side of the SRDF pair must be in the second column.

  It does not matter which side (R1 or R2) is in which column. The `-type` option of the `symrdf createpair` command determines which column is R1 or R2.

IMPORTANT

All devices for an SRDF side must be in the same column. That is, all R1 devices must be in either the left or right column, and all R2 devices must be in the other column.

HYPERMAX OS Solutions Enabler 8.0.1/HYPERMAX OS does not support meta-devices.

SRDF device pairs consisting of meta-devices on one side and non-meta-devices on the other side are allowed if the meta-devices are on an array running Enginuity 5876.

Example In the following example, the vi text editor creates the `RDFG148` device file consisting of 7 SRDF pairs for the local and remote arrays.

If the left column is R1, the first line pairs local device 0060 with remote device 0092:

```
vi RDFG148
0060 0092
0061 0093
0062 0094
0063 0095
0064 0096
0065 0097
0066 0098
```

When the `symrdf createpair -file` command references the device file, the `-type` option determines whether the devices in the left column are R1 or R2.

Note: The maximum symdev number that can be used on the HYPERMAX OS array is FFBF (65471).

Creating dynamic SRDF device pairs

This section explains the various ways to create dynamic SRDF device pairs. Topic include:

- "Supported devices (thin, standard, diskless)"
- "Blocking createpair when R2 is larger than R1"
- "Verifying host cannot write to target devices with -nowd option"
- "Creating dynamic pairs with -file option"
- "Creating dynamic pairs with -establish option"
- "Creating dynamic pairs with -format option"
- "Creating dynamic pairs with -invalidate option"
- "Creating dynamic pairs with -restore option"
- "Creating dynamic concurrent pairs"

Supported devices (thin, standard, diskless)

Table 10 lists the valid device type combinations for creating an SRDF pair.

Table 10 Device type combinations for creating SRDF pairs

Device 1	Device 2
Standard	Standard
Thin	Thin
Standard	Diskless[4]
Thin[1]	Diskless[1,4]
Thin[2]	Standard[3]

1. FBA devices require Enginuity 5875 or higher.
   CKD devices are not supported.
2. FBA devices require Enginuity 5875 or higher.
   CKD devices require Enginuity 5876 Q42012 SR or higher.
3. Only on Enginuity versions 5671, 5773.50154, or 5875 and higher.
4. 5876 diskless devices cannot be paired with devices on HYPERMAX OS.

Blocking createpair when R2 is larger than R1

IMPORTANT

R2 devices larger than their corresponding R1 devices cannot restore or failover to the R1.

`SYMAPI_RDF_CREATEPAIR_LARGER_R2` in the options file enables/disables creating SRDF pairs where R2 is larger than its corresponding R1. Valid values for the option are:

`ENABLE` - (default value) `createpair` for devices where R2 is larger than its R1 is allowed.

`DISABLE` - `createpair` for devices where R2 larger than its R1 is blocked.

symrdf createpair command syntax

```
symrdf -file Filename -sid SymmID -rdfg GrpNum [-bypass] [-noprompt]
   [-i Interval] [-c Count]
   [-v|-noecho] [-force] [-symforce] [-star]

createpair
   -type {R1|R2}
   -invalidate {R1|R2}
   -establish
   -restore>
   -nowd
   -rp
   -format
   -establish
   -rdf_mode {sync|acp_wp|acp_disk|async}
   -g NewDg
   -remote
   -cons_exempt
```

Where:

- `-type [R1|R2]` specifies whether the devices listed in the left column of the device file are configured as the R1 side or the R2 side.
- `-invalidate [R1|R2]` marks the R1 devices or R2 devices in the list to be the invalidated target for a full device copy once the SRDF pairs are created.
- `-establish` begins copying data to invalidated targets, synchronizing the dynamic SRDF pairs in the device file once the SRDF pairs are created.
- `-restore` begins copying data to the source devices, synchronizing the dynamic SRDF pairs in the device file once the SRDF pairs are created.
- `-nowd` bypasses the check explained in "Verifying host cannot write to target devices with -nowd option" on page 92.
- `-rp` allows the operation even when one or more devices are tagged for RecoverPoint.

  A non-concurrent R1 device can be tagged for RecoverPoint. A RecoverPoint tagged device can be used as an R1 device. A device tagged for RecoverPoint cannot be used as an R2 device (createpair) or swapped to become an R2 device (swap, half-swap).
- `-format` clears all tracks on the R1 and R2 sides to ensure no data exists on either side, and makes the R1 read write to the host.

  You can specify this option with `-establish`, `-type`, `-rdf_mode`, `-cons_exempt`, and `-g`.

  When used with `-establish`, the devices become read write on the SRDF link and are synchronized.
- `-rdf_mode` sets the SRDF mode of the pairs to be one of the following:
  - synchronous (sync),
  - asynchronous (async),
  - adaptive copy disk mode (acp_disk),
  - adaptive copy write pending mode (acp_wp).

  Note: Adaptive copy write pending mode is not supported when the R1 mirror of the RDF pair is on an array running HYPERMAX OS.

  Adaptive Copy Disk is the default mode unless overridden by the SYMAPI_DEFAULT_RDF_MODE options file setting. See "Blocking createpair when R2 is larger than R1" on page 90.
- `-g` specifies the device group name to be created with the devices in the device file.
- `-remote` requests a remote data copy. When the link is ready, data is copied to the SRDF mirror.
- `-cons_exempt` option allows devices to be added without affecting the state of an SRDF/A session or requiring that other devices in the session be suspended.

Verifying host cannot write to target devices with -nowd option

When the SYMAPI_RDF_CHECK_R2_NOT_WRITABLE parameter in the options file is enabled, it verifies that the host cannot write to the R2 devices during `createpair` operations (other than `createpair -invalidate <R1|R2>`). This parameter is disabled by default.

Use the `-nowd` option of the `symrdf createpair` command to bypass this check. The `-nowd` option applies to:

- R2 devices for all `createpair` actions
- R1 devices for the `createpair -invalidate R1`
- The source (R1) devices for `-invalidate R1` and to the target (R2) devices for `-invalidate R2`.

Creating dynamic pairs with -file option

"Creating a device file" on page 88 describes the steps to create a device file.

Example In the following example, the `createpair` command:

- Creates device pairs using device pairs listed in a device file `devicefile`,
- Ignores the check to see if the host can write to its targets (`-nowd`),
- Sets the mode to the default (adaptive copy disk) by not specifying another mode:

```
symrdf createpair -sid 123 -file devicefile -type r1 -rdfg 10 -nowd
```

Creating dynamic pairs with -establish option

Example In the following example, the `createpair -establish` command:

- Creates device pairs using device pairs listed in a device file `devicefile`,
- Begins copying data to its targets, synchronizing the device pairs listed in the device file.

```
symrdf createpair -file devicefile -sid 55 -rdfg 1 -type R1 -establish
```

Note: For the `createpair -establish` option, the R2 may be set to read/write disabled (not ready) if `SYMAPI_RDF_RW_DISABLE_R2=ENABLE` is set in the options file. For more information, refer to the *EMC Solutions Enabler Symmetrix Array Management CLI Product Guide*.

Creating dynamic pairs with -format option

The format option (`-format`) clears all tracks on the R1 and R2 sides to ensure no data exists on either side, and makes the R1 read write to the host. When you use this option to create dynamic pairs, an application cannot write to these devices.

The `symrdf createpair -format` option has the following restrictions:

- Enginuity 5876 and higher.
- Not supported in concurrent SRDF environments.
- The R1 and R2 cannot be mapped to a host.

Example In the following example, the `createpair -format` command:

- Creates device pairs using device pairs listed in a device file `devicefile`,
- Ignores the check to see if the host can write to its targets (`-nowd`),
- Sets the mode for the device pairs to synchronous (`-rdf_mode sync`)
- Clears tracks on the R1 and R2 sides to ensure no data exists on either side, and makes the R1 read write to the host (`-format`):

```
symrdf createpair -sid 66 -format -file devicefile -type r1 -rdfg 117
   -rdf_mode sync -nop
```

Creating dynamic pairs with -invalidate option

Example In the following example, the `symrdf createpair` command:

- Creates new SRDF pairs from the list of device pairs in the file `devicefile`.
- The `-type R1` option identifies the first-column devices in the device file in array 55 as R1 type devices.
- The `-invalidate` option indicates that the R2 devices are the targets to be refreshed from the R1 source devices.
- The `-nowd` option bypasses the validation check to ensure that the target of operation is write disabled to its host.
- The SRDF pairs become members of SRDF group 1.

```
symrdf createpair -sid 55 -file devicefile -rdfg 1 -type R1 -invalidate
   r2 -nowd
```

When the command completes, the pairing information is added to the SYMAPI database file on the host.

When the command completes, you can use the `establish` command to start copying data to the invalidated target devices:

```
symrdf -sid 55 -file devicefile establish -rdfg 1
```

Use the `query` command to check the progress of the establish operation:

```
symrdf -sid 55 -file devicefile query -rdfg 1
```

Once synchronized, you can perform various SRDF operations on SRDF pairs listed in the device file

Creating dynamic pairs with -restore option

Example Use the `-restore` option to perform a restore operation to copy data back to the R1 source devices:

```
symrdf createpair -sid 55 -file devicefile -rdfg 1 -type R1 -restore
```

Once the SRDF device pairs are created, the restore operation begins copying data to the source devices, synchronizing the dynamic SRDF device pairs listed in the device file.

The `createpair` with `-restore` operation is rejected when:

- The device is in one of the following BCV pair states: Synchronized, SyncInProg, Restored, RestoreInProg, SplitInProg.
- The device is the source or target of a TimeFinder/Snap operation.

- There is a background BCV split operation in progress.
- Devices are in the backend not ready state.
- There is an optimizer swap in progress on a device.
- The emulation type is not the same (such as, AS/400 has specific pairing rules).
- There are existing local invalid tracks on either the local or remote device.
- The SRDF/A session is active and `-cons_exempt` is not specified.
- The SRDF group is in asynchronous mode and the devices being added are not the same SRDF type R1 or R2.
- The SRDF group is in asynchronous mode with the SRDF links suspended and the `-establish` or `-restore` option is selected.
- The SRDF group is enabled for SRDF consistency protection.
- The operation involves one or more of the following unsupported devices: VCM DB, SFS, RAD, DRV, RAID-S, WORM-enabled devices, 4-way mirror, Meta member.

Creating dynamic concurrent pairs

In concurrent SRDF, R1 devices are mirrored concurrently to two R2 devices that reside in two remote arrays.

Use the `symrdf createpair` command to dynamically create concurrent SRDF pairs. This feature allows a second remote mirror to be dynamically added by converting a dynamic R1 device to a concurrent SRDF device.

Two remote mirrors are supported for any dynamic R1 device. With Enginuity 5875 or higher, both mirrors of a concurrent R1 device can be operating in SRDF/A mode.

"Concurrent Operations" on page 173 provides more information.

The following rules apply when creating a dynamic concurrent SRDF pair:

- The SRDF BCVs designated as dynamic SRDF devices are not supported.
- The two SRDF mirrors of the concurrent device must be assigned to different SRDF groups.
- The concurrent dynamic SRDF, dynamic SRDF, and concurrent SRDF states must be enabled.
- With the `-restore` selection, the `-remote` option is required if the link status for the first created remote mirror is read/write.

To dynamically create a second remote mirror using the `symrdf createpair` command, you must create two separate device files:

- One file containing the first set of R1/R2 device pairs, and
- A second device file listing the same R1 device paired with a different remote R2 device.

In a previous example, the `createpair` command created dynamic device pairs and RDF group 1 using a device file named `devicefile`. Devices in the first column of the device file were configured as R1 devices on array 55:

```
symrdf createpair -file devicefile -sid 55 -rdfg 1 -type R1
```

Use the `createpair` command and specify a second device file to create a second set of device pairs for the pairs created in the previous example.

Example In the following example:

- `symrdf createpair` command creates new SRDF pairs from the list of devices in a second device file `devicefile2`
- `-type R1` tells SRDF that devices listed in the first column of `devicefile2` are R1 type devices on array 55.

  Devices listed in the second-column become the second remote mirror devices.
- `-rdfg 2` configures the new SRDF device pairs as members of SRDF group 2.
- `-invalidate R1|R2` marks the R1 devices and R2 devices to invalidate for a full copy when the SRDF pair is created.

```
symrdf createpair -sid 55 -rdfg 2 -file devicefile2 -type R1
   -invalidate R1|R2
```

Use the `createpair` command with the **`-restore -remote`** options to restore the standard R2 type devices.

Example In the following example:

- `-restore` begins a full copy from the target to the source, synchronizing the dynamic SRDF pairs in the device file.
- `-remote` copies data is copied to the concurrent SRDF mirror when the concurrent link is ready.

  Note: These operations require the remote data copy option, or the concurrent link to be suspended.

```
symrdf createpair -file devicefile2 -sid 55 -type R1 -restore -remote
```

Note: The concurrent mirror device pairs must belong to a separate RA group than those defined in the first device file pairing.

Deleting dynamic SRDF device pairs

This section describes the steps to delete dynamic SRDF pairs. Topics include:

- "Deleting a dynamic SRDF pair"
- "-symforce option to clear local invalid tracks"
- "Deleting one-half of an SRDF pair"

Deleting a dynamic SRDF pair

The `deletepair` operation:

- Cancels the dynamic SRDF pairs,
- Removes the pairing information from the array and the SYMAPI database,

- If the device file option (`-file Filename`) is specified, changes the specified devices to non-SRDF devices (except for concurrent SRDF pairs).
- If the group option (`-g GroupName`) is specified, changes the device group to a regular device group (except when an SRDF concurrent pair exists).

  When deleting pairs using the group option:

  If additional devices were added to the device group before the `symrdf deletepair` command is issued, those added devices are also changed to non-SRDF devices, and the device group is changed to a regular device group, only if the added devices contained within it were dynamic devices. If the device group contained both SRDF and non-SRDF devices, the device group would be changed to an Invalid state.

The `deletepair` operation is rejected when:

- The device is in one of the following BCV pair states: Synchronized, SyncInProg, Restored, RestoreInProg, and SplitInProg.
- There is a background BCV split operation in progress.
- Devices in the backend are not in the ready state.
- There is an optimizer swap in progress on a device.
- SRDF consistency protection is enabled and the devices were not suspended with the `-cons_exempt` option.
- The SRDF links are not suspended.

  IMPORTANT

  You must suspend the SRDF links using the `symrdf suspend` command before performing the `symrdf deletepair` command.

Examples

To delete pairs for an SRDF group:

- `symrdf suspend` suspends the SRDF links for group NewGrp
- `symrdf deletepair` changes Newgrp to a non-SRDF group

```
symrdf suspend -sid 55 -g NewGrp

symrdf deletepair -sid 55 -g NewGrp
```

To delete pairs using a device file:

- `symrdf suspend` suspends the SRDF links for the devices listed in `devicefile`,
- `symrdf deletepair` deletes the specified SRDF pairs. The devices become non-SRDF devices.
- `-rdfg 2` specifies the SRDF group number:

```
symrdf suspend -sid 55 -file devicefile -rdfg 2

symrdf deletepair -sid 55 -file devicefile -rdfg 2
```

Note: To prevent a device group or a composite group from becoming invalid, first remove the devices from the group before performing the `deletepair` action on a device file.

After execution of the `symrdf deletepair` command, the dynamic SRDF pairs are canceled.

-symforce option to clear local invalid tracks

For Enginuity 5876 and higher, use `-symforce` with `symrdf deletepair` command to:

- Remove the SRDF relationship between the R1 and R2 devices
- Clear any local invalid tracks on these devices.

Example To suspend the SRDF relationship for device pairs listed in device file `devicefile`:

```
symrdf suspend -sid 55 -rdfg 112 -file devicefile
```

To delete the device pairs listed in device file `devicefile`:

```
symrdf deletepair -sid 55 -rdfg 112 -symforce -file devicefile
```

Note: This functionality is not supported for diskless devices and does not delete any device pairs containing R11, R21, or R22 devices.

Deleting one-half of an SRDF pair

The `half_deletepair` command allows you to dynamically remove the SRDF pairing relationship between R1/R2 device pairs. One-half of the specified device pair is converted from an SRDF device to a regular device.

The `half_deletepair` command can be specified using a device file or device group. When specified using a device file, all devices listed in the first column of the file are converted to regular devices (non-SRDF).

"half_deletepair" on page 250 lists the applicable SRDF pair states for `half_deletepair` operations.

The `symrdf half_deletepair` is rejected when:

- The device is in one of the following BCV pair states: Synchronized, SyncInProg, Restored, RestoreInProg, and SplitInProg.
- There is a background BCV split operation in progress.
- Devices in the backend are not in the ready state.
- There is an optimizer swap in progress on a device.
- SRDF consistency protection is enabled and the devices were not suspended with the `-cons_exempt` option.
- The SRDF links are not suspended.

Examples To remove the SRDF pairing from device group `Prod` and convert the devices assigned to `Prod` to regular (non-SRDF) devices, leaving their remote partners as SRDF devices:

```
symrdf suspend -g Prod
```

```
symrdf -g Prod half_deletepair
```

To remove the SRDF pairing of SRDF group 4 on array 1123 and convert one-half of those device pairs to regular (non-SRDF) devices:

```
symrdf suspend -sid 123 -rdfg 4 -file devicefile
```

```
symrdf half_deletepair -sid 123 -rdfg 4 -file devicefile
```

Note: Use the `symrdf list -half_pair` command to list all half pair devices for a specified SID or SRDF group. In addition to `half_deletepair` operations, half pairs can result from `symrdf failover` operations or a configuration changes.

Grouping, moving and swapping dynamic devices

This section describes the steps to group, move and swap dynamic SRDF devices, including:

- "Grouping dynamic pairs with a device file"
- "Moving dynamic SRDF device pairs"
- "Swapping SRDF devices"

Grouping dynamic pairs with a device file

Device groups are the primary method to manage SRDF devices.

An SRDF device file to allows you to manage the devices specified in the file as a single entity.

Procedure To create a device group:

1. Create a list of device pairings in a device file.
2. Use the `createpair` command to create the dynamic SRDF pairs,
3. Use the `-g` *`GroupName`* option to add the devices in the device file to a device group with the specified name.

   For example, to create dynamic devices as specified in file `devicefile` and add them to a group named Newgrp:

   ```
   symrdf createpair -sid 55 -rdfg 2 -file devicefile -type rdf1
   -invalidate r2 -g NewGrp
   ```

   All SRDF commands for these dynamic pairs can now be executed within the context of the `NewGrp` device group

4. Use the `-g` *`GroupName`* option to SRDF commands to perform operations on all the dynamic SRDF pairs in the group.

   For example, establish the group:

   ```
   symrdf -g NewGrp establish
   ```

Moving dynamic SRDF device pairs

Prior to Enginuity version 5773 SRDF pairs could not be moved between groups. To move pairs, original SRDF pairs had to be deleted, new pairs created, and then moved to a new group.

Beginning with Enginuity 5773, you can move dynamic SRDF devices from one SRDF group to another.

Note: There is no need to fully resynchronize the devices when performing the move. The current invalid track counters on both R1 and R2 stay intact.

This section describes moving dynamic SRDF pairs, including:

- "Moving SRDF pairs"
- "Moving half pairs"
- "SRDF mode after a movepair"

Moving SRDF pairs

Use the `movepair -new_rdfg GrpNum` command to move SRDF pairs.

For SRDF/A sessions, use the consistency exempt (`-cons_exempt`) option to allow devices to be moved into an active SRDF/A session without affecting the state of the session or requiring that other devices in the session be suspended.

To move devices out of an active SRDF/A session without affecting the state of the session, first suspend the devices using the `-cons_exempt` option.

After a successful move, the pair state is unchanged.

"movepair" on page 250 lists the applicable SRDF pair states for movepair operations.

symrdf movepair command syntax

SRDF pairs can be moved using a device file or device group:

```
symrdf -file Filename -sid SymmID -rdfg GrpNum movepair -new_rdfg
   GrpNum

symrdf -g GroupName movepair -new_rdfg GrpNum
```

Note: The `-new_rdfg GrpNum` option is required.

See "Move both sides of SRDF device pairs" on page 64 for an example.

The `movepair` operation has the following restrictions:

- A device cannot move when it is enabled for SRDF consistency.
- A device cannot move if it is in asynchronous mode when an SRDF/A cleanup or restore process is running.
- When moving one mirror of a concurrent R1 or an R21 device to a new SRDF group, the destination SRDF group must not be the same as the one supporting the other SRDF mirror.
- When issuing a full `movepair` operation, the destination SRDF group must connect the same two arrays as the original SRDF group.
- If the destination SRDF group is in asynchronous mode, the SRDF group type of the source and destination groups must match. In other words, in asynchronous mode, devices can only be moved from R1 to R1, or from R2 to R2.
- If the destination SRDF group is supporting an active SRDF/A session, the `-cons_exempt` option must be specified.

- If the original SRDF group is supporting an active SRDF/A session, the device pairs being moved must have been suspended using the `-cons_exempt` option.

Moving half pairs

The `half_movepair` operation moves only one side of an SRDF pair from one SRDF group to another.

See "Move one-half of an SRDF pair" on page 63 for an example.

"half_movepair" on page 251 lists the applicable SRDF pair states for control operations.

After a successful `half_movepair` the pair state can go from partitioned to a different state or vice versa.

For example, when a `half_movepair` action results in a normal SRDF pair configuration, the resulting SRDF pair state will be Split, Suspended, FailedOver or Partitioned.

SRDF mode after a movepair

After a `movepair` or `half_movepair` action, the resulting SRDF mode for the moved device is as follows:

- When moving a device *to* an SRDF group that is currently in asynchronous mode, the resulting SRDF mode for the moved device is asynchronous.
- When moving a device *from* an SRDF group that is in asynchronous mode to an SRDF group that is not in asynchronous mode, the resulting SRDF mode for the moved device will be adaptive copy disk.

Swapping SRDF devices

This section describes the following topics:

- "Required states before a swap operation"
- "Displaying SRDF swap-capable devices"
- "Impact on I/O"
- "Disabling SYMAPI control parameter"
- "Refreshing the data status"
- "Swapping dynamic R1/R2 devices"
- "Half-swapping dynamic R1/R2 devices"
- "Swapping cascaded SRDF devices"

The following general steps are required to perform an R1/R2 personality swap and resume SRDF operations:

- Suspend the SRDF remote mirroring.
- Perform a personality swap by converting the R1 to R2 and the R2 to R1 devices.
- Determine the synchronization direction and synchronize the R1 and the R2 devices.
- Resume remote mirroring.

Host I/Os are accepted at the secondary site (now R1 device) and are remotely mirrored to the R2 device at the primary site.

Required states before a swap operation

The current states of the various devices involved in the SRDF swap must be considered *before* executing a swap action. Table 11 lists which states are legal for this operation.

Table 11 SRDF device states before swap operation

SRDF state	Source R2 invalids	Target R2 invalids	State after swap
Suspended with R1 Write Disabled	Refresh R1|R2	Refresh R1|R2	Suspended
R1 Updated	Refresh R1	NA	Suspended
Failed Over	Refresh R1	NA	Suspended

Displaying SRDF swap-capable devices

Use the `symrdf list` command with the `-dynamic` option to display SRDF devices configured as dynamic SRDF-capable:

```
symrdf list -dynamic [-R1] [-R2] [-both]
```

- Use the command with no options to display all SRDF-capable devices.
- Use the **-R1** option to display all dynamic SRDF-capable devices that are configured as capable of becoming R1.
- Use the **-R2** option to display all dynamic SRDF-capable devices that are configured as capable of becoming R2.
- Use the **-both** option to display a list of dynamic SRDF-capable devices that are configured as capable of becoming R1 or R2

From the displayed list, determine which dynamic devices you want to swap.

Impact on I/O

When swapping source and target attributes I/O is not allowed to the R1 device, but I/O is allowed to the R2 device.

Disabling SYMAPI control parameter

In the options file, `SYMAPI_CTRL_OF_NONVISIBLE_DEVS` must be set to `DISABLE` to prevent control of devices that are not mapped to the user host. The default is `ENABLE`.

Once devices are dynamically swapped, an `incremental establish` operation is initiated and the devices become immediately available on the link. "Establish (incremental)" on page 54 explains this operation.

Refreshing the data status

Swapping the R1/R2 designation of the SRDF devices can impact the state of your stored data.

The refresh action indicates which device does **not** hold a valid copy of the data before the swap operation begins. If you determine that the R1 holds the valid copy, the action of `refresh R2` will obtain a count of the tracks that are different on the R2 and mark those tracks to refresh from the R1 to the R2 device.

The result will be the reverse if you choose to `refresh R1` as the option:

- `-refresh R1` — The R2 device holds the valid copy and the R1 device's invalid tracks are updated using the R2 data.
- `-refresh R2` — The R1 device holds the valid copy and the R2 device's invalid tracks are updated using the R1 data.

symrdf swap -refresh command syntax

You can issue the `swap` command for device groups, composite groups and device files:

```
symrdf -g DgName [-v | -noecho] [-force] [-symforce]
   [-bypass] [-noprompt] [-i Interval] [-c Count]
   [-hop2 | -bcv [-hop2] | -all | -rbcv | -brbcv]
   [-rdfg GrpNum] [-star]
```

Where:

- `-bcv` targets just the BCV devices associated with the SRDF device group for the `swap` action.
  - Use `-all` to target both BCV and standard devices.
  - Use nothing to target only standard devices.
- `-refresh` marks the source R1 devices or the target R2 devices to refresh from the remote mirror.
- `-hop2` targets the SRDF action at the group's second-hop devices in a cascaded SRDF relationship.
  - Use alone to target standard devices
  - Use `-bcv -hop2` to target BCV devices

Example The following example:

- Swaps the R1 designation of the associated BCV RDF1 devices within device group `ProdGrpB`.
- Marks to refresh any modified data on the current R1 side of these BCVs from their R2 mirrors:

```
symrdf -g ProdGrpB -bcv swap -refresh R1
```

Swapping dynamic R1/R2 devices

Dynamic R1/R2 swaps switch the SRDF personality of the SRDF device or composite group. Swaps can also be performed on devices in SRDF/A mode. Dynamic SRDF must be enabled to perform this operation.

With a dynamic swap, source R1 devices become target R2 devices and target R2 devices become source R1 devices.

Dynamic SRDF devices are configured as one of three types: RDF1 capable, RDF2 capable, or both. Devices must be configured as both in order to participate in a dynamic swap.

Dynamic swap operations have the following restrictions:

- Dynamic swap is not supported on arrays where the R2 device is larger than the R1 device.
- Do not perform a dynamic swap on SRDF/A devices enabled for consistency protection or if the SRDF/A session is actively copying.

HYPERMAX OS

- Adaptive copy write pending is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS. If the R2 side is on an array running HYPERMAX OS and the mode of the R1 is adaptive copy write pending, SRDF sets the mode to adaptive copy disk.

Half-swapping dynamic R1/R2 devices

You can issue a `half_swap` command that swaps one half of an SRDF relationship. This command changes an R1 mirror to an R2 mirror or an R2 mirror to an R1 mirror.

The `half_swap operation` has the following restrictions:

- The R2 device cannot be larger than the R1 device.
- A swap cannot be performed during an active SRDF/A session or when cleanup or restore is running.

Swapping cascaded SRDF devices

Swapping of an R21 device in a cascaded SRDF relationship is allowed as long as the R21 device is converted into a concurrent R1 (R11) device.

You can convert a concurrent R1 device into an R21 device.

For example, in an R2->R11->R2 configuration, you can swap either side of the relationship:

- Swap R2-> to get R1-> R21->R2
- Swap R11-> R2 to get R2-> R21->R1

The following swap is allowed:

Swap R1->R21 to get R2-> R11-> R2

The following swap is **not** allowed:

Swap R21->R2 to get R1->R22-> R1

Dynamic failover operations

SRDF dynamic devices can be quickly failed over, swapped, and then re-established all within a single command-line operation.

Note: This functionality requires that dynamic devices be both RDF1 and RDF2 capable.

Dynamic failover establish

The `symrdf failover -establish` command can be used as a composite operation on dynamic SRDF devices to quickly perform the following operations on SRDF devices in the specified group using a single command:

1. Failover the devices in the group.

   R2 devices in the group are made read/write enabled to their local hosts.

"Failover" on page 58 provides a detailed explanation of a failover operation.

2. After the failover operation has completed, swap the SRDF pair personalities.

   R1 devices become R2 devices and the R2 devices become R1 devices).

   "Swapping dynamic R1/R2 devices" on page 102 provides a detailed explanation with restrictions that apply when performing a dynamic swap operation.

3. Once the devices are dynamically swapped, perform an incremental establish operation.

   The devices become immediately available on the link.

   "Establish (incremental)" on page 54 explains this operation.

The `failover restore` operation has the following restrictions:

- Both the R1 and the R2 devices in the failover must be dynamic SRDF devices.
- The R2 device cannot be larger than its R1 device.
- The swap cannot result in a cascaded R21<-->R21 device pair.
- Cannot execute this command on both mirrors of a concurrent R1 device (composite group operation). This swap would convert the concurrent R1 into a concurrent R2, with a restore on both mirrors of that concurrent R2.

Dynamic failover restore

IMPORTANT

The `symrdf failover -establish` operation does not support devices operating in asynchronous mode with a read/write link because the R2 data is two cycle switches behind the R1 data, and swapping these devices would result in data loss.

Restrictions

- If an SRDF group being failed over is operating in asynchronous mode, then all devices in the group must be failed over in the same operation.
- The R1 and the R2 devices in the failover must be dynamic SRDF devices.
- The R2 device cannot be larger than its R1 device.
- The SRDF swap cannot result in a cascaded R21<-->R21 device pair.
- Not supported by any device group operations with more than one SRDF group.
- Cannot execute this command on both mirrors of a concurrent R2 device (composite group operation). This swap would convert the concurrent R2 into a concurrent R1, with a restore on both mirrors of that concurrent R1.

The `symrdf failover -restore` swaps the R1 and R2 and restores the invalid tracks on the new R2 side (formerly R1) to the new R1 side (formerly R2).

You can execute this command for device groups, composite groups and device files. The devices in this failover can be using synchronous or asynchronous links.

symrdf failover command syntax

For device groups:

```
symrdf -g DgName [-v | -noecho] [-force] [-symforce]
   [-bypass] [-noprompt] [-i Interval] [-c Count]
   [-hop2 | -bcv [-hop2] | -all | -rbcv | -brbcv]
   [-rdfg GrpNum] [-star]

failover [- immediate | -establish | -restore [-remote]]
```

Where:

- `-immediate` - deactivates the SRDF/A session immediately, without waiting for the two cycle switches to complete before starting the `failover -restore` operation.
- `-restore` - causes the dynamic SRDF device pairs to swap personality and start an incremental restore.
- `-remote` - requests a remote data copy flag with failback, failover, restore, update, and resume. When the concurrent link is ready, data is copied to the concurrent SRDF mirror. These operations require the remote data copy option, or the concurrent link to be suspended.

Usage notes

- Enginuity 5773 and higher is required to perform a swap resulting in a cascaded R21 device.
- Enginuity 5773.150 or higher is required to perform a swap resulting in a concurrent R2 device.

CHAPTER 4
SRDF/Asynchronous Operations

This chapter describes the following topics:

- Overview 108
- Transitioning to asynchronous mode 115
- Setting SRDF/A group cycle time, priority, and transmit idle 116
- Enabling/disabling consistency for SRDF/A devices 118
- Adding/removing devices using consistency exempt 119
- Managing Delta Set Extension 123
- Managing transmit idle 131
- Managing SRDF/A write pacing 131
- Displaying SRDF/A 138

Overview

SRDF/Asynchronous (SRDF/A) is a long distance disaster restart solution with fast application response times.

SRDF/A cycle modes

SRDF/A provides an R2 copy that is slightly behind its associated R1. Host writes are collected for a configurable interval (`-cycle_time`) into delta sets. Delta sets are transferred to the remote array in predefined timed cycles.

Control of SRDF/A cycles varies depending on whether the array is running in legacy mode (Enginuity 5773 - 5876) or multi-cycle mode (HYPERMAX OS):

Enginuity 5876

If either array in the solution is running Enginuity 5876, there are 2 cycles on the R1 side, and 2 cycles on the R2 side.

Each cycle switch moves the delta set to the next cycle in the process. This mode is referred to as “legacy mode”.

A new capture cycle cannot start until the transmit cycle completes its commit of data from the R1 side to the R2 side, and the R2 apply cycle is empty.

The basic steps in the life of a delta set in legacy mode include:

1. On the R1 side, host writes collect in the Capture cycle's delta set for a specified number of seconds.

   The length of the cycle is specified using the -cycle_time attribute.

   If a given track is overwritten multiple times, only the last write is preserved.

2. Once the cycle timer expires, and both the R1's Transmit cycle and the R2's Apply cycle are empty:

   - The delta set in the R2's Receive cycle is moved to the R2's Apply cycle, from which it is transferred to disk.
   - The delta set in the R1's Capture cycle is moved to the R1's Transmit cycle, from which it begins transferring to the R2's Receive cycle.
   - A new delta set is created as the R1 Capture cycle, to collect host writes. The delta set is received on the R2 side.

   Subsequent host writes are collected into the next delta set.

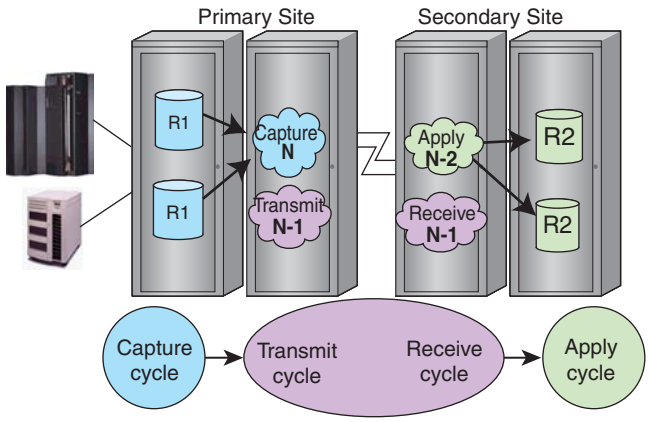


Figure 11 SRDF/A legacy mode

Mixed configurations

When one array in an SRDF configuration is running HYPERMAX OS, and one or more other arrays are running Enginuity 5876:

- SRDF/A single sessions (SSC) have only two cycles on the R1 side (legacy mode)
- SRDF/A multi-session consistency sessions (MSC) operate in legacy mode.

When a delta set is applied to the R2 target device, the R1 and R2 are in the *consistent* pair state. The R2 side is consistently 2 cycles behind the R1 site.

In the event of a failure at the R1 site or of the SRDF links, a partial delta set of data can be discarded, preserving consistency on the R2. The maximum data loss of for such failures is two SRDF/A cycles or less.

Multiple devices or device groups that require consistency can be grouped into *consistency groups*. Members of consistency groups cycle at the same time, to ensure consistency among the members, and if one member is interrupted, all other members suspend.

"SRDF consistency group operations" on page 145 provides more information.

HYPERMAX OS

If both arrays in the solution are running HYPERMAX OS, both SSC and MSC operate in multi-cycle mode. There can be 2 *or more* cycles on the R1, but only 2 cycles on the R2 side. Cycle switches are decoupled from committing delta sets from the R1 to the R2.

When the preset Minimum Cycle Time is reached, the R1 data collected during the capture cycle is added to the transmit queue and a new R1 capture cycle is started. There is no wait for the commit on the R2 side before starting a new capture cycle.

The transmit queue holds cycles waiting to be transmitted to the R2 side. Data in the transmit queue is committed to the R2 receive cycle when the current transmit cycle and apply cycle are empty.

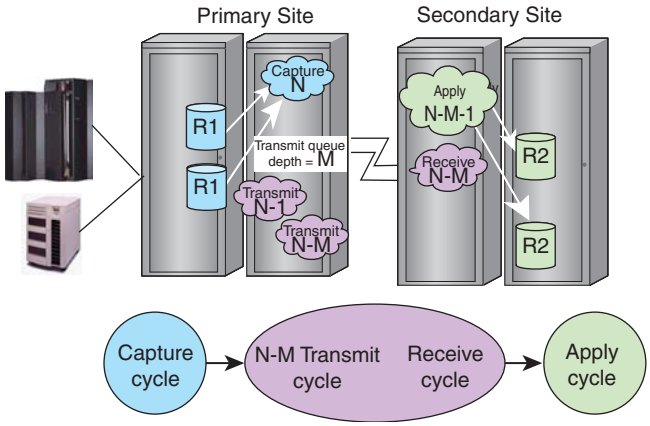


Figure 12 SRDF/A multi-cycle mode

Queuing allows smaller cycles of data to be buffered on the R1 side and smaller delta sets to be transferred to the R2 side.

The SRDF/A session can adjust to accommodate changes in the solution. If the SRDF link speed decreases or the apply rate on the R2 side decreases, more SRDF/A capture cycles can be added to the R1 side.

Data on the R2 side can be more than 2 cycles behind the R1.

In event of R1 failure or link failure, a partial delta set of data can be discarded, preserving consistency on the R2. The maximum data loss of for such failures can be more than two SRDF/A cycles.

EMC VMAX3 Family Product Guide provides a detailed description of SRDF/A multi-cycle mode.

SRDF/A write pacing, transmit idle, and Delta Set Extension

Unexpected write volume, transient link outages and other problems can result in the SRDF/A session being dropped. The following features help prevent session drop:

- Write pacing throttles host I/O to prevent cache overflow.

  You can enable/disable write pacing for a group or a device.

  "Managing SRDF/A write pacing" on page 131 provides more information.

- Transmit idle allows SRDF/A to wait a specified interval before dropping a session due to an SRDF link outage.

  Transmit idle allows the SRDF/A session to survive transient link outages.

  "Managing transmit idle" on page 131 provides more information.

- Delta Set Extension (DSE) pools page tracks to save pools when the array's write pending count exceeds the array's write pending limit (`-threshold`).

  DSE pools help SRDF/A manage unexpected spikes in I/O.

  "Managing Delta Set Extension" on page 123 provides more information.

SRDF/A benefits

- Provides remote mirroring at extended distances without adding response time to the R1 host. SRDF/A response times are equivalent to local non-SRDF devices.
- Supports efficient bandwidth usage.

  During the interval when SRDF collects host writes into a delta set, if the host writes the same track multiple times, SRDF sends only the latest change over the link.
- Supports all current SRDF topologies, including point-to-point and switched fabric.

  Supports all hosts and data emulation types supported by the array (FBA, CKD, AS 400).
- Provides dependent write consistency.

  Many applications (in particular, DBMS), use dependent write logic to ensure data integrity in the event of a failure. A dependent write is a write that is not issued by the application unless some prior I/O has completed. For example, when a DBMS updates a database:

  1. It first writes to the disk containing the log to indicate an update is coming, then
  2. It writes the data to the database dataset, and finally
  3. It writes again to the log volume to indicate that the database update is complete.

  These three writes (log, database, and log again) are related and each successive I/O is not issued until the prior I/O has successfully completed. If the writes are out of order, and an event such as a failure, or a creation of a point in time copy happens at that exact time, unrecoverable data loss may occur.

  SRDF/A ensures that dependent writes are in the same delta set or that later writes are in later delta sets.

SRDF /A features

- Supports multiple SRDF/A sessions per array.

  You can configure *all* your SRDF groups as SRDF/A-capable.
- Supports database consistency with SRDF/A-capable devices in composite groups using Multi-session Consistency (MSC).

  "SRDF consistency group operations" on page 145 provides more information.
- Maintains data consistency when mode is switched from asynchronous to synchronous.

  "Enabling/disabling consistency for SRDF/A devices" on page 118provides more information.
- Allows you to dynamically modify session priority, minimum cycle time, transmit idle and DSE pools.

HYPERMAX OS

For arrays running HYPERMAX OS, use the `symrdf set` command to dynamically change SRDF/A settings.

Enginuity 5773 - 5876

For arrays running Enginuity 5773-5876, use the `symconfigure` command to dynamically change SRDF/A settings.

- Allows you to dynamically add/remove device pairs from an active SRDF/A session without affecting the state of the session, or the reporting of SRDF pair states for devices that are not the target of the operation.

  "Adding/removing devices using consistency exempt" on page 119provides more information.

- Improves link utilization using software and hardware compression (Enginuity 5875 and higher).

Restrictions

- If either the R1 side or R2 side of an SRDF/A session is running HYPERMAX OS, Solutions Enabler 8.0.1 is required to monitor MSC groups.
- All SRDF/A-capable devices running in asynchronous mode must be managed together in an SRDF/A session.
- For SRDF/A-capable devices enabled for consistency group protection, consistency must be disabled before attempting to change the mode from asynchronous.
- SRDF Automated Replication (SRDF/AR) control operations are currently **not** supported for SRDF/A-capable devices running in asynchronous mode.
- All SRDF/A sessions enabled within a consistency group operate in the same mode, multi-cycle or legacy. For example, if:
  - SRDF group 1 connects Site A and Site B, both running HYPERMAX OS, and
  - SRDF group 2 Site A running HYMPERMAX OS and Site C running 5876.
    - Group 1 can run in multi-cycle mode.
    - Group 2 must run in legacy mode.

  If both groups are in the same consistency group and are enabled together, then group 1 will transition from multi-cycle to legacy mode as a part of the enable.

- If there are tracks owed from the R2 to the R1, do not switch to asynchronous mode.

  Note: If tracks are owed to the R1 device, the -force option is required to make SRDF/A-capable devices in asynchronous mode Ready on the link.

TimeFinder snaps and clones

- TF/Snap and TF/Clone operations affect whether SRDF devices are allowed to be set in asynchronous mode.
- TF/Snap and TF/Clone pair states impact setting SRDF devices to asynchronous mode.

  "SRDF operations and TimeFinder sessions" on page 261 provides more information.

- Some Snap and Clone operations are not be allowed SRDF/A-capable devices operating in asynchronous mode.

  EMC Solutions Enabler Symmetrix TimeFinder Family CLI Product Guide provides more information.

Move operations

After a `movepair` or `half_movepair` action, the resulting SRDF mode for the moved device will be as follows:

- When moving a device *to* an SRDF group that is currently in asynchronous mode, the resulting SRDF mode for the device being moved will be asynchronous.
- When moving a device *from* an SRDF group in asynchronous mode, the resulting SRDF mode for the device being moved is synchronous.

Using BCVs to protect the R2 side

EMC recommends that you use TimeFinder BCVs at the remote site to mirror R2 devices. This practice preserves a consistent image of data before resynchronization operations.

R2 device BCVs can be consistently split off of the R2 without dropping the SRDF links or disrupting 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.

EMC Solutions Enabler TimeFinder Family CLI Product Guide provides more information.

Immediate option

By default, the `failover`, `split`, and `suspend` operations cause SRDF to wait until the current cycle completes before dropping the session and making the devices Not Ready on the link. Completion time for these operations may be quite long.

Use the `-immediate` option in conjunction with `failover, split,` or `suspend` commands to immediately drop the SRDF/A session and make the devices Not Ready on the link.

The devices remain in asynchronous mode and pending tracks are converted to invalid tracks.

Use the `symrdf query -rdfa` command to display the number of tracks not committed to the R2 side and invalid tracks.

Restrictions

- The `-immediate` option applies only to devices participating in an active SRDF/A session.
- The `-immediate` option may result in remote invalid tracks on both the R1 and the R2 sides.
- The `-immediate` option does not compromise the consistency of data on the R2 side, but requires operator intervention to resolve any invalid tracks by using the correct `symrdf` command and pair state.
- If consistency is enabled on SRDF/A-capable devices, the `-force` option must used.

SRDF/Asynchronous operations

All SRDF/A operations (with the exception of consistency exempt, discussed later) must be performed on all devices in an SRDF group.

Thus, all devices in an SRDF group must be in the same SRDF device group. This is in contrast with SRDF/S, where operations can be performed on a subset of devices in an SRDF group.

Table 12 summarizes the operations described in this chapter.

Table 12 SRDF/A control operations

Control operation	Command	Description
Transition to asynchronous mode page 115	symrdf set mode async	Change the mode of the an SRDF group, composite group or device list to asynchronous mode.
Set SRDF group attributes page 116	symrdf set rdfa	Set the cycle time, session priority, and transmit idle for and SRDF/A group.
Check R1 devices for invalid tracks page 117	symrdf verify -noinvalids -consistent	Verify whether invalid tracks exist on both the R1 and R2 devices for a SRDF group, composite group or devices in a device list.
Enable/disable consistency for SRDF/A devices page 118	symrdf enable	Enable/disable consistency for a device group or devices in a device list.
Add/remove devices using the consistency exempt option page 119	symrdf createpair symrdf suspend symrdf movepair symrdf resume symrdf verify	Dynamically add and remove device pairs from an active SRDF/A session.
Display checkpoint complete status page 122	symrdf checkpoint	Display a checkpoint complete status when the data in the current cycle is committed to the R2 side.
Manage DSE page 123	symrdf set rdfa_dse symconfigure commit symcfg show	Set the SRDF/A DSE attributes for an SRDF group. Enginuity 5786 only: Add/remove/enable devices in DSE pools. Associate a DSE pool with and SRDF group. Monitor/display DSE pools.
Activate/deactivate DSE page 129	rdfa_dse_autostart symrdf activate/deactivate	Activate/deactivate SRDF/A DSE.
Enable transmit idle page 131	symrdf set rdfa -transmit_idle	Allow SRDF/A sessions to manage transient link outages without dropping.
Throttle host I/O page 133	symrdf set rdfa_pace symrdf -rdfa_pace activate symrdf -rdfa_pace deactivate symrdf -rdfa_wpace_exempt	Enable SRDF/A write pacing for groups or devices.
Display SRDF/A sessions page 138	symdg show symrdf -g *DgName* query -rdfa	Display SRDF/A sessions. Display SRDF/A groups.
Display SRDF devices page 138	symrdf list -rdfa	List SRDF/A capable devices.

Transitioning to asynchronous mode

You can create a new device group and specify the mode as asynchronous, or you can transition an existing SRDF device or group to asynchronous from another mode. The time it takes for devices or groups to transition from one mode to asynchronous mode varies depending on the original mode:

- From synchronous mode:

  If the devices are in a Synchronized state, the R2 devices already have a consistent copy.

  Enabling SRDF/A provides consistent data on the R2 immediately.

- From adaptive copy disk mode:

  Invalid tracks owed to the R2 are synchronized.

  Enabling SRDF/A provides consistent data on the R2 in two cycles.

- From adaptive copy write pending mode:

  Write pending slots are merged into the SRDF/A cycles.

  Enabling SRDF/A provides consistent data on the R2 two cycles after there are no more write pending slots.

symrdf set mode async command syntax

You can set the mode to asynchronous for a device group, composite group, or devices in a device file:

```
symrdf -g DgName set mode async
symrdf -cg CgName set mode async
symrdf -file Filename set mode async
```

Examples To set device group `prod` to asynchronous mode:

```
symrdf -g prod set mode async
```

To set composite group `Comp` to asynchronous mode:

```
symrdf -cg Comp set mode async
```

To set the devices listed in `Device.txt` to asynchronous mode:

```
symrdf -file device.txt set mode async
```

Note: This operation may not be allowed on TimeFinder/Snap and TimeFinder/Clone device pairs. "set mode async" on page 292, and page 293 provides more information.

Transitioning to synchronous mode

You can transition an SRDF/A device or device group to synchronous mode without losing consistency. Consistency on the R2 side is preserved.

The amount of time to complete the transition varies depending on whether the mode is legacy or multi-cycle:

- In legacy mode, the switch from asynchronous to synchronous requires two SRDF/A cycle switches to complete.

- In multi-cycle mode, the amount of time required includes the time to commit the current capture cycle and all cycles currently in the transmit queue to the R2 side.

Switching from asynchronous to synchronous is supported on Enginuity 5671 and higher.

symrdf set mode sync -consistent command syntax

You can set the mode to synchronous for a device group or devices in a device file:

```
symrdf -g DgName -consistent set mode sync
symrdf -file Filename -consistent set mode sync
```

Examples

To switch modes from asynchronous to synchronous and maintain R2 data consistency in group `prod`:

```
symrdf -g prod -consistent set mode sync
```

To switch modes from asynchronous to synchronous and maintain R2 data consistency for devices listed in device file `devfile1`:

```
symrdf -f devfile1 -consistent set mode sync
```

Setting SRDF/A group cycle time, priority, and transmit idle

SRDF/A configuration parameters include array-wide parameters, and group level settings.

EMC Solutions Enabler Array Management CLI Product Guide describes the steps to set the following SRDF/A array-wide parameters:

- SRDF/A cache usage - The percentage of write pending slots available to SRDF/A. Raising the value increases how much cache SRDF/A can use. Lowering the value reserves additional cache for non-SRDF/A cache usage.
- Maximum host throttle time - When the write pending limit is reached, delays writes from the host until a cache slot becomes free.

symrdf set rdfa command syntax

To set the SRDF/A group-level attributes on an SRDF group:

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
     [-noprompt] [-i Interval] [-c Count]

.............

set rdfa
   [-cycle_time 1 - 60]
   [-priority 1 - 64]
   [-transmit_idle {on|off}]
   [-both_sides]
```

Where:

- `-cycle_time` (`-cyc`) is the minimum time to wait before attempting an SRDF/A cycle switch.

  Valid values are 1 through 60 seconds.

  Default value for Enginuity 5874 and higher is 15.

  Default value for Enginuity lower than 5874 is 30.

See "Enabling and disabling SRDF consistency protection" on page 148 for details about how minimum cycle times are handled when you enable consistency protection.

- `-priority` (`-pri`) determines which SRDF/A sessions are dropped if the cache becomes full.

  Valid values are 1 (highest priority, last to be dropped) through 64 (lowest priority).

  The default value is 33.

- `-transmit_idle` (`-tra`) allows the SRDF/A session to wait (not drop) when the link cannot transmit data.

  Valid state values are on and off.

  Default is on.

- `-both_sides` applies the SRDF/A attributes to both the source and target sides of an SRDF/A session. If `-both_sides` is not specified, attributes are only applied to the source side.

Examples

To set the minimum cycle time for both sides of SRDF/A group 160:

```
symrdf -sid 134 -rdfg 160 set rdfa -cycle_time 32 -both_sides
```

To set the session priority for both sides of SRDF/A group 160:

```
symrdf -sid 134 -rdfg 160 set rdfa -priority 55 -both_sides
```

To set the cycle time and session priority for only the source side of SRDF/A group 12:

```
symrdf -sid 134 -rdfg 12 set rdfa -cycle_time 32 -priority 20
```

```
An RDF Set 'Attributes' operation execution is in progress for RDF group 12.
Please wait...

    SRDF/A Set Min Cycle Time(1134,012)..........................Started.
    SRDF/A Set Min Cycle Time (1134,012).........................Done.
    SRDF/A Set Priority (1134,012)...............................Started.
    SRDF/A Set Priority (1134,012)..............................,,,,,Done.
The RDF Set 'Attributes' operation successfully executed for RDF group 12.
```

Checking for R1 invalid tracks

Under normal operations, the `symrdf verify -consistent` command verifies that SRDF device pairs are in the R2 Consistent pair state. No invalid tracks are owed to the R2 side from its R1 side.

When an SRDF pair is in the Split state and the host writes to its R2 device, invalid tracks are owed to its R1 device.

When the pair is restored, the pair is still in the Consistent state because no invalid tracks are owed to the R2 device. SRDF does not recognize invalid tracks owed from R2 to R1.

Use the `-noinvalids` and `-consistent` options to perform an addition check to verify whether invalid tracks exist on both the R1 and R2 devices.

symrdf verify command syntax

You can issue the `symrdf verify` command with `-noinvalids` and `-consistent` options, to device groups, composite groups, and device files.

```
symrdf verify -g Dgname -consistent -noinv
symrdf verify -cg Cgname -consistent -noinv
symrdf verify -file DeviceFile -consistent -noinv
```

Example To monitor the clearing of invalid tracks every 60 seconds for the device group `dg1`:

```
symrdf verify -g dg1 -consistent -noinv -i 60

None of the devices in the group 'dg1' are in 'Consistent with no
   invalid tracks' state.

Not all devices in the group 'dg1' are in 'Consistent with no invalid
   tracks' state.

All devices in the group 'dg1' are in 'Consistent with no invalid
   tracks' state.
```

Enabling/disabling consistency for SRDF/A devices

The consistency feature ensures the dependent-write consistency of the data distributed across multiple R1 devices. The R1 and R2 devices can be distributed across multiple primary and secondary arrays.

Consistency groups are groups of SRDF devices enabled for database consistency. SRDF devices that belong to the same consistency group act in unison to preserve dependent-write consistency of a database distributed across multiple devices within the consistency group.

The consistency group ensures that remote mirroring is suspended for all SRDF devices in a consistency group as soon as one SRDF device in the group fails to send data across the SRDF links.

- Use the `enable` argument to enable consistency protection for devices in SRDF/Asynchronous mode by device group or device list.

  When consistency is enabled, and data cannot be copied from the R1 to the R2, all devices in the group will be made not ready on the links.

- Use the `disable` argument to disable consistency protection for devices in SRDF/Asynchronous mode by device group or device list.

  When consistency is disabled, and data cannot be copied from the R1 to the R2, only the devices in the group that are experiencing problems will be made not ready on the links. The device state for any remaining devices in the group will remain the same.

"SRDF consistency group operations" on page 145 describes how to enable consistency protection for SRDF/A pairs in a composite group.

symrdf enable command syntax

You can enable consistency for SRDF/A device pairs in a device group, or devices in a device file:

```
symrdf -g DgName -sid SID -rdfg RdfgNum enable
symrdf -file Filename -sid SID -rdfg RdfgNum enable
```

IMPORTANT

For concurrent SRDF configurations, you must enable consistency for each R2 mirror separately.

To use the `-file Filename` option:

- All device pairs in that SRDF group must be in the device file.
- If the device file includes concurrent devices, only the R2 side specified by the `-sid SID -rdfg` arguments is enabled.

  The device group on the second R2 side is not enabled.

To use the `-g DgName` option:

- All device pairs in that SRDF group must be in the device group.
- If `the device group i`ncludes concurrent devices, only the R2 side specified by the `-sid SID -rdfg` arguments is enabled.

Restrictions

Because you must enable consistency for each R2 mirror separately in a concurrent relationship, you cannot use the `-rdfg all` option.

Examples

To enable consistency protection for SRDF/A pairs in device group `prod`:

```
symrdf -g prod enable
```

To enable consistency protection for SRDF/A pairs listed in device file `devfile1`:

```
symrdf -file devfile1 -sid 123 -rdfg 10 enable
```

To disable consistency protection for SRDF/A pairs in device group `prod`:

```
symrdf -g prod disable
```

To disable consistency protection for SRDF/A pairs listed in device file `devfile1`:

```
symrdf -file devfile1 -sid -rdfg 10 disable
```

To enable consistency for devices in device file `file1`:

```
symrdf -f file1 -sid 123 -rdfg 55 enable
```

To enable consistency for R2 devices in a concurrent configuration (SRDF group 56 and SRDF group 57) of `devgroup2`:

```
symrdf -g devgroup2 -rdfg 56 enable
symrdf -g devgroup2 -rdfg 57 enable
```

Adding/removing devices using consistency exempt

Note: The consistency exempt option is available with Enginuity 5773.150 and higher.

The consistency exempt option (`-cons_exempt`) allows you to dynamically add and remove device pairs from an active SRDF/A session without affecting:

- the state of the session, or
- the reporting of SRDF pair states for devices that are not the target of the operation

When enabled, the consistency exempt option places devices into a consistency exempt state. Exempt devices are excluded from the group's consistency check.

After the operation is complete, the consistency exempt state is automatically terminated. Specifically, consistency is terminated when:

- The target devices are resumed and fully synchronized and
- Two full cycle switches have occurred, or

  The devices are removed from the group.

You can use the `-cons_exempt` option with the following commands:

- `createpair`

  The SRDF pairs become consistency exempt in the SRDF group in which they are created.

- `movepair`, `half_movepair`

  The SRDF pairs become consistency exempt in the target SRDF group into which they are moved.

- `suspend`

  Device pairs become consistency exempt in their current SRDF group. Device pairs moved from one group to another can be suspended with consistency exempt without effecting other devices in their group.

When devices are suspended and consistency exempt (within an active SRDF/A session) they can be controlled apart from other devices in the session. This is useful for `resume`, `establish`, `deletepair`, `half_deletepair`, `movepair`, and `half_movepair` operations.

Restrictions

- The consistency exempt option cannot be used:
  - For devices that are part of an SRDF/Star configuration.
  - For an SRDF/A session that is in the Transmit Idle state.
- If the device is an R2 device and the SRDF/A session is active, the `half_movepair` and `half_deletepair` commands are blocked.
- If the session is deactivated before the consistency exempt state is cleared, when re-activated, the device remains in the consistency exempt state until the device has no invalid tracks that need to be synchronized.
- A `movepair` operation of an SRDF pair to another SRDF group with an active SRDF/A session is only allowed when the SRDF pair state is suspended and can be blocked if in the failed over or split pair state.
- The `createpair` and `movepair` operations are allowed without the `-cons_exempt` option if the new SRDF group is operating in the asynchronous mode but the SRDF/A session is not active.

Adding device pairs to an active SRDF/A session

Procedure The following example uses device file “Myfile” to add device pairs to an active SRDF/A session:

1. Use the `createpair -establish` command to create the new device pairs, add them to a temporary SRDF group (10), and synchronize:

```
symrdf createpair -file Myfile -sid 1234 -rdfg 10 -type RDF1
-establish
```

2. Use the `verify -synchronized` command to monitor synchronization:

```
symrdf verify -file MyFile -sid 1234 -rdfg 10 -synchronized
```

When the device pairs are synchronized:

3. Use the `suspend` command to suspend the device pairs in th temporary group so they can be moved to the active SRDF/A group:

```
symrdf suspend -file MyFile -sid 1234 -rdfg 10
```

Note: The temporary group is synchronous, thus the consistency exempt option is not allowed.

4. Use the `movepair` command with the `-cons_exempt` option to move the device pairs from the temporary SRDF group to the active SRDF/A group:

```
symrdf movepair -file MyFile -sid 1234 -rdfg 10 -new_rdfg 20
-cons_exempt
```

5. Use the `resume` command to resume the device pairs:

```
symrdf resume -file MyFile -sid 1234 -rdfg 20
```

6. Use the `verify -consistent -noinvalids` command to display when the device pairs become consistent and there are no invalid tracks on the R1 and R2 sides:

```
symrdf verify -file MyFile -sid 1234 -rdfg 20 -consistent -noinvalids
```

CAUTION

Do not enable host access to the R1 side until the pair state for the devices reaches Consistent.

Removing device pairs from an active SRDF/A session

Procedure The following example uses device file “Myfile” to remove device pairs from an active SRDF/A session:

1. Use the `suspend` command with the `-cons_exempt` option to suspend the device pairs to be removed:

```
symrdf suspend -file MyFile -sid 1234 -rdfg 20 -cons_exempt
```

2. Use the `movepair` command to move the device pairs from the current SRDF group to another SRDF group:

```
symrdf movepair -file MyFile -sid 1234 -rdfg 20 -new_rdfg 30
```

3. Use the `resume` command to resume the devices in their new group:

```
symrdf resume -file MyFile -sid 1234 -rdfg 30
```

4. Use the `verify -synchronized` command to monitor synchronization:

```
symrdf verify -file MyFile -sid 1234 -rdfg 30 -synchronized
```

CAUTION

Do not enable host access to the R1 side until the pair state for the devices reaches Consistent.

Displaying checkpoint complete status

Use the `checkpoint` argument to display a checkpoint complete status when the data in the current cycle is committed to the R2 side.

The target devices must be in an active SRDF/A session.

symrdf checkpoint command syntax

You can issue the `checkpoint` operation on a device group, composite group, and device file:

```
symrdf -g DgName [-i Interval] [-c Count] [-rdfg GrpNum]
     [ -hop2 | -bcv [-hop2] | -all | -rbcv | -brbcv] checkpoint

symrdf -cg CgName [-i Interval] [-c Count]
     [ -hop2 ] [-rdfg SymmID:GrpNum | name:RdfGroupName] checkpoint

symrdf -file Filename -sid SymmID -rdfg GrpNum
     [-offline] [-i Interval] [-c Count] checkpoint
```

Where:

- `-c Count` specifies the number of times (Count) to repeat the operation before exiting.
- `-i Interval` specifies the number of seconds to wait between successive iterations of the operation.

  Default: 10 seconds.

  Minimum interval: 5 seconds.

If `-c Count` is not specified and `-i Interval` is specified, the operation repeats continuously at the specified interval.

If `-c Count` is specified and `-i Interval` is not specified, the operation repeats the specified number of iterations using the default interval.

Restrictions

- For All specified devices must be in the same SRDF/A session.

Examples

To confirm R2 data copy for device group `prod`:

```
symrdf -g prod checkpoint
```

To confirm the R2 data copy for devices in device group `Test` in RA group 7 on the second hop of a cascaded SRDF configuration:

```
symrdf -g Test -rdfg 7 -hop2 checkpoint
```

Managing Delta Set Extension

When many SRDF/A groups run on the same array, complex I/O profiles and link availability/bandwidth issues complicate the task of calculating cache requirements.

SRDF/A Delta Set Extension (DSE) extends the cache space available for SRDF/A session cycles by off loading cycle data from cache to preconfigured pool storage. DSE helps SRDF/A to ride through larger and longer throughput imbalances than cache-based buffering alone.

DSE is enabled by default on arrays running HYPERMAX OS, and disabled by default on arrays running Enginuity 5773 - 5876.

Note: DSE is not designed to solve permanent or persistent problems such as unbalanced/insufficient cache, host writes that consistently overrun cache, and long link outages.

When the SRDF/A session is activated, DSE is activated (on the R1 and R2 sides) if the autostart for DSE is set to enabled on both the R1 and the R2 sides. Autostart for DSE can be enabled/disabled, but the change does not take effect until the SRDF/A session is dropped and re-activated. By default, autostart for DSE is enabled regardless of whether the side is the R1 or R2 side.

DSE starts paging SRDF/A tracks to the DSE pool when the array write pending count crosses the DSE threshold (-threshold attribute). The default threshold is 50 percent of the System Write Pending Limit. After a cycle switch, Enginuity reads tracks from the DSE pool back into the array cache so that they can be transferred to the R2.

Enginuity 5733 -5876

Arrays running Enginuity 5773 - 5876, can share SRDF/A DSE pools among multiple SRDF/A groups. A single SRDF/A group can have up to 4 DSE pools associated with it (one for each device emulation type).

"Managing DSE pools (Enginuity 5773 - 5876)" on page 125 describes best practices, restrictions and procedures to manage DSE pools.

HYPERMAX OS

Arrays running HYPERMAX OS come preconfigured with one or more Storage Resource Pools (SRPs) containing all the storage available to the array. SRDF/A DSE allocations are made against one SRP per array designated as the SRP for DSE.

The SRP designated for DSE supports the DSE allocations for all SRDF/A sessions on the array.

The default SRP for DSE is the default SRP for FBA devices.

You can change which SRP is associated with DSE, and you can change the capacity of the SRP associated with DSE.

- "Managing the DSE SRP capacity (HYPERMAX OS)" on page 124 describes the procedures to manage the capacity of the DSE SRP.
- *EMC Solutions Enabler Array Management CLI User Guide* describes the steps to modify which SRP is associated with DSE.

Managing the DSE SRP capacity (HYPERMAX OS)

This section describes the steps to modify the capacity of the DSE SRP for arrays running HYPERMAX OS, including:

- "Restrictions" on page 124
- "About the DSE SRP" on page 124
- "Modify the DSE SRP capacity" on page 124

Restrictions

- CFGSYM access rights and Storage Admin authorization rights are required to run the symconfigure set command.
- If DSE requests for allocations exceed the maximum capacity of the DSE SRP, the SRDF/A session may drop.
- HYPERMAX OS does not support user defined DSE pools, and the following symrdf set commands are not supported:
  - symrdf set rdf_dse –fba_pool
  - symrdf set rdf_dse –ckd3390_pool
  - symrdf set rdf_dse –ckd3380_pool
  - symrdf set rdf_dse –as400_pool

About the DSE SRP

The default SRP associated with DSE is configured prior to installation. You can create another SRP for use with DSE, but only one SRP per array can be associated with DSE. All SRDF/A sessions on the array use the one SRP designated for use with DSE.

- If you enable SRDF/A DSE (rdfa_dse attribute) on another SRP, that SRP becomes the SRP for all DSE allocations.

  The SRP that was previously designated to support DSE is automatically modified not to support DSE (its rdfa_dse attribute is set to disabled).
- If you disable the rdfa_dse attribute on the DSE SRP without designating another SRP to support DSE, the default SRP for FBA emulation automatically becomes the DSE SRP.

Modify the DSE SRP capacity

Use the `symconfigure set symmetrix dse_max_cap` command to modify the capacity of the DSE SRP.

symconfigure set symmetrix dse_max_cap command syntax

```
symconfigure -sid SID commit -cmd "set symmetrix dse_max_cap =
MaxCap;"
```

Where:

- *`MaxCap`* specifies the maximum capacity in the array's DSE SRP. Valid values are:
  - 1 - 100000 - Specifies the maximum number of GB in the specified SRP that can be used by DSE.

- NoLimit - Specifies that DSE can use the entire capacity of the specified SRP.

Examples To set the maximum DSE capacity on SID 230 to a value of 100 GB:

```
symconfigure -sid 230 commit -cmd "set symmetrix dse_max_cap = 100;"

Execute a symconfigure operation for symmetrix '000197100230' (y/[n]) ? y

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

    Establishing a configuration change session...............Established.
    Processing symmetrix 000197100230
    {
          set symmetrix dse_max_cap=100;
    }

    Performing Access checks..................................Allowed.
     . . .
    Terminating the configuration change session..............Done.

The configuration change session has successfully completed.
```

To set the maximum DSE capacity on SID 230 to unlimited:

```
symconfigure -sid 230 commit -cmd "set symmetrix dse_max_cap = nolimit;"

Execute a symconfigure operation for symmetrix '000197100230' (y/[n]) ? y
.
.
.
The configuration change session has successfully completed.
```

Managing DSE pools (Enginuity 5773 - 5876)

This section describes DSE pool management for arrays running Enginuity 5773 - 5876, including:

- "DSE pool best practices"
- "Setting SRDF/A DSE attributes for an SRDF group"
- "Clearing existing DSE pool names"
- "Adding devices to an SRDF/A DSE pool"
- "Removing devices from an SRDF/A DSE pool"
- "Enabling/disabling devices in an SRDF/A DSE pool"
- "Associating an SRDF group with a DSE pool"
- "Displaying/monitoring SRDF/A DSE pool usage"

IMPORTANT

The procedures in this section are not applicable to VMAX 3 arrays running HYPERMAX OS. For the procedures applicable to VMAX 3 arrays, refer to "Managing the DSE SRP capacity (HYPERMAX OS)" on page 124

Restrictions

- A DSE pool cannot have the same name as a Snap pool on the same array.
- Each DSE pool can only contain one type of device emulation: FBA, CKD3390, CKD3380, or AS400.

- Each SRDF group can have at most one pool of each emulation.

DSE pool best practices

- Configure DSE pools on both R1 and R2 arrays.
- Plan for peak workloads.
- Spread the DSE pool devices across as many disks as possible.
- Ensure that sufficient DA and RA CPU resources are available for the DSE task.
- To simplify management and make the most efficient use of resources, use as small a number of DSE pools as possible.
- Configure DSE pools and enable DSE on the primary and on the secondary array. When TimeFinder/Snap sessions are used to replicate either R1 or R2 devices, create two separate preconfigured storage pools: DSE and Snap pools.
- Configure a separate DSE pool for each device emulation type (FBA, IBMi, CKD3380 or CKD3390). You can create multiple DSE pools for different SRDF/A groups.

Best Practices for EMC® SRDF®/A Delta Set Extension Technical Note provides more information.

Setting SRDF/A DSE attributes for an SRDF group

Use the `set rdfa_dse` command to set the SRDF/A DSE attributes for an SRDF group.

Note: The remote array must be reachable to complete this task.

symrdf set rdfa_dse command syntax

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
     [-noprompt] [-i Interval] [-c Count]

.............

set rdfa_dse
   [-autostart {on | off}]
   [-threshold 20 - 100]
   [-fba_pool PoolName]
   [-ckd3390_pool PoolName]
   [-ckd3380_pool PoolName]
   [-as400_pool PoolName>]
   [-both_sides]
```

Where:

- `-autostart` (`-aut`) specifies whether SRDF/A DSE is automatically enabled or disabled when an SRDF/A session is activated for an SRDF group.

  Valid values are on or off.

  Default is off.

- `-threshold` (`-thr`) specifies the percentage of the array's write pending limit. If cache usage of all active SRDF/A groups in the array exceeds this limit, data tracks for this SRDF group start to spill over to disks.

  Valid values 20 - 100.

Default is 50.

- `-fba_pool` (`-fba`) *`PoolName`* associates the pool *`PoolName`* containing SAVE devices with FBA emulation with the specified SRDF group.

  If the argument *`PoolName`* is not specified, the currently associated FBA pool is removed from the group.

- `-ckd3380_pool` (`-ckd3380`) *`PoolName`* associates the pool *`PoolName`* containing SAVE devices with CKD 3380 emulation with the specified SRDF group.

  If the argument *`PoolName`* is not specified, the currently associated CKD 3380 pool is removed from the group.

- `-ckd3390_pool` (`-ckd3380`) *`PoolName`* associates the pool *`PoolName`* containing SAVE devices with CKD 3390 emulation with the specified SRDF group.

  If the argument *`PoolName`* is not specified, the currently associated CKD 3390 pool is removed from the group.

- `-as400_pool` (`-as400`) *`PoolName`* associates the pool *`PoolName`* containing SAVE devices with an AS400 emulation with the specified SRDF group.

  If the argument *`PoolName`* is not specified, the currently associated AS400 pool is removed from the SRDF group.

- `-both_sides` sets the SRDF/A DSE attributes on both the source and target sides of an SRDF/A session.

  If `-both_sides` is not specified, attributes are only applied to the source side.

For arrays running Enginuity 5876, you can also use the `symconfigure` command to set these SRDF/A DSE attributes. See the *EMC Solutions Enabler Array Management CLI User Guide.*

Clearing existing DSE pool names

Use the `-_pool` commands with no *`PoolName`* argument to remove the association between the specified SRDF group and DSE pools.

Example To clear the DSE pool names for all 4 emulation types:

```
symrdf -sid 432 -rdfg 75 set rdfa_dse -fba_pool -ckd3390_pool -ckd3380_pool
   -as400_pool

An RDF Set 'Attributes' operation execution is in progress for
RDF group 75. Please wait...

    SRDF/A Set FBA Pool (0432,075)....................................Started.
    SRDF/A Set FBA Pool (0432,075)....................................Done.
    SRDF/A Set CKD3380 Pool (0432,075)................................Started.
    SRDF/A Set CKD3380 Pool (0432,075)................................Done.
    SRDF/A Set CKD3390 Pool (0432,075)................................Started.
    SRDF/A Set CKD3390 Pool (0432,075)................................Done.
    SRDF/A Set AS400 Pool (0432,075)..................................Started.
    SRDF/A Set AS400 Pool (0432,075)..................................Done.

The RDF "Attributes'' operation successfully executed for RDF group 75.
```

Adding devices to an SRDF/A DSE pool

Devices can be added to a DSE pool if they are:

- Disabled
- Inactive
- Do not belong to another pool

Command syntax

To add and enable SAVE devices to a DSE pool:

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

Example

```
add dev 018B:018C to pool finance,
type = rdfa_dse,
member_state=ENABLE;
```

Removing devices from an SRDF/A DSE pool

You can remove SAVE devices from an SRDF/A DSE pool, only if the devices are disabled and drained.

When you remove a device from a pool, it becomes available for use by other SAVE device pools.

The last device cannot be removed from an SRDF/A DSE pool if the pool is associated with an SRDF group.

Command syntax

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

Example

```
remove dev 018B from pool finance, type = rdfa_dse;
```

Enabling/disabling devices in an SRDF/A DSE pool

Devices in a DSE pool do not all have to be in the same state (enabled or disabled):

- If all the devices in a pool are disabled, the pool is disabled.
- If at least one device in a pool is enabled, the pool is enabled.

To enable or disable a range of devices, all the devices must be in the same pool.

You cannot disable all the devices in an SRDF/A DSE pool if the pool is currently associated with an SRDF group and SRDF/A DSE is active for the group.

Command syntax

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

Example

```
enable dev 018C in pool finance,
type = rdfa_dse
```

Associating an SRDF group with a DSE pool

SRDF/A DSE pools are created and managed in command files that are executed using the `symconfigure` command.

Procedure To set the SRDF/A DSE threshold, associate an SRDF group with a pool, and activate DSE:

1. Use the `symcfg list -sid SID -pools -rdfa_dse` command to list the configured DSE pools.
2. Create a text file containing the commands to set attributes for an SRDF group.

   The first command in the file must be to set the threshold.

   To set the threshold, associate with DSE pool `r1pool`, and specify FBA emulation, and enable autostart for SRDF group 7:

   ```
   set rdf group 7 rdfa_dse_threshold=20;
   set rdf group 7 rdfa_dse_pool=r1pool, emulation=fba;
   set rdf group 7 rdfa_dse_autostart=enable;
   ```

3. Use the `symconfigure commit` command to perform the operation:

   ```
   symconfigure commit -sid 12 -file setup_dse.cmd
   ```

EMC Solutions Enabler Array Management CLI User Guide provides more information about the `symconfigure` command.

Displaying/monitoring SRDF/A DSE pool usage

Use the `symcfg show` command to display the pool utilization for a specified SRDF/A DSE pool.

Command syntax

```
symcfg show [-sid SymmID] -pool PoolName -rdfa_dse
```

Example To display the utilization for DSE pool BC_DSE:

```
symcfg show -sid 03 -pool BC_DSE -rdfa_dse
```

```
Symmetrix ID: 000194901103

Symmetrix ID                    : 000194901103
Pool Name                       : BC_DSE
Pool Type                       : Rdfa DSE
Disk Location                   : Internal
Technology                      : FC
Dev Emulation                   : FBA
Dev Configuration               : 2-Way Mir
Pool State                      : Enabled
# of Devices in Pool            :   4
# of Enabled Devices in Pool    :   4
# of Related RDF Groups         :   1

Enabled Devices(4):
  {
   ----------------------------------------------------
   Sym          Usable        Free        Used Full Device
   Dev          Tracks      Tracks      Tracks  (%) State
   ----------------------------------------------------
   0228          17250        8682        8568   49 Enabled
   0229          17250       17062         188    1 Enabled
   022A          17250        8413        8837   51 Enabled
   022B          17250        8579        8671   50 Enabled
            ---------- ---------- ---------- ----
   Tracks        69000       42736       26264   38
```

Activating/deactivating SRDF/A DSE

There are several methods to activate SRDF/A DSE:

- Set the SRDF/A group parameter `rdfa_dse_autostart` to ENABLE.

SRDF/A DSE becomes active when the SRDF/A session is activated.

- Modify the SRDF/A DSE status for a device group, composite group, or file when the SRDF link status is Read Write,

  This activates or deactivates SRDF/A DSE for groups on both the R1 and R2 sides.

  Note: The SRDF links must be in asynchronous mode and SRDF/A must be active for activate or deactivate actions to succeed.

  Use the following commands to modify the device group, composite group, or file:

  ```
  symrdf [-g DgName | -cg CgName | -f FileName]
  activate | deactivate -rdfa_dse
  ```

- Modify the SRDF/A DSE status using RA group operations when the SRDF link status is Read Write.

  Use the following commands to modify the group:

  ```
  symrdf -sid SymmID -rdfg GrpNum [-v][-noprompt] [-i Interval] [-c
  Count]

  activate   -rdfa_dse  [-both_sides]
  deactivate  -rdfa_dse [-both_sides]
  ```

  The `-both_sides` option activates/deactivates SRDF/A DSE for groups on both the source and target sides. Otherwise, the activate/deactivate is only performed on the source side.

- Set the group mode to `sync` or `acp` when SRDF/A DSE is active for an SRDF group.

  This method does not require deactivating SRDF/A DSE.

  Deactivating SRDF/A in a group automatically deactivates SRDF/A DSE for that group.

Requirements to activate SRDF/A DSE with Dynamic Cache Partitioning

Requirements for activating SRDF/A DSE with dynamic cache partitioning include:

- All devices in the SRDF/A session must be in the same DCP.
- The `rdfa_dse_threshold` must be set, and must be lower than the `rdfa_cache_percentage` setting.
- The SRDF group must have at least one associated DSE pool with SAVE devices enabled.

Use the following syntax to activate SRDF/A DSE when dynamic cache partitioning is enabled:

```
symrdf type activate -rdfa_dse
```

Valid values for *type* are `-dg`, `-cg`, `-file`, or `-rdfg`.

Note: After activation, R1 and R2 cache usage is reported as a percent of DCP Write Pending Limit.

Managing transmit idle

Transmit idle allows an SRDF/A session to manage transient link outages without dropping. If transmit idle is not enabled, the SRDF/A session drops when the link cannot transmit data.

If transmit idle is enabled, a link failure starts the link limbo timer. If the link status is still Not Ready after the link limbo time expires, devices are marked Not Ready to the link. When the link recovers, the SRDF pair state returns to Consistent or SyncInProg.

See also "Link limbo" on page 81 and "Setting SRDF group attributes" on page 85.

Restrictions

When the SRDF pair is in the Transmit Idle state, only the following operations are allowed from the R1 side:

- rw_enable -r1
- write_disable -r1
- ready -r1
- not_ready -r1
- suspend -immediate

When the SRDF pair is in the Transmit Idle state, only the following operations are allowed from the R2 side:

- suspend -immediate
- failover -immediate

If at the beginning of a control action, all SRDF/A groups are not in the Transmit Idle state, the action fails if one of the groups enters the Transmit Idle state during processing.

symrdf set_rdfa command syntax

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
[-noprompt] [-i Interval] [-c Count]
.............

   set rdfa
      [-transmit_idle {on | off}]
      [-both_sides]
```

Example To enable transmit idle on both sides for SRDF/A group 12:

```
symrdf -sid 134 -rdfg 12 set rdfa -transmit_idle on -both_sides
```

Managing SRDF/A write pacing

SRDF/A write pacing extends the availability of SRDF/A by preventing conditions that result in cache overflow on both the R1 and R2 sides. Write pacing balances cache utilization by extending the host write I/O response time to prevent SRDF/A operational interruptions.

There are two types of write pacing:

- **Group-level pacing** is dynamically enabled for the entire SRDF/A group when slowdowns in host I/O rates, transmit cycle rates, or apply cycle rates occur.

  SRDF/A group-level write pacing monitors and responds to:

- Spikes in the host write I/O rates
- Slowdowns in data transmittal between R1 and R2
- R2 restore rates.

Group-level pacing controls the amount of cache used by SRDF/A. This prevents cache overflow on both the R1 and R2 sides, and helps the SRDF/A session to stay up and running.

Group-level pacing requires Enginuity 5874 or greater.

HYPERMAX OS

HYPERMAX OS introduces enhanced group-level pacing. Enhanced group-level pacing paces host I/Os to the DSE transfer rate for an SRDF/A session.

When DSE is activated for an SRDF/A session, host-issued write I/Os are throttled so their rate does not exceed the rate at which DSE can offload the SRDF/A session's cycle data.

Enhanced group-level pacing requires HYPERMAX OS on the R1 side. The R2 side can be running either HYPERMAX OS or Enginuity 5876.

Enhanced group-level pacing responds only to the spillover rate on the R1 side. It is not affected by spillover on the R2 side.

Enginuity 5773-5876

- **Device-level pacing** is for SRDF/A solutions in which the SRDF/A R2 devices participate in TimeFinder copy sessions.

  Note: device-level pacing is not supported in HYPERMAX OS.

  SRDF/A device-level write pacing addresses conditions that lead to cache overflow specifically due to TimeFinder/Snap and TimeFinder/Clone sessions on an R2 device running in asynchronous mode.

  Device-level write pacing requires Enginuity version 5875.135.91 or higher on both arrays.

You can enable/disable either or both write pacing options. Both write pacing options are compatible with each other and with other SRDF/A features including tunable cache utilization, Reserve Capacity, and MSC.

Enginuity version 5876.82.57 or higher includes a global write pacing statistics report.

You can activate and control group-level and device-level write pacing individually or simultaneously at the group, device group, composite group, or file level on the R1 side.

Each has an autostart capability that automatically activates write pacing whenever an SRDF/A session becomes active. If an SRDF group has both group-level and device-level pacing configured to autostart, both are activated when the SRDF/A session becomes active.

For information on using group-level and device-level write pacing, see "Write pacing operations" on page 133.

Requirements

- Group-level pacing is supported on Enginuity 5874.207.166 and higher.
  - The group-level exemption capability requires that the R1 side be running Enginuity 5875 and higher.

- Enhanced group-level write pacing requires that the R1 and R2 arrays be running Enginuity 5876 and higher.
- Device-level write pacing is supported on Enginuity 5875 and higher.
- The `activate` argument requires that the SRDF/A session be active and contain at least one participating device.

  This requirements does not apply to the autostart capability.

Write pacing operations

SRDF/A write pacing bases some of its actions on the following:

- R1 side cache usage
- Transfer rate of data from transmit delta set to receive delta set
- Restore rate on the R2 side

Starting with Enginuity 5876 SRDF/A group-level write pacing can respond to the following conditions:

- The write-pending level on an R2 device in an active SRDF/A session reaches the device's write-pending limit.
- The restore (apply) cycle time on the R2 side is longer than the capture cycle time.

The enhanced group-level write pacing feature can effectively pace host write I/Os in the following operational scenarios:

- Slower restore (apply) cycle times on specific R2 devices that are managed by slower-speed physical drives.
- FAST operations that lead to an imbalance in SRDF/A operations between the R1 and R2 sites.
- Sparing operations that lead to R2-side DAs becoming slower in overall restore operations.
- Production I/Os to the R2 side that lead to DAs and/or RAs becoming slower in restore operations.
- Restore delays during the pre-copy phase of TimeFinder/Clone sessions before activation.

The configuration and management of group-level write pacing are unaffected by this enhancement.

Write pacing considerations

Group level write pacing

- Only the group-level pacing values configured for the SRDF group on the R1 side of the SRDF/A session are used.
- In a cascaded SRDF environment:
  - With Enginuity 5876 and lower, group-level write pacing is only supported on the R1->R21 hop of the relationship.

- With Enginuity 5876 Q4 2012 SR and higher, group-level write pacing is supported on both the R1->R21 and R21->R2 hops of the relationship.

- In a concurrent SRDF/A environment, group-level pacing is supported on both mirrors of the concurrent R1. In this case, write pacing calculations are performed independently for the two SRDF/A sessions, and the host write I/Os sessions are subject to the greater of the two calculated delays.

Device-level write pacing

- Only the device-level pacing values configured for the SRDF group on the R1 side of the SRDF/A session are used.
- In a cascaded SRDF environment:
  - With Enginuity 5876 and lower, device-level write pacing is only supported on the R1->R21 hop of the relationship.
  - With Enginuity 5876 Q4 2012 SR and higher, device-level write pacing is supported on both the R1->R21 and R21->R2 hops of the relationship.
- There is no exemption from device-level pacing as there is for group-level pacing, and the R1 group-level exempt state does not affect device-level pacing.
- In a concurrent SRDF/A environment, device-level pacing is supported on both mirrors of the concurrent R1. In this case, write pacing calculations are performed independently for the two SRDF/A sessions, and the host write I/Os sessions are subject to the greater of the two calculated delays.
- If both group-level pacing and device-level pacing are active for an SRDF/A session, the group-level and device-level delays are calculated independently, and the greater calculated value is used for pacing. Note that as many as four different calculation results may be taken into account for a concurrent R1 device with both mirrors operating in asynchronous mode (group-level pacing for each mirror, device-level pacing for each mirror), using the greatest calculated delay in the calculation.

Identifying devices that cannot be paced in a cascaded SRDF configuration

A source device might not be paced because it has been set exempt from group-level write pacing or because it is not currently pace-capable.

- Exempt source devices (R1 or R21) have been excluded from group-level write pacing using the `-rdfa_wpace_exempt` option of the `symrdf` command. Exempt devices can be paced by device-level write pacing.
- R21 devices (in an R21>R2 pair) are not pace-capable if the corresponding R1>R21 SRDF pair is read/write (RW) on the SRDF link and operating in an adaptive copy mode. A device that is not pace-capable cannot be paced by device-level write pacing or group-level write pacing. The `-force` option is required for actions that will cause a device to become not pace-capable.

To identify devices that cannot be paced:

1. Use the `symcfg list` command with the `-rdfa` option to determine if the SRDF/A session includes devices that cannot be paced. This command provides the following information related to write pacing:
   - The state of write pacing (group-level and device-level) for the SRDF group

- Whether write pacing is currently activated and supported
- Whether write pacing is configured for autostart
- Whether there are devices in the SRDF/A session that might not be paced either because they have been set exempt from group-level write pacing or because they are not pace-capable.

  To view write pacing information for SRDF group 153:

  symcfg list -sid 1134 -rdfg 153 -rdfa

```
Symmetrix ID : 000195701134

                  S Y M M E T R I X   R D F A   G R O U P S

-------- ---------- -------- ----- --- --- --------- ------------------------
                                                        Write Pacing
 RA-Grp  Group      Flags    Cycle Pri Thr Transmit   Delay  Thr GRP DEV FLGS
         Name       CSRM TDA time          Idle Time (usecs) (%) SAU SAU P
-------- ---------- -------- ----- --- --- --------- ------- --- --- --- ----
153 (98) lc153142   .IS- XI.    15  33  50 000:00:00   50000  60 I.- I.- X
.
.
    (FLGS) Flags for Group-Level and Device-Level Pacing:
      Devs (P)aceable : X = All devices, . = Not all devices, - = N/A
```

An X in the FLGS P column indicates that all of the devices in the SRDF group can be paced. A period in the FLGS P column indicates that some of the devices in the SRDF group cannot be paced either because they have been set exempt from group-level write pacing or because they are not pace-capable.

2. Use the `symrdf list` command to determine which devices cannot be paced.

   a. Use the `symrdf list` command with the `-rdfa_wpace_exempt` option to identify devices that are exempt from group-level write pacing.

   b. Use the `symrdf list` command with the `-rdfa_not_pace_capable` option to identify devices participating in the SRDF/A session that are not pace-capable.

3. Use the `symdev show` command to obtain additional information about the devices identified in the previous step. This command provides the following information related to write pacing:

   - Whether the device is exempt from group-level write pacing
   - Whether write pacing is currently activated and supported
   - Whether the device is pace-capable

     To view write pacing information for device 00d1:

     symdev show -sid 230 00d1

```
.
.
.
Write Pacing Information
            {
            Pacing Capable                     : Yes
            Configured Group-level Exempt State: Disabled
            Effective Group-level Exempt State : Enabled
            Group-level Pacing State           : Enabled
            Device-level Pacing State          : Disabled
```

.
.
.

Setting SRDF/A group-level write pacing attributes

Use the `symrdf set rdfa_pace` command to set the SRDF/A write pacing attributes for an SRDF group.

symrdf set rdfa_pace command syntax

```
symrdf -sid SymmID -rdfg GrpNum [-v] [-symforce]
     [-noprompt] [-i Interval] [-c Count]

.............

   set rdfa_pace
       [-dp_autostart {on | off}]
       [-wp_autostart {on | off}]
       [-delay 1 - 1000000]
       [-threshold 1 - 99]>
       [-both_sides]
```

To set these group attributes, the remote side must be reachable.

Where:

- `-dp_autostart` (`-dp_aut`) specifies whether SRDF/A device-level pacing is automatically enabled or disabled when an SRDF/A session is activated or deactivated for an SRDF group.

  Valid state values are on or off.

  Default is off.

- `-wp_autostart` (`-wp_aut`) specifies whether the SRDF/A group-level pacing feature is automatically enabled or disabled when an SRDF/A session is activated for an SRDF group.

  Valid state values are on or off.

  Default is off.

- `-delay` (`-del`) sets the maximum host I/O delay, in microseconds, that the SRDF/A write pacing can cause.

  Valid values are 1 through 1000000 microseconds.

  The default is 50000 microseconds.

- `-threshold` (`-thr`) sets the minimum percentage of the array write-pending cache at which the array begins pacing host write I/Os for an SRDF group.

  Valid values are between 1 and 99.

  The default is 60.

- `-both_sides` sets the SRDF/A write pacing attributes on both the source and target sides of an SRDF/A session. Otherwise, these attributes are only set on the source side.

Note: If you plan on swapping the personalities of the R1 and R2 devices, configure the same SRDF/A write pacing values on both sides.

Examples In the following example, SRDF/A group-level write pacing is enabled for SRDF group 12 with:

- A maximum of a 1000 microsecond delay
- A write pending cache threshold of 55 percent

  If the calculated delay is less than the specified delay (1000), then the calculated delay is used.

```
symrdf -sid 134 -rdfg 12 set rdfa_pace -delay 1000 -threshold 55 -wp_autostart on
```

Use the `-both_sides`, option to display two entries for each attribute being applied; one for the source side and one for the target side:

```
symrdf -sid 432 -rdfg 75 set rdfa_pace -delay 500 -threshold 10 -wp_autostart on
   -dp_autostart on -both_sides
```

Activating write pacing

symrdf activate command syntax

To activate and deactivate SRDF/A write pacing at the device-group level:

```
symrdf -g DgName [-v | -noecho] [-force] [-symforce]

   activate [-rdfa_dse | -rdfa_pace | -rdfa_wpace | -rdfa_devpace] |
   -rdfa_wpace_exempt [LdevName [LevdevName....]]

   deactivate [-rdfa_dse | -rdfa_pace | -rdfa_wpace | -rdfa_devpace]|
   -rdfa_wpace_exempt [LdevName [ LevdevName....]]
```

Examples To activate group-level write pacing for SRDF group 76:

```
symrdf -sid 123 -rdfg 76 activate -rdfa_wpace
```

To exempt `DEV001` in the `prod` group from SRDF/A write pacing:

```
symrdf -g prod -rdfg 76 -rdfa_wpace_exempt DEV001
```

To deactivate device-level write pacing for `DEV012` in the `prod` device group:

```
symrdf -g prod deactivate -rdfa_devpace DEV012
```

Controlling simultaneous group-level and device-level write pacing

When write pacing is activated at both group-level and device-level, Enginuity monitors both the SRDF link performance of the SRDF/A session and the performance of the devices on the R2 side.

Requirements for controlling group and device-level write pacing

- The `symrdf activate/deactivate -rdfa_pace` commands are targeted at all devices in the SRDF group.
- The R1 array is accessible.

- The SRDF/A session being controlled is active and contains at least one participating device.
- The `symrdf deactivate -rdfa_pace` command requires the following:
  - The R2 array is accessible to verify that there are no TimeFinder/Snap or TimeFinder/Clone sessions off the R2 devices before deactivating device-level pacing.
  - If the SRDF/A session is in the transmit idle state, issue `symrdf deactivate -rdfa_pace -symforce` from the R1 side.

Examples To activate group-level and device-level write pacing simultaneously for the `ConsisGrp` CG:

```
symrdf -cg ConsisGrp activate -rdfa_pace
```

To exempt `DEV001` in the `prod` group from both group-level and device-level write pacing:

```
symrdf -g prod -sid 55 -rdfg 76 -rdfa_pace_exempt DEV001
```

To deactivate both group-level and device-level write pacing on the devices in `DeviceFile2`:

```
symrdf -file DeviceFile2 -sid 55 -rdfg 2 deactivate -rdfa_pace
```

Displaying SRDF/A

Note: Output of srdf list and query commands varies depending on whether SRDF/A is in multi-cycle mode (HYPERMAX OS) or legacy mode (Enginuity 5876).

Display SRDF/A group information

Use the `symdg show DgName` command to display SRDF/A session status information.

Use the `symrdf -g DgName query -rdfa` command to display SRDF/A group information.

SRDF/A-capable devices in an SRDF group are considered part of the SRDF/A session. The session status is displayed as active or inactive, as follows:

- Active indicates the SRDF/A mode is activated and that SRDF/A session data is currently being transmitted in operational cycles to the R2.
- Inactive indicates the SRDF/A devices are either Ready or Not Ready on the link and working in their basic mode (synchronous, semi-synchronous, or adaptive copy).

Note: If the links are suspended or a split operation is in process, SRDF/A is disabled and will show a session status of Inactive.

List SRDF/A- capable devices

Use the `symrdf list -rdfa` command to list SRDF/A-capable devices (R1, R2 and R21 devices) that are currently configured in SRDF groups.

Note: SRDF/A-capable does not mean the device is actually operating in asynchronous mode, only that it is capable of doing so. There is no command that lists devices that are actually operating in asynchronous mode.

The device type is shown as *R1* for SRDF/A-capable devices on the R1 and type *R2* for SRDF/A-capable devices on the R2.

The *R21* device type represents a cascaded SRDF device configuration.

CHAPTER 5
Consistency Group Operations

This chapter describes the following topics:

- Overview 142
- Creating an SRDF consistency group 145
- Enabling and disabling SRDF consistency protection 148
- Deleting an SRDF consistency group 154
- Suspending SRDF consistency protection 155
- Using the msc_cleanup command 156
- Modifying consistency groups 158
- Consistency groups with a parallel database 170
- Consistency groups with BCV access at the target site 171

Overview

SRDF consistency preserves the dependent-write consistency of devices within a group by monitoring data propagation from source devices to their corresponding target devices. If a source R1 device in the consistency group cannot propagate data to its corresponding R2 device, SRDF consistency suspends data propagation from *all* the R1 devices in the group.

SRDF consistency allows you to quickly recover from certain types of failures or physical disasters by retaining a consistent, DBMS-restartable copy of your database.

SRDF consistency group protection is available for SRDF/S and SRDF/A.

An SRDF *consistency group* is a composite group comprised of SRDF devices with consistency enabled.

The devices in the consistency group are configured to act in unison to maintain the integrity of a database when distributed across multiple arrays or across multiple devices within an array.

Domino mode also ensures consistency of a remote database.

Consistency protection using the SRDF daemon

The SRDF daemon (storrdfd) provides consistency protection for:

- SRDF/A Multi-Session Consistency (MSC) consistency groups in multi-array environments
- SRDF/S RDF-Enginuity Consistency Assist (ECA) consistency groups in multi-array environments
- Multiple SRDF groups within the same array
- For MSC consistency groups, the SRDF daemon performs cycle switching and cache recovery for all SRDF/A sessions within a consistency group, and manages the R1 -› R2 commits for SRDF/A sessions in multi-cycle mode.

  If a data flow interruption (such as a trip event) occurs, storrdfd:

  - Halts R1-›R2 data propagation
  - Analyzes the status of all SRDF/A sessions.
  - Either commits the last cycle of data to the R2 targets or discards it.
- For RDF-ECA consistency groups, storrdfd continuously polls SRDF/S sessions for data flow interruptions.

  If any R1 device is unable to propagate data to its R2 target, storrdfd:

  - Halts all R1-›R2 data flow within an RDF-ECA consistency group.

storrdfd ensures that you always have a consistent R2 copy of a database at the point in time in which a data interruption occurs.

Before you begin

Before storrdfd can monitor and manage a consistency group, you must:

- Create a composite group with SRDF consistency enabled (`-rdf_consistency` option)
- Enable the composite group (`symcg enable` command).

"Creating an SRDF consistency group" on page 145 provides more information.

Enabling the SRDF daemon

The storrdfd daemon is required for SRDF consistency group operations.

By default, the storrdfd daemon is disabled and must be enabled for all applications using the SYMAPI configuration database file and SRDF consistency protection.

Enable storrdfd using the following SYMAPI options file setting:

```
SYMAPI_USE_RDFD=ENABLE
```

Each host running the SRDF daemon must also be running the base daemon (storapid).

EMC Solutions Enabler CLI Command Reference explains common daemon tasks, including how to start and stop daemons.

Enabling the Group Naming Services daemon

The storrdfd daemon runs on each host for which SRDF consistency is required.

If the Group Naming Services (GNS) daemon is enabled, storrdfd relies on GNS to propagate updated CG definitions to all hosts locally attached to the same set of arrays.

If GNS is not enabled, you must manually recreate the updated CG definition on each one of these hosts.

Enable GNS on each host using the following SYMAPI options file setting:

```
SYMAPI_USE_GNS=ENABLE
```

IMPORTANT

When using GNS, enabling the gns_remote_mirror option in the daemon_options file will not mirror the CG if it includes any devices listed in "Mirroring exceptions" in the *EMC Solutions Enabler Array Management CLI User Guide*.

Redundant consistency protection

You can run two instances of the SRDF daemon simultaneously on separate control hosts to create redundant consistency protection for composite groups.

Simultaneous SRDF daemons perform independent monitoring and switching operations. If one fails, the other SRDF daemon takes it place, and completes all pending tasks (commit the last cycle to the target site).

Redundant SRDF daemons allow you to avoid service interruptions caused by:

- Performance bottlenecks on one of the control hosts
- Link failures of the redundant SRDF daemons
- Failure of one control hosts

Each control host must have a common view of the composite group being monitored. To give each control host a common view, do one of the following:

- Run the GNS daemon on each control hosts, as shown in Figure 13, or
- Manually define the composite group on all control hosts.

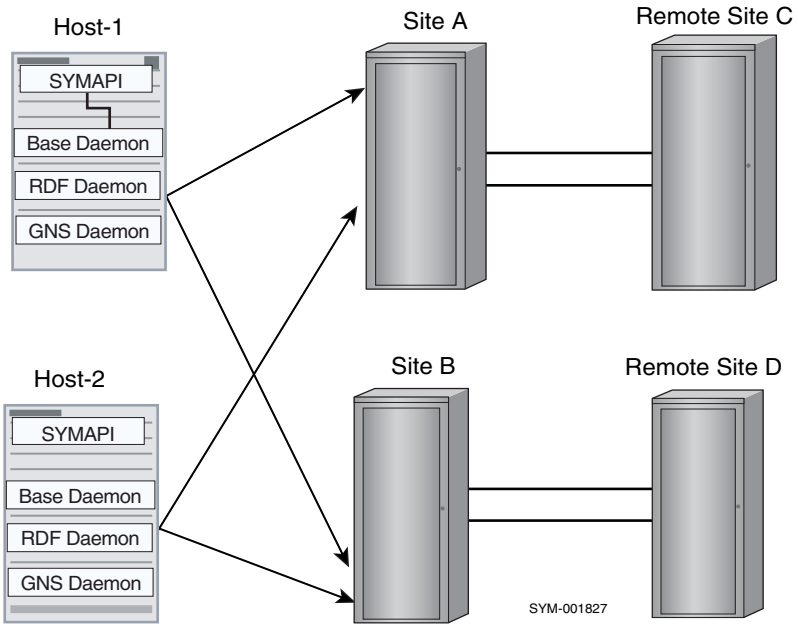


Figure 13 Running redundant hosts to ensure consistency protection

In Figure 13, Host-1 and Host-2 run all three daemons: base daemon, SRDF daemon, and GNS daemon to ensure data consistency protection

IMPORTANT

EMC strongly recommends running redundant SRDF daemons on at least two control hosts at each site. This ensures at least one SRDF daemon is available to perform time-critical, consistency monitoring operations.

EMC recommends that you do not run the SRDF daemon on the same control host running the database applications. Use this control host to issue other control commands (such as SRDF, TimeFinder, and Clone operations).

If the control host is powerful enough to efficiently handle all CPU operations, and is configured with sufficient gatekeeper devices for all your management applications, you can run ECC and Unisphere for VMAX with the Solutions Enabler daemons.

SRDF consistency group operations

SRDF composite groups are initially created using the `symcg create` command. Once they are created, they are populated with devices and device groups.

In order to be enabled as an SRDF consistency group, the composite group must be:

- Defined as a type RDF1, RDF2, or RDF21
- Have consistency enabled using the option`-rdf_consistency` option.

`symrdf` control operations can change a composite group. For example, a device personality swap operation can change an RDF1 CG to an RDF2 CG. SRDF control operations (`failover -establish` and `swap` operations) cannot change the type of an ANY composite group but can affect the devices in that CG.

"Consistency group operations and applicable pair states" on page 260 provides a list of control actions and the required SRDF pair states for consistency group operations.

Note: Enginuity 5773.150 and higher is required for thin devices in a composite group in SRDF/S and SRDF/A configurations.

Creating an SRDF consistency group

The following steps illustrate how to build a consistency group when devices in the group are either all synchronous or all asynchronous.

Note: All devices containing application and array data must be included in the consistency group for each DBMS or across the DBMS controlling the multi-database transactions.

1. Use the `symcfg list` command to list all SRDF (RA) groups on the source arrays connected to the local hosts to determine which devices to include in the CG:

   ```
   symcfg list -rdfg all
   ```

2. Use the `symcg create` command to create a consistency group (`ConsisGrp)` on one of the local hosts.

   Specify the SRDF type of the group and the `-rdf_consistency` option:

   ```
   symcg create ConsisGrp -type rdf1 -rdf_consistency
   ```

3. Use the `symcg addall` command to add the devices from an SRDF (RA) group, such as `RDG 64`, into the consistency group (`ConsisGrp)`:

   ```
   symcg -cg ConsisGrp -sid 3264 addall dev -rdfg 64
   ```

4. In a database configuration with multiple local hosts, you must build the same consistency group on all local hosts in the configuration.

   You can use the `symcg export` command to manually transfer the consistency group definition, or if enabled, use GNS to automatically transfer it.

   The following commands create the `consisgrp.txt` text file containing the new `ConsisGrp` composite group definition and then transfer it to `Host-1`:

```
symcg export ConsisGrp -f consisgrp.txt
rcp consisgrp.txt Host-1:/.
```

In the following command, the `-rdf_consistency` option adds the imported `ConsisGrp` definition to the SRDF consistency database on `Host-1`:

```
symcg import ConsisGrp -f consisgrp.txt -rdf_consistency
```

5. Verify that all devices in the group are either all synchronous or all asynchronous.

```
symrdf -cg ConsisGrp verify -async
```

6. If the devices are currently operating with synchronous replication and you want them to be operating asynchronously, set the composite group for asynchronous replication:

```
symrdf -cg ConsisGrp set mode async
```

7. If the SRDF pairs are not in the Consistent or Synchronized state at this time (the Split or Suspended state), you can use the `symrdf establish` command to initiate SRDF copying of R1 data to the R2 side.

```
symrdf -cg ConsisGrp establish
```

The device state is SyncInProg until the Consistent or Synchronized state is reached.

With asynchronous replication, it may take two cycle switches for all devices to reach the Consistent state.

In multi-cycle mode, if either the link is or destaging the R2Apply cycle is slow, it may take more than 2 cycle switches for all devices to reach Consistent state.

8. From one of the local hosts, use the `symcg enable` command to enable the composite group for consistency protection:

```
symcg -cg ConsisGrp enable
```

The `ConsistGrp` CG becomes an SRDF consistency group managed by the SRDF daemon. The SRDF daemon watches for any problems with R1->R2 data within the `ConsisGrp` CG.

Creating composite groups from various sources

Topics in this section include:

- "Creating a composite group from an existing device group"
- "Creating a composite group from an RDBMS database"
- "Creating a composite group from a logical volume group"

Note: The E-Lab™ Interoperability Navigator at http://elabnavigator.EMC.com provides detailed interoperability information.

Creating a composite group from an existing device group

You can use the `symdg` command with the `-rdf_consistency` option to translate the devices of an existing device group to a new or existing composite group.

In the following example the `symdg` command:

- Translates devices to SRDF
- Adds all devices from a device group Symm64DevGrp to a composite group `ConsisGrp`.
- Adds the composite group to the SRDF consistency database on the host
- Enables the group for SRDF consistency protection:

```
symdg dg2cg Symm64DevGrp ConsisGrp -rdf_consistency
```

Creating a composite group from an RDBMS database

You can use the `export` command to translate the devices of an existing RDBMS database or tablespace to a new or existing composite group.

Note: For SYMCLI to access a specified database, you must set the `SYMCLI_RDB_CONNECT` environment variable to the username and password of the array administrator's account.

In the following example, a local connect is used, and the `export` command sets the variable to a username of "array" and a password of "manager".

```
export SYMCLI_RDB_CONNECT=array/manager
```

Note: The Bourne and Korn shells use the `export` command to set environment variables. The C shell uses the `setenv` command.

Connecting by network

When connecting by the network, you need to add a database-specific variable to the `RDB_CONNECT` definition.

When connecting through the network in an Oracle environment, Oracle has a network listener process running.

You must add an Oracle connection string such as the Transparent Network Substrate (TNS).

Examples

In the following example, the `export` command adds the TNS alias name "api217":

```
export SYMCLI_RDB_CONNECT=array/manager@api217
```

When connecting through the network in an SQL Server 2000 environment you must add a string to indicate the ODBC data source administrator.

In the following example, the `export` command adds string "HR":

```
set SYMCLI_RDB_CONNECT=array/manager@HR
```

Optionally, you can set the `SYMCLI_RDB_TYPE` environmental variable to a specific type of database (`oracle`, `informix`, `sqlserver`, or `ibmudb`) so that you do not have to include the `-type` option on the `symrdb rdb2cg` command line.

In the following example, the `export` command sets this variable to `oracle`:

```
export SYMCLI_RDB_TYPE=oracle
```

Translating devices

You can translate the devices in a database to a composite group.

Examples In the following example, the `symrdb rdb2cg` command:

- Translates the devices of an Oracle-type database named oradb to an RDF1 type composite group named `ConsisGrpDb`.
- The `-rdf_consistency` option adds the composite group to the SRDF consistency database on the host:

```
symrdb -type oracle -db oradb rdb2cg ConsisGrpDb -cgtype rdf1 -rdf_consistency
```

You can translate the devices in an Oracle type tablespace to a composite group.

In the following example, the `symrdb tbs2cg` command translates the devices of an oracle type tablespace orats to an RDF1 type composite group named `ConsisGrpTs`:

```
symrdb -type oracle -tbs orats tbs2cg ConsisGrpTs -cgtype rdf1 -rdf_consistency
```

With most RDBMS database arrays, you must set up environment variables specific to that array.

Oracle arrays use `ORACLE_HOME` and `ORACLE_SID`.

Sybase arrays use `SYBASE` and `DSQUERY`.

Creating a composite group from a logical volume group

You can use the `symvg` command to translate the devices of an existing logical volume group to a new or existing composite group. This command does not require environment variables.

Example In the following example, the `symvg` command:

- Translates the devices of a logical volume group named LVM4vg to an RDF1 type composite group named `ConsisGrp`.
- The `-rdf_consistency` option adds the composite group to the SRDF consistency database on the host:

```
symvg vg2cg LVM4vg ConsisGrp -cgtype rdf1 -rdf_consistency
```

Enabling and disabling SRDF consistency protection

You can enable or disable consistency protection for all the devices in a composite group. When you enable the composite group for consistency, the group is referred to as an SRDF consistency group.

Restrictions

- You can have either consistency protection or the domino effect mode enabled for a device, but not both.
- When a composite group is enabled for consistency protection:
  - Its name cannot be changed without first disabling the consistency protection. After the name change, re-enable the composite group using the new name.

- If the composite group is enabled for SRDF/A consistency protection, the SRDF daemon immediately begins cycle switches on the SRDF groups within the composite group (or named subset in MSC mode).

  The cycle switches for all SRDF groups will be performed at the same time. The interval between these cycle switches is determined by the smallest minimum cycle time defined on the R1 SRDF groups in the composite group (or named subset).

  The smallest minimum cycle time supported by the SRDF daemon is 3 seconds. This value is used if the smallest minimum cycle time across all component groups is less than 3 seconds.

◆ If you change the minimum cycle time for any of the R1 SRDF groups while the composite group (or named subset) is enabled for SRDF/A consistency protection, the new minimum cycle time will not take effect until you disable consistency protection and then re-enable it.

◆ If you change the minimum cycle time, the new minimum cycle time will not take effect until you disable consistency protection and then re-enable it.

◆ You can change contents of a composite group by doing one of the following:

  - Disable consistency protection on a composite group while you add or remove devices, and then re-enable consistency protection after editing the composite group.

    Devices in the composite group are unprotected during the time required to edit and then re-enable the composite group.

  - For RDF1 composite groups, you can dynamically modify the composite group while maintaining consistency protection during the editing process.

    "Modifying consistency groups" on page 158 provides more information.

Enabling consistency: composite group vs. SRDF group name

You can enable and disable consistency protection at the composite group level or at the SRDF group name level.

Composite group level

When consistency is enabled at the composite group level, all devices within the consistency group operate as a single unit.

If one R1 device in a CG is unable to propagate data to its R2 target, the SRDF links of all the devices within that CG are suspended.

To enable consistency protection at the composite group level, all device mirrors must be operating in the same SRDF mode: all device mirrors must be operating either synchronously or asynchronously.

Use the `symcg enable` and `symcg disable` commands to enable/disable consistency protection at the composite group level. All device pairs in the specified group are enabled/disabled.

Examples To enable consistency protection for all device pairs in composite group `prod` CG:

```
symcg -cg prod enable
```

To disable consistency protection for all device pairs in `prod` CG:

```
symcg -cg prod disable
```

SRDF group name level

When consistency protection is enabled at the SRDF group level, only the devices in the specified SRDF group operate as a unit.

If an R1 device in a CG cannot send data to its R2 target, the SRDF links for only those devices in the specified SRDF group of the CG are suspended.

SRDF group protection is useful for concurrent devices with one mirror operating in synchronous mode and the other mirror operating in asynchronous mode.

To enable consistency protection at the SRDF group name level, you must first define one or more named subsets of devices within the composite group.

A subset can consist of one or more of the SRDF groups within the composite group.

Restrictions

When a subset of a CG is enabled for consistency protection at the SRDF group name level:

- You must disable consistency protection on the subset before you can:
  - Change the name of the subset.
  - Add or remove SRDF groups to the subset.

    Note: For an RDF1 composite group, you can dynamically modify the contents of a subset while consistency protection is enabled. "Modifying consistency groups" on page 158 provides more information.

- You cannot enable a composite group at the CG level and a member SRDF group at the same time.
  - If a composite group is enabled at the CG level, no part of it can be simultaneously enabled at the SRDF group name level.
  - If a subset of the group is enabled at the SRDF group name level, the group cannot be enabled at the CG level.

Example In the following example, composite group SALES consists of a set of concurrent SRDF devices distributed across two arrays, 076 and 077.

On array 076:

- SRDF group 100 operates in asynchronous mode, and
- SRDF group 120 operates in synchronous mode.

On array 077:

- SRDF group 101 operates in asynchronous mode, and
- SRDF group 121 operates in synchronous mode.

The following commands create two named subsets of the composite group:

- One containing the asynchronous SRDF groups:

```
symcg -cg SALES set -name sales1 -rdfg 76:100
symcg -cg SALES set -name sales1 -rdfg 77:101
```

- One containing the synchronous SRDF groups:

```
symcg -cg SALES set -name sales2 -rdfg 76:120
symcg -cg SALES set -name sales2 -rdfg 77:121
```

The following commands enable independent consistency protection for the two subsets:

```
symcg -cg SALES enable -rdfg name:sales1
symcg -cg SALES enable -rdfg name:sales2
```

SRDF/S devices

The `enable` action enables consistency protection either:

- Across all synchronous-mode devices in a consistency group, or
- Across all synchronous-mode devices in a named subset of a composite group.

If any R1 devices in an SRDF/S consistency group cannot propagate data to their corresponding R2 targets, the SRDF daemon suspends data propagation from all R1 devices in the consistency group, halting all data flow to the R2 targets.

Examples

To enable consistency protection for SRDF/S pairs in the `prod` CG:

```
symcg -cg prod enable
```

To disable consistency protection for SRDF/S pairs in the `prod` CG:

```
symcg -cg prod disable
```

SRDF/A devices

The `enable` action enables consistency protection either:

- Across all asynchronous-mode devices in a consistency group, or
- Across all asynchronous-mode devices in a named subset of a composite group.

If an SRDF/A session that was enabled for consistency protection cannot propagate data from the R1 devices to their corresponding R2 target, Enginuity deactivates that session, suspending data propagation for all devices in the SRDF/A session and preserving R2 consistency.

If the consistency group or named subset of a composite group is comprised of multiple SRDF/A sessions, the SRDF daemon suspends data propagation for the other SRDF/A sessions, halting all data flow to the R2 targets in order to preserve R2 consistency.

Examples

To enable consistency protection for SRDF/A pairs in the `prod2` CG:

```
symcg -cg prod2 enable
```

To disable consistency protection for SRDF/A pairs in the `prod2` CG:

```
symcg -cg prod2 disable
```

Enabling SRDF consistency protection for concurrent SRDF devices

You can enable and disable consistency protection for concurrent devices at the composite group level or at the SRDF group name level.

Composite group level

To enable consistency protection for concurrent devices at the composite group level, all device mirrors must be operating in the same SRDF mode: all device mirrors must be operating either synchronously or asynchronously.

If the two groups are operating in asynchronous mode, they cycle-switch together.

In either asynchronous or synchronous mode, the SRDF daemon suspends the SRDF links for both groups if a concurrent R1 device is unable to propagate its data to either of its remote R2 partners. This preserves the consistency of R2 data.

Use the `symcg enable` and `symcg disable` commands to enable/disable consistency protection at the composite group level. All device pairs in the specified group are enabled/disabled.

If the concurrent mirrors are in asynchronous mode, the `enable` command enables consistency with MSC consistency protection.

If the concurrent mirrors are in synchronous mode, the `enable` command enables consistency with RDF-ECA consistency protection.

Examples

In the following example, composite group `prod` contains a concurrent R1 with two asynchronous target mirrors.

To enable consistency protection with MSC consistency protection for the two target mirrors:

```
symcg -cg prod enable
```

To disable consistency protection for all device pairs in `prod` CG:

```
symcg -cg prod disable
```

Group name level

When consistency is enabled at the SRDF group name level, the SRDF daemon monitors the SRDF groups separately.

If a concurrent R1 device is unable to propagate its data to one of its remote R2 partners, the daemon suspends the SRDF links for only the group representing that R2 mirror.

Restrictions

- If the two mirrors of the concurrent R1 devices in the composite group are operating in different modes (one mirror in synchronous mode and the other mirror in asynchronous mode), SRDF consistency protection cannot be enabled at the composite group level.

  You must individually enable each group representing the device mirrors by its group name.
- Enginuity 5874 or higher is required to enable RDF-ECA for both mirrors of a concurrent R1.
- Enginuity 5875 or higher is required to enable MSC for both mirrors of a concurrent R1.

- Table 13 lists the combinations of consistency protection modes allowed for the mirrors of a concurrent relationship.

Table 13 Consistency modes for concurrent mirrors

R1->R2 (first mirror)	R1->R2 (second mirror)
MSC	None
MSC	RDF-ECA
MSC	MSC
RDF-ECA	None
RDF-ECA	RDF-ECA
RDF-ECA	MSC
None	None
None	MSC
None	RDF-ECA

Example

1. Use the `symcg` command to define the group name to associate with the SRDF group number.

   In the following example, the name `cGrpA` is associated with SRDF group 55 on array 123:

   ```
   symcg -cg prod set -name cGrpA -rdfg 123:55
   ```

2. Use the `symcg` command to enable consistency protection for the SRDF group.

   In the following example, the name `cGrpA` is associated with SRDF group 55 on array 123:

   ```
   symcg -cg prod enable -rdfg name:cGrpA
   ```

   - If the mirrors in SRDF group 55 are operating in asynchronous mode, the SRDF group is enabled with MSC consistency protection.
   - If the mirrors in SRDF group 55 are operating in synchronous mode, the SRDF group is enabled with RDF-ECA protection.

3. Repeat the steps above to enable consistency protection for the second concurrent SRDF group

   Use a unique name for the second group.

Checking if device pairs are enabled for consistency protection

Use the `symrdf verify -enabled` command to validate whether device pairs are enabled for consistency protection.

"ping command" on page 34 provides more information.

Example

To verify whether the device pairs in the STAGING group are enabled for consistency protection:

```
symrdf -g STAGING verify -enabled
```

If none of the device pairs in the STAGING group are enabled for consistency protection, the following message displays:

```
None of the devices in the group 'STAGING' are 'Enabled'.
```

If all devices in the STAGING group were enabled for consistency protection, the following message displays:

```
All devices in the group 'STAGING' are 'Enabled'.
```

Use the `symrdf verify -enabled -synchronized -consistent` command to verify whether the device pairs are enabled for consistency protection and are in the synchronized OR consistent pair state.

Example To verify whether the device pairs in the STAGING group are enabled for consistency protection and are in the synchronized or consistent pair state:

```
symrdf -g STAGING verify -enabled -synchronized -consistent
```

If all devices are enabled and in the synchronized OR consistent pair state, the following message displays:

```
"All devices in the group 'STAGING' are 'Enabled' and in
'Synchronized, Consistent' states."
'Synchronized, Consistent' states."Blocking symcg enable on R2 side
```

Blocking symcg enable on R2 side

You can execute the `symcg enable` command from the R1 or R2 side of an SRDF relationship.

The `SYMAPI_ALLOW_CG_ENABLE_FROM_R2` in the options file allows you to prevent the `symcg enable` operation from being executed on the R2 side.

The default for `SYMAPI_ALLOW_CG_ENABLE_FROM_R2` is enabled. When enabled, this option allows the SDRF daemon running on the R2 side to close the RDF-ECA window due to a link failure, even though the failure prevents the R2 side from communicating with the R1 side.

This option can be set as:

- `ENABLE` – (Default) Allows the composite group to be enabled on the R2 side.
- `DISABLE` - Blocks the composite group from being enabled on the R2 side.

Deleting an SRDF consistency group

When you delete an SRDF consistency group from a CG, the SRDF daemon stops monitoring the CG.

⚠CAUTION

After deletion, SRDF consistency protection on the R2 data cannot be guaranteed even though the devices formerly in the CG may remain enabled.

Best practice is to disable consistency protection before deleting a group. "Enabling and disabling SRDF consistency protection" on page 148 provides more information.

The `-force` option is required if the group is disabled and there are members in the group.

The `-symforce` option is required if the group is enabled. The composite group remains enabled but is removed from the SYMAPI database.

Examples To delete a disabled SRDF consistency group `mycg1` (with members):

```
symcg delete mycg1 -force
```

Suspending SRDF consistency protection

You can use the `suspend`, `split` or `failover` commands to suspend consistency protection for all devices in an SRDF consistency group where all devices are either synchronous or asynchronous.

The state of the R2 devices at the end of the deactivation varies depending on whether the `suspend` or `split` command is used:

- `symrdf -cg suspend` - The R2 devices are in the write disabled state and cannot be accessed by the target-side hosts.

  R2 database copy is consistent with the production copy on the R1 side.

- `symrdf -cg split` - The R2 devices are enabled for both reads and writes by the target-side hosts.

The `-force` option is required:

When the same consistency group is defined on multiple hosts, you can initiate a suspend operation from any host provided the consistency group is enabled.

Note: If you execute the `failover` command on both mirrors of a concurrent R1 device, the concurrent R1 is converted into a concurrent R2 with a restore on both mirrors of the concurrent R2.

Examples To deactivate consistency in a consistency group named `ConsisGrp`:

```
symrdf -cg ConsisGrp suspend -force
```

To resume the SRDF links between the SRDF pairs in the SRDF consistency group and I/O traffic between the R1 devices and their paired R2 devices:

```
symrdf -cg ConsisGrp resume
```

Consistency protection is automatically restored upon resumption of the link.

Consistency protection is not disabled unless you specify `symcg -cg disable`.

For asynchronous replication, use the `symrdf -cg verify` command with the `-cg_consistent` option to ensure that the SRDF consistency group is SRDF-consistency enabled and in a consistent state.

A consistent state means that at least two cycle switches have occurred and all devices in each SRDF (RA) group have reached a consistent state.

Verify consistency

Example To verify that the SRDF consistency group ConsisGroup is SRDF-consistency enabled and in a consistent state.

```
symrdf -cg ConsisGrp verify -cg_consistent
```

(For synchronous operations) To verify if the device pairs in ConsisGroup are in Synchronized state:

```
symrdf -cg ConsisGrp verify -synchronized
```

Using the msc_cleanup command

When an SRDF/A *single* mode session is dropped, Enginuity automatically starts a cleanup process:

- The primary array marks new incoming writes as being owed to the secondary array.
- The capture and transmit delta sets are discarded, but the data is marked as being owed to the secondary array. All of these owed tracks are sent to the secondary array once SRDF is resumed, as long as the copy direction remains primary to secondary.
- The secondary array marks and discards the receive delta set only. Data is marked as tracks owed to the primary array.
- The secondary array makes sure the apply (N-2) delta set is safely applied to disk; this is the dependent-write consistent image.

When a SRDF/A *multiple* mode session with Multi-Session Consistency (MSC) is dropped, MSC cleanup operations either:

- Discards any incomplete SRDF/A data, or
- Commits completed data to the R2 to maintain dependent write consistency.

When a SRDF/A *multiple* mode session with MSC is dropped, additional cleanup is required in fault scenarios where all delta sets of a transition have not been fully applied or discarded.

Use the `msc_cleanup` command to cleanup after a session is dropped for devices operating in SRDF/A mode with consistency enabled MSC.

For example, if a link failure causes protection to be triggered, the daemon may not be able to process all cleanup operations for the R2 devices where the receive and apply delta sets reside. Run the `symrdf msc_cleanup` command manually from the R2 site. If no consistency group definition is available at the R2 site, direct the cleanup operation to an SRDF (RA) group that was included as part of the consistency group.

The command can be executed by composite group from the R1 or R2 site or by SRDF group from the R2 site.

Use the `symcfg list` command check whether a MSC cleanup operation is required.

Use the `symcfg list` command with the `-rdfg all` option to display whether a MSC cleanup operation is required for only SRDF (RA) groups on the specified array.

Output of the `symcfg list` command includes flag information for SRDF groups operating in SRDF/A mode. An X in the RDFA Flags “M" column denotes that an MSC cleanup operation is required.

Examples To cleanup a composite group (`mycg`):

```
symrdf -cg mycg msc_cleanup
```

To cleanup from the remote host at the R2 site for array 1123 and direct the command to SRDF group 4:

```
symrdf -sid 123 -rdfg 4 msc_cleanup
```

Modifying consistency groups

You can dynamically add or remove the following device types for an RDF1 consistency group without first disabling consistency protection:

- Simple R1
- Concurrent R11

Use the `symcg modify` command with the `add` and `remove` options to modify SRDF consistency groups.

Before you begin

Before you begin, you must understand how the SRDF daemon maintains consistency protection during dynamic modification:

- On the local host, the SRDF daemon continuously monitors the consistency group being changed.

  The SRDF daemon must be running locally on the host where the `symcg modify` command is issued.

- On other hosts, the SRDF daemons do the following:
  - On hosts running GNS - SRDF daemons monitor the consistency group as it is being modified as long as these hosts are locally attached to the same set of arrays as the control host.

    Depending on the timing of the GNS updates, there may be a brief period during which the SRDF daemon stops monitoring the consistency group while waiting for the updated consistency group definition to propagate to the local GNS daemon.

  - On hosts *not* running GNS - If the SRDF daemons are running Solutions Enabler versions lower than 7.3.1, the daemons stop monitoring the CG during dynamic modification. These older daemons see the old CG definition until the `symstar buildcg -update` command is issued.

    IMPORTANT

    EMC strongly recommends running GNS on your hosts to ensure consistency protection while dynamically modifying CGs.

    To restart monitoring the modified consistency group on these hosts, perform the following steps after you complete the modification operation:

    1. Use the `symcg export` operation on the control host to export the new definition to a file.
    2. Send the file to the other host.
    3. Use the `symcg import` operation on the control host to import the new definition file.

Restrictions

The following apply to dynamic `add` and `remove` options of the `symcg modify` command:

- Enginuity 5773 or higher is required on all arrays of the CG.

- All arrays are reachable.
- The SRDF daemon must be running locally on the host where the `symcg modify` command is issued.
- The `symcg modify` command only applies to RDF1 composite groups.

  It is not allowed for RDF2, RDF21, or type=ANY composite groups.
- The `symcg modify` command is not allowed for:
  - CGs consisting of device groups.
  - CGs containing concurrent SRDF devices.
  - Any devices in SRDF/Star mode.

    Use the `symstar modifycg` command to modify devices in the CG are in STAR mode. "SRDF/Star consistency group operations" on page 249 provides more information.
- The SRDF groups affected by the `symcg modify` command cannot contain any devices enabled for consistency protection by another CG.
- Devices within SRDF groups of the CG to be modified must be in one of the following SRDF pair states:
  - Synchronized
  - SyncInProg with invalid tracks owed to the R2
  - Consistent with no invalid tracks
  - Within an affected SRDF group, device pairs can be a mixture of Synchronized and SyncInProg or a mixture of Consistent and SyncInProg.

Note: If the `symcg modify` command fails, you can rerun the command or issue `symcg modify -recover`. No control operations are allowed on a CG until after a recover completes on that CG.

Preparing the staging area

Before you can dynamically modify SRDF consistency groups, you must create a staging area that mirrors the configuration of the CG. The staging area consists of:

- SRDF groups containing the device pairs to be added to a consistency group (`symcg modify -add` operations),
- SRDF groups for receiving the device pairs removed from a consistency group (`symcg modify -remove` operations).
- The SRDF groups in the staging area must be established between the same arrays as the SRDF groups in the consistency group.

  For concurrent CGs, the SRDF groups in the staging area must be established among three arrays.

Restrictions: SRDF groups and devices in the staging area

- SRDF groups cannot be part of an SRDF/Star configuration.

- Devices cannot be enabled for consistency protection.
- Devices cannot be defined with SRDF/Star SDDF (Symmetrix Differential Data Facility) sessions.
- BCVs are not allowed.
- All devices must be SRDF dynamic and of the same type:
  - Simple R1 devices
  - Concurrent R11 devices
- All device pairs must set in the same mode:
  - Adaptive copy disk
  - Adaptive copy write pending for diskless R21->R2 device pairs

  Note: Adaptive copy write pending mode (acp_wp) is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS, and diskless R21 devices are not supported on arrays running HYPERMAX OS.

Restrictions: SRDF groups and devices for dynamic add operations

The dynamic modify add operation moves device pairs from the staging area into the SRDF groups of a consistency group.

All devices in the staging area must be in one of the following SRDF pair states for each SRDF group:

- Synchronized
- SyncInProg with invalid tracks owed to the R2
- Suspended
- Suspended with invalid tracks owed to the R2

If any device pair is Suspended (with or without invalid tracks on any of its SRDF groups), then the device pairs in the same SRDF group must all be Suspended.

Figure 14 shows a staging area for an R1->R2 configuration:

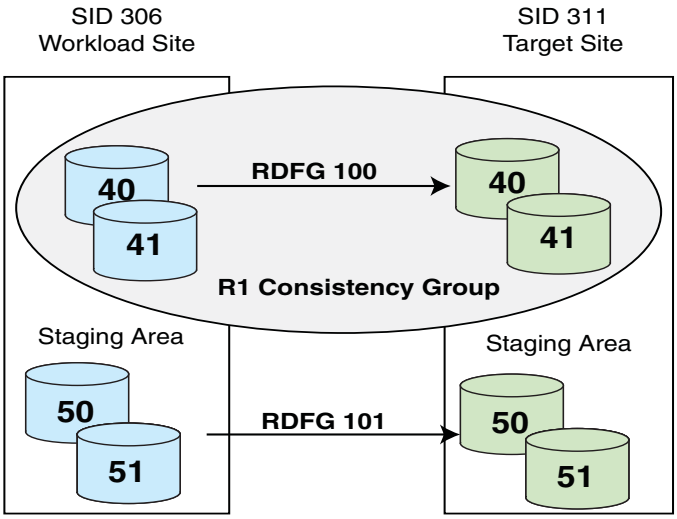


Figure 14 Staging area for adding devices to the R1CG consistency group

RDFG 101 is established between the same array as the RDFG 100 in the R1CG consistency group.

Figure 15 shows the R1CG consistency group after the dynamic add operation:.

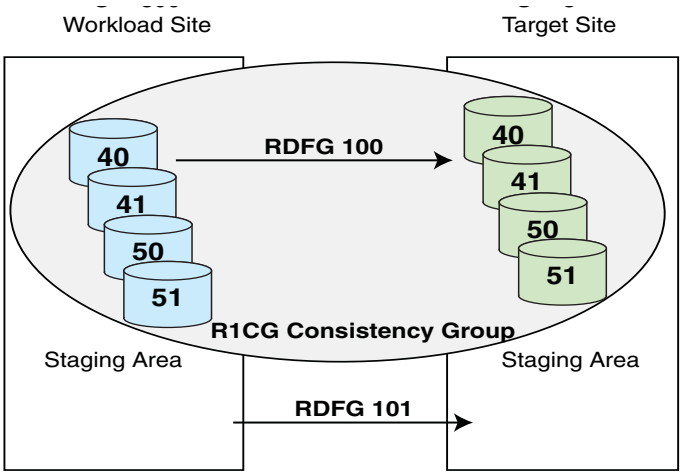


Figure 15 R1CG consistency group after a dynamic modify add operation

Devices 50 and 51 were moved to R1CG.

The staging area contains the empty RDFG 101.

Dynamic remove operation

The dynamic modify remove operation moves the device pairs from the consistency group into the SRDF groups in the staging areas.

To prepare the staging area for this operation, create the SRDF groups for receiving the device pairs removed from a consistency group.

Note: The dynamic modify remove operation must never leave an SRDF group empty.

Example Figure 16 shows empty group RDFG 34 configured to receive devices removed from RDFG 32:

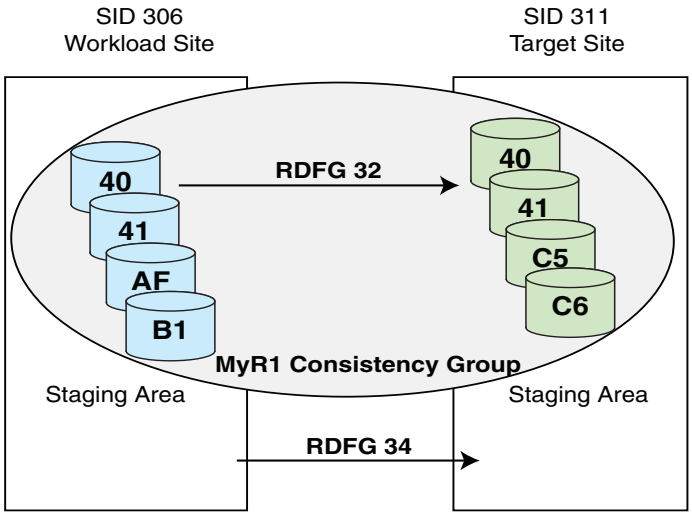


Figure 16 Preparing the staging area for removing devices from the MyR1 CG

The staging area consists of RDFG 34, an R1->R2 configuration established between the same array as RDFG 32 in the MyR1 consistency group.

Figure 17 shows the MyR1 consistency group and its staging area after the dynamic modify remove operation has completed.

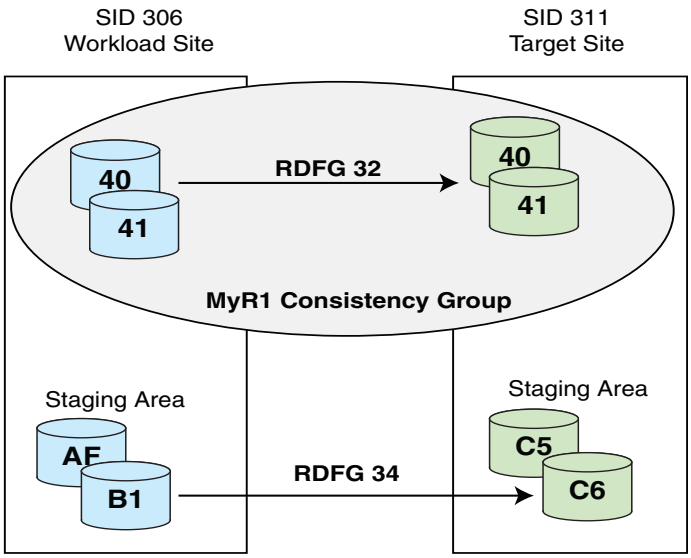


Figure 17 MyR1 CG after a dynamic modify remove operation

Restrictions - add operations

The following are restrictions for dynamically adding devices to an SRDF consistency group using the `symcg modify -add` command:

- The `symcg modify -add` command:
  - Cannot add new SRDF groups to the CG.
  - Cannot add a concurrent R11 device to a CG enabled at the composite group level.
  - Prohibits adding both mirrors of a concurrent R11 device to the same SRDF group name.
  - Cannot add a triangle of devices to a CG. In other words, a concurrent R11 device cannot have one R1 mirror paired with an R21 device, which is then paired with an R22 device that is paired with the other R1 mirror of the concurrent R1 device.
  - Prohibits adding a cascaded R1 device to a concurrent CG.
  - Prohibits adding a concurrent R1 device to a cascaded CG.
- If the target is a cascaded CG, the operation must be enabled by CG hop 1 or by the SRDF group name hop 1.
- If the target is a cascaded CG and the devices to be added are simple R1 devices, the CG cannot be enabled by CG hop 2 or by SRDF group name hop 2.
- If the target is a cascaded CG and the devices to be added are cascaded R1 devices paired with diskless R21 devices, then all R21 devices in the affected SRDF group must also be diskless.
- If the target is a cascaded CG and the devices to be added are cascaded R1 devices paired with non-diskless R21 devices, then all R21 devices in the affected SRDF group must be non-diskless.

Restrictions - remove operations

The following are restrictions for dynamically removing devices from an SRDF consistency group using the `symcg modify -remove` command:

- The dynamic modify remove operation must never leave an SRDF group empty.
- The `symcg modify -remove` command cannot remove SRDF groups from a consistency group.
- The `symcg modify -remove` command prohibits a cascaded R1 device from being removed from a consistency group enabled at the composite group level.
- The `symcg modify -remove` command cannot remove both legs of a concurrent R11 device if they are enabled for consistency protection by the same SRDF group name.

Adding devices to an RDF1 consistency group

Table 14 lists the allowable device types for a dynamic modify add operation on a composite group enabled for consistency protection at the composite group level and the SRDF group name level. This RDF1 CG is not concurrent or cascaded.

Table 14 Allowable device types for adding devices to an RDF1 CG

Device type in staging area	Enabled at CG level	Enabled at SRDF group name level
Simple R1 (R1->R2)	Allowed	Allowed
Concurrent R11	Not allowed	Only allowed if both affected SRDF groups in the CG already exist and are assigned to different SRDF group names.
Cascaded R1	Not allowed	Not allowed

Examples To move devices 50 and 51 from SRDF group 101 in the staging area to SRDF group 100 in `R1CG` on array 306:

```
symcg -cg R1CG modify -add -sid 306 -stg_rdfg 101 -devs 50:51 -cg_rdfg 100
```

To check if the devices were added to R1CG:

```
symrdf -cg R1CG query -detail
```

Adding devices to a concurrent RDF1 consistency group

Before you perform this procedure, review "Enabling SRDF consistency protection for concurrent SRDF devices" on page 151.

Table 15 shows the allowable device types for a dynamic modify add operation on a concurrent RDF1 composite group enabled for consistency protection at the composite group level and the SRDF group name level.

Table 15 Allowable device types for adding devices to a concurrent RDF1 CG

Device type in staging area	Enabled at CG level	Enabled at SRDF group name level
Simple R1 (R1->R2)	Allowed	Allowed
Concurrent R11	Not allowed	Only allowed if each mirror is assigned to a different SRDF group
Cascaded R1	Not allowed	Not allowed

Table 16 lists the allowable consistency modes for the SRDF groups of a concurrent CG.

Table 16 Supported consistency modes for concurrent SRDF groups

SRDF group 1 (first mirror)	SRDF group 2 (second mirror)
RDF-ECA	RDF-ECA
RDF-ECA	MSC
RDF-ECA	Not enabled

Table 16 Supported consistency modes for concurrent SRDF groups

SRDF group 1 (first mirror)	SRDF group 2 (second mirror)
Not enabled	RDF-ECA
MSC	RDF-ECA
MSC	MSC
MSC	Not enabled
Not enabled	MSC

Example In this example, device 20 is added to two independently-enabled SRDF groups of a CG.

Figure 18 on page 165 shows the staging area shared by array 306, 311, and 402 in a concurrent SRDF configuration:

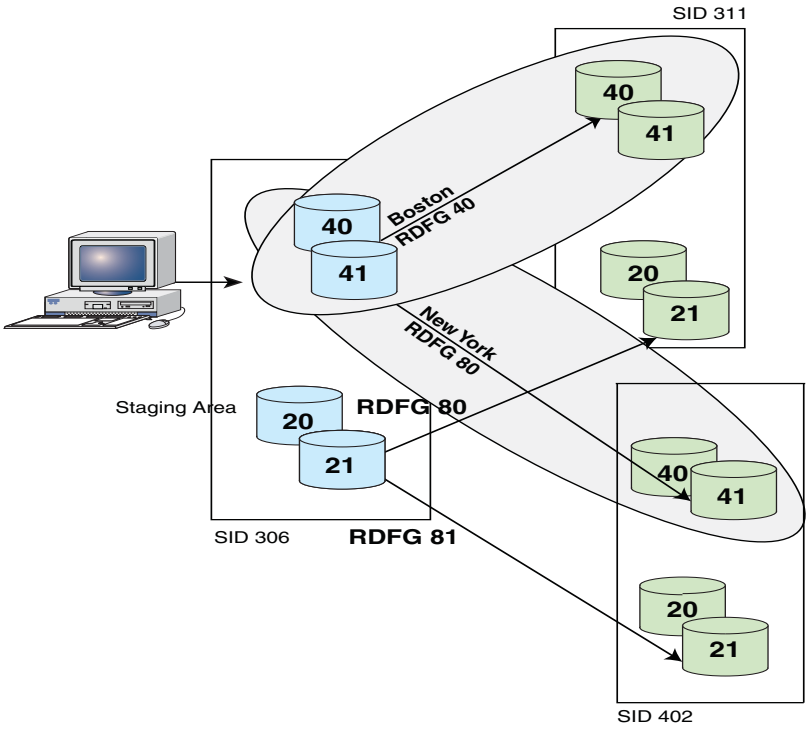


Figure 18 Adding a device to independently-enabled SRDF groups of a concurrent CG

The staging area contains devices 20 and 21.

Since the SRDF groups 70 and 71 of `ConCG` operate in different SRDF modes, they were enabled independently for consistency protection using the following SRDF group names:

- *Boston*: device pairs operate in SRDF/S mode and are set for RDF-ECA consistency protection.
- *New York*: device pairs operate in SRDF/A mode and are enabled for MSC consistency protection

To add only device 20 from the staging area into SRDF groups 70 and 71 of `ConCG`:

```
symcg -cg ConCG modify -add -sid 306 -stg_rdfg 80,81 -devs 20 -cg_rdfg 70,71
```

To check if the devices were added to `ConCG`:

```
symrdf -cg ConCG query -detail
```

Adding devices to a cascaded RDF1 consistency group

Before you perform this procedure, review "Checking if device pairs are enabled for consistency protection" on page 153.

Table 17 shows the allowable device types for a dynamic modify add operation on a cascaded R1 composite group enabled for consistency protection at the composite group level and the SRDF group name level.

Table 17 Allowable device types for adding devices to a cascaded RDF1 CG

Device type in staging area	Enabled at CG level			Enabled at SRDF group name level		
	Hop 1 enabled Hop 2 not enabled	Hop 1 enabled Hop 2 enabled	Hop 1 not enabled Hop 2 enabled	Hop 1 enabled Hop 2 not enabled	Hop 1 enabled Hop 2 enabled	Hop 1 not enabled Hop 2 enabled
Simple R1 (R1->R2)	Allowed	Not allowed	Not allowed	Allowed	Not allowed	Not allowed
Concurrent R11	Not allowed	Not allowed	Not allowed	Not allowed	Not allowed	Not allowed
Cascaded R1	Allowed	Allowed	Not allowed	Allowed	Allowed	Not allowed

Table 18 lists the allowable consistency modes for the hops of a cascaded CG.

Table 18 Supported consistency modes for cascaded hops

R1->R21 (hop 1)	R21->R2 (hop 2)
RDF-ECA	MSC
RDF-ECA	Not enabled
MSC	Not enabled

Example Figure 19 shows a cascaded SRDF configuration sharing the staging area among array 306, 311, and 402:

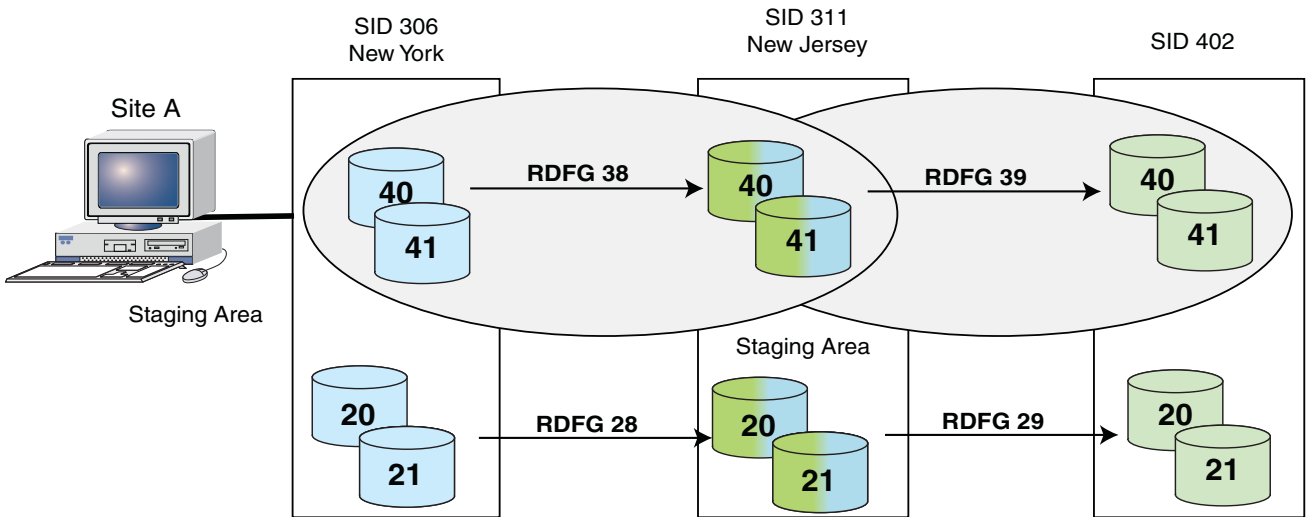


Figure 19 Adding devices to independently-enabled SRDF groups of a cascaded CG

.The staging area contains devices 20 and 21 to be added to `CasCG`.

The hops were independently enabled for consistency protection using the following SRDF group names:

- *New York*: device pairs operate in SRDF/S mode and are set for RDF-ECA consistency protection.
- *New Jersey*: device pairs operate in SRDF/A mode and are enabled for MSC consistency protection.

To add devices 20 and 21 from the staging area into SRDF groups 38 and 39 of `CasCG`:

```
symcg -cg CasCG modify -add -sid 306 -stg_rdfg 28 -devs 20:21 -stg_r21_rdfg 29
   -cg_rdfg 38 -cg_r21_rdfg 39
```

To check if the devices were added to `CasCG`:

```
symrdf -cg CasCG query -detail -hop2
```

Removing devices from an RDF1 consistency group

Table 19 shows the allowable device types for a dynamic modify remove operation on a composite group enabled for consistency protection at the composite group level and the SRDF group name level. This RDF1 CG is not concurrent or cascaded.

Table 19 Allowable device types for removing devices from an RDF1 CG

Device type in CG	Enabled at CG level	Enabled at SRDF group name level
Simple R1 (R1->R2)	Allowed	Allowed
Concurrent R11	Not applicable	Not applicable
Cascaded R1	Not applicable	Not applicable

Example To remove devices `50` and `51` from `RDFG 100` of `R1CG` on array 306 to `RDFG 101` in the staging area:

```
symcg -cg R1CG modify -remove -sid 306 -stg_rdfg 101 -devs 50:51 -cg_rdfg 100
```

Removing devices from a concurrent RDF1 consistency group

Table 20 shows the allowable device types for a dynamic modify remove operation on a concurrent R1 composite group enabled for consistency protection at the composite group level and the SRDF group name level.

Table 20 Allowable device types for removing devices from a concurrent RDF1 CG

Device type in CG	Enabled at CG level	Enabled at SRDF group name level
Simple R1 (R1->R2)	Allowed	Allowed
Concurrent R11	Not allowed	Only allowed if both mirrors are not enabled by the same SRDF group name.
Cascaded R1	Now allowed	Not allowed

Example To remove devices `20` through `30` from SRDF groups `70` and `80` of `ConCG` on array 306 into SRDF groups `71` and `81` in the staging area:

```
symcg -cg ConCG modify -remove -sid 306 -stg_rdfg 71,81 -devs 20:30 -cg_rdfg 70,80
```

Removing devices from a cascaded RDF1 consistency group

Table 21 shows the allowable device types for performing a dynamic modify remove operation on a cascaded R1 composite group enabled for consistency protection at the CG level and the SRDF group name level.

Table 21 Allowable device types for removing devices from a cascaded RDF1 CG

Device type in CG	Enabled at CG level			Enabled at SRDF group name level		
	Hop1 enabled Hop2 not enabled	Hop1 enabled Hop2 enabled	Hop1 not enabled Hop2 enabled	Hop1 enabled Hop2 not enabled	Hop1 enabled Hop2 enabled	Hop1 not enabled Hop2 enabled
Simple R1 (R1->R2)	Allowed	Not applicable	Not applicable	Allowed	Not applicable	Not applicable
Concurrent R11	Not allowed	Not allowed	Not allowed	Not allowed	Not allowed	Not allowed
Cascaded R1	Allowed	Allowed	Not allowed	Allowed	Allowed	Not allowed

Example To remove device 20 of SRDF groups 38 (R1->R21) and 39 (R21->R2) of `CasCG` on array 306 into SRDF groups 28 and 29 in the staging area:

```
symcg -cg CasCG modify -remove -sid 306 -cg_rdfg 38 -devs 20 -cg_r21_rdfg 39 -stg_rdfg 28
-stg_r21_rdfg 29
```

Recovering from a failed dynamic modify operation

Details about dynamic modify operations (target CG, SRDF groups, staging area, and operation type) are stored in the Symmetrix File System (SFS).

If a dynamic modify operation fails and all sites are reachable:

1. Re-run the command with the *exact* parameters.

2. If the command fails again, execute the `symcg modify -recover` command:

   symcg modify -cg *CasCG* -recover

This command uses the dynamic modify command information in SFS.

The recover operation either:

- Completes the unfinished steps of the dynamic modify operation, or
- Rolls back any tasks performed on the CG before failure, placing the CG into its original state.

For example, if a concurrent R11 loses a link to one of its mirrors during a dynamic modify add operation, the recover operation may remove all devices added to the CG by this operation. This ensures that the CG device pairs are consistent at all three sites.

Consistency groups with a parallel database

Figure 20 illustrates an SRDF consistency group with a parallel database such as Oracle Parallel Server (OPS).

- The production database array spans two hosts and two arrays, A and C.
- A SRDF consistency group includes R1 devices from arrays A and C.

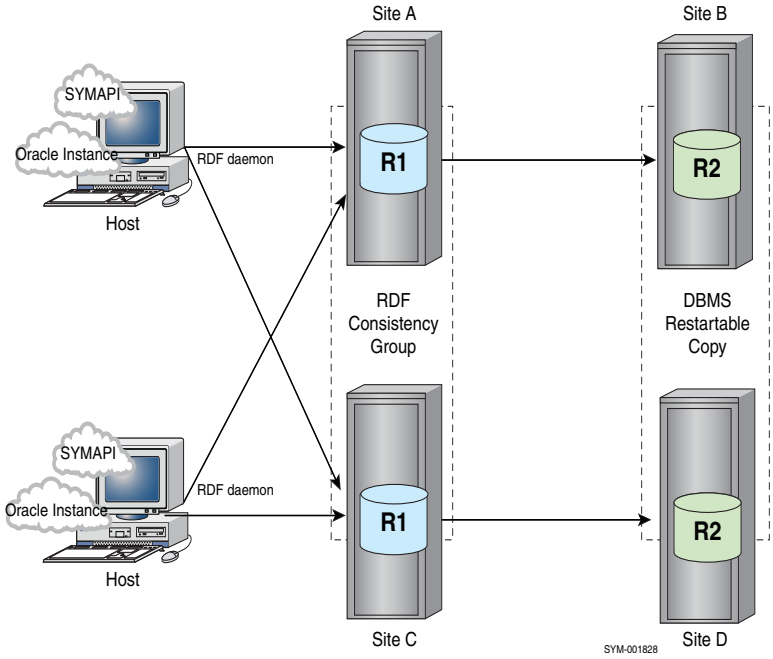


Figure 20 Using an SRDF consistency group with a parallel database configuration

The same consistency group definition must exist on both hosts. If enabled, Group Name Services (GNS) automatically propagates a composite group definition to the arrays and to all locally-attached hosts running the GNS daemon.

Although each production host can provide I/O to both R1 devices in the configuration, the DBMS has a distributed lock manager that ensures two hosts cannot write data to the same R1 device at the same time.

The SRDF links to two remote arrays (B and D) enable the R2 devices on those arrays to mirror the database activity on their respective R1 devices.

A typical remote configuration includes a target-side host or hosts (not shown in the illustration) to restart and access the database copy at the target site.

Figure 20 shows the SRDF daemons located on the production hosts.EMC recommends that you do not run the SRDF daemon on the same control host running database applications. "Redundant consistency protection" on page 143 provides more information.

Consistency groups with BCV access at the target site

When an SRDF consistency group includes devices on one or more source arrays propagating production data to one or more target arrays, TF BCVs at the target site can be indirectly involved in the consistency process.

Figure 21 illustrates a configuration with target-side BCVs that mirror the R2 devices:

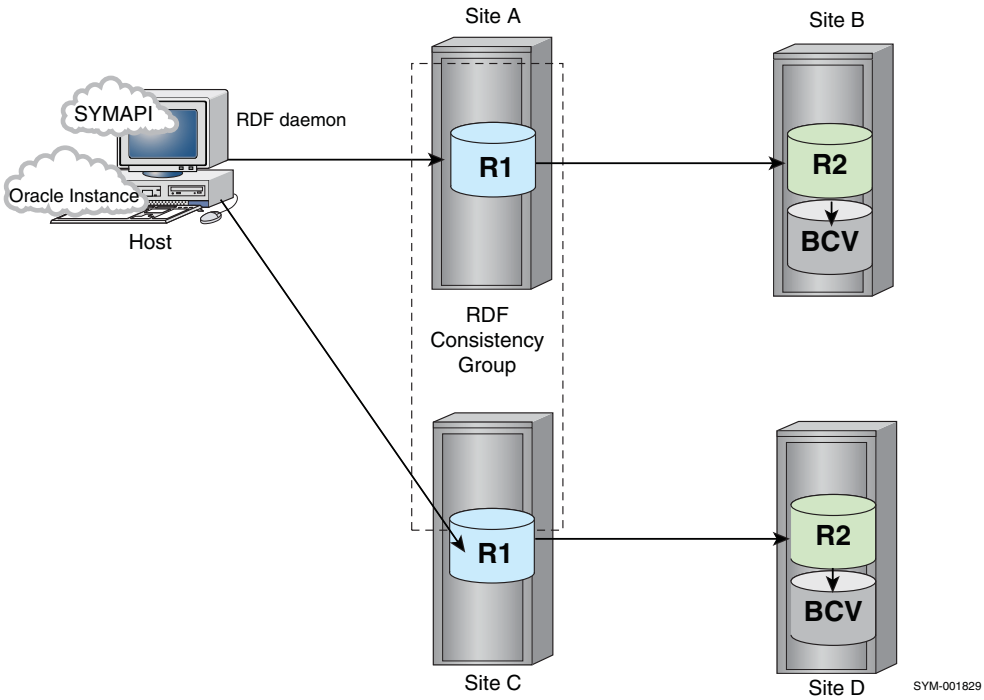


Figure 21 Using an SRDF consistency group with BCVs at the target site

You must split the BCV pairs at the target sites to access data on the BCVs from the target-side hosts.

The recovery sequence in a configuration that includes BCVs at the target site is the same as described in "Recovering from a failed dynamic modify operation" on page 168 with the following exception:

At the end of the sequence, the DBMS-restartable copy of the database exists on the target R2 devices and on the BCVs if the BCVs were synchronized with the target site's R2 devices at the time the interruption occurred.

When data propagation is interrupted, the R2 devices of the suspended SRDF pairs are in a Write Disabled state. The target-side hosts cannot write to the R2 devices, thus protecting the consistent DBMS-restartable copy on the R2 devices.

You can perform disaster testing and business continuance tasks by splitting off the BCV version of the restartable copy, while maintaining an unchanged R2 copy of the database. The R2 copy can remain consistent with the R1 production database until normal SRDF mirroring between the R1 and R2 sides resumes.

This configuration allows you to split off and access the DBMS-restartable database copy on the BCVs without risking the data protection that exists on the R2 devices when propagation of data is interrupted.

To manage the BCVs from the R2 side, associate the BCVs with a single SRDF consistency group defined on the target-site host that is connected to arrays B and D.

Figure 21 shows the SRDF daemons located on the production hosts.EMC recommends that you do not run the SRDF daemon on the same control host running database applications. "Redundant consistency protection" on page 143 provides more information.

CHAPTER 6
Concurrent Operations

This chapter describes the following topics:

- Overview 174
- Other documentation 176
- Creating/establishing concurrent SRDF devices 176
- Splitting concurrent SRDF devices 177
- Restoring concurrent devices 177
- Viewing concurrent SRDF devices 179

Overview

In a concurrent SRDF configuration, the source R1 device is mirrored to two R2 devices on two different remote arrays.

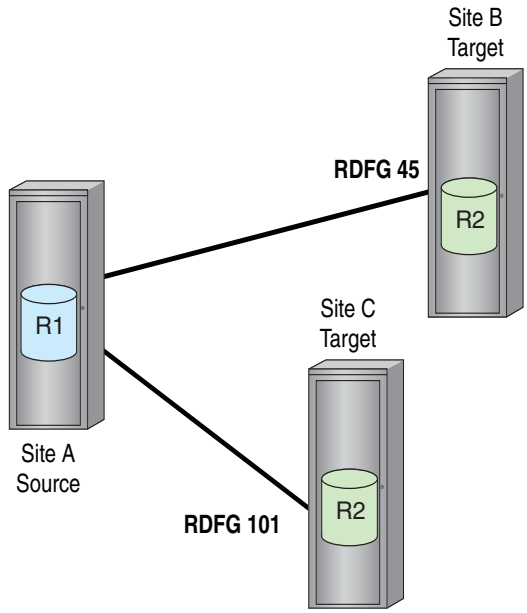


Figure 22 Concurrent SRDF

The two R2 devices operate independently but concurrently using any combination of SRDF modes.

Note: Starting with Enginuity 5875, both legs of the concurrent SRDF configuration can be in asynchronous mode

If both R2 mirrors are synchronous:

- A write I/O from the host at the R1 device side is returned as completed when both remote array' signal that the I/O is in cache at the remote side.

If one R2 is synchronous and the other R2 is adaptive copy:

- I/O from the R2 operating in synchronous mode must present ending status to the sending array before a second host I/O can be accepted.

  The host does not wait for the R2 operating in adaptive copy mode.

Restrictions

- The R2 devices at each remote array must belong to a different SRDF group.
- Asynchronous SRDF to both R2 devices requires Enginuity 5875 or higher on the R1 side.
- Simultaneous restore from both R2 devices to the R1 device cannot be performed.
- SRDF swap cannot be performed.

Both R2 devices in synchronous mode

If both R2 devices are in synchronous mode, both target sites have exact replicas of the source data. For this configuration, all three sites must be within synchronous distances.

Figure 23 shows three sites that are within synchronous distance:

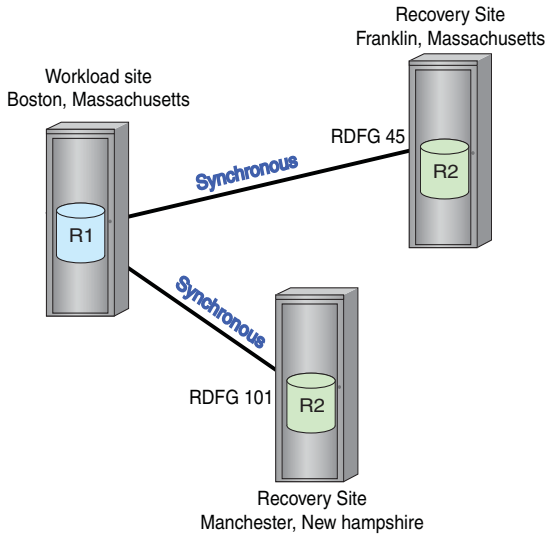


Figure 23 Concurrent SRDF/S to both R2 devices

Both R2 devices in asynchronous mode

You can configure concurrent SRDF/A to asynchronously mirror to recovery sites located at extended distances from the workload site.

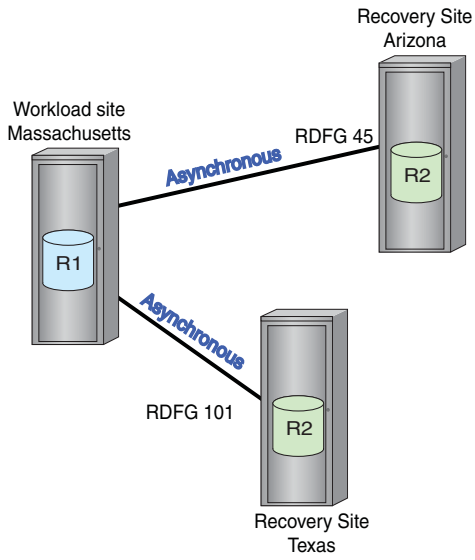


Figure 24 Concurrent SRDF/A to both R2 devices

With concurrent SRDF, you can build a device group or a composite group containing devices that only belong to the two SRDF groups representing the concurrent remote mirrors.

Your device group can also include BCV devices and SRDF devices that are not concurrent SRDF devices but that belong to either one of the concurrent SRDF groups.

Each mirror in a concurrent relationship must belong to a different SRDF group. When controlling or setting concurrent SRDF devices:

- `-rdfg n` performs the operation on the specified SRDF group number (remote mirror)
- `-rdfg ALL` performs the operation on the both SRDF groups.

Other documentation

Applicable pair states for concurrent SRDF operations

You can perform a control operation on one of these legs only if the other leg is in an acceptable pair state.

"Concurrent SRDF operations and applicable pair states" on page 254 provides more information.

Consistency protection

You can enable consistency protection for devices in a concurrent configuration.

"Enabling SRDF consistency protection for concurrent SRDF devices" on page 151 provides more information.

Note: Consistency protection for two mirrors participating in different consistency groups with SRDF/S requires Enginuity 5874 and higher.

Concurrent SRDF operations

To configure a concurrent SRDF relationship:

1. Create the initial R1 -> R2 pair between the first array and second array.
2. Create the R11 -> R2 pair between first array and the third array.

Creating/establishing concurrent SRDF devices

Procedure

To create a device group for the concurrent SRDF devices and initially synchronize (establish) the devices across the concurrent SRDF links:

1. Use the `symdg` command to create an R1 device group.

```
symdg [-i Interval] [-c Count] [-v]
.....
create DgName -type RDF1
```

```
symdg create ConcGrp -type RDF1
```

2. Use the `symdg add` command to add all concurrent SRDF devices to the device group:

```
symdg -g DgName [-i Interval] [-c Count] [-v]
.....
add dev SymDevName
```

```
symdg add dev 0001 -g ConcGrp -sid 0001
```

```
symdg add dev 0021 -g ConcGrp
symdg add dev 002A -g ConcGrp
```

3. Use the `symrdf establish` command to establish concurrent SRDF pairs that belong to the device group for the first R2 devices:

   ```
   symrdf -g DgName [-v | -noecho]
   ......
   -rdfg GrpNum establish
   ```

   ```
   symrdf -g ConcGrp establish -rdfg 1
   ```

4. Repeat Step 3 to establish concurrent SRDF pairs that belong to the device group for the second R2 devices:

   ```
   symrdf -g ConcGrp establish -rdfg 2
   ```

   You can also use the `-rdfg ALL` option to simultaneously establish both mirrors of each SRDF pair in one command:

   ```
   symrdf -g concGrp -full establish -rdfg ALL
   ```

   Note: Business Continuance Volume (BCV) devices cannot contain concurrent SRDF mirrors.

Splitting concurrent SRDF devices

Use the `symrdf split` command to split concurrent SRDF pairs, either one at a time or at the same time.

Examples To split the concurrent pairs one at a time:

```
symrdf -g DgName split -rdfg GroupNum of first mirror
symrdf -g DgName split -rdfg GroupNum of second mirror
```

For example, to split the concurrent pairs for device group `concGrp` one at a time:

```
symrdf -g concGrp split -rdfg 1
symrdf -g concGrp split -rdfg 2
```

To split the concurrent pairs simultaneously:

```
symrdf -g DgName split -rdfg All
```

For example, to split the concurrent pairs for device group `concGrp` at the same time:

```
symrdf -g concGrp split -rdfg ALL
```

Note: Concurrent R1 devices can have two mirrors participating in different consistency groups with MSC consistency protection enabled. This applies only to Enginuity 5875 and higher.

Restoring concurrent devices

In concurrent configuration, there are two RDFG groups of R2 devices.

- You can restore the R1 device from either of the R2 devices.

  To restore the R1 device from either of the R2 devices, you must specify which R2 device to use.

- You can restore both the R1 and one R2 device from the second R2 device.

Restoring R1 from a concurrent R2

Use the `restore` command to restore only the R1 device from the specified R2:

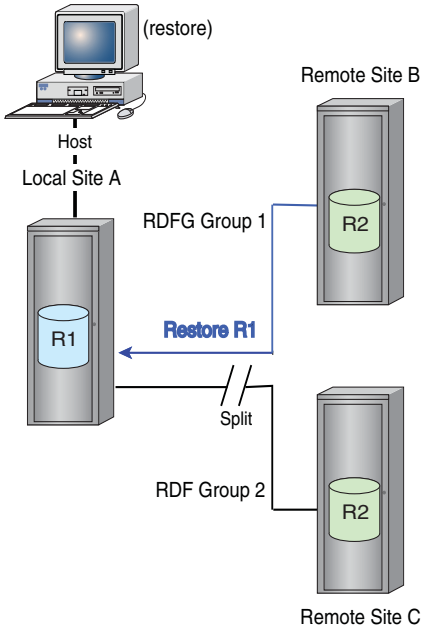


Figure 25 Restoring the R1 a concurrent configuration

When the `restore` command is executed:

- Both remote mirrors are split.
- The R1 device is restored from and synchronized with the R2 device in the specified RDFG group specified in the command.
- The R2 device belonging to SRDF group *not* used in the restore operation remains in the split state.

Use the `symrdf restore` command to restore from the specified RDFG group:

```
symrdf -g DgName restore -rdfg GroupNum of selected R2 mirror
```

Examples

To restore devices in group `concGrp` from RDFG group 1:

```
symrdf -g concGrp restore -rdfg 1
```

Use the `establish` command to re-establish the R2 devices not used in the restore operation:

```
symrdf -g DgName restore -rdfg GroupNum of group not used to restore
```

To re-establish second mirror (RDFG 2) for group `concGrp`:

```
symrdf -g concGrp establish -rdfg 2
```

Restoring both R1 and R2 from the second concurrent R2

Use the `restore` command with the `remote` option to restore both the R1 devices and the R2 devices on one leg from the R2 devices on the second leg:

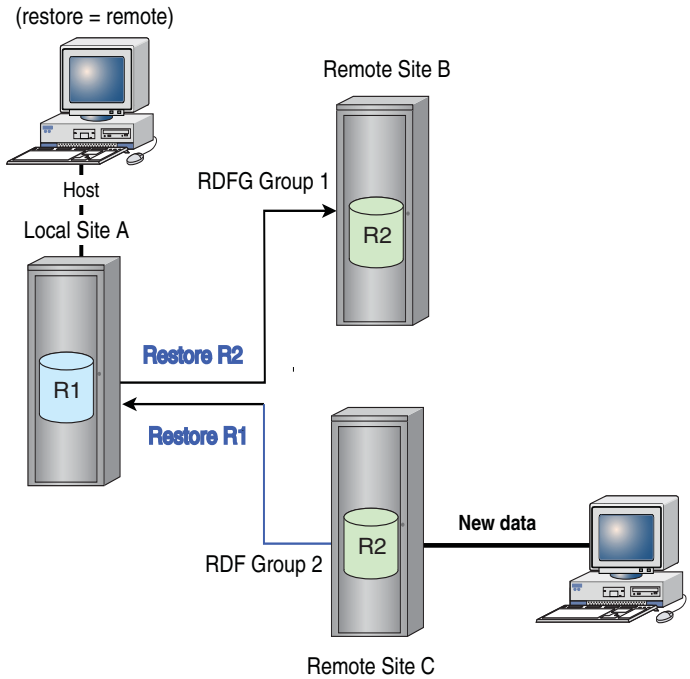


Figure 26 Restoring the source device and mirror in a concurrent SRDF configuration

When the `restore` command with the `remote` option is executed:

- Data from the specified R2 SRDF group 2 propagates data to the R1.
- The R1 SRDF group uses this data to restore the other R2 mirror, synchronizing all concurrent SRDF mirrors.

Note: You cannot simultaneously restore from both remote mirrors to the R1 device.

Use the `symrdf restore` command with the `remote` option to restore both the R1 devices and R2 devices on the second leg from the specified RDFG group:

```
symrdf -g DgName restore -rdfg GroupNum -remote
```

Example To restore the both the R1 and the R2 devices in RDF group 1 using the data in RDF group 2:

```
symrdf -g ConcGrp restore -rdfg 2 -remote
```

Viewing concurrent SRDF devices

Use the `symrdf list` command with the `-concurrent` option to display concurrent SRDF devices on the local array.

Each device of a concurrent pair belongs to a different RDF group, as shown in the RDF Typ:G column.

```
symrdf list -concurrent -sid 321

Symmetrix ID: 000192600321

                              Local Device View
-------------------------------------------------------------------------------
```

```
                        STATUS     MODES                      RDF  S T A T E S
Sym          RDF        ---------  -----  R1 Inv   R2 Inv ----------------------
Dev   RDev   Typ:G      SA RA LNK  MDATE  Tracks   Tracks Dev RDev Pair
---- ---- --------      ---------  ----- -------  ------- --- ---- -------------

00060 00060   R1:128    RW RW RW   S..1.       0        0 RW  WD   Synchronized
      00060   R1:228    RW RW RW   S..1.       0        0 RW  WD   Synchronized
00061 00061   R1:128    RW RW RW   S..1.       0        0 RW  WD   Synchronized
               .
               .
               .
```

Use the `query -rdfg all` command to display the state of concurrent SRDF pairs.

In the following example, concurrent SRDF pairs are in the process of synchronizing (SyncInProg):

```
symrdf -g conrdf query -rdfg all

Device Group (DG) Name             : conrdf
DG's Type                          : RDF1
.
.
.
Source (R1) View                   Target (R2) View         MODES
--------------------------------   ------------------------ ----- ------------
                ST                 LI      ST
Standard        A                  N       A
Logical         T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev     E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDAE  STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  00060 RW       0    69030 RW 0060 WD       0        0 S...  SyncInProg
              RW       0    69030 RW 0060 WD       0        0 S...  SyncInProg
DEV002  00061 RW       0    69030 RW 0061 WD       0        0 S...  SyncInProg
              RW       0    69030 RW 0061 WD       0        0 S...  SyncInProg
DEV003  00062 RW       0    69030 RW 0062 WD       0        0 S...  SyncInProg
```

During synchronization, use the `symrdf verify -summary` command to displays a summary message every 30 seconds until both concurrent mirrors of each SRDF pair are synchronized:

```
symrdf -g conrdf verify -summary -rdfg all -i 30 -synchronized
.
.
None of the devices in the group 'conrdf' are in 'Synchronized'
state.
.
.
Not All devices in the group 'conrdf' are in 'Synchronized' state.
.
.
All devices in the group 'conrdf' are in 'Synchronized' state.
```

CHAPTER 7
Device Migration Operations

This chapter describes the following topics:

- Overview 182
- Requirements 182
- R1 device migration 183
- R2 device migration 186
- Restrictions and limitations 190
- Sample procedure: migrating R1 devices 191
- Sample procedure: migrating R2 devices 199
- Pair states for migrate -setup 202
- Pair states for migrate -replace for first leg of concurrent SRDF 203
- Pair states for migrate -replace for second leg of concurrent SRDF 205

Overview

SRDF device migration allows you to replace an existing device in an SRDF pair with a new device on a different array.

During migration, a concurrent SRDF relationship is established to transfer data from an existing R1 device to a new device in adaptive copy disk mode.

When data transfer completes, the R1 device or the R2 device is replaced with the newly-populated device in the SRDF pair.

Requirements

- Each array must have a unique ID (sid).
- The existing SRDF device and the new devices must be dynamic R1 or R2 capable.

HYPERMAX OS

- Adaptive copy write pending mode is not supported when the R1 side of the RDF pair is on an array running HYPERMAX OS.

  For configurations where the R1 side is on an array running HYPERMAX OS, and the R2 side is running Enginuity 5876, the mode of the new device pair is set to the RDF mode of the R1 device being replaced.

- The Geometry Compatibility Mode attribute (-gcm) allows devices on arrays running HYPERMAX OS to be paired with devices on arrays running Enginuity 5876 that have an odd number of cylinders. When GCM is set, migration operations are subject to the following restrictions:
  - If the new device is on an array running HYPERMAX OS:
    - If the R1 device is being replaced:

      If the existing R2 device is on an array running Enginuity 5876 with an odd number of cylinders, then the migration is allowed if the new device can be made the same size using the GCM attribute.

      If the existing R2 device is on an array running HYPERMAX OS with GCM set, then the migration is allowed if the new device can be made the same size by setting the GCM attribute.

    - If the R2 is being replaced:

      If the existing R1 device is on an array running Enginuity 5876 with an odd number of cylinders, then the migration is allowed if the new device can be made the same size by setting the GCM attribute.

      If the existing R1 device is on an array running HYPERMAX OS with GCM set, then the migration is allowed if the new device can be made the same size by setting the GCM attribute.

  - If the new device is on an array running Enginuity 5876 and has an odd number of cylinders:
    - If the R1 is being replaced:

      If the existing R2 device is on an array running Enginuity 5876, then the new device must be the same configured size.

If the existing R2 device is on an array running HYPERMAX OS with GCM set, then the migration is allowed if the new device has the same GCM size as the R2 device.

– If the R2 is being replaced:

  If the existing R1 device is on an array running Enginuity 5876, then the new device must be the same configured size.

  If the existing R1 device is on an array running HYPERMAX OS with GCM set, then the migration will be allowed if the new device has the same GCM size as the R1.

R1 device migration

Before you can migrate an R1 device to a new array, you must create a temporary concurrent SRDF configuration with the new array as one of the R2 sites.

This section describes the steps to complete an R1 migration, including:

- "Configure a temporary SRDF group" and R1 device to enable the migration.
- "Establish a concurrent SRDF relationship"to transfer data to the from the old R1 device to the device that will become the new R1.
- "Replace the R1 device" with the newly-populated device in the SRDF pair.

Configure a temporary SRDF group

Configure a temporary SRDF group to synchronize data from the existing R1 device to the new R1 device.

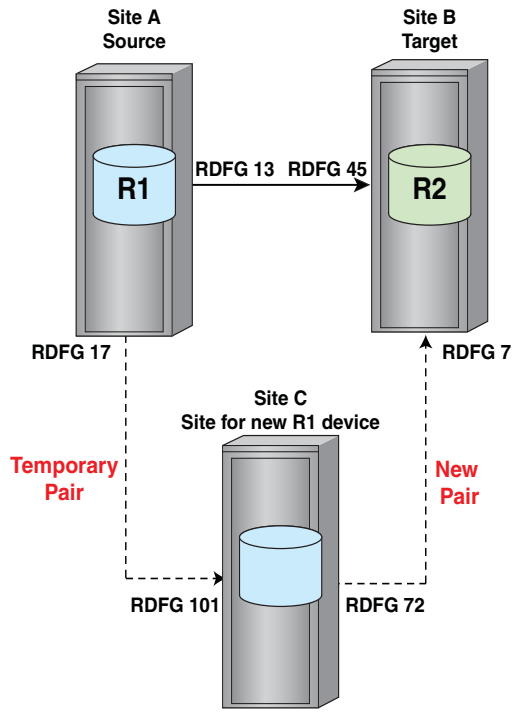


Figure 27 R1 migration: configuration setup

In Figure 27:

- Site A contains the existing R1 device paired with the R2 device in Site B,
- Site C contains the new non-SRDF device you want replace the existing R1 device.

  The dotted lines indicate that there are no SRDF relationships to Site C.
- A temporary SRDF group (RDFG 17) is used to synchronize data from the existing R1 to the new device in Site C.

The new R1 device replaces the existing R1 device during the migration.

Establish a concurrent SRDF relationship

Use the `symrdf migrate -setup` command to establish a concurrent relationship between the source device and two target devices.

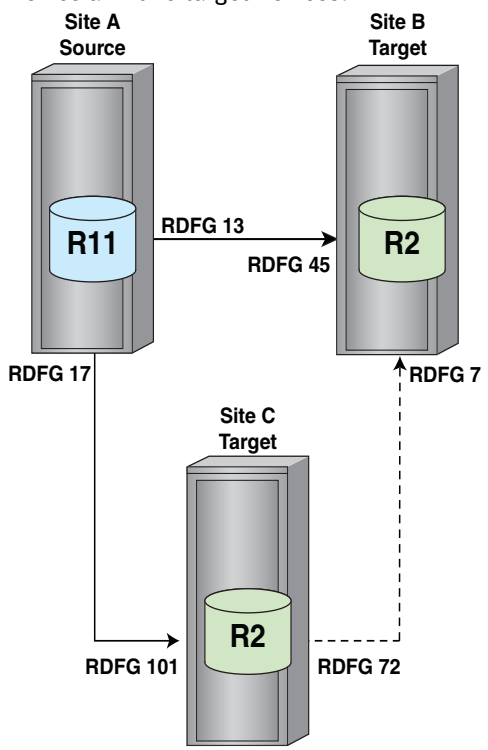


Figure 28 R1 migration: establishing a concurrent relationship

In Figure 28:

- The R1 device becomes the concurrent R11 device writing to two R2devices.
- Data synchronization in adaptive copy disk mode begins between the device and the R2 device on Site C.
- No SRDF pairing exists between the devices on Site C and Site B.

Note: You may need to modify existing device group or composite group scripts to accommodate the new R11 configuration.

Replace the R1 device

1. Wait until the two R2 devices are near synchronization with the R11 device.
2. Shut down any applications writing to the source device.

3. Use the `symrdf migrate -replace R1` command to replace the source device.

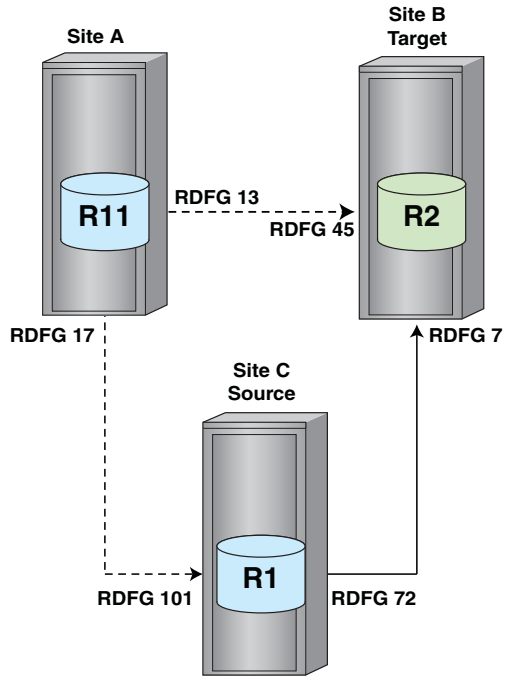


Figure 29 R1 migration: replacing the source device

The `symrdf migrate -replace R1` command executes the following actions:

1. Sets the source device to USR-NR (user not ready).

   This prevents applications writing to or reading from the R1 device.

2. Verifies the devices are in the correct pair state for replacement.

   See also "SRDF pair states for migration" on page 201.

3. (If applicable) Waits until all invalid tracks are cleared.
4. (If applicable) Drains the SRDF/A session.
5. Removes the SRDF pairing between the devices on the current R11 (Site A) and the original R2 (Site B).
6. Removes the SRDF pairing between the devices on the current R11 (Site A) and the new R2 (Site C).
7. Sets an SRDF pairing between the devices on Site C and B using the original SRDF mode of Site A and B.

   No additional copying of data is required between this SRDF pair because data is already the same on both devices.

8. Makes the devices read/write on the SRDF links.

The new R1 device is ready. You can restart the applications writing to the new R1 device on Site C.

The original R1 device remains USR-NR.

R2 device migration

R2 device migration allows you to replace the original R2 devices with new R2 devices. It shows the initial two-site topology, the migration process, and the final SRDF topology.

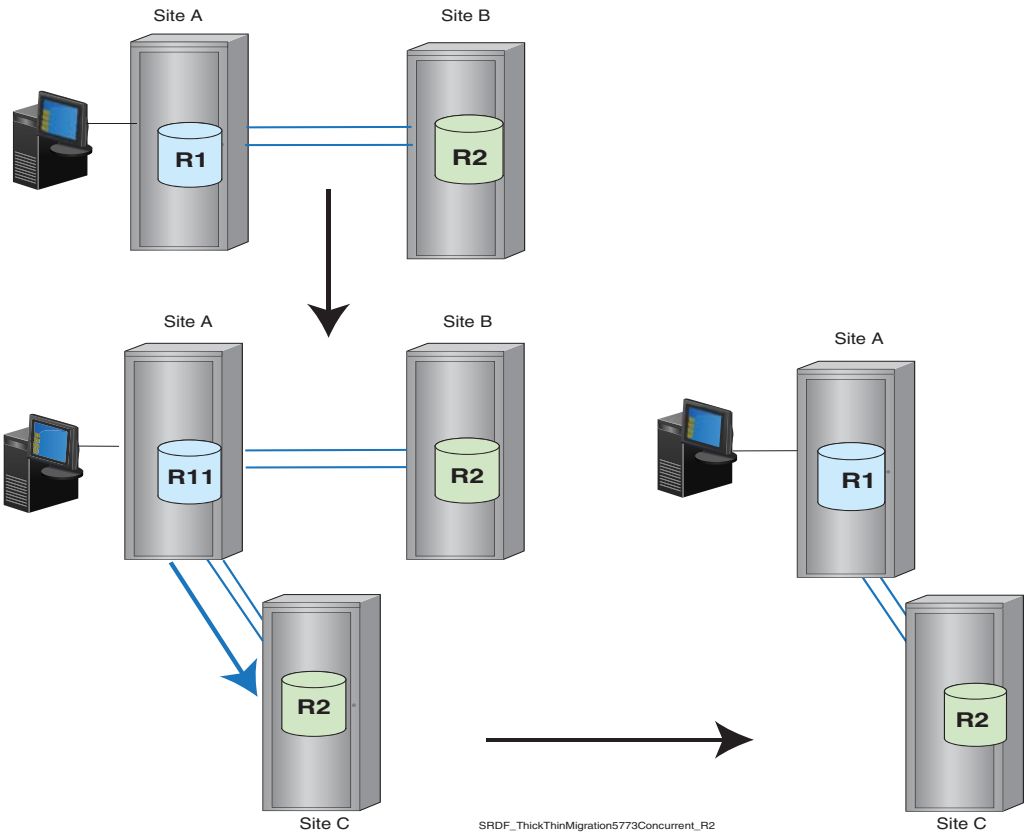


Figure 30 Migrating R2 devices

This section describes the steps to complete an R2 migration, including:

- "Configure a replacement R2 as a non-SRDF device:"
- "Establish a concurrent SRDF relationship"to transfer data to the from the R1 device to the device that will become the new R2.
- "Replace the R2 device" with the newly-populated device in the SRDF pair.

Configure setup for R2 migration

Configure a replacement R2 as a non-SRDF device:

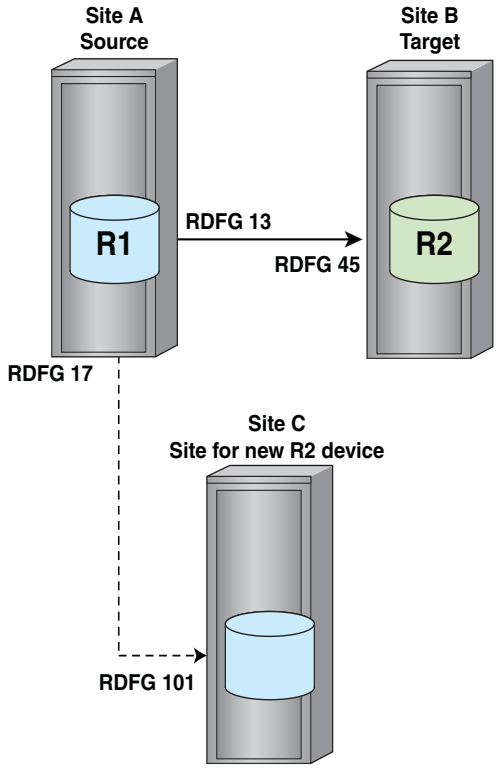


Figure 31 R2 migration: configuration setup

In Figure 31:

- Site A contains the R1 device paired with the existing R2 device in Site B,
- Site C contains the new non-SRDF device that will replace the R2 device.

  The dotted lines indicate no SRDF pairing exists with Site C.

Establish a concurrent SRDF relationship

Use the `symrdf migrate -setup` command to establish a concurrent SRDF relationship among the three sites:

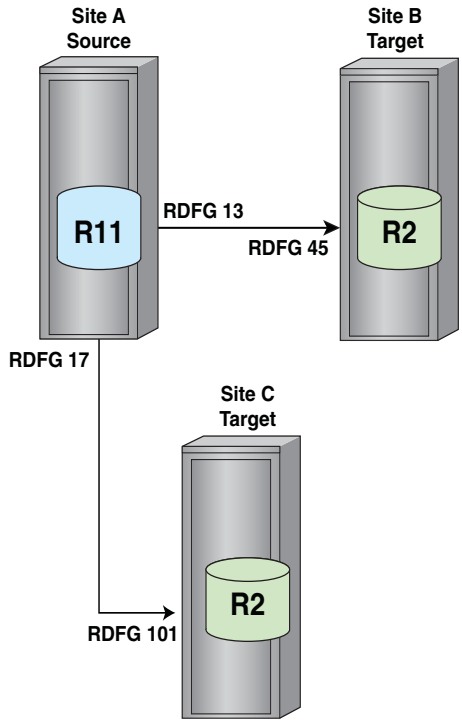


Figure 32 R2 migration: establishing a concurrent relationship

The establish action creates a concurrent SRDF relationship to transfer data from the existing source device to both target devices.

In Figure 32, the R1 becomes the R11 device writing to two target R2 devices.

- The source site continues to accept I/Os from the host.
- There is no need to shut down the applications writing to R1.
- No temporary pairing (like an R1 migration) is required.
- The source and target devices do not have to be close to synchronization.

Note: You may need to modify existing device group or composite group scripts to accommodate the new configuration.

Replace the R2 device

Use the `symrdf migrate -replace R2` command to replace the existing R2 device with the new R2 device in the SRDF pair:

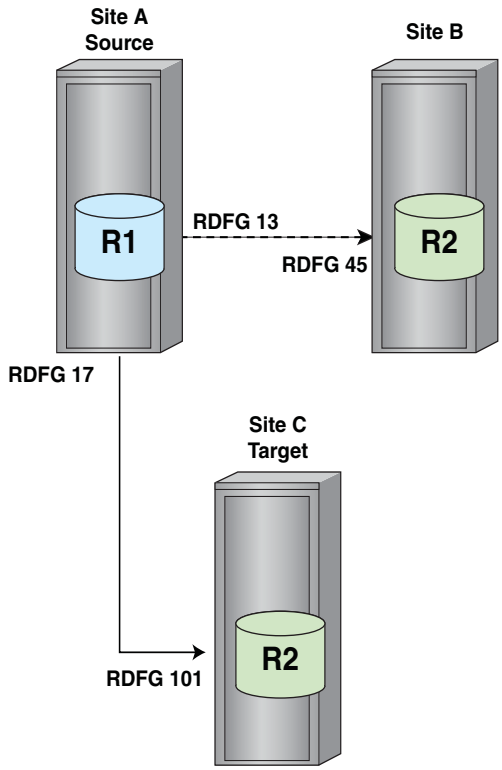


Figure 33 R2 migration: replacing the target device

The `symrdf migrate -replace R2` command executes the following actions:

1. Verifies the devices are in the correct pair state for replacement.

   "SRDF pair states for migration" on page 201 provides more information.

2. Removes the SRDF pairing between the devices on Site A and B.
3. Sets the mode of Site A and C using the original SRDF mode of Site A and B.

R1 and R2 migration procedures

Before you begin

- Plan for each migration.

  If you have defined scripts for your existing R1/R2 pair, evaluate how you may need to modify those scripts with new SIDs, SRDF device pairings, device groups, and composite groups.

  Keep in mind that during a device migration, the R1/R2 pair transforms into a concurrent SRDF relationship (R2<-R11->R2), and then back into an R1->R2 relationship.

- An SRDF group must exist for the new device.

If R1 is being replaced, this is the SRDF group between the new R1 and the existing R2.

If R2 is being replaced, this is the SRDF group between the new R2 and the existing R1.

- For an R1 migration *only*, a temporary SRDF group is required to synchronize data from the existing R1 device to the new device.

  If performing an R1 migration, create this temporary SRDF group.

- Before replacing the R1 device, you must shut down all applications using it.

  Application shutdown is not required when replacing an R2 device.

- Review "SRDF pair states for migration" on page 201.

Restrictions and limitations

SRDF/A device pairs

- The attributes associated with an existing SRDF group pertaining to an SRDF/A session are not automatically associated with the new SRDF group after migration.

  You must issue the `symconfigure` command on the new SRDF group and set the appropriate attributes, such as the minimum_cycle_time and the DSE (Delta Set Extension) autostart settings.

- If replacing a device of an SRDF pair in SRDF/A mode, all existing rules for DSE apply if DSE autostart is enabled on the new SRDF group.

  For example, the DSE threshold must be less than the maximum cache usage for the new SRDF group.

- If replacing the R1 device of an SRDF pair in SRDF/A mode, the new SRDF group in the new R1 array must be SRDF/A capable.
- If replacing a device of an SRDF pair in SRDF/A mode and Cache partitioning is enabled on the new array, all new devices must belong to the same cache partition.
- If the existing device is in SRDF/A mode, the entire SRDF group must be migrated.
- If the existing device is in SRDF/A mode, the new SRDF group must be empty.
- If replacing the R1 device, the temporary SRDF group must not be in SRDF/A mode.
- The existing SRDF device pair cannot be in semi-synchronous mode.

Devices

- The new device (R1 or R2) cannot be an SRDF device before migration.
- The existing device (R1 or R2) and the replacement device cannot be diskless.
- The new R1 device cannot be larger than the existing R1 device.
- The existing R1 device cannot have any local invalid tracks.
- After migration, the R2 device cannot be larger than the R1 device.
- The existing (R1 or R2) and the new device cannot be configured for SRDF/Star.
- The existing device and the replacement device cannot be a source or a target device for TF/Mirror, TF/Snap, TF/Clone, Open Replicator, and Federated Live Migration.

This restriction does not apply to the SRDF partner of the existing device.

- The existing R1/R2 device pair cannot be in a concurrent SRDF relationship.

  Set the `-config` option to equal `pair` in `symrdf migrate -setup` to indicate this pair is not part of such a configuration.

- An SRDF consistency protection group must be enabled at the RDFG-name level, **NOT** at the composite-group level.

  Otherwise, the `migrate -setup` command stops the monitoring/cycle switching of your composite group.

  "Sample procedure: migrating R1 devices", explains the procedure for an SRDF consistency protection group enabled at the composite-group level.

Sample procedure: migrating R1 devices

For this sample procedure, the SRDF consistency protection group is enabled at the composite-group level.

This procedure shows the steps to change this setting and enable SRDF consistency protection at the RDFG-name level.

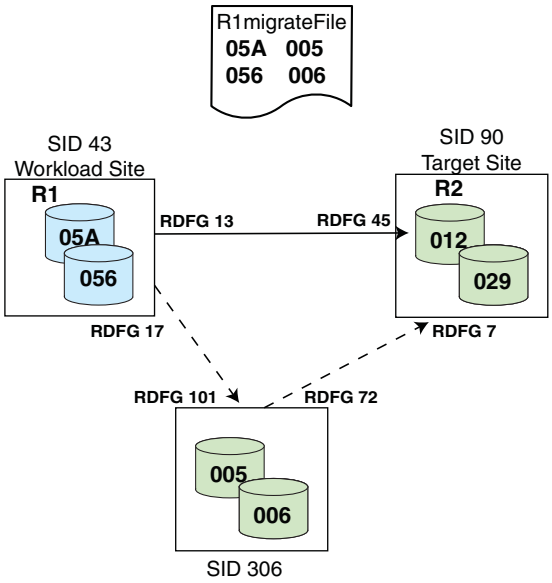


Figure 34 R1 migration example: Initial configuration

Figure 34 shows an R1 and R2 relationship between array 43 and array 90.

After R1 migration, the devices in array 306 will become the source devices for array 90.

Step 1: Querying the sample SRDF/A configuration

Use the `symrdf query -detail` command to query a configuration with SRDF consistency protection enabled at the composite-group level.

```
symrdf -cg MigrateRDF query -detail

Composite Group Name       :  MigrateRDF
Composite Group Type       :  RDF1
Number of Symmetrix Units  :  1
```

```
Number of RDF (RA) Groups :  1
RDF Consistency Mode      :  MSC

RDFA MSC Consistency Info:{
    Session Status                      : Active
    Consistency State                   : CONSISTENT
    }

Symmetrix ID                      : 000192600043 (Microcode Version: 5874)
Remote Symmetrix ID               : 000192600090 (Microcode Version: 5874)
RDF (RA) Group Number             :   1 (00) 13 (0C)
RDFA Info:
    {
    Cycle Number                        : 29
    Session Status                      : Active - MSC
    Consistency Exempt Devices          : No
    Minimum Cycle Time                  : 00:00:30
    Avg Cycle Time                      : 00:00:30
    Duration of Last cycle              : 00:00:30
    Session Priority                    : 33
    Tracks not Committed to the R2 Side: 0
    Time that R2 is behind R1           : 00:00:42
    R2 Image Capture Time               : Mon Sep 21 13:28:44 2009
    R2 Data is Consistent               : True
    R1 Side Percent Cache In Use        : 0
    R2 Side Percent Cache In Use        : 0
    R1 Side DSE Used Tracks             : 0
    R2 Side DSE Used Tracks             : 0
    Transmit Idle Time                  : 00:00:00
    }

        Source (R1) View             Target (R2) View       MODES
--------------------------------  ------------------------- ----- ------------
               ST                 LI       ST
  Standard      A                  N        A
Logical  Sym    T  R1 Inv  R2 Inv  K        T  R1 Inv  R2 Inv       RDF Pair
Device   Dev    E  Tracks Tracks   S Dev    E  Tracks  Tracks MDACE STATE
-------------------------------- -- ------------------------- ----- ------------
DEV001   0005A NR       0       0 RW 00012 WD       0       0 A..X. Consistent
DEV002   000F8 NR       0       0 RW 00029 WD       0       0 A..X. Consistent

Total            ------- -------            ------- -------
  Track(s)             0       0                  0       0
  MBs                0.0     0.0                0.0     0.0
```

Step 2: Changing the SRDF consistency protection setting

To maintain consistency protection after establishing a concurrent SRDF relationship:

- Remove the SRDF consistency protection enabled at the composite-group level, and then
- Enable consistency protection at the RDFG-name level.

In the following example:

- The `symcg set -name siteb` command sets the SRDF group name to `siteb`.
- The `symcg disable` command disables SRDF consistency protection at the composite-group level
- The `symcg enable` command enables SRDF consistency protection at the RDFG-name level.

```
symcg -cg MigrateRDF -rdfg 043:13 set -name siteb
symcg -cg MigrateRDF disable
```

```
A consistency 'Disable' operation execution is
in progress for composite group 'MigrateRDF'. Please wait...

The consistency 'Disable' operation successfully executed for
composite group 'MigrateRDF'.
```

```
symcg -cg MigrateRDF -rdfg name:siteb enable
```

```
A consistency 'Enable' operation execution is
in progress for composite group 'MigrateRDF'. Please wait...

The consistency 'Enable' operation successfully executed for
composite group 'MigrateRDF'.
```

Verifying the changes

Use the `symrdf query -detail` command to verify that the changes and additions were made to the SRDF/A configuration.

In the following example, SRDF consistency protection is now enabled using the SRDF group name of `siteb`.

```
symrdf -cg MigrateRDF query -detail

Composite Group Name      :  MigrateRDF
Composite Group Type      :  RDF1
Number of Symmetrix Units :  1
Number of RDF (RA) Groups :  1
RDF Consistency Mode      :  NONE

RDFG Names:
    {
    RDFG Name                          : siteb
    RDF Consistency Mode               : MSC
    MSC Consistency Info:
        {
        Session Status                 : Active
        Consistency State              : Consistent
        }
    }
```

Step 3: Pairing devices

Create a device file to pair SRDF devices with the new non-SRDF devices.

"Creating a device file" on page 88 provides more information.

This pairing is used temporarily to transfer data from the existing R1 devices to the devices that will eventually replace them in an SRDF pair.

In the following example, device file R1MigrateFile contains two pairs:

```
05A 005
056 006
```

R1 devices 05A and 056 in array 43 are paired with the new devices 005 and 006 in array 306.

Step 4: Establishing a concurrent SRDF relationship

The `symrdf migrate -setup` command establishes a concurrent SRDF relationship between the existing R1 devices and the new devices in adaptive copy disk mode, and begins the synchronization of these devices.

Note: You may need to modify existing device group or composite group scripts to accommodate the temporary change of the existing R1 devices to R11 devices.

The `symrdf -migrate -setup -config pair -force` command establishes a concurrent SRDF relationship between the R1 devices in array 43 and the new devices in array 306 using SRDF group 17.

This is a temporary relationship to transfer data from the existing R1 to its replacement.

Using the -force option

You must use the `-force` option because SRDF consistency protection is enabled.

```
symrdf -sid 043 -rdfg 17 -f R1MigrateFile migrate -setup -config pair -force

An RDF 'Migrate Setup' operation execution is
in progress for device file 'R1migrateFile'. Please wait...

    Migrate Setup for R1 device(s) in (043,017).......................Started.
    Create RDF Pair in (0043,017)....................................Started.
    Create RDF Pair in (0043,017)....................................Done.
    Mark target device(s) in (0043,017) for full copy from source....Started.
    Devices: 06F0-06FF in (0043,017)............................... Marked.
    Mark target device(s) in (0043, 017) for full copy from source...Done.
    Merge track tables between source and target in (0043,017).......Started.
    Devices: 06F0-06FF in (0043,017)............................... Merged.
    Merge track tables between source and target in (0043,017).. ....Done.
    Resume RDF link(s) for device(s) in (0043,017)...................Started.
    Resume RDF link(s) for device(s) in (0043,017)...................Done.
    Migrate Setup for R1 device(s) in (0043,017) ....................Done.

The RDF 'Migrate Setup' operation finished successfully
for device file 'R1MigrateFile'.
```

Note: If the host is reading and writing to the R1 device during this action, a synchronized pair state may not be attainable because the pair is operating in adaptive copy disk mode.

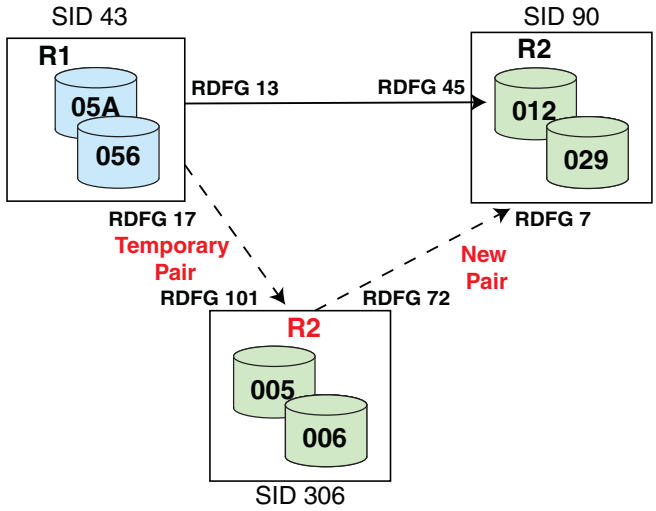


Figure 35 Concurrent SRDF relationship

In Figure 35:

- Devices 05A and 056 are paired with devices 005 and 006 in a concurrent SRDF relationship using SRDF group 17.
- Devices 005 and 006 are made read/write on the SRDF links in adaptive copy disk mode.
- SRDF group 17 is used temporarily to transfer data from the R1 devices to the new devices.

Step 5: Replacing R1 devices with new devices

1. If consistency is enabled, use the `symcg disable` command to disable it.

   To disable SRDF consistency protection for composite group `MigrateRDF`:

   ```
   symcg -cg MigrateRDF -rdfg name:siteb disable

   A consistency 'Disable' operation execution is
   in progress for composite group 'MigrateRDF'. Please wait...

   The consistency 'Disable' operation successfully executed for
   composite group 'MigrateRDF'.
   ```

2. Terminate any TF/Mirror, TF/Snap, TF/Clone, Open Replicator, and Federated Live Migration sessions.
3. Use the `symrdf migrate -replace` command to set R1 (R11) device as USR-NR, complete the final synchronization of data between the existing and the new device, and reconfigure the devices into a new SRDF pair.

   The device pairings of the replaced devices are removed. The new devices become R1 devices paired with the existing R2 devices using the original SRDF mode of the replaced pair.

   Note: The `migrate -replace R1` command waits for synchronization to finish and may take a long time. To avoid the locking of the SYMAPI database for this entire time, set the environment variable SYMCLI_CTL_ACCESS=PARALLEL. If you set this variable, you may need to run the `symcfg sync` command after the R1 migration is complete.

In the following example, the `migrate -replace R1` command specifies the new SRDF group 72 to reconfigure and connect the new R1 devices 005 and 006 in array 306 with the R2 devices 012 and 029 in Symmetix 90:

```
symrdf -sid 043 -rdfg 17 -f R1migrateFile migrate -replace r1 -config pair -new_rdfg 72

An RDF 'Migrate Replace R1' operation execution is
in progress for device file 'R1migrateFile'. Please wait...

    Migrate Replace R1 for new R1 device(s) in (0306, 072)...........Started.
    Waiting for invalid tracks to reach 0 in   (0043, 013)...........Started.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........Started.
    Waiting for invalid tracks to reach 0 in   (0043, 013)...........Done.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........915994 remaining.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........519572 remaining.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........245889 remaining.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........107613 remaining.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........1110 remaining.
    Waiting for invalid tracks to reach 0 in   (0043, 017)...........Done.
    Suspend RDF link(s) for device(s) in (0043,013)..................Started.
    Suspend RDF link(s) for device(s) in (0041,013)..................Done.
    Suspend RDF link(s) for device(s) in (0043,017)..................Done.
```

```
    Delete RDF Pair in (0043,013)..................................Started.
    Delete RDF Pair in (0043,017)..................................Started.
    Delete RDF Pair in (0043,013)..................................Done.
    Delete RDF Pair in (0043,017)..................................Done.
    Create RDF Pair in (0306,072)..................................Started.
    Create RDF Pair in (0306,072)..................................Done.
    Resume RDF link(s) for device(s) in (0306,072)..................Started.
    Merge track tables between source and target in (0306,072).......Started.
    Devices: 0690-069F in (0306,072)............................... Merged.
    Merge track tables between source and target in (0306,072).......Done.
    Resume RDF link(s) for device(s) in (0306,072)..................Done.
    Migrate Replace R1 for new R1 device(s) in (0306, 072)..........Done.

The RDF 'Migrate Replace R1' operation finished successfully
for device file 'R1migrateFile'.
```

After replacing the R1 devices:

- Recreate your device groups and/or composite groups,
- Possibly update your scripts, since the devices are no longer concurrent SRDF.
- Recreate any TF/Mirror, TF/Snap, TF/Clone, Open Replicator, and Federated Live Migration sessions (used on the original R1 devices) on the new R1 devices.

In the following example, the `MigrateRDF` consistency group is deleted and re-created:

- The `symcg delete` command deletes the `MigrateRDF` consistency group.
- The `symcg create` command recreates `MigrateRDF` as an RDF1 with consistency.
- The `symcg addall dev` command add devices `MigrateRDF`.
- The `symcg enable` command enables consistency protection.

```
symcg -force delete MigrateRDF
symcg create MigrateRDF -type rdf1 -rdf_consistency
symcg -cg MigrateRDF -sid 306 -rdfg 72 addall dev
symcg -cg MigratRDF enable

A consistency 'Enable' operation execution is
in progress for composite group 'MigrateRDF'. Please wait...

The consistency 'Enable' operation successfully executed for
composite group 'MigrateRDF'.
```

When migration is complete (as shown in Figure 36):

- SID 306 devices are the R1 devices.
- SID 306 devices are paired with the R2 devices in SID 90.

- This new SRDF pair uses the original SRDF mode of the replaced pair.

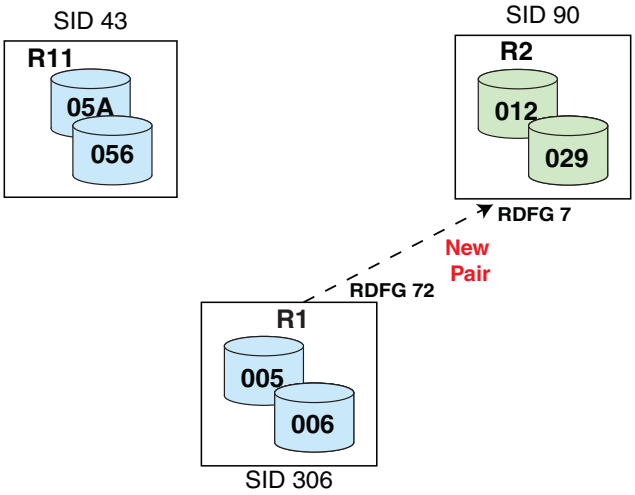


Figure 36 Migrated R1 devices

Step 6: Verifying the new pair and setting changes

Use the `symrdf query -detail` to verify that:

- The SID 306 devices are now the source devices for SID 90,
- Consistency protection is rebuilt.

```
symrdf -cg MigrateRDF query -detail

Composite Group Name       :  MigrateRDF
Composite Group Type       :  RDF1
Number of Symmetrix Units  :  1
Number of RDF (RA) Groups  :  1
RDF Consistency Mode       :  MSC

RDFG MSC Consistency Info:{
   Session Status                  : Active
   Consistency State               : CONSISTENT
    }

Symmetrix ID                       : 000190100306 (Microcode Version: 5773)
Remote Symmetrix ID                : 000192600090 (Microcode Version: 5874)
RDF (RA) Group Number              :   3 (02) - siteb
RDFA Info:
    {
    Cycle Number                        : 3
    Session Status                      : Active - MSC
    Consistency Exempt Devices          : No
    Minimum Cycle Time                  : 00:00:30
    Avg Cycle Time                      : 00:00:33
    Duration of Last cycle              : 00:00:30
    Session Priority                    : 33
    Tracks not Committed to the R2 Side: 0
    Time that R2 is behind R1           : 00:00:34
    R2 Image Capture Time               : Mon Sep 21 13:52:03 2009
    R2 Data is Consistent               : True
    R1 Side Percent Cache In Use        : 0
    R2 Side Percent Cache In Use        : 0
    R1 Side DSE Used Tracks             : 0
    R2 Side DSE Used Tracks             : 0
    Transmit Idle Time                  : 00:00:00
```

```
    }

        Source (R1) View             Target (R2) View       MODES
--------------------------------  ------------------------- ----- ------------
               ST                 LI       ST
  Standard      A                  N        A
Logical  Sym    T  R1 Inv  R2 Inv  K        T  R1 Inv  R2 Inv       RDF Pair
Device   Dev    E  Tracks Tracks   S Dev    E  Tracks  Tracks MDACE STATE
-------------------------------- -- ------------------------- ----- ------------
DEV001   00005 RW       0       0 RW 00012 WD       0       0 A..X. Consistent
DEV002   00006 RW       0       0 RW 00029 WD       0       0 A..X. Consistent

Total            ------- -------            ------- -------
  Track(s)             0       0                  0       0
  MBs                0.0     0.0                0.0     0.0
```

Sample procedure: migrating R2 devices

In this migration example, the devices in array 306 will become the R2 devices for array 43.

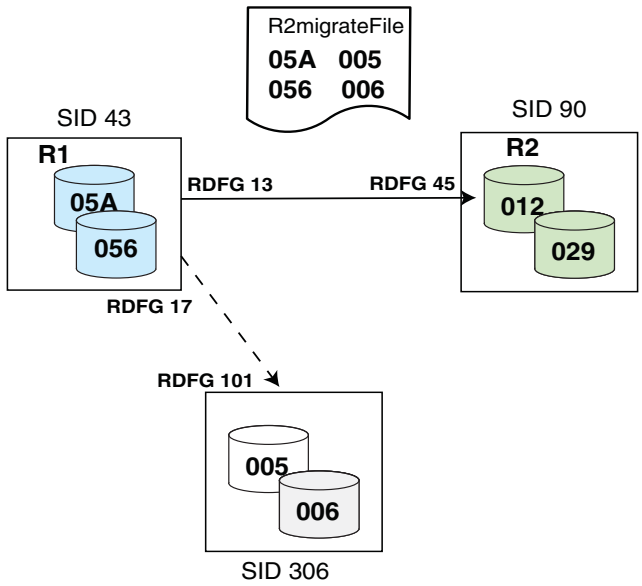


Figure 37 R2 migration example: Initial configuration

Figure 37 shows the R1 and R2 relationship between array 43 and array 90.

Step 1: Pairing devices

Create a device file to pair SRDF devices with the new non-SRDF devices.

"Creating a device file" on page 88 provides more information.

In the following example, device file R2MigrateFile contains two pairs:

```
05A 005
056 006
```

When migration is complete, R1 devices 05A and 056 in array 43 will be paired with the new devices 005 and 006 on array 306.

Step 2: Establishing a concurrent SRDF relationship

The `symrdf migrate -setup` command establishes a concurrent SRDF relationship between the existing R1 devices and the new devices in adaptive copy disk mode, and begins the synchronization of these devices.

Because this is an R2 migration, the R1 continues to process I/Os from its host, and synchronization is not required between the R1 and the new device.

Note: You may need to modify existing device group or composite group scripts to accommodate the temporary change of the existing R1 devices to R11 devices.

The `symrdf migrate -setup -config pair` command establishes a concurrent SRDF relationship between the R1 devices 05A and 056 in array 43 and the new devices 005 and 006 in array 306 using SRDF group 17:

```
symrdf -file R2migrateFile -sid 043 -rdfg 17 migrate -setup -config pair
```

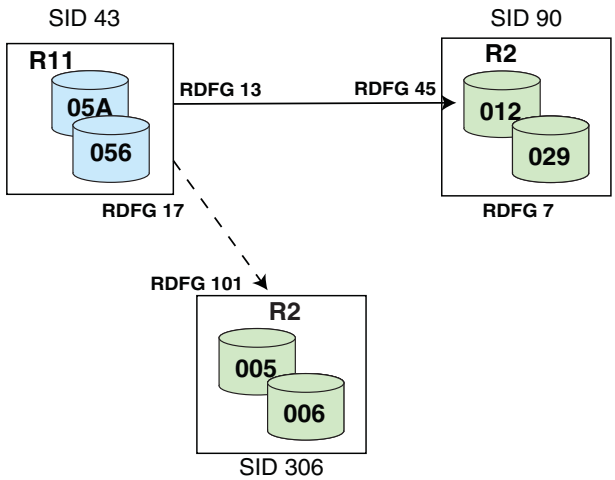


Figure 38 Concurrent SRDF relationship

In Figure 38:

- Devices 05A and 056 are paired with devices 005 and 006 in a concurrent SRDF relationship using the SRDF group 17,
- Devices 005 and 006 are made read/write on the SRDF links in adaptive copy disk mode.

Unlike an R1 device migration, the SRDF group 17 is permanent, and synchronizes data from the source to the target devices.

Step 3: Replacing R2 devices with new devices

1. If SRDF consistency protection is enabled, disable it.
2. Terminate any TF/Mirror, TF/Snap, TF/Clone, Open Replicator, and Federated Live Migration sessions.
3. Use the `symrdf migrate -replace R2` command to delete the SRDF pairing between array 43 and array 90.

   Note: After replacing R2, you must modify device groups and/or composite groups to remove all BCVs, VDEVS, TGTs from the original R2 and then add appropriate counterparts to the new R2.

   You must also recreate any TF/Mirror, TF/Snap, TF/Clone, Open Replicator, and Federated Live Migration sessions on the new R2.

In the following. example, the `symrdf migrate -replace R2 -config pair` command uses the SRDF group 17 to reconfigure and connect the R1 devices 05A and 056 with the new R2 devices 005 and 006:

```
symrdf -file R2migrateFile -sid 043 -rdfg 17 migrate -replace R2 -config pair.
```

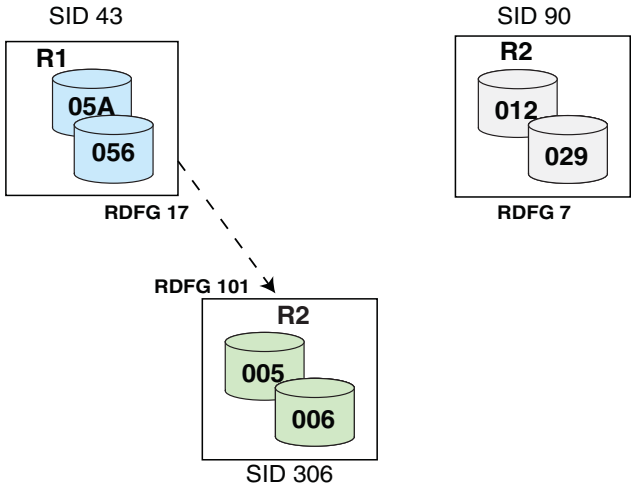


Figure 39 Migrated R2 devices

When migration is complete, the array 306 devices become the R2 devices and are paired with the R1 devices in Symmetix 43.

This new pair uses the original SRDF mode of the replaced pair.

SRDF pair states for migration

An existing R1 and R2 pair must in a specific SRDF state to perform certain migration control operations.

Table 22 shows the applicable pair states for `symrdf migrate -setup` for an R1 and an R2 migration.

Table 22 SRDF migrate -setup control operation and applicable pair states

Control operation:	Pair state: existing R1->R2											
	SYNC IN PROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILED OVER	PARTITIONED[1]	PARTITIONED[2]	R1 UPDATED	R1 UPD IN PROG	INVALID	CONSISTENT	TRANSMIT IDLE
migrate -setup	P	P	P[3]	P[3]							P	

1. The remote array is in the SYMAPI database (it was discovered).
2. The remote array is not in the SYMAPI database (it was not discovered or was removed).
3. Only when replacing the R2 devices.

Pair states for migrate -setup

Figure 40 shows a sample configuration for an R1 migration:

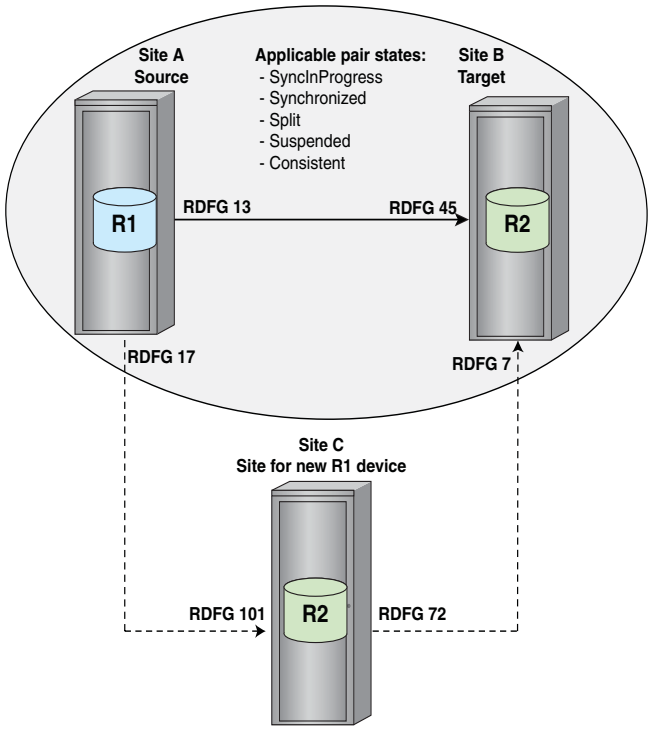


Figure 40 R1 migration: applicable R1/R2 pair states for migrate -setup

The R1 in array A and the R2 in array B must be in one of the applicable pair states before issuing the `symrdf migrate -setup` command, which establishes a concurrent SRDF relationship among the three sites.

Figure 41 shows a sample configuration for an R2 migration:

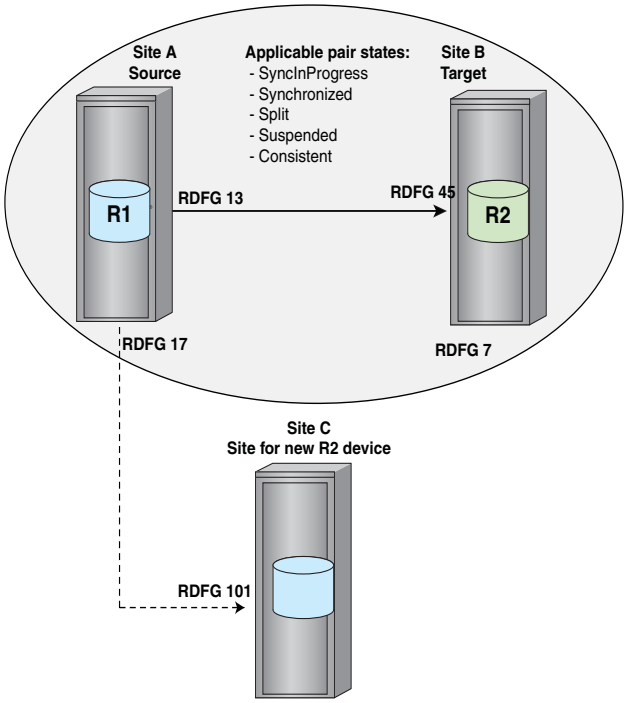


Figure 41 R2 migration: applicable R1/R2 pair states for migrate -setup

The R1 in array A and the R2 in array B must be in one of the applicable pair states before issuing the `symrdf migrate -setup` command, which establishes a concurrent SRDF relationship among the three sites.

Pair states for migrate -replace for first leg of concurrent SRDF

Figure 42 shows the SRDF pair state required before replacing an R1, the R11 and its existing device.

Figure 43 shows the SRDF pair state required when replacing R2, the R11 and its existing R2 device. For the purpose of this discussion, this is the first leg of the concurrent SRDF relationship for both R1 and R2 migrations.

Table 23 shows the applicable pair states for `symrdf migrate -replace` for an R1 and an R2 migration.

Table 23 SRDF migrate -replace control operation and applicable pair states

	Pair state: Existing ->R2											
Control operation:	SYNC IN PROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILED OVER	PARTITIONED[1]	PARTITIONED[2]	R1 UPDATED	R1 UPD IN PROG	INVALID	CONSISTENT	TRANSMIT IDLE
migrate -replace	P	P	P	P							P	

1. The remote array is in the SYMAPI database (it was discovered).

2. The remote array is not in the SYMAPI database (it was not discovered or was removed).

Figure 42 shows a sample concurrent SRDF configuration for an R1 migration:.

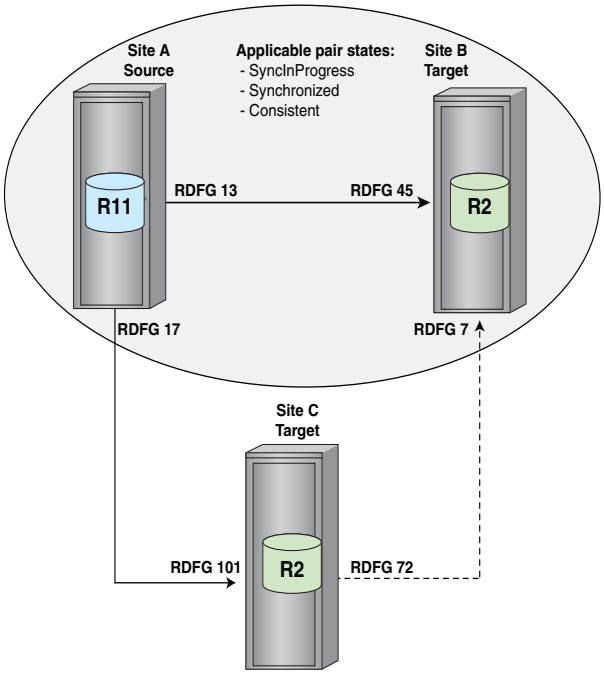


Figure 42 R1 migration: R11/R2 applicable pair states for migrate -replace (first leg)

The R11 in array A and the R2 device in array B must be in one of the applicable pair states before issuing the `symrdf migrate -replace` command.

Figure 43 shows a sample concurrent SRDF configuration for an R2 migration:

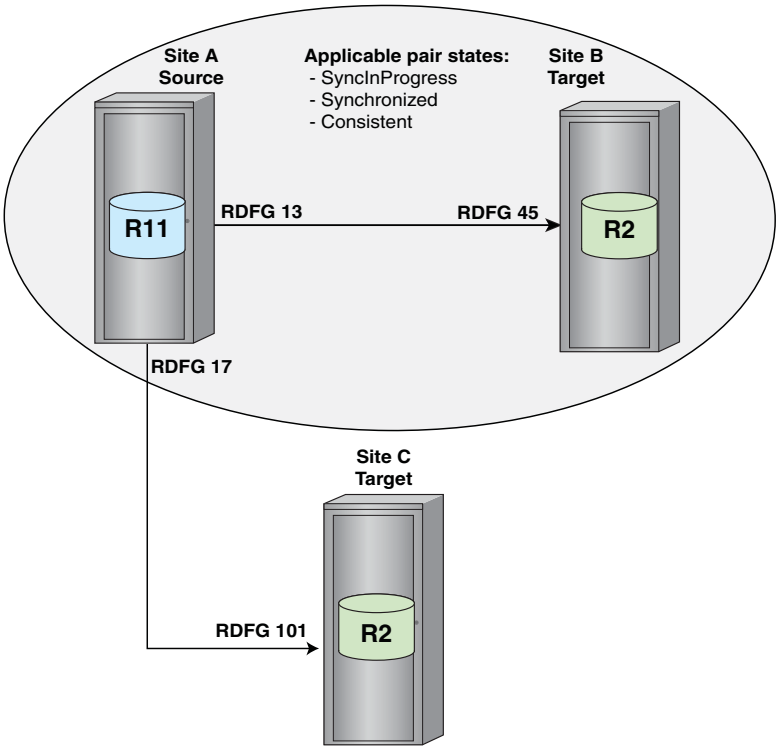


Figure 43 R2 migration:R11/R2 applicable pair states for migrate -replace (first leg)

The R11 in array A and the R2 device in array B must be in one of the states before issuing the `symrdf migrate -replace` command

Pair states for migrate -replace for second leg of concurrent SRDF

Before replacing an R1, the R11 and its replacement device must in a specific SRDF pair state shown in Figure 44. This temporary pairing was used to perform the concurrent SRDF data transfer to the new device. When replacing R2, the R11 and the new R2 device (new pair) must also be in a certain pair state shown in Figure 45.

Table 24 shows the applicable pair states for `symrdf migrate -replace` for an R1 and an R2 migration.

Table 24 SRDF migrate -replace control operation and applicable pair states

Control operation:	Pair state: Temporary or New ->R2											
	SYNC IN PROG	SYNCHRONIZED	SPLIT	SUSPENDED	FAILED OVER	PARTITIONED[1]	PARTITIONED[2]	R1 UPDATED	R1 UPD IN PROG	INVALID	CONSISTENT	TRANSMIT IDLE
migrate -replace	P	P									P	

1. The remote array is in the SYMAPI database (it was discovered).
2. The remote array is not in the SYMAPI database (it was not discovered or was removed).

Figure 44 shows a sample concurrent SRDF configuration for an R1 migration.

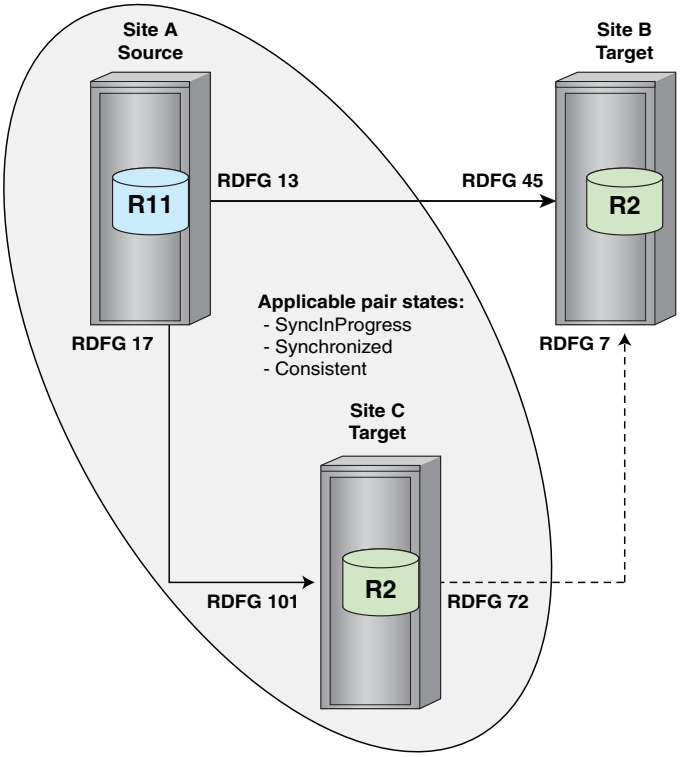


Figure 44 R1 migration: applicable R11/R2 pair states for migrate -replace (second leg)

The R11 device in array A and the R2 device in array C must be in one of the applicable pair states before issuing the `symrdf migrate -replace` command.

Figure 45 shows a sample concurrent SRDF configuration for an R2 migration:

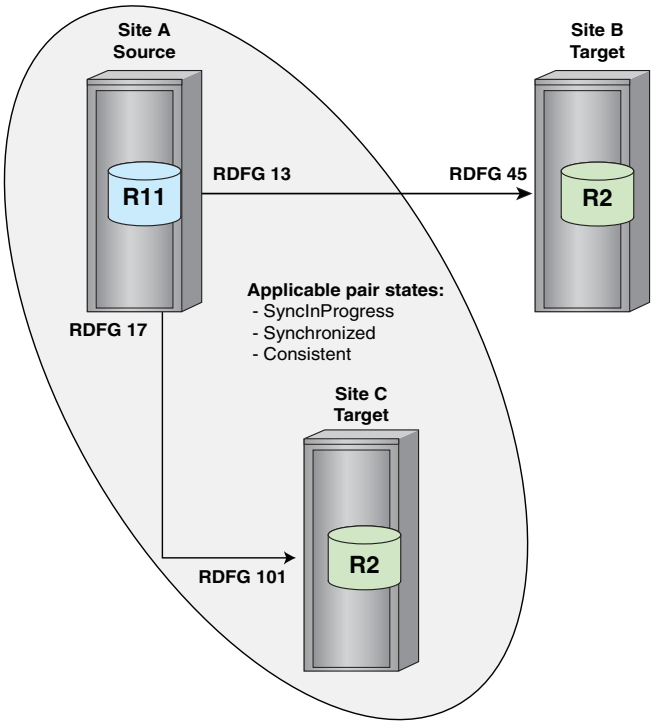


Figure 45 R2 migration: applicable R11/R2 pair states for migrate -replace (second leg)

The R11 in array A and the R2 device in array C must be in one of the states before issuing the `symrdf migrate -replace` command.

CHAPTER 8
SRDF/Automated Replication

This chapter describes the following topics:

- Overview 210
- Restrictions 210
- Configuring single-hop sessions 211
- Configuring multi-hop sessions 215
- Concurrent BCVs with SRDF/AR 217
- Setting replication cycle parameters 218
- Clustered SRDF/AR 220
- Setting symreplicate parameters in the options file 222
- Managing locked devices 228

Overview

SRDF/Automated Replication (SRDF/AR) provides a long-distance disaster restart solution. SRDF/AR can operate:

- In two-site topologies that use SRDF/DM in combination with TimeFinder.
- In three-site topologies that use a combination of SRDF/S, SRDF/DM, and TimeFinder.

  Three-site topologies operate in synchronous mode in the first hop and in adaptive copy mode in the second hop.

  Note: Note: Multi-hop SRDF/AR requires Enginuity version 5876.159.102 or higher.

SRDF/AR provides automated consistent replication of data from standard devices and RDF1 BCV devices over SRDF links to remote SRDF pairs.

SRDF/AR is invoked using the `symreplicate` command.

- `symreplicate` supports single-hop and multi-hop SRDF configurations.
- You can start, stop, or restart a `symreplicate` session without degrading the data copy.
- You can set up a concurrent BCV to have access to an independent copy of the replicating data during a `symreplicate` session.

By default, the `symreplicate` replication process is performed in the background.

Restrictions

- SRDF/AR does not support SRDF/Asynchronous-capable devices.
- The `symreplicate` command operates on device groups and composite groups.

  Scope for the `symreplicate` command cannot be limited to a specific SRDF group using the `-rdfg` option.
- When running `symreplicate` against device groups and composite groups of type ANY:
  - Concurrent SRDF devices are not supported for device groups (DG) or composite groups (CG).
  - The following combinations of standard devices are supported when using the `-consistent` option:
    - All STDs are non-SRDF
    - All STDs are R1 devices
    - All STDs are R2 devices
    - STDs contain a mixture of R1s and non-SRDF devices
    - STDs contain a mixture of R2 and non-SRDF devices

Note: Device external locks in the array are held during the entire `symreplicate` session. Locks are necessary to block other applications from altering device states while the session executes. "Managing locked devices" on page 228 provides more information.

SRDF/Automated Replication operations

Configuring single-hop sessions

Figure 46 shows how `symreplicate` copies data in a single-hop configuration for a complete copy cycle:

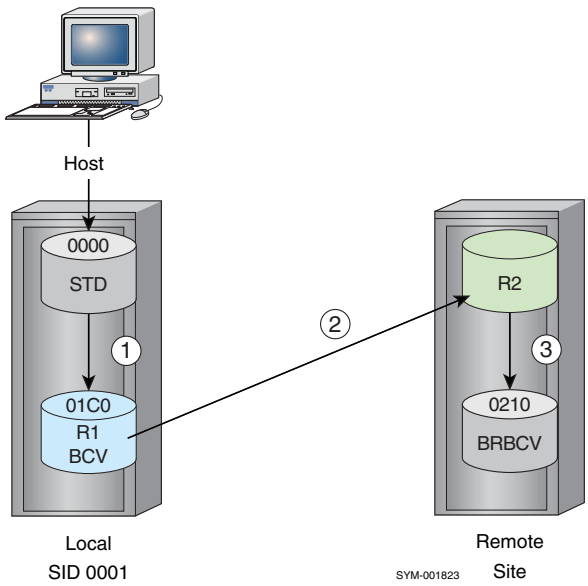


Figure 46 Automated data copy path in single-hop SRDF systems

The copy process includes the following steps:

- **1** - From the standard device to the BCV of the local array.
- **2** - From the BCV device of the local array to the standard device of the remote array.
- **3** - From the remote standard device to its BRBCV device.

Before you begin: setting the hop type parameter

You must set the replication type parameter in the replicate options file before you can configure a single-hop `symreplicate` session.

"SYMCLI_REPLICATE_HOP_TYPE=‹RepType›" on page 224provides more information.

Set the parameter as follows:

```
SYMCLI_REPLICATE_HOP_TYPE=SINGLE
```

The `symreplicate` session:

- Incrementally establishes SRDF and BCV pairs, and

- Differentially splits BCV pairs to reduce data transfers.

Setting up single-hop data replication

To set up a single-hop `symreplicate` session:

1. Select any number of standard devices of the same type (R1, R2, or non-SRDF).
2. Use the `symdg create` command to create a device group or composite group of the same type.

   ```
   symdg create newdg
   ```

3. Use the `symdg add dev` command to add the devices to the device group.

   ```
   symdg add dev 0000 -g newdg -sid 35002
   symdg add dev 0001 -g newdg
   ```

4. Use the `symbcv associate` command to associate an equal number of R1-BCV devices of matching sizes.

   ```
   symbcv associate dev 01C0 -g newdg
   symbcv associate dev 01C1 -g newdg
   ```

5. Use the `symbcv associate` command to associate an equal number of BRBCV devices (remote BCVs), also of matching sizes.

   ```
   symbcv associate dev 0210 -g newdg -bcv -rdf
   symbcv associate dev 0211 -g newdg -bcv -rdf
       .
       .
       .
   ```

Note: The `symreplicate` command uses composite groups (`-cg`) to implement single-hop or multi-hop configurations for devices that span multiple arrays.

The following must be true before you start a `symreplicate` session:

- Both sets of BCV pairs must have a pairing relationship.
- The local BCV pairs must be established.
- The SRDF pairs must be in the Suspended pair state.
- The remote BCVs (BRBCVs) must be in the split pair state.
- No writes are allowed to the BRBCV by any directly attached host at the remote site.

Setting up pair states automatically

You can set up the required pair state pair for SRDF/AR automatically using either:

- `symreplicate setup` command
- `symreplicate start` command with the `-setup` option

Auto-replication setup sets up the required pair states for devices and executes one copy (auto-replication) cycle.

Setting up the device states ahead of time reduces replication processing time.

The `setup` commands execute one cycle of the `symreplicate` session (regardless of the number of cycles defined in the options file), and then exits.

The default setup operation provides no I/O optimization, and does not engage any special algorithm changes in the selection of pair assignments. For standard devices encountered without BCVs, the first unassigned BCV device found is paired with the standard.

Setup operations correct only pair states of devices in the group. If a BCV in the group is paired with a standard device outside of the group, setup does not correct it.

The setup command does not exit until the devices are in the required pair state to run the `symreplicate` session. This may take some time.

Note: Optionally, you can manually reproduce the single-hop replication cycle using a sequence of SRDF and TimeFinder CLI commands.

The following topics provide more information:

- "Manually setting up single hop" on page 214
- "Manually setting up multi-hop" on page 216
- "Setting the symreplicate control parameters" on page 224

Examples To execute the `symreplicate setup` command on a device group (`DevGrp1`) using an options file (`OpFile`):

```
symreplicate -g DevGrp1 setup -options Opfile
```

The first cycle of the `symreplicate start -setup` command puts the devices into the required pair state.

To execute the `symreplicate start` command with the `-setup` option:

```
symreplicate -g DevGrp1 start -options Opfile -setup
```

Using the -exact option

Use the `-exact` option to start the `symreplicate` session with the STD-BCV pair relationships in the exact order that they were associated/added to the device group or composite group.

Using the -optimize option

Use the `-optimize` option in conjunction with the `-setup` option or the `setup` argument to optimize the disk I/O on standard/BCV pairs in the device or composite group.

The `-optimize` option splits all pairs and performs an optimized STD-BCV pairing within the specified group.

If you use the `-optimize` option with device groups, the device pair selection attempts to distribute I/O by pairing devices in the group that are not on the same disk adapter.

Example

```
symreplicate setup -g DgName -optimize
```

Use the `-optimize` option with composite groups to specify the same pairing behavior for an RA group.

Use the `-optimize_rag` option with either the `-setup` option or the `setup` argument to configure pair assignments for RA groups that provide remote I/O optimization (distribution by using different remote disk adapters).

Example: **symreplicate setup -cg *CgName* -optimize_rag**

Note: Single-hop replication does a full optimization on all RA groups.

Using the consistent split option

Use the `-consistent` option with the `start` action to:

- Consistently split all of the BCV pairs on the local array in a typical SRDF configuration
- Consistently split all of the BCV pairs on the Hop 1 remote array in a multi-hop configuration.

Note: This requires a TimeFinder/CG license.

Consistent split operations are automatically retried if the split fails to complete within the allotted window. If a consistent split operation fails due to the consistency timing window closing before the split can complete (`SYMAPI_C_CONSISTENCY_WINDOW_CLOSED`):

- The first-hop local BCV device pairs are automatically resynchronized, and
- The split operation is reattempted.

The consistent split error recovery operation is attempted the number of times specified in the `SYMCLI_REPLICATE_CONS_SPLIT_RETRY` file parameter, defined in the replicate options file.

If a value is not specified, then the recovery operation is attempted 3 times before terminating the `symreplicate` session.

"SYMCLI_REPLICATE_CONS_SPLIT_RETRY=‹NumRetries›" on page 227 provides more information.

Manually setting up single hop

To manually reproduce the single-hop replication cycle using a sequence of SRDF and TimeFinder CLI commands:

1. Wait for any ongoing establish to complete.
2. Split the BCV pairs:

   symmir split -g newdg

3. Establish the SRDF pairs:

   symrdf establish -g newdg -bcv

4. Wait for any ongoing establish to complete.
5. Suspend the SRDF pairs:

   symrdf suspend -g newdg -bcv

6. Establish the BCV pairs:

   symmir establish -g newdg -exact

7. Establish the remote BRBCV pairs:

   symmir establish -g newdg -bcv -rdf -exact

8. Wait for any ongoing establish to complete.
9. Split the remote BRBCV pairs:

```
symmir split -g newdg -bcv -rdf
```

Note: You may have to include additional command options in some of the above steps (for example, `establish -full` for BCV pairs without relationships).

Configuring multi-hop sessions

Figure 47 shows a complete `symreplicate` copy cycle in a multi-hop configuration:

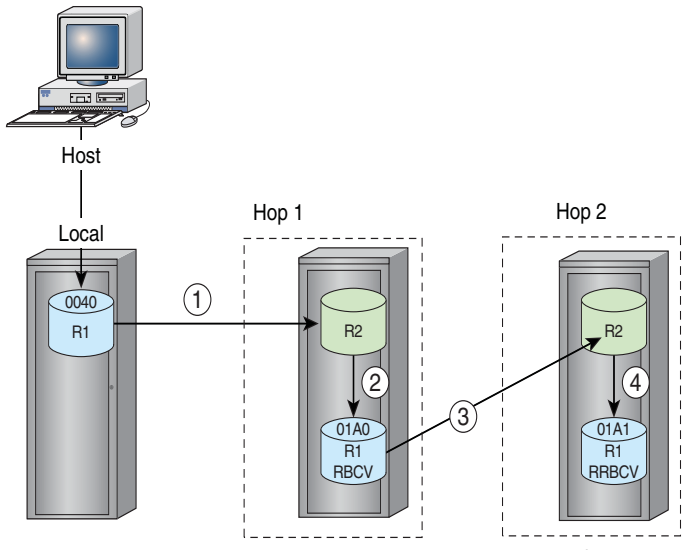


Figure 47 Automated data copy path in multi-hop SRDF

Data copy paths in Figure 47 are:

- **1** - From the local standard device to a standard device on the array at Hop 1
- **2**- From the Hop 1 standard device to its BCV (RBCV)
- **3** - From the RBCV device at Hop 1 to the standard device on the array at Hop 2
- **4** - From the Hop 2 standard device to its BCV (RRBCV)

  Path 2d requires a BCV in the array at Hop 2. The BCV must not be disabled.

Before you begin: setting the hop type and use final parameters

Set the replication type parameter in the replicate options file before you configure a multi-hop `symreplicate` session.

Set the parameter as follows:

```
SYMCLI_REPLICATE_HOP_TYPE=MULTI
```

"SYMCLI_REPLICATE_HOP_TYPE=‹RepType›" on page 224 provides more information.

Set the replication use final BCV parameter in the replicate options file to `FALSE` to prevent the final Hop 2 BCV from being updated:

```
SYMCLI_REPLICATE_USE_FINAL_BCV=FALSE
```

"Setting the symreplicate control parameters" on page 224 provides more information.

Setting up for a multi-hop configuration

To set up a multi-hop `symreplicate` session:

1. Use the `symdg create` command to create an R1 device group (`-g`) or composite group (`-cg`).

   `symdg create newdg2 -type RDF1`

2. Use the `symdg add dev` command to add any number of R1 devices.

   `symdg add dev 0040 -g newdg2 -sid 0001`

3. Use the `symdg add dev` command to remotely associate an equal number of matching sized R1-BCVs or Hop 1 RBCV devices.

   ```
   symbcv associate dev 01A0 -g newdg2 -rdf
   symbcv associate dev 01A1 -g newdg2 -rrdf
   ```

The following must be true before you start a `symreplicate` session without a setup operation:

- The local SRDF pairs must be synchronized
- The BCV pairs must be established
- The remote SRDF pairs must be suspended.
- If the final BCVs in the second-hop array are used, the BCVs must be in the split state.

Device pair state can be configured automatically using the `symreplicate setup` command or the `-setup` option with the `symreplicate start` command.

"Setting up pair states automatically" on page 212 provides more information.

Manually setting up multi-hop

To manually reproduce the multi-hop replication cycle using a sequence of SRDF and TimeFinder CLI commands:

1. Wait for any ongoing establish to complete.
2. Split the BCV pairs (2b in Figure 47):

   `symmir split -g newdg2 -rdf -remote`

   The `-remote` option specifies that the remote SRDF pairs establish.

3. Wait for the establish to complete.
4. Suspend the remote SRDF pairs (2c in Figure 47), and establish the BCV pairs (2b in Figure 47):

   `symmir establish -g newdg2 -rdf -exact`

5. Use either a device file or the `-rrbcv` option to establish the BCV pairs in the second hop (2d in Figure 47):

   `symmir establish -f 2nd_hop_devs.txt -sid SymmID`

or

```
symmir establish -g newdg2 -rrbc
```

Note: To use the `-rrbcv` option, the SRDF BCV devices must have been previously associated with the group, using `symbcv -rrdf`.

6. Wait for any ongoing establish to complete.
7. Split the 2nd hop BCV pairs:

```
symmir split -f 2nd_hop_devs.txt
```

or

```
symmir split -g newdg2 -rrbcv
```

Perform Steps 5 and 7 when you want to use the final hop 2 BCVs in the replicate cycle.

You can use the `-preaction` and `-postaction` options to specify scripts for `symreplicate` to run before and after splitting the BCVs (step 2).

Note: You may have to include additional command options in some of the above steps (such as `establish -full` for BCV pairs without relationships).

Concurrent BCVs with SRDF/AR

Set up concurrent BCVs if you need an independent copy of your data during a replication cycle.

- One BCV copy is associated with the SRDF/AR device group and
- The other BCV copy is not.

  The BCV not associated with the replication cycle receives the same data as the one associated with the SRDF/AR devices. This BCV can be accessed by its host during the `symreplicate` cycle.

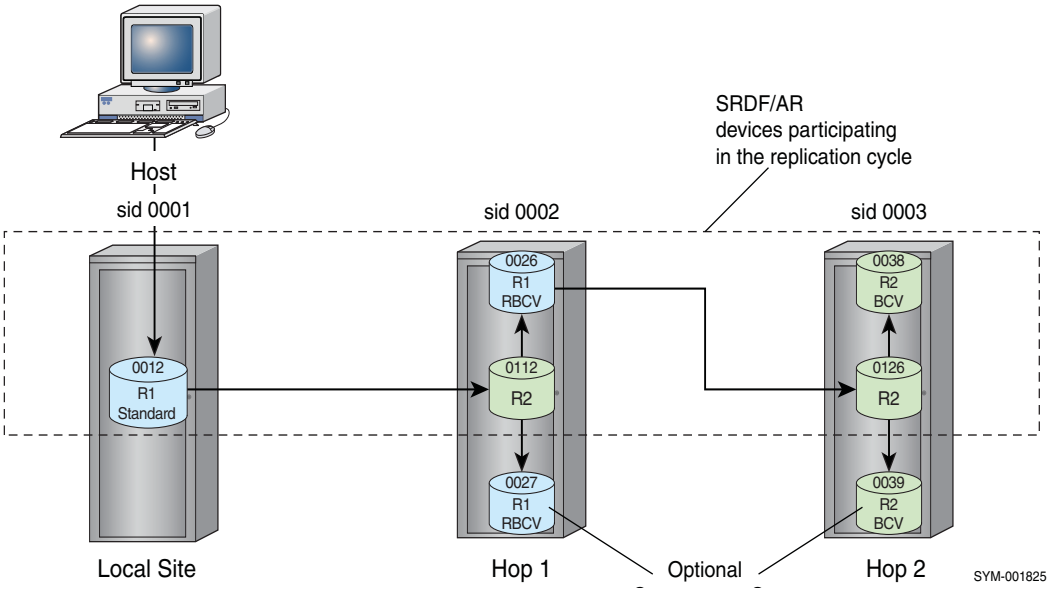


Figure 48 Concurrent BCV in a multi-hop configuration

In Figure 48, Devices 0027 and 0039 are not part of the SRDF/AR copy cycle.

To access these devices from the production host during the SRDF/AR copy cycle, you must define separate device files on the host that include the standard R2 device and the R2 BCV on Hop 1 and Hop 2.

The device files are used to establish the BCV pairs, split BCV pairs, and access the BCV devices.

Setting replication cycle parameters

You can manipulate the replication cycle patterns to fit your needs by setting the following parameters in the symreplicate options file:

`SYMCLI_REPLICATE_CYCLE=`*`CycleTime`*

CycleTime is a timer that indicates the period of time in *minutes* or *hours*:*minutes*(*hh*:*mm*) between when each copy action starts and when it starts again (how often the copy reoccurs). For example, a *CycleTime* of 120 would initiate a new copy every 2 hours.

`SYMCLI_REPLICATE_NUM_CYCLES=`*`NumCycles`*

NumCycles indicates the number of replication cycles (copies) to perform before `symreplicate` exits. For example, a zero value (the default value) results in continuous cycling until the `symreplicate stop` command is issued.

`SYMCLI_REPLICATE_CYCLE_DELAY=`*`Delay`*

Delay indicates the minimum amount of time to wait between the end of one copy cycle and the beginning of the next. For example, a *Delay* of 20 would always force a wait of 20 minutes or more between cycles.

`SYMCLI_REPLICATE_CYCLE_OVERFLOW=`*`OvfMethod`*

OvfMethod indicates how to behave when the actual copy time of your data and/or data transfer throughput is so large as to exceed the *CycleTime* value. Here, the initial copy event has overflowed into the period that should be for the next copy cycle. Possible behavior values are:

`IMMEDIATE` — When overflowed, starts a new cycle immediately after the current copy finishes.

`NEXT` — When overflowed, waits for the copy to finish, and then starts at the next expiration time (*CycleTime*). (Starts the copies on multiples of the *CycleTime* parameter.)

For example, if a 1-hour copy cycle completed in 1.5 hours, the next cycle could be set to begin immediately (`IMMEDIATE`) or in half an hour (`NEXT`).

Setting the first time cycle parameters

You may not have enough information to set the exact cycle time parameters when you first create the SRDF configuration.

Best practice

- Start the `symreplicate` session with the basic parameters set.
- Use `symreplicate query` to monitor session progress, and record the timing results of the initial copies.
- Adjust the various timing parameters to best accommodate the copy requirements for your needs.

Table 25 provides two parameter setups for an initial `symreplicate` session trial:

Table 25 Initial setups for cycle timing parameters

SYMCLI_REPLICATE_CYCLE=60 SYMCLI_REPLICATE_CYCLE_DELAY=0 SYMCLI_REPLICATE_CYCLE_OVERFLOW=NEXT	Every hour if possible, or every 2, or 3 hours based on data throughput and size.
SYMCLI_REPLICATE_CYCLE=0 SYMCLI_REPLICATE_CYCLE_DELAY=60	Cycle through the first copy, then wait 60 minutes (delay), and then another cycle, delay, and so on.

Viewing cycle time and invalid track statistics

Use the `symreplicate stats` command to display statistical information for cycle time and invalid tracks.

Use the command to display cycle time and invalid tracks for a specified:

- Device group (`-g`)
- Composite group (`-cg`)
- Symmetrix ID (`-sid`)

Use the `-log` option to write information to a specified log file.

Use the `-cycle` option to display only cycle time statistics for the last SRDF/AR cycle time, the maximum cycle time and the average cycle time.

Use the `-itrks` option to display only invalid track statistics for the last SRDF/AR cycle, the maximum invalid tracks and the average number of invalid tracks per SRDF/AR cycle.

Use the `-all` option (default) to display both the cycle time and invalid tracks statistics.

Example To display both cycle time and invalid track statistics for device group `srdfar` on SID 1123:

```
symreplicate -g srdfar -sid 123 -all stats

Group Name: srdfar

Cycle Time (hh.mm.ss):
---------------------------------------

Last Cycle Time:  06:10:01
Max Cycle Time:   08:00:00
Avg Cycle time:   06:00:00

Invalid Tracks:
---------------------------------------

Last Cycle:  12345 ( 9055.5 MB)
Maximum:     10780 ( 8502.3 MB)
Average:     11562 ( 7500.0 MB)
```

Logging symreplicate steps

To track the steps in a `symreplicate` session, set the log step entry in the options file to `TRUE`:

```
SYMCLI_REPLICATE_LOG_STEP=TRUE
```

When this option is enabled, `symreplicate` writes an entry to the SYMAPI log file after each step is completed.

Log entries contain the time that the step ended and whether it was successful.

"SYMCLI_REPLICATE_LOG_STEP=‹TRUE|FALSE›" on page 225 provides more information.

Clustered SRDF/AR

Clustered SRDF/AR enables you to start, stop, and restart `symreplicate` sessions from any host connected to any local array participating in the `symreplicate` session.

In the clustered SRDF/AR environment, you can write the replication log file directly to the Symmetrix File System (SFS) instead of the local host directory of the node that began the session.

If the primary node should fail, then any locally attached host to the array containing the log file can restart the SRDF/AR session from where it left off.

Writing log files to a specified SFS

Use the `symreplicate start` command with the `-sid` and `-log` options to write the log file to the SFS. You must specify:

- `-sid` – ID of the array where the log file is to be stored at the start of the `symreplicate` session.

- `-g` or `-cg` - Group name.
- `-log` *`LogFilename`* - (Optional) User log filename.

Usage notes

- If you do not specify the Symmetrix ID (`-sid`) at the start of the session, the log file is written to local disk using the default SYMAPI log directory. This is not restartable from another node.
- If you begin a session and specify a user log file name (`-log` *`LogFilename`*), you must specify the `-log` option for all other commands in the session sequence.
- If you begin a session and specify only the group name (`-g`, `-cg`):
  - The log file is given the same name as the group,
  - You must specify only the `-g` or `-cg` option for all other commands in the session sequence.

HYPERMAX OS

- In HYPERMAX OS/Solutions Enabler 8.0.1, the following options for the `symreplicate start` command are not supported, and the command fails with the message “Illegal option”.
  - - vxfs
  - -rdb

To write the log file for device group `session1` to a file named `srdfar1.log` at the SFS on array 201:

```
symreplicate start -g session1 -log srdfar1.log -sid 201
```

Restarting from another host

When log files are sent to the SFS, then any locally attached host to the array containing the log file can restart the SRDF/AR session from where it left off.

Use the `symreplicate restart` command with the `-recover` option to restart the session using the specified log and recover the device locks from the previous session.

Command usage notes

The `-recover` option recovers the device locks from the previously started session. Verify that no other currently running `symreplicate` session is using the same devices before using the `-recover` option.

You do not need to specify the device or composite group name (`-g`, `-cg`) on the host where the session is restarted.

Example

To restart the SRDF/AR session from another local host:

```
symreplicate restart -g session1 -log srdfar1.log -sid 201 -recover
```

Displaying log files written to the SFS

symreplicate list

Use the `symreplicate list` command with the `-sid` option to display a list of the current SRDF/AR log files written to the SFS at the specified SID.

Use the `symreplicate list` command with the `-sort` option to sort the log file list by name (default) or type.

Example To list the log files at SID 201:

```
symreplicate list -sid 201
```

symreplicate show

Use the `symreplicate show` command to display the information content of a particular log file.

Usage notes

- Log filename (`-log`) and the Symmetrix ID (`-sid`) are required.
- Use the `-args` option to display only command line arguments.
- Use the `-devs` option to display only devices.
- Use the `-opts` option to display only options.
- Use the `-all` option (default) to display all available information contained in the log.

EMC Solutions Enabler Symmetrix CLI Command Reference provides more information.

Example To display the log file `srdfar1.log` at SID 201:

```
symreplicate show -log srdfar1.log -sid 201 -all
```

Deleting a log file written to SFS

Use the `symreplicate delete -log` command to delete the specified log file written to SFS.

Usage notes

Specify either the group name (`-g`, `-cg`) or the log filename (`-log`) depending on whether you defined a user log name when you began the session.

Example To delete log file `srdfar1.log` written to the SFS:

```
symreplicate delete -log srdfar1.log
```

Setting symreplicate parameters in the options file

You can modify parameters in the `symreplicate` options file to:

- Set replication retry and sleep timers
- Control replicate behavior

Note: If you specify an options file on `restart`, you may not change the following options:
"SYMCLI_REPLICATE_USE_FINAL_BCV=‹TRUE|FALSE›"
"SYMCLI_REPLICATE_HOP_TYPE=‹RepType›"
If you attempt to change these options, an error message is displayed. All other options may be changed, and the new values take effect immediately.

IMPORTANT

You must specify the *RepType*. See:

"SYMCLI_REPLICATE_HOP_TYPE=‹RepType›" on page 224.

Set a nonzero value for either a *CycleTime* or a *Delay* time, (even though their default values are zero). See:

"SYMCLI_REPLICATE_CYCLE=CycleTime" on page 218

"SYMCLI_REPLICATE_CYCLE_DELAY=Delay" on page 218

Option file format

Make sure that your changes conform to the syntax in the example below.

The desired value is entered for the italicized text.

Lines beginning with a "#" (comment) are ignored by SYMCLI:

```
#Comment
SYMCLI_REPLICATE_HOP_TYPE=<RepType>
SYMCLI_REPLICATE_CYCLE=<CycleTime>
SYMCLI_REPLICATE_CYCLE_OVERFLOW=<OvfMethod>
SYMCLI_REPLICATE_CYCLE_DELAY=<Delay>
SYMCLI_REPLICATE_NUM_CYCLES=<NumCycles>
SYMCLI_REPLICATE_USE_FINAL_BCV=<TRUE|FALSE>
SYMCLI_REPLICATE_LOG_STEP=<TRUE|FALSE>
SYMCLI_REPLICATE_GEN_TIME_LIMIT=<TimeLimit>
SYMCLI_REPLICATE_GEN_SLEEP_TIME=<SleepTime>
SYMCLI_REPLICATE_RDF_TIME_LIMIT=<TimeLimit>
SYMCLI_REPLICATE_RDF_SLEEP_TIME=<SleepTime>
SYMCLI_REPLICATE_BCV_TIME_LIMIT=<TimeLimit>
SYMCLI_REPLICATE_BCV_SLEEP_TIME=<SleepTime>
SYMCLI_REPLICATE_MAX_BCV_SLEEP_TIME_FACTOR=<Factor>
SYMCLI_REPLICATE_MAX_RDF_SLEEP_TIME_FACTOR=<Factor>
SYMCLI_REPLICATE_PROTECT_BCVS=<Protection>
SYMCLI_REPLICATE_TF_CLONE_EMULATION=<TRUE|FALSE>
SYMCLI_REPLICATE_PERSISTENT_LOCKS=<TRUE|FALSE>
SYMCLI_REPLICATE_CONS_SPLIT_RETRY=<NumRetries>
SYMCLI_REPLICATE_R1_BCV_EST_TYPE=<EstablishType>
SYMCLI_REPLICATE_R1_BCV_DELAY=<EstablishDelay>
SYMCLI_REPLICATE_FINAL_BCV_EST_TYPE=<EstablishType>
SYMCLI_REPLICATE_FINAL_BCV_DELAY=<EstablishDelay>
SYMCLI_REPLICATE_ENABLE_STATS=<TRUE|FALSE>
SYMCLI_REPLICATE_STATS_RESET_ON_RESTART=<TRUE|FALSE>
```

Setting replication retry and sleep times

Control how long and how often `symreplicate` executes control operations by setting the following parameters in the `symreplicate` options file:

`SYMCLI_REPLICATE_GEN_TIME_LIMIT=`*`TimeLimit`*

Controls how long errors of a general nature, such as waiting for a lock, are retried.

`SYMCLI_REPLICATE_RDF_TIME_LIMIT=`*`TimeLimit`*

Controls how long to wait for SRDF devices to enter a specific state.

`SYMCLI_REPLICATE_BCV_TIME_LIMIT=`*`TimeLimit`*

Controls how long to wait for BCV devices to enter a specific state.

`SYMCLI_REPLICATE_GEN_SLEEP_TIME=`*`SleepTime`*

Controls how long `symreplicate` should sleep before retrying a general operation.

`SYMCLI_REPLICATE_RDF_SLEEP_TIME=`*`SleepTime`*

Controls the minimum time `symreplicate` should sleep before retrying an SRDF operation.

`SYMCLI_REPLICATE_BCV_SLEEP_TIME=`*`SleepTime`*

Controls the minimum time `symreplicate` should sleep before retrying a BCV operation.

`SYMCLI_REPLICATE_MAX_BCV_SLEEP_TIME_FACTOR=`*`Factor`*

Controls the maximum time that `symreplicate` sleeps before checking the BCV device state.

`SYMCLI_REPLICATE_MAX_RDF_SLEEP_TIME_FACTOR=`*`Factor`*

Controls the maximum time that `symreplicate` sleeps before checking the SRDF device state.

Setting the symreplicate control parameters

You can modify the following parameters in the `symreplicate` options file to control replicate behavior:

`SYMCLI_REPLICATE_HOP_TYPE=<`*`RepType`*`>`

Defines your configured environment in which to operate the data `symreplicate` session. This parameter is not optional and must be specified. Possible *RepType* values are:

`SINGLE` — Single-hop configuration

`MULTI` —Multi-hop configuration

`SYMCLI_REPLICATE_USE_FINAL_BCV=<`*`TRUE`*`|`*`FALSE`*`>`

Indicates whether to update the BCV in the final (last) remote array (for multi-hop only).

`TRUE` –(default) Replicates data copy the BCV in the final (last) remote array.

`FALSE` -The second hop BCV devices will be omitted.

```
SYMCLI_REPLICATE_PROTECT_BCVS=
          <NONE|BOTH|LOCAL|REMOTE|FIRST_HOP|SECOND_HOP>
```

`NONE` - (default) Establishes BCV-STD pairs without the protective establish behavior, relating to two-way mirrored BCV devices.

`LOCAL` or `REMOTE` - Causes the two mirrors of the BCV to be moved or joined to the standard device.

`BOTH` - Both the local BCV mirrors and the remote BCV mirrors get joined to their standard device.

`FIRST_HOP` or `SECOND_HOP` - Performs the protect BCV establish for first or second hop devices only in a multi-hop configuration.

`SYMCLI_REPLICATE_CYCLE=<CycleTime>`

Defines the period to wait between copy operations in total *minutes* or in an *hours*:*minutes* (*hh:mm*) format.

`SYMCLI_REPLICATE_CYCLE_DELAY=<Delay>`

Specifies the minimum time to wait between adjacent cycles. Even if a cycle overruns the specified *CycleTime* and *OvfMethod* is set to `IMMEDIATE` when *Delay* is specified, the session waits this delay time before beginning another cycle.

`SYMCLI_REPLICATE_NUM_CYCLES=<NumCycles>`

Specifies the number of cycles to perform before exiting. If you specify a value of zero, the `symreplicate` session cycles forever. The *NumCycles* default value is zero.

`SYMCLI_REPLICATE_CYCLE_OVERFLOW=<OvfMethod>`

Describes what to do if the cycle overruns the specified *CycleTime*. Possible *OvfMethod* values are:

`IMMEDIATE` — Begins next cycle immediately (the default)

`NEXT` — Skips this copy cycle and wait for the next to begin

`SYMCLI_REPLICATE_LOG_STEP=<TRUE|FALSE>`

`TRUE` - Writes a log entry to the SYMAPI log file after each step of the `symreplicate` cycle is completed. The entry displays the time that the step ended and whether the step was successful.

`SYMCLI_REPLICATE_GEN_TIME_LIMIT=<TimeLimit>`

Indicates how long errors of a general nature should be retried (for example, attempting to acquire a array lock). Currently, the general *TimeLimit* only applies when initiating an SRDF split or establish operation.

The default general *TimeLimit* is 00:30 if not specified.

The *TimeLimit* value enables you to control how long `symreplicate` retries certain types of operations. *TimeLimit* must be specified using one of the following formats:

hh:mm — Specifies the number of hours and minutes

sss — Specifies the number of seconds

A *TimeLimit* specified as zero (0) indicates that no time limit applies, causing the operation to be retried indefinitely.

`SYMCLI_REPLICATE_RDF_TIME_LIMIT=<TimeLimit>`

Indicates how long to wait for SRDF devices to enter a specific state. For example, after successfully issuing the command to establish an R2 BCV device with the corresponding R1 standard device, `symreplicate` waits the indicated length of time for the devices to become synchronized.

The default SRDF *TimeLimit* is 04:00 if not specified.

`SYMCLI_REPLICATE_BCV_TIME_LIMIT=<TimeLimit>`

Indicates how long to wait for BCV devices to enter a specific state. For example, after successfully issuing the command to establish a BCV device with the corresponding standard device, `symreplicate` waits the indicated length of time for the devices to become synchronized.

The default BCV *TimeLimit* is 02:00 if not specified.

`SYMCLI_REPLICATE_GEN_SLEEP_TIME=<SleepTime>`

Indicates how long `symreplicate` should sleep before retrying a general operation (for example, attempting to acquire a array lock). Currently, the general *SleepTime* only applies when initiating an SRDF split or establish operation. The default general *SleepTime* is 10 seconds if not specified.

The *SleepTime* value enables you to control how long `symreplicate` sleeps before retrying certain types of operations. *SleepTime* must be specified using one of the following formats:

hh:mm — Specifies the number of hours and minutes

sss — Specifies the number of seconds

A *SleepTime* must be specified as greater than zero (0).

`SYMCLI_REPLICATE_RDF_SLEEP_TIME=<SleepTime>`

Indicates the minimum length of time that `symreplicate` should sleep before retrying an SRDF device operation. For example, after issuing the command to establish an R2 BCV device with the corresponding R1 standard device, `symreplicate` sleeps the indicated length of time before retrying the operation.

The default SRDF *SleepTime* is 15 seconds if not specified.

`SYMCLI_REPLICATE_BCV_SLEEP_TIME=<SleepTime>`

Indicates the minimum length of time that symreplicate should sleep before retrying a BCV device operation. For example, after issuing the command to establish a BCV device with the corresponding standard device, `symreplicate` sleeps the indicated length of time before retrying the operation.

The default BCV *SleepTime* is 10 seconds if not specified.

`SYMCLI_REPLICATE_MAX_BCV_SLEEP_TIME_FACTOR=<Factor>`

Provides a way to specify the maximum time that `symreplicate` sleeps before checking again to see if BCV devices have entered a specific state. The product of this value multiplied by the sleep time gives the maximum time that `symreplicate` sleeps. The factor is specified using a nonzero integer. If not specified, the default factor is 3.

By default, `symreplicate` sleeps between 10 and 30 seconds when checking on the state of BCV devices, up to a maximum time of 2 hours.

`SYMCLI_REPLICATE_MAX_RDF_SLEEP_TIME_FACTOR=<Factor>`

Provides a way to specify the maximum time that `symreplicate` sleeps before checking again to see if SRDF devices have entered a specific state. The product of this value multiplied by the sleep time gives the maximum time that `symreplicate` sleeps. The factor is specified using a nonzero integer.

If not specified, the default factor is 4.

Note: By default, `symreplicate` sleeps between 15 and 60 seconds when checking on the state of SRDF devices, up to a maximum time of 4 hours.

`SYMCLI_REPLICATE_TF_CLONE_EMULATION=<TRUE|FALSE>`

Indicates that TF/Clone emulation is enabled/disabled.

`FALSE` – (default) The TF/Clone emulation default is disabled.

`TRUE` - Indicates that clone emulation is enabled.

`SYMCLI_REPLICATE_PERSISTENT_LOCKS=<TRUE|FALSE>`

Allows device locks to persist in the event of a system crash or component failure.

`TRUE` - Causes symreplicate to acquire the device locks for the `symreplicate` session with the `SYMAPI_DLOCK_FLAG_PERSISTENT` attribute.

`FALSE` - The persistent attribute will not be used to acquire the device locks for the session. If the base daemon (storapi daemon) is running and persistent locks are not set, the base daemon will release the device locks in the event of a failure.

`SYMCLI_REPLICATE_CONS_SPLIT_RETRY=<NumRetries>`

Specifies the number of error recovery attempts that will be made when a consistent split operation fails because the timing window closed before the split operation completed. A default retry value of 3 will be used if the SYMCLI_REPLICATE_CONS_SPLIT_RETRY option parameter is not specified when a consistent split (`-consistent`) is requested. A retry value of 0 indicates that no retry attempts should be made.

`SYMCLI_REPLICATE_R1_BCV_EST_TYPE=<EstablishType>`

Specifies the establish type for the local/first hop BCV devices. *EstablishType* specifies the way that BCV establish operations will be executed by TimeFinder. One of the following values may be specified:

`SINGULAR` — BCV devices will be established one at a time; the next device will not be established until the previous device has been established.

`SERIAL` — BCV devices will be established as fast as the establish requests can be accepted by the array.

`PARALLEL` — BCV devices establish requests will be passed in parallel to each of the servicing DA directors.

`SYMCLI_REPLICATE_R1_BCV_DELAY=<EstablishDelay>`

Denotes how long to wait between issuing establish requests. Establish types of SINGULAR and PARALLEL, for an ‹*EstablishDelay*› can be specified through the SYMCLI_REPLICATE_R1_BCV_DELAY file parameter.

SYMCLI_REPLICATE_FINAL_BCV_EST_TYPE=<*EstablishType*>

Identifies the establish type for the remote/second hop BCV devices.

SYMCLI_REPLICATE_FINAL_BCV_DELAY=<*EstablishDelay*>

Indicates how long to wait between issuing establish requests for the remote/second hop BCV devices. For an establish type of PARALLEL the delay value indicates how long to wait before passing the next establish request to an individual servicing DA director. An establish delay of 0 to 30 seconds may be specified with a value of 0 being the default.

SYMCLI_REPLICATE_ENABLE_STATS=<*TRUE*|*FALSE*>

Enables or disables the gathering of statistics.

TRUE - (default) Indicates that statistics gathering is enabled.

FALSE - Indicates that statistics gathering is to be disabled.

SYMCLI_REPLICATE_STATS_RESET_ON_RESTART=<*TRUE*|*FALSE*>

Resets statistics when a restart action is executed.

TRUE - Indicates that statistics are to be reset when restarting a symreplicate session.

FALSE - (default) Statistics are not reset upon restart of a symreplicate session.

Managing locked devices

Device external locks in the array are held during the entire symreplicate session. Device external locks block other applications from altering device states while the symreplicate session executes.

When a symreplicate session terminates because the SRDF link goes down unexpectedly, the locked devices prevent session restart when the SRDF link is restored.

You can recover, release or acquired to persist device locks.

Recovering locks

Use the symreplicate start or restart command with the -recover option to recover the device locks and restart the session.

Note: Device locks can be recovered as long as exactly the same devices are still locked under the lock holder ID of the previous symreplicate session.

Releasing locks

Optionally, you can release the device external locks held in the array for a terminated SRDF/AR session.

Locks may need to be released manually if a session is terminated unexpectedly due to a system crash or component failure. Device locks for a terminated session can be released manually for a device group, composite group or log file without restarting the session.

Use the `symreplicate release` command to release any device external locks associated with devices in the specified device group that are still held from when they were locked from the terminated SRDF/AR session.

Example To release devices locks on a terminated session for device group `prod` on array `35002`:

```
symreplicate -g prod release -sid 35002
```

Restrictions to release locks

- The SRDF/AR session for the targeted devices must not be active.
- Devices must have been locked by the previous session and the lock holder ID must match the previous session's ID.
- The number of devices to be unlocked must be less than or equal to the total number of devices in the previous SRDF/AR session.

The force (`-force`) option is required to release device locks in the following situations:

- If the release action is requested in a clustered SRDF/AR environment on a host that did not initiate the session and the status of the session cannot be determined.
- If any of the devices' lock holder ID in the targeted SRDF/AR session do not match the session's lock hoder ID, and the user wants to release the devices locked with the session's lock holder ID.
- If the lock holder ID for some devices in the targeted SRDF/AR session do not match the lock holder ID of that session, and the user wants to release the devices locked with the session's original lock holder ID.

Acquiring persistent locks

If the base daemon (SYMAPI daemon) is running, device locks are automatically released in the event of a system crash or component failure.

To acquire the device using the persistent attribute, set the persistent locks parameter in the symreplicate options file to `TRUE`:

```
SYMCLI_REPLICATE_PERSISTENT_LOCKS=TRUE
```

See "SYMCLI_REPLICATE_PERSISTENT_LOCKS=‹TRUE|FALSE›" on page 227.

CHAPTER 9
TimeFinder and SRDF operations

This chapter describes the following topics:

- TimeFinder consistent splits in SRDF configurations 232
- Multi-hop operations 233
- Before you begin: system-wide device groups 233
- System-wide splits 235
- Controlling basic operations in a multi-hop configuration 236

TimeFinder consistent splits in SRDF configurations

TimeFinder consistent split allows you to split off a consistent, restartable copy of a database management array within seconds with no service interruption.

A concurrent split helps to avoid inconsistencies and restart problems that can occur when splitting database-related BCVs without first quiescing the database.

Consistent split operations are implemented using the Enginuity Consistency Assist (ECA) feature. ECA allows you to consistently activate copy sessions across multiple heterogeneous hosts.

Consistent split operations can also be used in conjunction with SRDF Automated Replication (SRDF/AR) to set up automatic remote mirroring according to a predefined copy schedule.

SRDF/AR is explained in "SRDF/Automated Replication" on page 209.

Enginuity Consistency Assist

TimeFinder consistent split operations are performed using the `symir` command with `-consistent` option.

You can also use the `symreplicate` command with the `-consistent` option to run a copy cycle that freezes I/O to all devices in a device or composite group for both single-hop and multi-hop configurations.

Figure 49 on page 233 depicts how a control host can perform ECA consistent splits for three database hosts that access devices on a array.

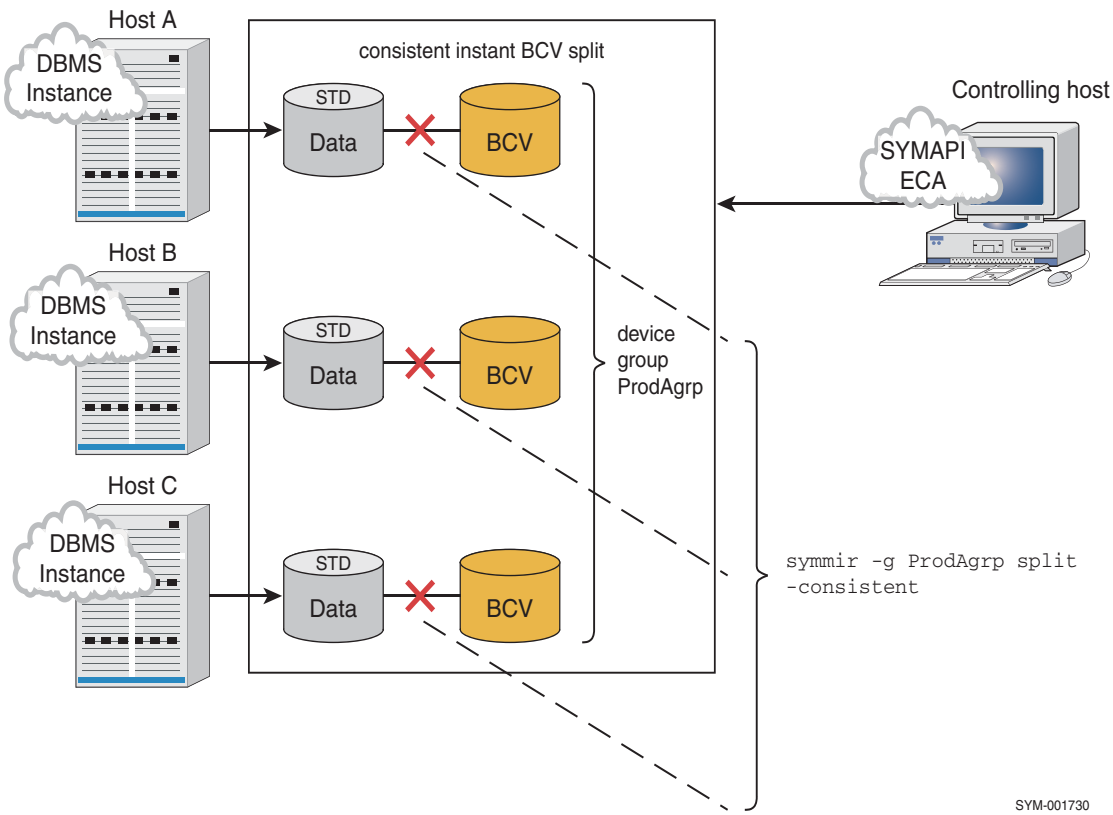


Figure 49 ECA consistent split

To consistently split BCV pairs using ECA:

- You must have either a control host with no database or a database host with a dedicated channel.
- Device or composite groups must be created on the controlling host for the target database to be consistently split.

  These 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, then you can consistently split all of the BCV pairs related to those hosts with a single command.

Multi-hop operations

You can manage various compounded remote configurations using both the TimeFinder and SRDF components of SYMCLI.

Figure 51 on page 237, shows multiple sites (remote Sites B and C) remotely mirroring to a local array at Site A.

The most typical configuration is a remote site (Site B in Figure 51), functioning as a remote mirror to standard devices (Site A).

A third site (Site C) can remotely mirror just the BCV devices at Site A.

Multi-hop SRDF

You can also configure a multi-hop to a second-level SRDF.

In Figure 51:

- Site D remotely mirrors standard devices at Site A, and
- Site E remotely mirrors Site A's BCV.
- The `symrdf` command manages the SRDF pairs within the SRDF link
- The `symmir` command manages the BCV pairs within any one site.

Before you begin: system-wide device groups

`symmir` operations require an existing group of SRDF devices.

To create a device group containing STD and BCV RDF1 devices:

1. Use the `symdg create` command to create an empty device group:

   ```
   symdg create prod -type RDF1
   ```

2. Use the `symdg add dev` command to add devices to the new device group:

   ```
   symdg -g prod add dev 0001 -sid 344402 DEV001
   ```

3. Use the `symbcv associate` commands to associate the devices with a local BCV, and remote BCVs:

```
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
```

All devices must be established with the `symmir` and `symrdf` commands.

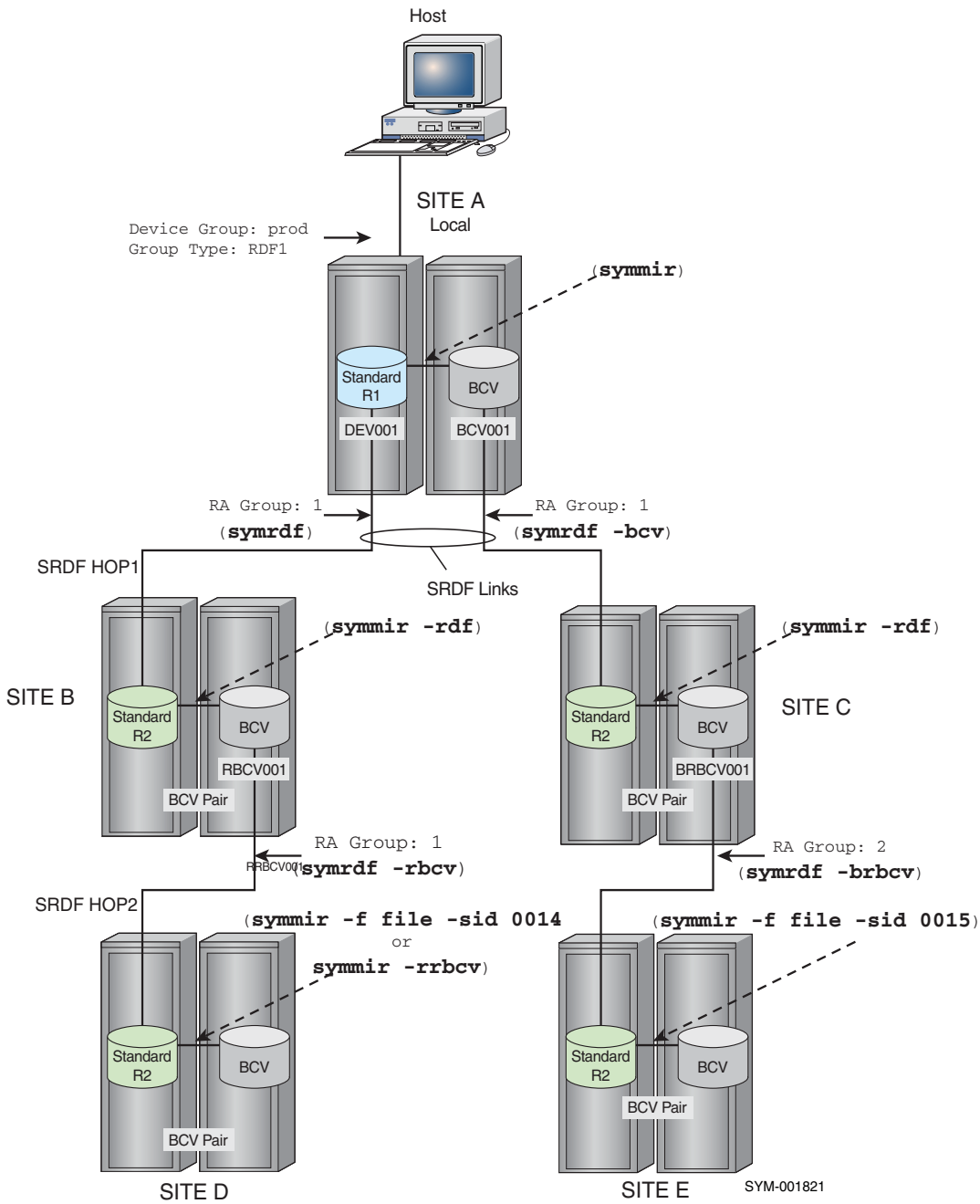


Figure 50 Commands used to perform splits in a complex configuration

System-wide splits

Figure 51 shows how the `symmir` and `symrdf` commands might be applied to split operations in a complex configuration.

Note: You must have established SRDF device groups before you perform any `symmir` and `symrdf` operations.

Examples

To split the BCV pair within Site A:

```
symmir -g prod split
```

To split SRDF pairs at Site B from host-connected Site A:

```
symrdf -g prod split
```

To split the BCV pairs within Site B:

```
symmir -g prod -rdf split
```

To split BCV SRDF pairs at Site C from host-connected Site A:

```
symrdf -g prod -bcv split
```

To split the BCV pairs within Site C:

```
symmir -g prod -rdf -bcv split
```

To split BCV SRDF pairs at Site D from host standard-associated Site B:

```
symrdf -g prod -rbcv split
```

To split the BCV pairs within Site D:

```
symmir -f dfile -sid 0014 split
```

or

```
symmir -g prod -rrbcv split
```

To split BCV SRDF pairs at Site E from host BCV-associated Site C:

```
symrdf -g prod -brbcv split
```

To split the BCV pairs within Site E (hop 2):

```
symmir -f dfile -sid 0015 split
```

Perform operations such as `establish` and `restore` in the same manner for remote sites.

EMC Solutions Enabler Symmetrix TimeFinder Family CLI Product Guide provides more information.

Controlling basic operations in a multi-hop configuration

Table 26 is a sequence of commands to perform basic control operations in a multi-hop configuration.

Each Step number correlates to a bubble number in Figure 51.

Table 26 Basic operations in a multi-hop configuration

Step	CLI control operation	Description
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 standard device 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 -f ‹› -sid 056 establish or symmir -g ‹› -rrbcv establish	Creates the BCV associated hop 2 BCV copy.
16	symmir -f ‹› -sid 056 split or symmir -g ‹› -rrbcv	Splits the BCV-associated hop 2 device pair.

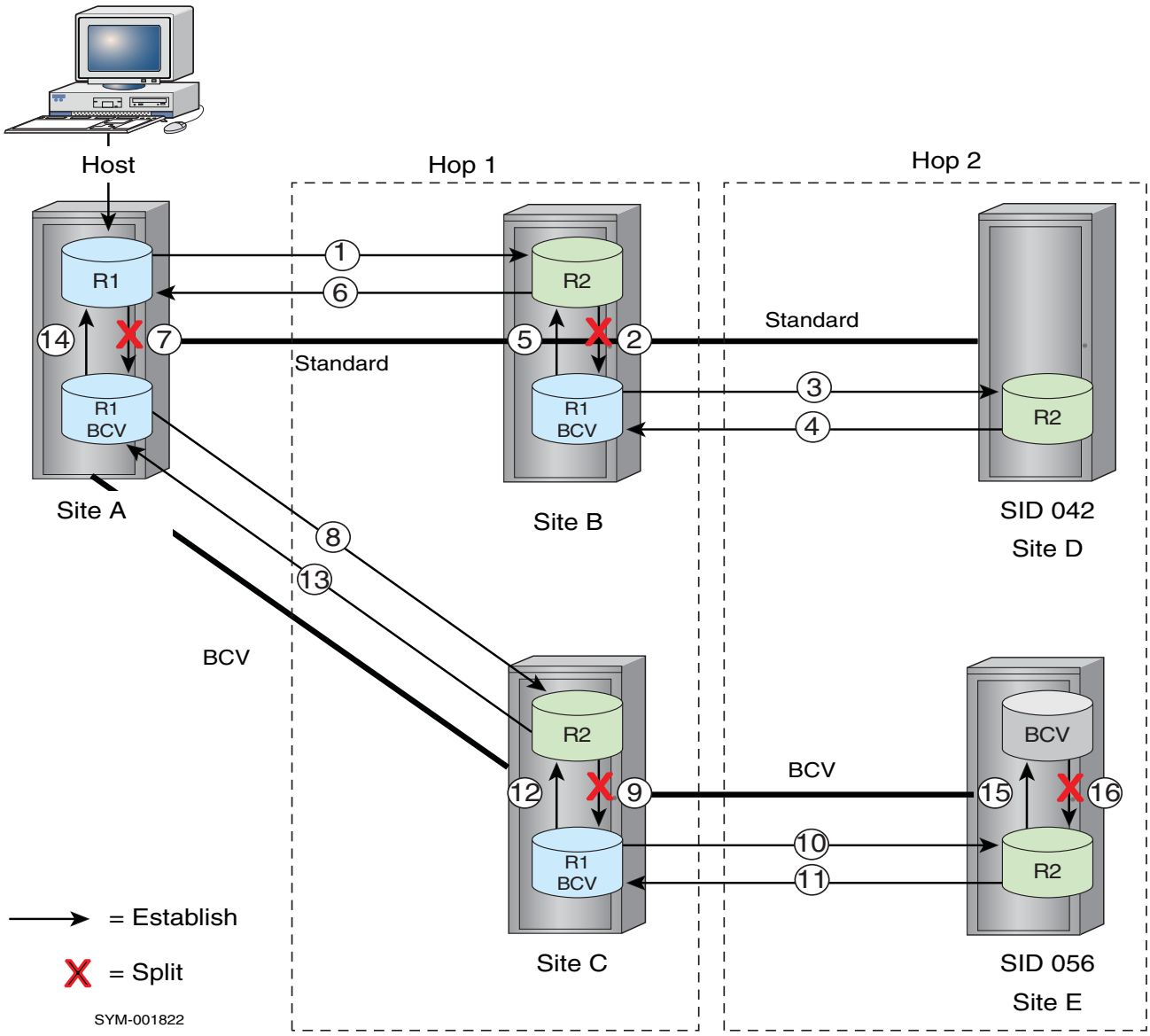


Figure 51 Basic operations in multi-hop SRDF configurations

CHAPTER 10
Automated Recovery Operations

This chapter describes the following topics:

- Overview 240
- Requirements and strategies 240
- symrecover command 241
- Starting SRDF Automated Recovery 242
- Stopping SRDF Automated Recovery 244
- symrecover options file parameters 244

Overview

SRDF Automated Recovery is a utility for optimizing ever-ready fault management responses in basic SRDF environments.

SRDF Automated Recovery runs in the background and monitors the state of various SRDF/S or SRDF/A sessions.

SRDF Automated Recovery is implemented by the `symrecover` command.

If SRDF Automated Recovery detects a session failure, it attempts an automatic recovery and restart of the session. The restart uses preconfigured settings specified in the `symrecover` options file.

The options file also provides parameters for:

- Setting up error logging and event notification through email, and
- Parameters for monitor, recovery strategies, and restart actions.

Note: SRDF Automated Recovery is not supported in SRDF/Star environments.

Figure 52 shows a basic SRDF recover environment:

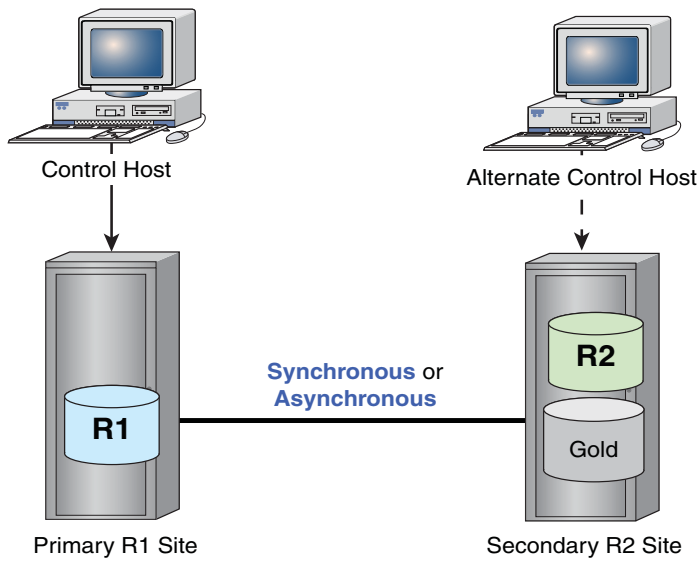


Figure 52 SRDF Recovery environment

In a basic recovery environment, a primary R1 site replicates to the secondary R2 site over a synchronous or asynchronous link.

A gold copy (BCV or clone) can be built on the R2 site to augment recovery restart strategies.

Requirements and strategies

- The `symrecover` session must be started either at the primary R1 site or the remote R2 site.
- Solutions Enabler binaries must either be in the PATH or specified as a parameter.

- The `symrecover` command can only be run with the Perl script shipped with Solutions Enabler.
- If the group is concurrent, then `symrecover` must be run from the R1 workload site.
- The initial group state must be CONSISTENT or SYNCHRONIZED, depending on the target SRDF state, unless the `restart_group_on_startup` option is specified (not the default).

Consistency protection restrictions

- If consistency protection is desired, it must be enabled prior to starting `symrecover`. A `symrecover` session must be started on the same site where consistency was enabled via a consistency group.

  Note: If you are managing using device groups, `symrecover` can be started at other sites.

Gold copy restrictions

- You can perform R2 gold copying with either Native Clones or Business Continuance Volumes (BCVs).
- BCV-to-STD association for the R2 gold copy is dynamic using the `symmir` defaults.

Restart restrictions

Note: See Table 27 on page 244 for a complete list of parameters and optional recovery actions to be set in the `symrecover` options file.

- A recovery fails if monitoring a leg that has an R22 device when the other SRDF mirror of the R22 is read/write (RW) on the link (such states as `synchronized`, `syncinprog`, or `consistent`).
- The recovery does not start when the `-restart_group_on_starup` parameters are specified, and an R22 device has another SRDF mirror that is already RW on the link.

symrecover command

SRDF Automated Recovery is launched and optimized by the `symrecover` command. The symrecover command processes the command line options and options file settings.

symrecover command syntax

```
symrecover [-h]
symrecover [-env | -version]

symrecover start {-g DgName | -cg CgName}
[-mode {SYNC | ASYNC}] [-out LogPath]
[-options OptnFile]
```

Where:

- Either a device group (`-g DgName`) or composite group (`-cg CgName`) must be specified.

- `-mode` - specifies the type of SRDF session. Valid values are SYNC and ASYNC. There is no default and this option must be specified.
- `-out` *`LogPath`* - specifies an alternate fully-qualified directory location to use for the log file.
- `-options` *`OptnFile`* - specifies the fully-qualified file containing program options.

Usage notes

- The symrecover command does not support the monitoring or recovery of a device group or composite group that is set with an ANY group type.
- Any options specified on the command line take precedence over the options specified by `-options` *`OptnFile`*.
- In a cascaded SRDF environment:
  - Specify the target composite group.
  - Do not use the `-mode` option.

Note: The `symrecover` command returns an error if used with an SRDF device pair containing thin and standard devices. The thin device must be on a array running Enginuity 5875 or higher. The standard device must be on a array running Enginuity 5671, 5773.50154, or 5875 and higher (but not Enginuity 5874).

Examples To start a recovery in a basic SRDF/S environment:

```
symrecover start -g DgName -mode sync -options OptnFile
```

To start a recovery in a cascaded SRDF environment:

```
symrecover start -cg CgName -cascaded_monitor_both_hops -options
OptnFile
```

Starting SRDF Automated Recovery

The `symrecover` command can be invoked manually from the command line, but more commonly is configured to run continuously in the background using:

- Windows Scheduled Tasks,
- UNIX CRON/scheduled task, or
- UNIX (RC.2) file.

You can run the `symrecover` command from either the R1 or the R2 side as long as all the SRDF standard devices in the device group or the composite group are local to the host.

When devices in groups are not local to a host, they are marked as invalid, stop all control operations from being performed against them.

If an SRDF/A group becomes synchronous (SRDF/S), `symrecover` attempts to reset the SRDF link to SRDF/A mode.

You can define devices in groups on the R2 side with a corresponding partner but `symrecover` cannot start in this environment because you cannot monitor groups on the R2 side when the remote partner is concurrent. You must monitor these groups from the host.

Examples

To manually start `symrecover` for an SRDF/A composite group named `RDFAmon`, using the options file named `cg_mon_opts`:

```
symrecover start -cg RDFAmon -mode async -options cg_mon_opts
```

where the `cg_mon_opts` options file includes the following settings and default values for a BCV gold copy:

```
# Options file for symrecover
#######################################################
goldcopy_clone_list = TGT
goldcopy_location = R2
goldcopy_max_wait = 1800
goldcopy_resync_interval = 0
goldcopy_state_post_restart = ACTIVATED
goldcopy_state_startup = ACTIVATED
goldcopy_type = CLONE
help = 0
log_level = 3
monitor_cycle_time = 300
monitor_only = 0
out = /var/symapi/log
restart_adcopy_resynch_threshold = 30000
restart_attempt_pause = 60
restart_delay = 30
restart_group_on_startup = 0
restart_max_attempts = 5
restart_max_wait_adcopy_sync = 0
restart_max_wait_state_change = 0
restart_max_wait_warn_interval = 600
restart_rdfa_min_cycle_warn_interval = 300
restart_rdfa_min_cycle_warn_value = 0
restart_state_syncinprog_wait_time = 120
restart_state_syncinprog_warn_interval = 300
restart_state_transmit_wait_time = 120
restart_state_transmit_warn_interval = 300
restart_sync_type = ADCOPY
restart_window = 3600
run_once = 0
run_until_first_failure = 0
```

To recover a cascaded SRDF environment, add the following parameter settings to the options file in the previous example:

```
cascaded_monitor_both_hops = 1
goldcopy_location = All
```

These options:

- Allow recovery on both hops, and
- Build gold copies at the R21 and R2 sites.

  The hop2 (R21->R2 link) restarts quickly and safely in ADCOPY mode, during the R2 resynchronization period.

Stopping SRDF Automated Recovery

To stop `symrecover` manually, enter a Ctrl/C.

To stop a `symrecover` task running in the background use one of the following options:

- Windows - Cancel the task in the Scheduled Tasks, or use End Task in the Task Manager.
- UNIX - Use the kill command.

symrecover options file parameters

Table 27 describes the valid settings in the `symrecover` options file.

Many of these settings in Table 27 have Boolean values of 0 or 1.

- 0 disables the setting
- 1 enables the setting

Table 27 symrecover options file parameters (page 1 of 5)

Setting	Description
`cascaded_monitor_both_hops= [0|1]`	0 - (default) Monitors/recovers a single hop only invoked from any site. 1 - For cascaded SRDF environments, the `symrecover` session monitors both hops linking the cascaded sites. The `symrecover` session ignores the `-mode` option and can be invoked at either the R1 primary or the remote R2 tertiary site only (not at R21 site).
`email_addr_target= <e_addr1, e_addr2, ..., ...>`	Email notification address on errors. If any of the `email_*` options are specified, then this option must also be specified to activate email alerts. Multiple comma delimited addresses may be specified. There is no default value.
`email_addr_source= e_addr1`	Specifies an address that will be used as the 'from' field of any e-mails that symrecover sends. No checks are done about the validity of the e-mail address. If this is not specified, then a default value is generated based on the array's hostname and current user account.
`email_server= e_srvr_addr`	Specifies the host target email server. If any of the `email_*` options are specified then this option must also be specified to activate email alerts. There is no default value.
`email_subject= err_subject_string`	Specifies the email notification subject on errors. The default value is: `SymRecover Alert: Host [HostName] Group [GrpName]`

Table 27 symrecover options file parameters (page 2 of 5)

Setting	Description
`email_log_level= SeverityLevel`	The severity level desired for the email alert triggering message. Valid values are: 0 = Off. 1 = Only Errors are reported. 2 = Errors and Warnings are reported. 3 = Errors, Warnings, and Informational messages are reported. 4 = All messages are reported, including all SYMCLI commands and responses. **Note:** For each message that meets the particular logging level requirement, an email is sent with that message. It is highly recommended that at most this be set to either a 1 or a 2. If the required email options (`email_server` and `email_addr_target`) are not specified, then the default value is 0. If they are specified then the default value is 1.
`goldcopy_location= LocationValue`	Specifies the location of the backup gold copy. Possible (case insensitive) values are: • NONE = No gold copy is desired. All other gold copy optional parameters in this list are ignored. • R2 = A gold copy on the R2 site is desired. R2 is the default setting. Any R2 BCV pairs need to be already defined before calling `symrecover`.

`goldcopy_type=CopyType`
Old alternate, if still necessary:
`goldcopy_type_r2= CopyType`
	Specifies the type of goldcopy to create on the R2 side. Valid (case insensitive) values are:
	none = No gold copy is desired. All other `goldcopy_*` options are ignored.
	bcv = BCV gold copy on the R2 side is created, which is the default.
	clone = Clone gold copy on the R2 is created.
	Note: For the BCV gold copy, the R2 BCVs must be paired with the R2 devices before starting `symrecover`.
	For the clone gold copy, the target devices must have a clone session with the R2 devices before starting `symrecover`.

`goldcopy_state_startup= CopyType`
Old alternate, if still necessary:
`goldcopy_bcv_r2_mirror_state_startup= CopyState`
	Specifies the desired state of the R2 gold copy upon routine startup. Valid (case insensitive) values are:
	establish = the devices must be established (BCV gold copy only).
	split = The devices must be split (BCV gold copy only).
	activated = The devices must be in the copied state (clone gold copy only).
	created = The devices must be in the precopy state (clone gold copy only).
	none = The devices must be unchanged, which is the default.
	Note: If the gold copy type is BCV and the default state of the BCVs is establish, this has been shown to increase SRDF/A session drops.

Table 27 symrecover options file parameters (page 3 of 5)

Setting	Description

`goldcopy_state_post_restart= CopyState`
 	
Old alternate, if still necessary:
`goldcopy_bcv_r2_mir_state_post_restart = CopyState`
	Following a successful SRDF/A session restart or BCV resync, specifies which state the R2 gold copy should be. Valid (case insensitive) values are:
	establish = The devices must be left established (BCV gold copy only).
	split = The devices must be split, which is the default (BCV gold copy only).
	activated = The devices must be in the copied state (clone only).
	created = The devices must be in the precopy state (clone only).
	
	Note: If the gold copy type is BCV and the default state of the BCVs is establish, this has been shown to increase SRDF/A session drops.

`goldcopy_max_wait= MaxWaitTime`
 	
Old alternate, if still necessary:
`goldcopy_max_wait_bcv= MaxWaitTime`
	Specifies the length of time, in seconds, for `symrecover` to wait for synchronization. Valid values are 0 to `maxint` (2147483647).
	The default is 0, which indicates for `symrecover` to wait forever.
	For clone gold copies, if the `goldcopy_state_post_restart` option is set to `activated`, it waits for the clone copied state to be reached before performing synchronization. If this option is set to `created`, it waits for the clone precopied state to be reached.

`goldcopy_resync_interval= resynctime`
 	
Old alternate, if still necessary:
`goldcopy_bcv_r2_mirror_resync_interval = resynctime`
	Defines the resync interval, in minutes, for `symrecover` to automatically create a new clone gold copy or a new BCV gold copy, which overrides the existing gold copy. This action only takes place during non-error periods.
	Valid values are 0, and 15 to `maxint`. Zero (0) indicates that the mirrors are never to be automatically synchronized outside of error-producing events.The default is 15.
	
	Note: If the gold copy type is `BCV`, the act of frequently synchronizing the R2 BCVs has been shown to increase SRDF/A session drops.

`goldcopy_clone_list= List`	For a clone gold copy, this option tells `symrecover` which list within the device group or the composite group to search for clone devices. Valid (case insensitive) values are: tgt = Uses the TGT list. bcv = Uses the BCV list.
`monitor_cycle_time= cycletime`	Defines the number of seconds to pause between monitor status scans. The minimum value is 30 seconds, the maximum is 3600 seconds. The default value is 300 seconds.
`monitor_only= [0|1]`	Specifies to only monitor the state of specified group. No recovery actions will take place. This option is not enabled by default. **Note:** `monitor_only`, `run_once`, and `run_until_first_failure` are mutually exclusive options.
`run_once= [0|1]`	Specifies to check the status of the group once. If the group needs recovery actions perform them. Exit after one check. This option is not enabled by default. This option ignores the setting of `restart_max_attempts`. **Note:** `monitor_only`, `run_once`, and `run_until_first_failure` are mutually exclusive options.

Table 27 symrecover options file parameters (page 4 of 5)

Setting	Description
`run_until_first_failure= [0|1]`	Specifies to monitor the group until the first failure occurs and then exit without performing any recovery action. This option is not enabled by default. This option ignores the setting of `restart_max_attempts`. **Note:** `monitor_only`, `run_once`, and `run_until_first_failure` are mutually exclusive options.
`rdfg= rdfgvalue`	When working with device groups or composite groups that contain concurrent devices, symrecover supports monitoring only one of the SRDF groups that contain mirrors of the concurrent devices. Use the rdfg option to indicate the SRDF group that symrecover should monitor. Note that monitoring of concurrent SRDF defined groups is only supported when symrecover is executed from the R1 side. The value is taken directly as specified and no data validation is performed on it. This option is not set by default and non-concurrent SRDF groups are assumed. **Note:** If the group is a composite group, and consistency is enabled, this must be of the "name:" format and this value is case sensitive.
`restart_adcopy_resynch_threshold= tracks`	Specifies the number of tracks outstanding that during recovery will trigger a switch over to SRDF/A or SRDF/S. The default value is 30000.
`restart_attempt_pause= time`	Inserts a specified wait time before an attempt is made to restart a failed session to allow for things to settle down. After the `restart_attempt_pause` is complete, `symrecover` redrives the overall monitor loop. If there is still a problem, the restart failure count is incremented and a restart is attempted. Valid values are 30 to 3600 seconds. The default is 60 seconds.
`restart_delay= time`	Inserts a specified wait time after an attempt is made to restart a failed session and the attempt itself fails. Valid values are 0 (no delay, immediately restart) to maxint. The default is 30 seconds.
`restart_group_on_startup= [0|1]`	On `symrecover` startup, if the group being monitored is not initially in a Consistent state (for SRDF/A) or a Synchronized state (for SRDF/S), `symrecover` considers this an error condition and exits. If this option is specified, `symrecover` will attempt to recover the group on startup. This option is not enabled by default.
`restart_max_attempts= attempts`	Specifies the maximum number of restart attempts that are performed within the `restart_window` interval. After this limit is reached the program will terminate. The range is from 0 to maxint. The value of 0 means to attempt indefinitely.The default is 5 attempts.
`restart_max_wait_adcopy_sync= time`	Specifies the length of time (in seconds) during a restart for a program to wait for a group to achieve the `restart_adcopy_resync_threshold` number of tracks pending. Valid values are 0 to maxint. The value of 0 means to infinitely wait. The default is 0.
`restart_max_wait_state_change= statetime`	Specifies the length of time (in seconds) during a restart for a program to wait for a group to change to a desired state (once requested). Valid values are 0 to maxint. The value of 0 means to infinitely wait. The default is 0.

Table 27 symrecover options file parameters (page 5 of 5)

Setting	Description
`restart_max_wait_warn_interval= warntime`	Specifies the length of time (in seconds) during a restart, while waiting for a state change to occur, to display a progress warning message. Valid values are 0 and 30 to maxint. The value of 0 means to wait forever. The default is 600 seconds.
`restart_rdfa_min_cycle_warn_interval= cyclewarntime`	Specifies the length of time (in seconds) before repetitively displaying a warning when the RDFA minimum cycle time exceeds the `restart_rdfa_min_cycle_warn_value` parameter. Valid values are 30 to `maxint`.The default is 600.
`restart_rdfa_min_cycle_warn_value= warntime`	Specifies the maximum value (in seconds) to which a trigger can occur with a warning message, indicating the RDFA minimum cycle time has exceeded this value. Valid values are 0 and 30 to `maxint`. The value of 0 means this feature is turned off, which is the default.
`restart_state_syncinprog_wait_time time`	The maximum length of time (in seconds) during a group syncinprog state that sleep is done before rechecking the group status. Valid values are [30] to [maxint]. The default is [120] seconds.
`restart_state_transmit_warn_interval= time`	Specifies the interval of time (in seconds) that while a group remains in a transmit idle state, to generate a warning message. Valid values are 0 to `maxint`. The default is 300 seconds.
`restart_state_transmit_wait_time= transwaittime`	Specifies the maximum length of time (in seconds) that during a group transmit idle state, a sleep is done before rechecking the group status. Valid values are 30 to maxint. The default is 120 seconds.
`restart_sync_type= synctype`	Specifies the type of synchronization to be used following the detection of a failed SRDF/A session. Valid values are: ADCOPY = adaptive copy disk (default). SYNC = synchronous mode. NONE = No intermediate track resynch stage will be attempted. A direct re-establish using the existing SRDF session mode will be attempted. Note that if `cascaded_monitor_both_hops` is set, `restart_sync_type` is ignored as ADCOPY is used in the R21->R2 link at restart.
`restart_window= time`	Specifies a time window (in seconds) during which no more than restart_max_attempts failures and accompanying restart attempts will be tolerated before monitoring is terminated. The window begins at the time of the first failure and ends restart_window seconds later. A new window begins with a failure after expiration of the previous window.
`log_level= level`	The desired logging level. Valid values are: 0 = Off. 1 = Only Errors are reported. 2 = Errors and Warnings are reported. 3 = Errors, Warnings, and Informational messages are reported (default). 4 = All messages are reported.

APPENDIX A
SRDF operations and pair states

This appendix describes the following topics:

- SRDF operations and applicable pair states 250
- Concurrent SRDF operations and applicable pair states 254
- Consistency group operations and applicable pair states 260

SRDF operations and applicable pair states

When you issue a command to perform an SRDF control operations, SRDF verifies the state of the device pairs.

If the device pair is not in a legal SRDF state to initiate the control operation, the action is blocked.

Use the `-force` option to perform the control operation, regardless of the pair state.

Note: If devices are running in SRDF/A mode, control operations for restore, update R1, and failback require the use of the -force option.

Examples:

To initiate a failover on all SRDF pairs in the `prod` group that are in the Split state:

```
symrdf -g prod -force failover
```

To initiate a failover on one SRDF pair, DEV001, in the `prod` group that is n the SyncInProg state:

```
symrdf -g prod -force failover DEV001
```

IMPORTANT

The `-force` option may place the SRDF pair into an undesirable state. After using this option, always check the pair state.

Control operations for R1 -> R2 pair states

The first column in Table 28 lists the control operations you can invoke for the listed pair states.

Allowed actions are noted by Ys.

The Partitoned1 pair state indicates that the remote array is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote array is not in the SYMAPI database and was not discovered, or was removed from this database.9

Table 28 SRDF control operations and applicable pair states (page 1 of 3)

Control operation:	R1 ->R2 pair state:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
deletepair			Y[28,36,37]	Y[28,36,37]	Y[28,36,37]							
half_deletepair			Y[28,36,37]	Y[28,36,37]	Y[28,36]	Y[28,36,37]	Y[28,36,37]					
movepair			Y	Y	Y							

Table 28 SRDF control operations and applicable pair states (page 2 of 3)

Control operation:	R1 ->R2 pair state:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
half_movepair			Y	Y	Y	Y	Y					
swap			Y[43]	Y[6,43]	Y[43]			Y[43]				
swap -refresh R1			Y[29,43]	Y[6,29,43]	Y[29,43]			Y[29,43]				
swap -refresh R2			Y[29,43]	Y[6,29,43]								
half_swap			Y[43]	Y[43]	Y[43]	Y[43]	Y[43]					
establish			Y[26]	Y[26]	Y[5,26]			Y[5,26]		Y[12]		
establish -full			Y[26]	Y[26]	Y[5,26]			Y[5,26]		Y[12, 17]		
split	Y[15,19,28,36,37]	Y		Y[19,28,36,37]				Y[19,28,36,37]			Y[19,28,36,37]	
restore			Y[29]	Y[3,29]	Y[5,29]			Y[5,29]		Y[13]		
restore -full			Y[29]	Y[3,29]	Y[5,29]			Y[5,29]		Y[13,14]		
update				Y[1,28,29]	Y[28,29]			Y[28,29]				
failback			Y[26,28,29]	Y[1,26,28,29]	Y[26,28,29]	Y[8,18,26]	Y[8,18,26]	Y[26,28,29]	Y[26,28,29]			
failover	Y[15,19,30,41,42]	Y[19,30,41,42]	Y[18,19,30,41,42]	Y[19,30,41,42]		Y[9,19,41,42]	Y[9,19,41,42]	Y[19,30,41,42]	Y[15,19,30,41,42]	Y[18,19 30, 41,42]	Y[19,30,41,42]	Y[19,30,41,42]
failover -establish	Y[15,17,29,30 43]	Y[29,30, 43]	Y[18,19,30,43]	Y[29,30,43]	Y[29,30,43]			Y[29,30,43]	Y[15,29,30,43]	Y[18,29,30,43]		
failover -restore	Y[15,26,31, 43]	Y[26,31,43]	Y[18,26,31,43]	Y[26,31,43]	Y[18,26,31, 43]					Y[18,26,31,43]	Y[26,31 43]	
invalidate R1				Y[2,29]								
invalidate R2				Y[26]								
merge				Y[3,33,34]	Y[18,33,34]							
msc_cleanup			Y	Y	Y	Y[9]	Y[9]					
not_ready R1	Y[6,21]	Y[6,21]	Y[6,21]	Y[6,21]	Y[21]	Y[6,8,21]	Y[6,8,21]	Y[21]	Y[21]	Y[6,21]	Y[6,21]	Y[6,8,21]
not_ready R2	Y[10,17,21]	Y[10,21]	Y[10,21]	Y[21]	Y[10,21]	Y[7,9,21]	Y[7,9,21]	Y[10,21]	Y[10,21]	Y[10,17,21]		
ready R1	Y[21]	Y[21]	Y[21]	Y[21]	Y[21]	Y[8,21]	Y[8,21]	Y[21]	Y[21]	Y[21]	Y[21]	Y[21]

Table 28 SRDF control operations and applicable pair states (page 3 of 3)

Control operation:	R1 ->R2 pair state:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
ready R2	Y[17,21]	Y[21]	Y[21]	Y[21]	Y[21]	Y[9,21]	Y[9,21]	Y[21]	Y[21]	Y[17,21]		
refresh R1				Y[1,20,39]	Y[18,39]							
refresh R2				Y[1,18,26]								
resume				Y								
suspend	Y[15,19,28,36,37,38]	Y	Y[16,18,19,28,36,37,38]	Y[7,19,28,36,37,38]	Y[5,19,28,36,37,38]	Y[19,28,36,37,38]	Y[19,28,36,37,38]	Y[19,28,36,37,38]	Y[15,19,28,36,37,38]	Y[18,19,28,36,37,38]	Y[19,20,28,36,37,38]	Y[19,20,28,36,37,38]
disable	Y[41]	Y[41]	Y[41]	Y[41]	Y[41]	Y[18,41]	Y[18,41]	Y[41]	Y[41]	Y[41]	Y[41]	Y[23]
enable	Y[24]		Y[24]	Y[24]	Y[24]			Y[24]	Y[24]	Y[24]	Y	Y[23]
rw_disable R2	Y[17]	Y	Y[11]	Y	Y[11]	Y[7,9]	Y[7,9]	Y[11]	Y[11]	Y[2,17]		
rw_enable R1	Y[1]	Y[1]		Y[1]	Y[18]	Y[1,8]	Y[1,8]			Y[1]	Y[1]	Y[1,8]
rw_enable R2				Y[7]		Y[7,9]	Y[7,9]					
write_disable R1	Y[18]	Y[18]	Y[18]	Y[18]		Y[4,8]	Y[4,8]			Y[18]	Y[18]	Y[4,8]
write_disable R2			Y[18]	Y[18]	Y[18]	Y[9,10]	Y[9,10]	Y[18]	Y[18]	Y[18]		Y[9,10]
activate -rdfa_dse	Y[22]										Y	Y[23]
deactivate -rdfa_dse	Y[22]										Y	Y[23]
activate -rdfa_devpace	Y[22,44]										Y[44]	Y[23,44]
deactivate -rdfa_devpace	Y[22,44]										Y[44]	Y[23,25,44]
activate -rdfa_pace	Y[22]										Y	Y[23]
deactivate -rdfa_pace	Y[22]										Y	Y[23,25]
activate -rdfa_wpace	Y[22]										Y	Y[23]
deactivate -rdfa_wpace	Y[22]										Y	Y[23]
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y			Y	Y		Y	
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y			Y	Y		Y	

1. SA is Write Disabled or is Not Ready on the source side.
2. SA is Write Disabled, or is Not Ready on the source side, or must use -nowd.
3. SA is Write Disabled, or is Not Ready on the source side, or must use -force.
4. SA is Ready on the source side.
5. Source is not visible to any host.
6. Write Disabled on the source.
7. SA or RA is Write Disabled or is Not Ready on the target side.
8. Host application running while connected to the source.
9. Host application running while connected to the target.
10. RA is Ready on the target side.
11. RA is Write Disabled on the target side.
12. Source and target are Not Ready but the SRDF link is Ready and there are no local or remote invalid tracks on the source or the target.
13. Source and target are Not Ready but the SRDF link is Ready and there are no remote invalid tracks on the source side.
14. Source and target are Not Ready but the SRDF link is Ready and there are no local or remote invalid tracks on the source side.
15. Can use -symforce.
16. Write Disabled on the SRDF link.
17. Not allowed when SRDF/A is active.
18. Must use -force.
19. If enabled for SRDF consistency protection, must use -force.
20. Must use -immediate.
21. Not allowed on a diskless device.
22. SRDF/A must be active.
23. Source must be reachable.
24. Must be in async mode.
25. Only allowed on the R1 side and must use -symforce.
26. No local invalid tracks on the source side.
27. No remote invalid tracks on the source side.
28. If remote invalid tracks are on the source side, must use -force.
29. No local invalid tracks on the target side.
30. If remote invalid tracks are on the source side, must use -symforce.
31. If remote invalid tracks are on the target side, must use -force.
32. No remote invalid tracks on the target side.
33. Source device is Read Write Enabled and there are no local and remote invalid tracks on the target side.
34. Target device is Read Write Enabled and there are no local and remote invalid tracks on the target side.
35. Source and target devices are Read Write Enabled.
36. If there are local invalid tracks on the source side, must use -symforce if the source is not an R11 or R21.
37. If there are local invalid tracks on the target side, must use -symforce if the target is not an R11 or R21.

38. Write Disabled on the SRDF link and must use -force.
39. No local invalid tracks on the target side and no remote invalid tracks on the source side and must use -force.
40. If the SRDF device pair is operating in adaptive copy mode, must use -symforce.
41. If enabled for CG SRDF consistency protection, must use -force.
42. If local invalid tracks are on the target side, must use –symforce.
43. Not allowed if enabled for SRDF consistency protection.
44. Not allowed if the R1 or R2 array is running Enginuity 5977 or higher

Concurrent SRDF operations and applicable pair states

This section provides the concurrent SRDF control operations and their applicable pair states for concurrent R1 (R11) and concurrent R2 (R22), including:

- "Concurrent SRDF: 1st leg R1-› 2nd leg R1 pair states"
- "Concurrent SRDF: 1st leg R2 -› 2nd leg R2 pair states"

About concurrent R1

In a concurrent R1 relationship, there are two separate links, or legs, sending data from one R1 device to two separate R2 mirrors. You can perform a control operation on one of these legs only if the other leg is in a certain pair state.

Note: If a concurrent R1 device is made RW (read write) from either of the SRDF relationships, it is also seen as RW from the other relationship. The commands to make a concurrent R1 device RW are: `rw_enable R2`, `split`, and `failover`.

Concurrent SRDF: 1st leg R1-› 2nd leg R1 pair states

Table 29 lists the allowable control operations for the first leg of the concurrent R1 pair (the one being controlled by an SRDF action) given the pair state of the second leg (the one not being controlled).

Allowed actions are noted by Ys.

The Partitoned1 pair state indicates that the remote array is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote array is not in the SYMAPI database and was not discovered, or was removed from this database.

Table 29 SRDF control operations and applicable states for concurrent R1 pairs (page 1 of 3)

Control operation of 1st leg of concurrent SRDF R1 pair:	Pair state of 2nd leg of concurrent SRDF R1 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
createpair -establish	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
createpair -restore	Y[1,5,6]	Y[1,6]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1,6]			Y[1,5]	Y[1,5,6]	Y[1,5,6]
createpair -invalidate R1	Y[1,5]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1,5]	Y[1]	Y[1,5]	Y[1,5]

Table 29 SRDF control operations and applicable states for concurrent R1 pairs (page 2 of 3)

Control operation of 1st leg of concurrent SRDF R1 pair:	Pair state of 2nd leg of concurrent SRDF R1 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
createpair -invalidate R2	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
deletepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
half_deletepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
movepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
half_movepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
swap	Y[1,12,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,12]	Y[1,12]
half_swap	Y[1,12,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]		Y[1,12]	Y[1]
swap -refresh R1	Y[1,5,12,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,5,17]	Y[1,17]	Y[1,5,12]	Y[1,5,12]
swap -refresh R2	Y[1,12,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,17]	Y[1,12]	Y[1,12]
establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
full establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
split	Y	Y	Y	Y	Y[2]	Y	Y	Y[3]		Y	Y	Y
restore	Y[5,6]	Y[6]	Y	Y	Y	Y	Y[6]			Y[6]	Y[5,6]	Y[5,6]
full restore	Y[5,6]	Y[6]	Y	Y	Y	Y	Y[6]			Y[6]	Y[5,6]	Y[5,6]
update	Y[5,6]	Y[6]	Y	Y	Y	Y	Y			Y[6]	Y[5,6]	Y[5,6]
failback	Y[5]	Y	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y	Y[5]	Y[6]	Y[5]	Y[5]
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y[1,12,13]	Y[1]	Y[1,8]	Y[1,8]	Y[1,8,15]	Y[1,8]	Y[1,8]				Y[1,12,13]	Y[1,12,13]
failover -restore	Y[1,11,12,13]	Y[1]	Y[1,8]	Y[1,8]	Y[1,8,15]	Y[1,8]	Y[1,8]				Y[1,12,13]	Y[1,4,11,12,13]
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y[5]	Y	Y	Y	Y	Y	Y	Y	Y[5]		Y[5]	Y[5]
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 29 SRDF control operations and applicable states for concurrent R1 pairs (page 3 of 3)

Control operation of 1st leg of concurrent SRDF R1 pair:	Pair state of 2nd leg of concurrent SRDF R1 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y[5]	Y	Y	Y	Y	Y	Y	Y	Y[5]		Y[5]	Y[5]
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y[5,7]	Y[7]	Y	Y	Y	Y	Y[6]	Y	Y[5]	Y[5,6]	Y[5,7]	Y[5,7]
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. If the other pair (the one not being controlled) is enabled for SRDF consistency protection, must use -force. This operation can change the composite group type, causing SRDF consistency monitoring to stop.
2. Must use -force. The state of the other pair changes to Suspended.
3. Changes the state of the other pair to Split.
4. If device will become an R21 and the other pair is in Transmit Idle pair state, data synchronization between R1->R21 cannot complete.
5. Not allowed if R2->R11<-R2 data resynchronization will result.
6. Must use -remote.
7. Only allowed if data flows from R2 to R11 and -remote is used.
8. Not allowed if R11 is diskless.

9. Not allowed if a diskless device and will become an R21 device.
10. Not allowed if tracks are owed to R21 from concurrent R2.
11. Not allowed if a diskless device and will become an R21 device and results in R1<-R21->R2 data resynchronization.
12. Not allowed if SRDF/A group-level write pacing or SRDF/A device-level write pacing is active and supported on the other R1 mirror what will become the R21 and the R21 array is running an Enginuity level lower than 5876 Q42012 SR.
13. If the pair being controlled is operating in adaptive copy mode, must use -force if the R1 mirror of what will become the R21 has SRDF/A group-level and/or device-level write pacing activated and supported.
14. If the pair being controlled is operating in asynchronous mode, with SRDF/A group-level and/or device-level write pacing enabled for autostart on what will be the R1 mirror of the resulting R21, must use -force if the other pair (that is not being controlled) is operating in adaptive copy mode.
15. The R11 is not visible to any host.
16. The other pair's (the one not being controlled) R1 is not visible to any host.
17. Not allowed when what will become the R21->R2 is in Synchronous mode.

Concurrent SRDF: 1st leg R2 -> 2nd leg R2 pair states

Concurrent R2 devices are intended for SRDF/Star configurations.

In a concurrent R2 configuration, an R2 device has two remote mirrors, only one of which can be active (read/write) at a given time. *EMC Solutions Enabler Symmetrix SRDF/Star CLI Product Guide* provides more information.

Table 30 lists the allowable control operations for the first leg (the one being controlled by an SRDF action) of the concurrent R2 pair given the pair state of the second leg (the one not being controlled).

Allowed actions are noted by Ys.

The Partitoned1 pair state indicates that the remote array is in the SYMAPI database and was discovered. The Partitioned2 pair state indicates the remote array is not in the SYMAPI database and was not discovered, or was removed from this database.

Table 30 SRDF control operations and applicable states for concurrent R2 pairs (page 1 of 3)

Control operation of 1st leg of concurrent SRDF R2 pair:	Pair state of 2nd leg of concurrent SRDF R2 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
createpair -establish			Y[1,2]	Y[1]	Y[1,2]	Y[1]						
createpair -restore			Y[1,2]	Y[1]	Y[1,2]	Y[1]						
createpair -invalidate R1	Y[1]	Y[1]	Y[1,2]	Y[1]	Y[1,2]	Y[1]		Y[1]	Y[1]		Y[1]	Y[1]
createpair -invalidate R2	Y[1]	Y[1]	Y[1,2]	Y[1]	Y[1,2]	Y[1]		Y[1]	Y[1]		Y[1]	Y[1]
deletepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
half_deletepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]

Table 30 SRDF control operations and applicable states for concurrent R2 pairs (page 2 of 3)

Control operation of 1st leg of concurrent SRDF R2 pair:	Pair state of 2nd leg of concurrent SRDF R2 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
movepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
half_movepair	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
swap	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]		Y[1]	Y[1]		Y[1]	Y[1]
half_swap	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]		Y[1]	Y[1]		Y[1]	Y[1]
swap -refresh R1	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]		Y[1]	Y[1]		Y[1]	Y[1]
swap -refresh R2				Y[1]		Y[1]						
establish			Y[2]	Y	Y[2]	Y						
establish -full			Y[2]	Y	Y[2]	Y						
split			Y	Y[3]	Y	Y		Y	Y	Y		
restore			Y[2]	Y	Y[2]	Y						
restore -full			Y[2]	Y	Y[2]	Y						
update			Y	Y	Y	Y						
failback			Y[2]	Y	Y[2]	Y						
failover			Y	Y[3]	Y	Y	Y			Y		
failover -establish			Y	Y[3]	Y[16]	Y						
failover -restore			Y[1]	Y[1,3]	Y[1,16]	Y[1]		Y[1,14]	Y[1,14]			
invalidate -R1	Y	Y	Y	Y	Y	Y		Y	Y		Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y		Y	Y		Y	Y
merge			Y	Y	Y	Y						
msc_cleanup			Y	Y	Y	Y	Y					
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y[2]	Y	Y[2]	Y	Y	Y[2]	Y[2]	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y		Y	Y		Y	Y
refresh R2				Y		Y						
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume			Y[2]	Y	Y[2]	Y						
rw_disable R2	Y	Y	Y[2]	Y	Y[2]	Y		Y[2]	Y[2]		Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y	Y

Table 30 SRDF control operations and applicable states for concurrent R2 pairs (page 3 of 3)

Control operation of 1st leg of concurrent SRDF R2 pair:	Pair state of 2nd leg of concurrent SRDF R2 pair:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
rw_enable R2			Y	Y[3]	Y	Y	Y	Y	Y	Y		
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y[2]	Y	Y[2]	Y		Y[2]	Y[2]		Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: Table 30 uses the same keys as Table 29 on page 254.

Consistency group operations and applicable pair states

This section provides the consistency group (SRDF/CG) control operations and the applicable pair states for devices within a consistency group.

Allowed actions are noted by Ys.

Table 31 SRDF control operations and applicable pair states for devices in an SRDF/CG

Control operation:	Pair state:											
	SyncInProg	Synchronized	Split	Suspended	Failed Over	Partitioned1	Partitioned2	R1 Updated	R1 UpdInProg	Invalid	Consistent	TransmitIdle
enable	Y	Y	Y	Y	Y			Y	Y	Y	Y	
disable	Y	Y	Y	Y	Y	Y[2]	Y[2]	Y	Y	Y	Y	Y[2]
modify -add	Y[1]	Y									Y	
modify -remove	Y[1]	Y									Y	
modify -recover	Y	Y		Y		Y[2]	Y[2]				Y	

1. There are no local invalid tracks on the source side and no remote invalid tracks on the remote side.
2. Must use -force.

APPENDIX B
SRDF operations and TimeFinder sessions

This appendix describes the following topics:

- SRDF operations during TimeFinder/Snap/VP Snap sessions 262
  - "SRDF operations when R1 is source of TimeFinder Snap/VP Snap"
  - "SRDF operations when R1 is target of TimeFinder Snap/VP Snap"
  - "SRDF operations when R2 is source of TimeFinder Snap/VP Snap"
  - "SRDF operations when R2 is target of TimeFinder Snap/VP Snap"
- SRDF operations for TimeFinder/Clone sessions 272
  - "SRDF operations when R1 is source of TimeFinder Clone"
  - "SRDF operations when R1 is target of TimeFinder Clone"
  - "SRDF operations when R2 is source of TimeFinder Clone"
  - "SRDF operations when R2 is target of TimeFinder Clone"
- SRDF operations for Extent-level TimeFinder/Clone sessions 282
  - "SRDF operations when R1 is source of Extent-level Clone"
  - "SRDF operations when R1 is target of Extent-level Clone"
  - "SRDF operations when R2 is source of Extent-level Clone"
  - "SRDF operations when R2 is target of Extent-level Clone"
- SRDF set operations for TimeFinder/Snap sessions 292
  - "SRDF set operations when R1 is source of TimeFinder/Snap"
  - "SRDF set operations when R1 is target of TimeFinder/Snap"
  - "SRDF set operations when R2 is source of TimeFinder/Snap"
  - "SRDF set operations when R2 is target of TimeFinder/Snap"
- SRDF set operations for TimeFinder/Clone sessions 295
  - "SRDF set operations when R1 is source of TimeFinder/Clone"
  - "SRDF set operations when R1 is target of TimeFinder/Clone"
  - "SRDF set operations when R2 is source of TimeFinder/Clone"
  - "SRDF set operations when R2 is target of TimeFinder/Clone"
- SRDF set operations for Extent-level TimeFinder/Clone sessions..... 298
  - "SRDF set operations when R1 is source of Extent-level Clone"
  - "SRDF set operations when R1 is target of Extent-level Clone"
  - "SRDF set operations when R2 is source of Extent-level Clone"
  - "SRDF set operations when R2 is target of Extent-level Clone"
- SRDF operations for TimeFinder Snapvx 300
  - "SRDF operations when R1 is source of Snapvx"
  - "SRDF operations when R1 is target for Snapvx"
  - "SRDF operations when R2 is source of Snapvx"
  - "SRDF operations when R2 is target for Snapvx"
- SRDF set operations for TimeFinder Snapvx sessions..... 307
  - "SRDF set operations when R1 is source of TimeFinder Snapvx"
  - "SRDF set operations when R1 is target of TimeFinder Snapvx"
  - "SRDF set operations when R2 is source of TimeFinder Snapvx"
  - "SRDF set operations when R2 is target of TimeFinder Snapvx"

SRDF operations during TimeFinder/Snap/VP Snap sessions

This section lists the allowable SRDF operations for TimeFinder/Snap and TimeFinder VP Snap copy sessions on the R1 source and target and the R2 source and target.

Note: TimeFinder/Snap and TimeFinder VP Snap are separate features. They are combined in this section because their interactions with SRDF are very similar.

Some footnotes in the tables below refer to devices that are not pace-capable. For additional details, see "Identifying devices that cannot be paced in a cascaded SRDF configuration" on page 134.

SRDF operations when R1 is source of TimeFinder Snap/VP Snap

Table 32 identifies the allowable SRDF actions when the R1 is the source of a TimeFinder/Snap or VP Snap session.

Table 32 Allowable SRDF operations when R1 is the source of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y	Y	Y	Y	Y[3]	Y	Y		
createpair -restore	Y	Y	Y	Y	Y[2]	Y[3]	Y	Y		
createpair -invalidate R1	Y	Y	Y	Y	Y	Y[3]	Y	Y		
createpair -invalidate R2	Y	Y	Y	Y	Y	Y[3]	Y	Y		
createpair -format	Y									
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
half_swap	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
swap -refresh R1	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
swap -refresh R2	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
establish	Y	Y	Y	Y	Y	Y[3]	Y	Y		
establish -full	Y	Y	Y	Y	Y	Y[3]	Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y	Y	Y	Y	Y[3]	Y	Y		
restore -full	Y	Y	Y	Y	Y[2]		Y	Y		

Table 32 Allowable SRDF operations when R1 is the source of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
update	Y	Y	Y	Y	Y	Y[3]	Y	Y		
failback	Y	Y	Y	Y	Y	Y[3]	Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
failover -restore	Y	Y[1]	Y[1]	Y[1]	Y[1]			Y[1]		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y	Y	Y	Y[3]	Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y[3]	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y	Y	Y	Y	Y[3]	Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 32 Allowable SRDF operations when R1 is the source of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if the devices are in asynchronous mode and R1 and R2 array are running Enginuity 5876 or less and there is a TimeFinder/Snap off of the R1 and either:
   a. The SRDF pair is an R22->R1 of a concurrent R2 setup in which either:
      i. The R22 array is running an Enginuity level less than 5876 or the R1 array is running an Enginuity level less than 5875..
      ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.
   b. The SRDF pair is not an R22->R1 of a concurrent R2 setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.
2. Allowed for TimeFinder VP Snap. If not TimeFinder VP Snap, must use -force.
3. Only allowed for TimeFinder VP Snap.

SRDF operations when R1 is target of TimeFinder Snap/VP Snap

Table 33 identifies the allowable SRDF actions when R1 is the target of a TimeFinder/Snap or VP Snap session.

Table 33 Allowable SRDF operations when R1 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -restore	Y						Y	Y		
createpair -invalidate R1	Y						Y	Y		
createpair -invalidate R2	Y						Y	Y		
createpair -format	Y									
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 33 Allowable SRDF operations when R1 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
swap	Y							Y		
half_swap	Y							Y		
swap -refresh R1	Y							Y		
swap -refresh R2	Y							Y		
establish	Y						Y	Y		
establish-full	Y						Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y						Y	Y		
restore -full	Y						Y	Y		
update	Y						Y	Y		
failback	Y						Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y							Y		
failover -restore	Y							Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y						Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y						Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y						Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 33 Allowable SRDF operations when R1 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for TimeFinder/Snap R1 targets.

SRDF operations when R2 is source of TimeFinder Snap/VP Snap

Table 34 identifies the allowable SRDF actions when the R2 is the source of a TimeFinder/Snap or VP Snap copy session.

Table 34 Allowable SRDF operations when R2 is the source of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y[2]	Y[2]	Y[2]	Y[2]			Y[2]		
createpair -restore	Y	Y[2]	Y[2]	Y[2]	Y[2]			Y[2]		
createpair -invalidate R1	Y	Y[2]	Y[2]	Y[2]	Y[2]			Y[2]		
createpair -invalidate R2	Y	Y[2]	Y[2]	Y[2]	Y[2]			Y[2]		
createpair -format	Y									
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 34 Allowable SRDF operations when R2 is the source of a TimeFinder/Snap or VP Snap (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y[4]	Y[4]	Y[4]	Y[4]	Y	Y	Y[4]		
half_movepair	Y	Y[4]	Y[4]	Y[4]	Y[4]	Y	Y	Y[4]		
swap	Y	Y	Y	Y	Y		Y	Y		
half_swap	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R1	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R2	Y	Y	Y	Y	Y		Y	Y		
establish	Y	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	Y[2]		
establish -full	Y	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	Y[2]		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y[2]	Y[2]	Y[2]	Y[2]	Y[1]	Y[2]	Y[2]		
restore -full	Y	Y[2]	Y[2]	Y[2]	Y[2]	Y[1]	Y[2]	Y[2]		
update	Y	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	Y[2]		
failback	Y	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	Y[2]		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y	Y	Y	Y		Y	Y		
failover -restore	Y	Y	Y	Y	Y		Y	Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y	Y	Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y		Y	Y		

Table 34 Allowable SRDF operations when R2 is the source of a TimeFinder/Snap or VP Snap (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y[2]	Y[2]	Y[2]	Y[2]		Y[2]	Y[2]		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Only allowed if the required Enginuity levels or patches are detected. If TimeFinder VP Snap, requires Enginuity version 5876 Q42012 SR and higher.
2. Not allowed if the devices are in asynchronous mode and R1 and R2 array are running Enginuity 5876 or less and there is a TimeFinder/Snap off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 array is running an Enginuity level less than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R21 side.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.

3. If the SRDF/A session is in the Transmit Idle state, you must issue the command with -symforce from the R1 side.
4. Not allowed if the devices are moving to a group operating in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less, and there is a TimeFinder/Snap off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 array is running an Enginuity level lower than 5876, or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
5. If a TimeFinder VP Snap, you must use -force.

SRDF operations when R2 is target of TimeFinder Snap/VP Snap

Table 35 identifies the allowable SRDF actions when the R2 is the target of a TimeFinder/Snap or VP Snap copy session.

Table 35 Allowable SRDF operations when R2 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y								Y		
createpair -restore	Y								Y		
createpair -invalidate R1	Y								Y		
createpair -invalidate R2	Y								Y		
createpair -format	Y										
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y					Y		Y	Y		
half_swap	Y					Y		Y	Y		
swap -refresh R1	Y					Y		Y	Y		
swap -refresh R2	Y					Y		Y	Y		
establish	Y					Y		Y	Y		
establish -full	Y					Y		Y	Y		

Table 35 Allowable SRDF operations when R2 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y					Y		Y	Y		
restore -full	Y					Y		Y	Y		
update	Y					Y		Y	Y		
failback	Y					Y		Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y					Y		Y	Y		
failover -restore	Y					Y		Y	Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y					Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y					Y		Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y					Y		Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 35 Allowable SRDF operations when R2 is the target of a TimeFinder/Snap or VP Snap

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

SRDF operations for TimeFinder/Clone sessions

This section lists the allowable SRDF operations for TimeFinder /Clone copy sessions on the R1 source and target and the R2 source and target.

SRDF operations when R1 is source of TimeFinder Clone

Table 36 identifies the allowable SRDF actions when the R1 is the source of a TimeFinder/Clone copy session.

Table 36 Allowable SRDF operations when R1 is the source of a TimeFinder/Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -invalidate R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -invalidate R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
half_swap	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
swap -refresh R1	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
swap -refresh R2	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
establish -full	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
restore -full	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
update	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
failback	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
failover -restore	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		

Table 36 Allowable SRDF operations when R1 is the source of a TimeFinder/Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if the devices are in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the R1 and

either:

a. The SRDF pair is an R22->R1 of a concurrent R2 setup in which either:

   i. The R22 array is running an Enginuity level less than 5876 or the R1 array is running an Enginuity level less than 5875.

   ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.

b. The SRDF device pair is not an R22->R1 of a concurrent R2 setup and either:

   i. The Enginuity level is lower than 5875 on either the R1 or R2 array.

   ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.

SRDF operations when R1 is target of TimeFinder Clone

Table 37 identifies the allowable SRDF actions when the R1 is the target of a TimeFinder/Clone copy session.

Table 37 Allowable SRDF operations when R1 is the target of a TimeFinder/Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y						Y			Y		Y	Y		
createpair -restore	Y						Y			Y		Y	Y		
createpair -invalidate R1	Y						Y			Y		Y	Y		
createpair -invalidate R2	Y						Y			Y		Y	Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y												Y		
half_swap	Y												Y		
swap -refresh R1	Y												Y		
swap -refresh R2	Y												Y		
establish	Y					Y	Y			Y		Y	Y		
full establish	Y					Y	Y			Y		Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y						Y			Y		Y	Y		
restore -full	Y						Y			Y		Y	Y		

Table 37 Allowable SRDF operations when R1 is the target of a TimeFinder/Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
update	Y						Y			Y		Y	Y		
failback	Y						Y			Y		Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y												Y		
failover -restore	Y												Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y						Y			Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y						Y			Y		Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y					Y	Y			Y		Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 37 Allowable SRDF operations when R1 is the target of a TimeFinder/Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for TimeFinder/Clone R1 targets.

SRDF operations when R2 is source of TimeFinder Clone

Table 38 identifies the allowable SRDF actions when the R2 is the source of a TimeFinder/Clone copy session.

Table 38 Allowable SRDF operations when R2 is the source of a TimeFinder/Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[1,3]	Y[3]	Y			Y[3]		
createpair -restore	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y			Y[3]		
createpair -invalidate R1	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y			Y		
createpair -invalidate R2	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y			Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y[5]	Y[5]	Y[5]	Y	Y	Y	Y[5]	Y[5]	Y	Y	Y	Y[5]		
half_movepair	Y	Y[5]	Y[5]	Y[5]	Y	Y	Y	Y[5]	Y[5]	Y	Y	Y	Y[5]		
swap	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
half_swap	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
establish	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
full establish	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3,1]	Y[3]	Y		Y	Y[3]		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[2]	Y	Y[3]		
restore -full	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[2]	Y	Y[3]		
update	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
failback	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
failover -restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 38 Allowable SRDF operations when R2 is the source of a TimeFinder/Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
merge	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[3]	Y	Y[3]		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[3]	Y	Y[3]		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Only allowed if required Enginuity levels or patches are detected and not a TimeFinder VP Snap.
2. Not allowed if the devices are in async mode and there is a TimeFinder/Clone off of the R2 either:
   a. The SRDF pair will become an R21->R2 for which any of the following apply:
      i. The R21 array is running an Enginuity level lower than 5876 Q42012 SR, or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.
      iii. If the R21->R2 pair will be read/write (RW) on the SRDF link, the R21 must be pace-capable.
   b. The SRDF device pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.
3. Not allowed if the devices are in asynchronous mode and R1 and R2 array are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 Symmetrix is running an Enginuity level less than 5876 or the R2 Symmetrix is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R21 side.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 Symmetrix array.
      i. ii.SRDF/A device-level write pacing is not configured for autostart on the R1 side.
4. Not allowed if the devices are moving to a group operating in async mode and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup and any of the following apply:
      i. The R21 array is running an Enginuity level lower than 5876 Q42012 SR, or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
   b. The SRDF device pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
5. Not allowed if the devices are moving to a group operating in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 array is running an Enginuity level less than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side

of the new group.

SRDF operations when R2 is target of TimeFinder Clone

Table 39 identifies the allowable SRDF actions when the R2 is the target of a TimeFinder/Clone copy session.

Table 39 Allowable SRDF operations when R2 is the target of a TimeFinder/Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y												Y		
createpair -restore	Y												Y		
createpair -invalidate R1	Y												Y		
createpair -invalidate R2	Y												Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y						Y			Y		Y	Y		
half_swap	Y						Y			Y		Y	Y		
swap -refresh R1	Y						Y			Y		Y	Y		
swap -refresh R2	Y						Y			Y		Y	Y		
establish	Y						Y			Y		Y	Y		
establish -full	Y						Y			Y		Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y						Y			Y		Y	Y		
restore -full	Y						Y			Y		Y	Y		
update	Y						Y			Y		Y	Y		
failback	Y						Y			Y		Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y						Y			Y		Y	Y		
failover -restore	Y						Y			Y		Y	Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 39 Allowable SRDF operations when R2 is the target of a TimeFinder/Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
merge	Y						Y			Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y						Y			Y		Y	Y		
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y						Y			Y		Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the TimeFinder/Clone R2 targets.

SRDF operations for Extent-level TimeFinder/Clone sessions

This section lists the allowable SRDF operations for Extent-level TimeFinder /Clone copy sessions on the R1 source and target and the R2 source and target.

SRDF operations when R1 is source of Extent-level Clone

Table 40 identifies the allowable SRDF actions when the R1 is the source of an Extent-level TimeFinder/Clone copy session.

Table 40 Allowable SRDF operations when R1 is the source of an Extent-level Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -invalidate R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -invalidate R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
swap	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
half_swap	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
swap -refresh R1	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
swap -refresh R2	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
establish -full	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
restore -full	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
update	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
failback	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		
failover -restore	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y			Y[1]		

Table 40 Allowable SRDF operations when R1 is the source of an Extent-level Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if the devices are operating in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the

R1 and either:

a. The SRDF pair is an R22->R1 of a concurrent R2 setup in which either:

   i. The R22 array is running an Enginuity level lower than 5876, or the R1 array is running an Enginuity level lower than 5875.

   ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.

b. The SRDF device pair is not an R22->R1 of a concurrent R2 setup and either:

   i. The Enginuity level is lower than 5875 on either the R1 or R2 array.

   ii. SRDF/A device-level write pacing is not configured for autostart on the R2 side.

SRDF operations when R1 is target of Extent-level Clone

Table 41 identifies the allowable SRDF actions when the R1 is the target of an Extent-level TimeFinder/Clone copy session.

Table 41 Allowable SRDF operations when the R1 is the target of an Extent-level Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y														
createpair -restore	Y														
createpair -invalidate R1	Y														
createpair -invalidate R2	Y														
createpair -format	Y														
deletepair	Y														
half_deletepair	Y														
movepair	Y														
half_movepair	Y														
swap	Y														
half_swap	Y														
swap -refresh R1	Y														
swap -refresh R2	Y														
establish	Y					Y	Y			Y		Y	Y		
establish -full	Y					Y	Y			Y		Y	Y		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y														
restore -full	Y														

Table 41 Allowable SRDF operations when the R1 is the target of an Extent-level Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
update	Y														
failback	Y														
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y														
failover -restore	Y														
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
merge	Y														
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not _ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y														
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y					Y	Y			Y		Y	Y		
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 41 Allowable SRDF operations when the R1 is the target of an Extent-level Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the Extent-level TimeFinder/Clone R1 targets.

SRDF operations when R2 is source of Extent-level Clone

Table 42 identifies the allowable SRDF actions when the R2 is the source of an Extent-level TimeFinder/Clone copy session.

Table 42 Allowable SRDF operations when R2 is the source of an Extent-level Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[1,3]	Y[3]	Y			Y[3]		
createpair -restore	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y			Y[3]		
createpair -invalidate R1	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y			Y		
createpair -invalidate R2	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y[3]			Y		
createpair -format	Y														
deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y[5]	Y[5]	Y[5]	Y	Y	Y	Y[5]	Y[5]	Y	Y	Y	Y[5]		
half_movepair	Y	Y[5]	Y[5]	Y[5]	Y	Y	Y	Y[5]	Y[5]	Y	Y	Y	Y[5]		
swap	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
half_swap	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
swap -refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
establish	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
establish -full	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[1,3]	Y[3]	Y		Y	Y[3]		
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[2]	Y	Y[3]		
restore -full	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[2]	Y	Y[3]		
update	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
failback	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
failover -restore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 42 Allowable SRDF operations when R2 is the source of an Extent-level Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
merge	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not _ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y		
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y		Y	Y[3]		
rw_disable R2	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[3]	Y	Y[3]		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y[3]	Y[3]	Y[3]	Y	Y	Y	Y[3]	Y[3]	Y	Y[3]	Y	Y[3]		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Only allowed if required Enginuity levels or patches are detected.
2. Not allowed if the devices are in async mode and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair will become an R21->R2 for which any of the following apply:
      i. The R21 array is running an Enginuity level lower than 5876 Q42012 SR, or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.
      iii. If the R21->R2 pair will be read/write (RW) on the SRDF link, the R21 must be pace-capable.
   b. The SRDF device pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.
3. Not allowed if the devices are in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 Symmetrix is running an Enginuity level less than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R21 side.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side.
4. Not allowed if the devices are moving to a group operating in asynchronous mode and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup and any of the following apply:
      i. The R21 array is running an Enginuity level lower than 5876 Q42012 SR, or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
   b. The SRDF device pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
5. Not allowed if the devices are moving to a group operating in asynchronous mode and R1 and R2 arrays are running Enginuity 5876 or less and there is a TimeFinder/Clone off of the R2 and either:
   a. The SRDF pair is an R21->R2 of a cascaded setup in which either:
      i. The R21 array is running an Enginuity level less than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.
   b. The SRDF pair is not an R21->R2 of a cascaded setup and either:
      i. The Enginuity level is lower than 5875 on either the R1 or R2 array.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 side of the new group.

SRDF operations when R2 is target of Extent-level Clone

Table 43 identifies the allowable SRDF actions when the R2 is the target of an Extent-level TimeFinder/Clone copy session.

Table 43 Allowable SRDF operations when the R2 is the target of an Extent-level Clone

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
createpair -establish	Y														
createpair -restore	Y														
createpair -invalidate R1	Y														
createpair -invalidate R2	Y														
createpair -format	Y														
deletepair	Y														
half_deletepair	Y														
movepair	Y														
half_movepair	Y														
swap	Y														
half_swap	Y														
swap -refresh R1	Y														
swap -refresh R2	Y														
establish	Y														
establish -full	Y														
split	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
restore	Y														
restore -full	Y														
update	Y														
failback	Y														
failover	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y														
failover -restore	Y														
invalidate -R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 43 Allowable SRDF operations when the R2 is the target of an Extent-level Clone (continued)

SRDF control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
merge	Y														
msc_cleanup	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
refresh R2	Y														
suspend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume	Y														
rw_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for Extent-level TimeFinder/Clone R2 targets.

SRDF set operations for TimeFinder/Snap sessions

This section lists the allowable SRDF set operations for TimeFinder /Snap copy sessions on the R1 source and target and the R2 source and target.

SRDF set operations when R1 is source of TimeFinder/Snap

Table 44 identifies the allowable SRDF set actions when the R1 is the source of a TimeFinder/Snap copy session.

Table 44 Allowable SRDF set operations when R1 is the source of a TimeFinder/Snap

SRDF set operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R1 is target of TimeFinder/Snap

Table 45 identifies the allowable SRDF set actions when the R1 is the target for a TimeFinder/Snap copy session.

Table 45 Allowable SRDF set operations when R1 is the target of a TimeFinder/Snap

SRDF set operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R2 is source of TimeFinder/Snap

Table 46 identifies the allowable SRDF set actions when the R2 is the source of a TimeFinder/Snap copy session.

Table 46 Allowable SRDF set operations when R2 is the source of a TimeFinder/Snap

SRDF set operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y[1]	Y[1]	Y[1]	Y[1]		Y[1]	Y[1]		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. If the R2 is not an extent-based TimeFinder/Snap source device and R1 and R2 arrays are running Enginuity 5876 or less, then not allowed if either of the following is true:
   a. The RDF device pair is the R21->R2 of a cascaded setup and either:
      i. The R21 array is running an Enginuity level less than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 group of the R21 device.
   b. The SRDF pair is not the R21->R2 of a cascaded setup and either:
      i. The R1 or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 group.

SRDF set operations when R2 is target of TimeFinder/Snap

Table 47 identifies the allowable SRDF set when the R2 is the target of a TimeFinder/Snap copy session.

Table 47 Allowable SRDF set operations when R2 is the target of a TimeFinder/Snap

SRDF set operation:	No Session	Create in Progress	Created	Recreated	Copy on Write	Copied	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y										
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations for TimeFinder/Clone sessions

This section lists the allowable SRDF set operations for TimeFinder /Clone copy sessions on the R1 and R2 source and the R1 and R2 target.

SRDF set operations when R1 is source of TimeFinder/Clone

Table 48 identifies the allowable SRDF set actions when the R1 is the source of a TimeFinder/Clone copy session.

Table 48 Allowable SRDF set operations when R1 is the source of a TimeFinder/Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R1 is target of TimeFinder/Clone

Table 49 identifies the allowable SRDF set actions when the R1 is the target of a TimeFinder/Clone copy session.

Table 49 Allowable SRDF set operations when R1 is the target of a TimeFinder/Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y		Y	Y	Y	Y		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. Not allowed if TimeFinder/Clone pair was created with -copy.

SRDF set operations when R2 is source of TimeFinder/Clone

Table 50 identifies the allowable SRDF set actions when the R2 is the source of a TimeFinder/Clone copy session.

Table 50 Allowable SRDF set operations when R2 is the source of a TimeFinder/Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y[1]	Y[1]	Y[1]	Y	Y	Y	Y[1]	Y[1]	Y		Y	Y[1]		
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

1. If the R2 is not an extent-based TimeFinder/Clone source device and R1 and R2 array are running Enginuity 5876 or less, then not allowed if either of the following is true:
   a. The RDF device pair is the R21->R2 of a cascaded setup and either:
      i. The R21 array is running an Enginuity level lower than 5876 or the R2 array is running an Enginuity level less than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 group of the R21 device.
   b. The SRDF pair is not the R21->R2 of a cascaded setup and either:
      i. The R1 or the R2 array is running an Enginuity level lower than 5875.
      ii. SRDF/A device-level write pacing is not configured for autostart on the R1 group.

SRDF set operations when R2 is target of TimeFinder/Clone

Table 51 identifies the allowable SRDF set actions when the R2 is the target of a TimeFinder/Clone copy session.

Table 51 Allowable SRDF set operations when R2 is the target of a TimeFinder/Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y														
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations for Extent-level TimeFinder/Clone sessions

This section lists the allowable SRDF set operations for Extent-level TimeFinder /Clone copy sessions on the R1 and R2 source and the R1 and R2 target.

SRDF set operations when R1 is source of Extent-level Clone

Table 52 identifies the allowable SRDF set actions when the R1 is the source of an Extent-level TimeFinder/Clone copy session.

Table 52 Allowable SRDF set operations when R1 is the source of an Extent-level Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R1 is target of Extent-level Clone

Table 53 identifies the allowable SRDF set actions when the R1 is the target of an Extent-level TimeFinder/Clone copy session.

Table 53 Allowable SRDF set operations when R1 is the target of an Extent-level Clone

SRDF SET control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R2 is source of Extent-level Clone

Table 54 identifies the allowable SRDF set actions when the R2 is the source of an Extent-level TimeFinder/Clone copy session.

Table 54 Allowable SRDF set operations when the R2 is the source of an Extent-level Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: There are no subscripts for the operations in this table.

SRDF set operations when R2 is target of Extent-level Clone

Table 55 identifies the allowable SRDF set actions when the R2 is the target of an Extent-level TimeFinder/Clone copy session.

Table 55 Allowable SRDF set operations when R2 is the target of an Extent-level Clone

SRDF set control operation:	No Session	Create in Progress	Created	Recreated	Precopy	Copy in Progress	Copied	Copy on Write	Copy on Access	Split	Restore in Progress	Restored	Term in Progress	Invalid	Failed
set mode async	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode sync	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

SRDF operations for TimeFinder Snapvx

This section lists the allowable SRDF operations for TimeFinder Snapvx sessions on VMAX 3 arrays running HYPERMAX OS.

SRDF operations when R1 is source of Snapvx

Table 56 identifies allowable SRDF operations when the R1 is the source of a TimeFinder Snapvx session

Table 56 Allowable SRDF operations when R1 is the source of a TimeFinder Snapvx

SRDF Control Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
createpair - establish	Y	Y	Y		Y	Y	
createpair - restore	Y	Y	Y		Y	Y	
createpair - invalidate R1	Y	Y	Y		Y	Y	
createpair - invalidate R2	Y	Y	Y		Y	Y	
createpair - format	Y						
deletepair	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	Y
half_movepair	Y	Y	Y	Y	Y	Y	Y
swap	Y	Y	Y			Y	
half_swap	Y	Y	Y			Y	
swap -refresh R1	Y	Y	Y			Y	
swap -refresh R2	Y	Y	Y			Y	
establish	Y	Y	Y		Y	Y	
establish -full	Y	Y	Y		Y	Y	
split	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y	Y		Y	Y	
restore -full	Y	Y	Y		Y	Y	
update	Y	Y	Y		Y	Y	
failback	Y	Y	Y		Y	Y	
failover	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y	Y			Y	

Table 56 Allowable SRDF operations when R1 is the source of a TimeFinder Snapvx

SRDF Control Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
failover -restore	Y	Y	Y			Y	
invalidate -R1	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y		Y	Y	
msc_cleanup	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y
ready R1	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y		Y	Y	
refresh R2	Y	Y	Y	Y	Y	Y	
suspend	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y	Y		Y	Y	
rw_disable R2	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y	Y	Y	Y	Y	
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y	Y	Y	Y	Y	
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y

SRDF operations when R1 is target for Snapvx

Table 57 identifies allowable SRDF operations when the R1 is the target of a TimeFinder Snapvx session

Table 57 Allowable SRDF operations when R1 is the target of a TimeFinder Snapvx

SRDF Control Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
createpair - establish	Y		Y		
createpair - restore	Y		Y		
createpair - invalidate R1	Y		Y		
createpair - invalidate R2	Y		Y		
createpair - format	Y				
deletepair	Y	Y	Y		Y
half_deletepair	Y	Y	Y		Y
movepair	Y	Y	Y		Y
half_movepair	Y	Y	Y		Y
swap	Y				
half_swap	Y				
swap -refresh R1	Y				
swap -refresh R2	Y				
establish	Y	Y	Y		
establish -full	Y	Y	Y		
split	Y	Y	Y		Y
restore	Y		Y		
restore -full	Y		Y		
update	Y		Y		
failback	Y		Y		
failover	Y	Y	Y		Y
failover -establish	Y				
failover -restore	Y				
invalidate -R1	Y	Y	Y		Y
invalidate -R2	Y	Y	Y		Y
merge	Y		Y		
msc_cleanup	Y	Y	Y		Y

Table 57 Allowable SRDF operations when R1 is the target of a TimeFinder Snapvx

SRDF Control Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
not_ready R1	Y	Y	Y		Y
not_ready R2	Y	Y	Y		Y
ready R1	Y	Y	Y		Y
ready R2	Y	Y	Y		Y
refresh R1	Y		Y		
refresh R2	Y	Y	Y		
suspend	Y	Y	Y		Y
resume	Y		Y		
rw_disable R2	Y	Y	Y		Y
rw_enable R1	Y	Y	Y		Y
rw_enable R2	Y	Y	Y		Y
write_disable R1	Y	Y	Y		Y
write_disable R2	Y	Y	Y		Y
activate -rdfa_dse	Y	Y	Y		Y
deactivate -rdfa_dse	Y	Y	Y		Y
activate -rdfa_devpace	Y	Y	Y		Y
deactivate -rdfa_devpace	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y		Y
deactivate -rdfa_pace	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y		Y
deactivate -rdfa_wpace	Y	Y	Y		Y
activate -rdfa_wpace_exempt	Y	Y	Y		Y
deactivate -rdfa_wpace_exempt	Y	Y	Y		Y

Note: There are no subscripts for the operations in this table.

SRDF operations when R2 is source of Snapvx

Table 58 identifies allowable SRDF operations when the R2 is the source of a TimeFinder Snapvx session

Table 58 Allowable SRDF operations when R2 is the source of a TimeFinder Snapvx

SRDF Control Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
createpair - establish	Y	Y	Y[2]			Y	
createpair - restore	Y	Y	Y			Y	
createpair - invalidate R1	Y	Y	Y			Y	
createpair - invalidate R2	Y	Y	Y			Y	
createpair - format	Y						
deletepair	Y	Y	Y	Y	Y	Y	Y
half_deletepair	Y	Y	Y	Y	Y	Y	Y
movepair	Y	Y	Y	Y	Y	Y	
half_movepair	Y	Y	Y	Y	Y	Y	
swap	Y	Y	Y		Y	Y	
half_swap	Y	Y	Y		Y	Y	
swap -refresh R1	Y	Y	Y		Y	Y	
swap -refresh R2	Y	Y	Y		Y	Y	
establish	Y	Y	Y		Y	Y	
establish -full	Y	Y	Y[2]		Y	Y	
split	Y	Y	Y	Y	Y	Y	Y
restore	Y	Y	Y	Y	Y	Y	
restore -full	Y	Y	Y	Y	Y	Y	
update	Y	Y	Y		Y	Y	
failback	Y	Y	Y		Y	Y	
failover	Y	Y	Y	Y	Y	Y	Y
failover -establish	Y	Y	Y		Y	Y	
failover -restore	Y	Y	Y		Y	Y	
invalidate -R1	Y	Y	Y	Y	Y	Y	Y
invalidate -R2	Y	Y	Y	Y	Y	Y	Y
merge	Y	Y	Y		Y	Y	
msc_cleanup	Y	Y	Y	Y	Y	Y	Y
not_ready R1	Y	Y	Y	Y	Y	Y	Y
not_ready R2	Y	Y	Y	Y	Y	Y	Y

Table 58 Allowable SRDF operations when R2 is the source of a TimeFinder Snapvx

SRDF Control Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
ready R1	Y	Y	Y	Y	Y	Y	Y
ready R2	Y	Y	Y	Y	Y	Y	Y
refresh R1	Y	Y	Y	Y	Y	Y	
refresh R2	Y	Y	Y		Y	Y	
suspend	Y	Y	Y	Y	Y	Y	Y
resume	Y	Y	Y		Y	Y	
rw_disable R2	Y	Y	Y	Y	Y	Y	Y
rw_enable R1	Y	Y	Y	Y	Y	Y	Y
rw_enable R2	Y	Y	Y	Y	Y	Y	Y
write_disable R1	Y	Y	Y	Y	Y	Y	Y
write_disable R2	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_dse	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_devpace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_devpace	Y	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	
activate -rdfa_pace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_pace	Y	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	
activate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace	Y	Y	Y	Y	Y	Y	Y
activate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y
deactivate -rdfa_wpace_exempt	Y	Y	Y	Y	Y	Y	Y

1. If the SRDF/A session is in Transmit Idle state, you must use symforce from the R1 side.
2. Must use -force

SRDF operations when R2 is target for Snapvx

Table 59 identifies allowable SRDF operations when the R2 is the target of a TimeFinder Snapvx session.

Table 59 Allowable SRDF operations when R2 is the target of a TimeFinder Snapvx

SRDF Control Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
createpair - establish	Y				
createpair - restore	Y				
createpair - invalidate R1	Y				
createpair - invalidate R2	Y				
createpair - format	Y				
deletepair	Y	Y	Y		Y
half_deletepair	Y	Y	Y		Y
movepair	Y	Y	Y		Y
half_movepair	Y	Y	Y		Y
swap	Y		Y		
half_swap	Y		Y		
swap -refresh R1	Y		Y		
swap -refresh R2	Y		Y		
establish	Y		Y		
establish -full	Y		Y		
split	Y	Y	Y		Y
restore	Y		Y		
restore -full	Y		Y		
update	Y		Y		
failback	Y		Y		
failover	Y	Y	Y		Y
failover -establish	Y		Y		
failover -restore	Y		Y		
invalidate -R1	Y	Y	Y		Y
invalidate -R2	Y	Y	Y		Y

Table 59 Allowable SRDF operations when R2 is the target of a TimeFinder Snapvx

SRDF Control Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
merge	Y		Y		
msc_cleanup	Y	Y	Y		Y
not_ready R1	Y	Y	Y		Y
not_ready R2	Y	Y	Y		Y
ready R1	Y	Y	Y		Y
ready R2	Y		Y		
refresh R1	Y	Y	Y		
refresh R2	Y	Y	Y		
suspend	Y	Y	Y		Y
resume	Y		Y		
rw_disable R2	Y	Y	Y		Y
rw_enable R1	Y	Y	Y		Y
rw_enable R2	Y	Y	Y		Y
write_disable R1	Y	Y	Y		Y
write_disable R2	Y	Y	Y		Y
activate -rdfa_dse	Y	Y	Y		Y
deactivate -rdfa_dse	Y	Y	Y		Y
activate -rdfa_devpace	Y	Y	Y		Y
deactivate -rdfa_devpace	Y	Y	Y		
activate -rdfa_pace	Y	Y	Y		Y
deactivate -rdfa_pace	Y	Y	Y		
activate -rdfa_wpace	Y	Y	Y		Y
deactivate -rdfa_wpace	Y	Y	Y		Y
activate -rdfa_wpace_exempt	Y	Y	Y		Y
deactivate -rdfa_wpace_exempt	Y	Y	Y		Y

SRDF set operations for TimeFinder Snapvx sessions

This section lists the allowable SRDF set operations for Timefinder Snapvx sessions on the R1 and R2 source, and the R1 and R2 target.

SRDF set operations when R1 is source of TimeFinder Snapvx

Table 60 identifies allowable SRDF set operations when the R1 is the source of a TimeFinder Snapvx session.

Table 60 Allowable SRDF set operations when R1 is the source of a TimeFinder Snapvx

SRDF Set Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
set mode async	Y	Y	Y	Y	Y	Y	
set mode sync	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y

SRDF set operations when R1 is target of TimeFinder Snapvx

Table 61 identifies allowable SRDF set operations when the R1 is the target of a TimeFinder Snapvx session.

Table 61 Allowable SRDF set operations when R1 is the target of a TimeFinder Snapvx

SRDF Set Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
set mode async	Y	Y	Y	Y	
set mode sync	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y

SRDF set operations when R2 is source of TimeFinder Snapvx

Table 62 identifies allowable SRDF set operations when the R2 is the source of a TimeFinder Snapvx session.

Table 62 Allowable SRDF set operations when R2 is the source of a TimeFinder Snapvx

SRDF Set Operation:	No Snapshot	Establish in Progress	Established	Restore in Progress	Restored	Term in Progress	Failed
set mode async	Y	Y	Y		Y	Y	
set mode sync	Y	Y	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y	Y	Y

SRDF set operations when R2 is target of TimeFinder Snapvx

Table 63 identifies allowable SRDF set operations when the R2 is the target of a TimeFinder Snapvx session.

Table 63 Allowable SRDF set operations when R2 is the target of a TimeFinder Snapvx

SRDF Set Operation:	No Link	Link Copy in Progress	Link Copied	Linked	Failed
set mode async	Y				
set mode sync	Y	Y	Y	Y	Y
set mode acp_disk	Y	Y	Y	Y	Y
set mode acp_wp	Y	Y	Y	Y	Y