Engineering White Paper

EMC² where information lives

Using SYMCLI to Perform Control Operations with SRDF Family Products

Abstract

This paper provides an introduction to the SRDF functionality that allows you to transmit copies of the data from a Symmetrix unit located at the production site to a remotely located Symmetrix unit.

Published 4/5/2004

Part Number 300-000-076 REV H H604.6

Table of Contents

Introduction

The SRDF® product family (referred to as "SRDF" in this document) provides a mirrored data storage solution that allows you to duplicate production site data on one or more remote target Symmetrix® systems. When your main systems are down for a planned or unplanned event, SRDF enables fast switchover from the source data to the target copy.

SRDF over IP (Internet Protocol) can create copies of the data and transmit these copies over high-bandwidth IP-based Virtual Private Networks to a remote Symmetrix system. When distance impacts performance, SRDF can use delayed synchronization methods of data replication to ensure data mobility.

SRDF products currently support the following methods of data replication:

- *Synchronous Replication* provides real-time mirroring of data between the source Symmetrix and the target Symmetrix systems. Data is written simultaneously to the cache of both systems in real time before the application I/O is completed, thus ensuring the highest possible data availability.
- *Semi-Synchronous Replication* writes data to the source system, completes the I/O, and then synchronizes the data with the target system. Since the I/O is completed prior to synchronizing data with the target system, this method provides an added performance advantage. A second write will not be accepted on a Symmetrix source device until its target device has been synchronized.
- *Adaptive Copy Replication* transfers data from the source devices to the remote devices without waiting for an acknowledgment. This is especially useful when transferring large amounts of data during data center migrations, consolidations, and in data mobility environments.
- *Asynchronous Replication* places host writes into cycles or "chunks" and then transfers the entire chunks to the target system. When a complete chunk is received on the R2 side, the copy cycle is committed. If the RDF links are lost during data transfer, any partial chunk is discarded, preserving consistency on the R2. Beginning with Solutions Enabler version 5.3 running on Symmetrix units using Enginuity™ version 5670, this method provides a consistent point-in-time R2 image that is not far behind the R1 side and results in minimal data loss if there is a disaster at the source site.

All methods of replication can co-exist in a Symmetrix unit, allowing you to specify the method on a per-device basis. No special application coding is required and no CPU overhead is incurred.

The local SRDF device, known as the *source (R1) device*, is configured in a pairing relationship with a remote *target (R2) device*, forming an *SRDF pair*. While the R2 device is mirrored with the R1 device, the R2 device is write disabled or not ready to its host. (*Not ready* means disabled for both reads and writes.) After the R2 device becomes synchronized with its R1 device, you can split the R2 device from the R1 device at any time, making the R2 device fully accessible again to its host. After the split, the target (R2) device contains valid data and is available for performing business continuance tasks through its original device address or restoring (copying) data to the source (R1) device if there is a loss of data on that device.

Purpose and Scope

This document describes SRDF functionality in versions of EMC® Solutions Enabler up to version 5.4 running on Symmetrix units using Enginuity versions 5x65 to 5x67, 5568, 5669, and 5670.

Related Documentation

The following documents provide information related to the concepts discussed in this paper:

- *EMC Solutions Enabler Symmetrix SRDF CLI Product Guide*
- *EMC Solutions Enabler Symmetrix Base Control Product Guide*
- *EMC Solutions Enabler Symmetrix TimeFinder CLI Product Guide*

- *Using SYMCLI to Query and Verify with SRDF Family Products* (P/N 300-000-077)
- *Using SYMCLI to Implement RDF Consistency Protection with SRDF Family Products* (P/N 300-000-284)
- *Using SYMCLI to Set Up TimeFinder/Mirror BCV Pairs* (P/N 300-000-072)
- *Using SYMCLI to Perform TimeFinder/Mirror Control Operations* (P/N 300-000-074)
- *Using SYMCLI to Perform SRDF/AR* (P/N 300-000-078)

Practical Uses

Practical uses of *suspend* and *resume* operations usually involve unplanned situations in which you want to immediately suspend I/O between the R1 and R2 devices over the RDF links. In this way, you can stop any data propagation problems or perform immediate backups without affecting I/O from the local host application. You can then resume I/O between the R1 and R2 and return to normal operation.

Practical uses of *establish* and *split* operations usually involve planned situations in which you want to use the R2 copy of the data without interfering with normal write operations to the R1 device. Splitting a point-in-time copy of data allows you to access that data on the R2 device for various business continuance tasks. The ease of splitting SRDF pairs into separate database instances providing exact copies makes it convenient to perform scheduled backup operations, data warehousing, or new application testing from the target Symmetrix data while normal operations continue on the source Symmetrix.

You can also use the R2 copy to test disaster recovery plans without manually intensive recovery drills, complex procedures, and business interruptions. You can also test upgrades to new versions or change actual code without affecting your online production server. Simply run the experimental server on the R2 copy of the database until the upgraded code runs with no errors. Then upgrade the production server.

In cases where an absolute realtime database is not essential, you can split the SRDF pair periodically and use the R2 copy for queries and report generation. Then you can re-establish the SRDF pair periodically to provide incremental updating of data on the R2 device. The ability to refresh the R2 device periodically allows you to provide the latest information for data processing and reporting.

Practical uses of *failover* and *failback* operations usually involve situations in which you need to switch business operations from the production site to a remote site. Once the switch occurs, normal operations continue using the remote (R2) copy of synchronized application data. Scheduled maintenance at the production site is one reason you might want to failover to the R2 site. Scheduled maintenance can include such tasks as operating system upgrades, host processor upgrades, and environmental disruptions.

Disaster recovery is another reason to temporarily failover to a remote site. The typical recovery routine involves customized software and complex procedures. Offsite media must be either electronically transmitted or physically shipped to the recovery site. Time-consuming restores and the application of logs usually follow. SRDF's failover/failback operations significantly reduce the restore time by incrementally updating only the specific tracks that have changed, accomplishing in minutes what might take hours for a complete load from a dumped database volume. Moreover, you can start the server and run it to its full production capability while the synchronization is still in progress.

Practical uses of the R1 *update* operation usually involve situations in which you want the R1 to become almost synchronized with the R2 before a failback, while the R2 side is still online to its host. You can include the `–until` *`track_threshold`* option to identify a number of invalid tracks that can build up from the active local I/O on the R2 side before retriggering another update operation. Note, however, that SYMCLI does not retrigger another update until the previous batch of updates is fully copied to the R1 side.

Concurrent RDF means having two target R2 devices configured as concurrent mirrors of one source R1 device. Using a concurrent SRDF pair like this allows you to easily generate two copies of the same data at remote locations. When you split the two R2 devices from their source R1 device, each target site's host can access its own data.

Swapping R1/R2 devices of an SRDF pair causes the source R1 device to become a target R2 device, and the former target device becomes the source device. Swapping SRDF devices allows the R2 side to take over operations while retaining a remote mirror on the R1 side. Swapping is especially useful after failing over an application from the R1 side to the R2 side. *Dynamic Swap* (a very fast version) is available with Enginuity version 5567 or higher.

Data Mobility replication is an SRDF configuration that restricts SRDF devices to operating only with Adaptive Copy replication. This is useful when you need to ensure that certain devices are used only for transferring large amounts of data in data mobility environments.

Parallel processing allows you to perform SRDF operations in a less time-consuming manner than performing the same operations sequentially, especially establish operations that need to mark and merge track tables between the R1 and R2 devices. If an application requires establishing multiple device groups across multiple Symmetrix units, doing so in parallel can significantly reduce the time required for the operation to complete. Beginning with the SRDF component of EMC Solutions Enabler version 5.0, if you want to perform SRDF parallel processing within a single Symmetrix unit, you can perform up to sixteen parallel control operations within a single Symmetrix unit across sixteen different RDF groups. (With EMC Solutions Enabler version 4.3, you can perform up to four parallel control operations within a single Symmetrix unit.) Beginning with Solutions Enabler version 5.2 running on Symmetrix units using Enginuity version 5669, parallel processing is controlled at the device level rather than at the RDF group level, allowing you to perform up to 64 parallel processing operations on different devices.

Dynamic SRDF allows you to create your own dynamic SRDF pairs from non-SRDF devices while the Symmetrix unit is in operation ("on the fly"). Historically, source and target SRDF device pairing has been static once set at configuration time, and this is still true of devices that are configured as non-dynamic SRDF devices. However, beginning with the SRDF component of EMC Solutions Enabler version 5.0 running on Symmetrix units using Enginuity version 5568, you now have the capability of using RDF-capable, non-SRDF devices in creating and synchronizing SRDF pairs. This feature provides greater flexibility in deciding where to copy protected data.

Beginning with Solutions Enabler version 5.2 running on Symmetrix units using Enginuity version 5669, you can create *dynamic RDF groups* in a Switched Fabric SRDF environment. An RDF group (RA group) represents a logical connection between two Symmetrix units. Historically, RDF groups were limited to those static RDF groups defined at configuration time. However, you can now create, modify, and delete RDF groups while the Symmetrix unit is in operation. This feature provides greater flexibility in forming SRDF-pair-associated links.

SRDF Asynchronous replication (SRDF/A) allows you to maintain a consistent point-in-time R2 image with only a slight lag behind the R1 side, which results in minimal data loss if there is a break in communication between the source and target. Beginning with Solutions Enabler version 5.3 running on Symmetrix units using Enginuity version 5670, SRDF/A allows you to configure an SRDF environment in which R1 devices transfer data to R2 devices in chunk cycles. One difference between Asynchronous replication and other methods of replication is that less data is transferred, and the data is handled fewer times. If the same data is updated multiple times in the same cycle (for example, the same track is written to ten times in a given time period), that data is sent across the RDF links only once and does not have to be copied within the cache for each update as in other ordered write solutions. This performance benefit can result in more efficient link utilization and potentially lower link bandwidth requirements when compared to a traditional ordered write solution that must handle each update write discretely. This saving in cycle size is most relevant for applications that can tolerate a loss of up to twice the size of a cycle if the RDF links are lost.

SRDF Configurations and Methods of Implementation

An SRDF configuration has at least one source unit and one target unit. SRDF configurations may transfer data in a uni-directional or bi-directional manner. In a uni-directional configuration, all R1 devices reside in the source Symmetrix unit and all R2 devices in the target Symmetrix unit. Under normal conditions, data flows from the R1 devices to the target R2 devices. In a bi-directional configuration shown in Figure 1, both R1 and target R2 devices reside in each Symmetrix unit. Data flows from the source (R1) devices in their respective Symmetrix unit to their corresponding target (R2) devices in the other Symmetrix unit.

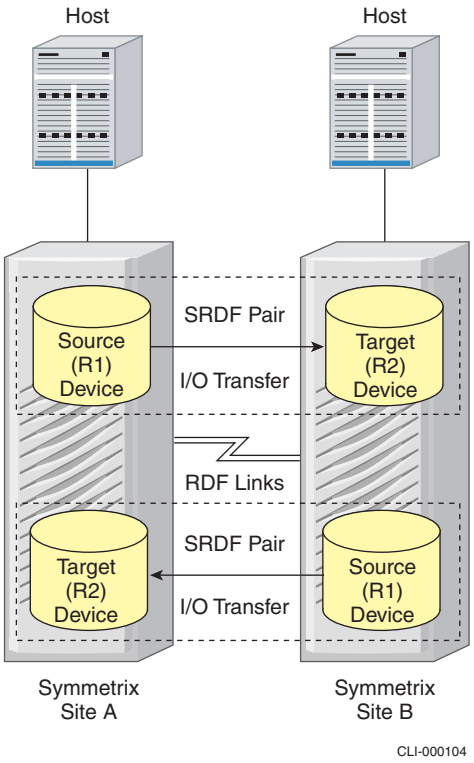

Figure 1. SRDF Bi-Directional Configuration

SRDF connectivity implementations encompass several solutions, depending on geographical requirements. The ESCON-based Campus Solution provides mirroring operations for Symmetrix units located up to 60 kilometers (37.5 miles) apart using fiber-optic links. SRDF over Fiber Channel topology supports high-speed synchronous mirroring among Symmetrix systems in campus situations.

With the Extended Distance Solution for sites farther removed geographically, SRDF FarPoint™ software supports mirroring across SRDF-supported telecommunications links, including T1/E1, T3/E3, and ATM. For businesses that have an intranet based on IP, SRDF over IP is another solution.

An SRDF topology can incorporate open network switching (*fabric*) in the RDF links. The switched RDF involves non-blocking switching devices that interconnect two or more nodes. Symmetrix units in a switched RDF topology can have each port pair running full-duplex.

During configuration, SRDF pairs are usually configured such that the R1 device and its paired R2 device are the same size. Although it is possible to reconfigure devices and migrate data from an R1 device to a larger size R2 device, your ability to access that data may require file system operations on the target-side host. For information about reconfiguring existing devices, refer to the white paper *Using the SYMCLI Configuration Manager* (P/N 300-000-475).

RDF Groups

The Remote Link Directors (RLDs), which manage the data transfers between Symmetrix units, have either an RA1 or RA2 designation. RA1s reside in the source Symmetrix, RA2s in the target Symmetrix. These RLDs have to be assigned to an RDF group as part of each Symmetrix configuration. An *RDF group* (also called an *RA group*) is a logical grouping of RDF devices that are serviced by the same set of RLDs.

RDF groups are related to physical RDF connections. Each link is logically associated with an RDF group at the time the Symmetrix is configured. An RDF group is configured and assigned an RDF group number by Enginuity. Typically, there are two or more RA/RF directors per RDF group.

The `symcfg list` command displays all Symmetrix units attached to your host and reachable via RDF links. Adding the `–RA all` option to the command displays how many RA/RF directors exist.

```
symcfg list
symcfg list –RA all
```

Setting Up Device Groups and Composite Groups

During configuration, SRDF devices are configured in pairing relationships and usually established so that the SRDF pair's mirroring activity becomes operational as soon as the RDF links are online. The Symmetrix global memory stores information about the pair state of operational SRDF devices.

Unlike the RDF group described in the previous section, the *device group* and the *composite group* are entities that you can create and use to manage and control SRDF pairs. Your host's SYMAPI[1] database file or the GNS-managed global repository stores information about a device group or composite group and the devices contained in the group.

Beginning with Solutions Enabler version 5.4, you can create a composite group to control a set of SRDF pairs and BCV pairs[2] that spans multiple Symmetrix units. When an RDF composite group is enabled for consistency protection, it is known as an RDF *consistency group*. A composite group provides greater flexibility than a device group, which can define devices only on a single Symmetrix unit. However, unlike the device group, the composite group cannot currently operate on specific pairs within the group but must perform an operation on the entire group. For more information about composite groups, refer to the section "Using a Composite Group to Control a Set of Devices That Spans Multiple Symmetrix Units."

An SRDF device group or composite group can hold one of two types of standard devices: RDF1 (source) or RDF2 (target). An SRDF device (R1 or R2) can be moved to another device group or composite group only if the source and destination groups are the same group type.

You can build an RDF1 type group on a host attached to a Symmetrix that contains those RDF1 devices. If a host is attached to a Symmetrix that contains RDF2 devices, you can build an RDF2 type group on that host. You can perform the same SRDF operations from any attached host that contains the group definition. For information about propagating group definitions globally, refer to the white paper *Using SYMCLI and GNS to Propagate Group Definitions to Multiple Hosts* (P/N 300-000-384).

When adding RDF standard devices to a device group or composite group, all devices in the device group:

- Must be SRDF devices
- Must be either all RDF1 or RDF2 type devices, as specified by the group type

[1] Symmetrix API

[2] In versions of Solutions Enabler prior to version 5.4, you can use a consistency group to control SRDF pairs only.

- Must belong to the same Symmetrix RDF group number

When adding TimeFinder™/Mirror RDF BCV devices to a device group or composite group:

- You can associate RDF BCV devices with any group type, including the Regular type
- You can choose RDF BCV devices that belong to a different RDF group number than the SRDF standard devices
- You cannot mix RDF1 and RDF2 type RDF BCV devices in the same group
- You can associate RDF BCV devices locally, remotely via the SRDF standard devices, or remotely via the locally associated RDF BCV devices

To check the configuration of SRDF devices before adding them to a device group or composite group, you can use the `symrdf list` command to list SRDF devices configured on Symmetrix units attached to your host:

```
symrdf list
```

When you add a device to a device group, a logical device name is assigned to it either by your specifying a logical name on the command line or by default.

The following sequence creates an RDF1 type device group and adds an R1 device to the group:

1. Create a device group named Rdf1Grp:

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

2. Add an R1 device (Symmetrix device name 085) to the device group on Symmetrix number 000000003264. A default logical name of the form DEV001 is assigned to the R1 device:

   ```
   symld -g Rdf1Grp –sid 3264 add dev 085
   ```

Invalid Tracks

The concept of *invalid tracks* in SRDF systems indicates what data is not synchronized between the two devices that form an SRDF pair. On both the source and target sides of an SRDF setup, the Symmetrix keeps an account of the tracks that are “owed” to the other side. The owed tracks are known as *remote invalids*.

For example, consider the case of an R1 device whose logical connection to its R2 has been suspended. If both devices are made write accessible, hosts on both sides of the RDF link can write to their respective devices, creating R2 invalid tracks on the R1 side and R1 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 link between the R1 and R2, the invalid tracks have to be resolved.

Resolution of invalid tracks depends on which operation you perform. For instance, you can have remote invalids on both sides prior to an establish or a restore operation. If so, performing an establish operation indicates to SRDF that you want to copy modified R1 tracks to the R2 side. In the process, any tracks that were modified on the R2 side are overwritten with data from corresponding tracks on the R1 side.

Performing a restore operation indicates the opposite — that you want to copy modified R2 tracks to the R1 side. In the process, any tracks that were modified on the R1 side are overwritten with data from corresponding tracks on the R2 side.

For more information on conditions governing composite operations, refer to *EMC Solutions Enabler Symmetrix SRDF CLI Product Guide*.

It is possible, though not common, to end up with local invalid tracks at the end of a series of singular RDF operations. When that happens, exercise care in determining which data to preserve and which to discard.

SRDF Control Operations

The `symrdf` command performs the high level control operations of the SRDF environment with two types of low level control operations: composite and singular operations. You perform most SRDF operations using composite operations. A composite operation is made up of several singular operations.

Table 1 lists the singular operations that make up each composite operation.

Table 1. Decomposition of Composite Operations into Singular Operations

Composite Operation	symrdf **options**	**Individual operations**	**When used**
Full establish	–full establish	- Write disable R2 devices on RA - Suspend RDF link traffic - Mark target device invalid - Merge track tables - Resume RDF link traffic	- Initial synchronization of RDF mirrors - Replacement of failed drive on the R2 side
Incremental establish	establish	- Write disable R2 devices on RA - Suspend RDF link traffic - Refresh tracks on target - Merge track tables - Resume RDF link traffic	Resynchronize RDF mirrors after they have been split and target data can be discarded
Split	split	- Suspend RDF link traffic - Read write enable R2 to its local host	When both sides need to be independently accessible (e.g., for testing)
Full restore	–full restore	- Write disable R1 to host - Write disable R2 devices on RA - Suspend RDF link traffic - Mark all source tracks invalid - Merge track tables - Resume RDF link traffic - Read write enable R1 to host	- Initial (reverse) synchronization of RDF mirrors - Replacement of failed drive on R1 side
Incremental restore	restore	- Write disable R1 to host - Write disable R2 devices on RA - Suspend RDF link traffic - Refresh source invalid tracks - Merge track tables - Resume RDF link traffic - Read write enable R1 to host	Resynchronize RDF mirrors after they have been split and the source data can be discarded
Failover	failover	- Write disable R1 to hosts - Suspend RDF link traffic - Read write enable R2 to hosts	In the event of a failure of the source site
Failback	failback	- Write disable R2 on RA - Suspend RDF link traffic - Refresh source invalid tracks (requires use of the `–force` option) - Merge track tables - Resume RDF link traffic - Write enable R1 to hosts	To return to the source site from the target site after the cause of failure has been remedied
Update	update	- Suspend RDF link traffic - Refresh source invalid tracks (requires use of the `–force` option) - Merge track tables - Resume RDF link traffic	To get the R1 side close to synchronized with the R2 side before a failback, while the R2 side is still online to the host

SRDF Pair States

The SRDF pair state encompasses the state of the R1 device, state of the R2 device, and the status of the RDF links between them. Before you can perform an SRDF control operation successfully, the SRDF pair must be in a state that is valid for that operation.

Table 2 lists the possible SRDF pair states that the `symrdf query` command can display. Not Ready means read/write disabled. Ready means enabled for both reads and writes. WD is write disabled.

Table 2. SRDF Pair States

SRDF Pair State	R1 State	RDF Links Status	R2 State	Invalid Tracks
SyncInProg	Ready	Ready	Not Ready or WD	>0
Synchronized	Ready	Ready	Not Ready or WD	0
Consistent	Ready	Ready	Not Ready or WD	0 (None owed to the R2)
Split	Ready	Not Ready or WD	Ready	—
Failed Over	Not Ready or WD	Not Ready	Ready	—
R1 Updated	Not Ready or WD	Ready or WD	Ready	0 (R1 tracks on source)
R1 UpdinProg	Not Ready or WD	Ready or WD	Ready	>0 (R1 tracks on source)
Suspended	Any status	Not Ready or WD	Not Ready or WD	—
Partitioned (from R1)	Any status	Not Ready	Not Available	—
Partitioned (from R2)	Not Available	Not Ready	Any status	—
Mixed	—	—	—	—
Invalid	Any status	Any status	Any status	—

The Partitioned state means that the Symmetrix API (SymmAPI™ or SYMAPI) is unable to communicate though the corresponding RDF paths to the remote Symmetrix unit. This state does not necessarily mean that two Symmetrix units have lost communication. For example, if one Symmetrix unit has two RA groups and SYMAPI is unable to communicate to a remote Symmetrix unit via one of those RA groups, only RDF devices belonging to that group are marked Partitioned; RDF devices belonging to the other RA group are not.

The Mixed state appears only with the `symdg show` display to indicate that there are different SRDF pair states in the device group.

The Invalid state is the default when no other SRDF pair states apply. In this case, the combination of R1, R2, and RDF link statuses do not match any other SRDF pair state. This state can also occur when something goes wrong on the device at the DA level (since `symrdf query` does not display DA status).

When you initiate an SRDF control operation, SYMAPI checks the state of each SRDF pair involved in the operation. If a pair is not in an SRDF pair state that is valid for that operation, the operation will fail unless you have included the `–force` option with the command. Using the `–force` option with an SRDF control operation forces an SRDF pair to a specified state. For example, the following command initiates a failover operation on all SRDF pairs that are currently in the Split state, which is not a typical state for failover:

```
symrdf –g Rdf1Grp –force failover
```

For more information about which SRDF control operations can use the `–force` option, refer to *EMC Solutions Enabler Symmetrix SRDF CLI Product Guide*.

Suspending the RDF Links of an SRDF Pair

The only singular control operations that you are likely to use on a routine basis are `symrdf suspend` and `symrdf resume`. The suspend operation suspends I/O traffic on the RDF links between one or more SRDF pairs in a device group. The resume operation re-establishes the RDF links for those SRDF pairs that were suspended.

Suspending I/O between the R1 and R2 devices is useful if one or more R1 devices cannot propagate data to its corresponding R2 device. By suspending all data propagation from the R1 devices, you ensure that all data flow to the R2 side is instantly and completely halted. By doing this, you ensure that a consistent database (up to the point in time of the data propagation failure) exists on the remote side of the configuration. This enables applications to still use the remote database.

While the RDF links between an SRDF pair are suspended, local I/O to the R1 devices can still occur. While these updates are not sent to the R2 devices immediately, the system does propagate these updates to the R2 side once you initiate a resumption of normal SRDF mirroring activity.

To initiate a suspend operation, an SRDF pair must be in one of the following states (use the `symrdf query` command to check the state of SRDF pairs in a device group):

- Synchronized
- R1 Updated
- SyncInProg and the `–symforce` option was specified[3]
- R1 UpdInProg and the `–symforce` option was specified[3]

You can initiate the suspend from either host. The following command initiates a suspend operation on all SRDF pairs in the device group named Rdf1Grp:

```
symrdf –g Rdf1Grp suspend
```

To resume the RDF links between an SRDF pair, the pair must be in the Suspended state. Assuming that all SRDF pairs in the device group are in the Suspended state, the following command resumes I/O traffic between all SRDF pairs in the device group.

```
symrdf –g Rdf1Grp resume
```

There are many reasons why a track table may need to be merged while an SRDF pair is in the Suspended state. If the `symrdf resume` command returns an error code of 21, the track tables of the R1 and R2 devices need to be merged before you can resume the RDF links. Commands that merge and resume are:

1. `symrdf –g Rdf1Grp merge`
   `symrdf –g Rdf1Grp resume`
2. `symrdf –g Rdf1Grp resume –force`
3. `symrdf –g Rdf1Grp establish`

For a device group with many SRDF pairs, SYMCLI uses these commands to determine which track tables need to be merged and acts only on those tables.

[3] To accept `–symforce` with an SRDF command, SYMCLI must recognize that you have enabled `-symforce` in the options file. It is recommended that you do not enable this option except for an emergency.

Establishing an SRDF Pair

Establishing an SRDF pair initiates remote mirroring — the copying of data from the source (R1) device to the target (R2) device. SRDF pairs come into existence in two different ways:

- At configuration time through the pairing of SRDF devices. This is a static pairing configuration.
- Any time during Symmetrix operation through your own pairing of dynamic, non-SRDF devices. This is a dynamic pairing configuration in which SRDF pairs can be created and deleted on the fly. See the section “Creating SRDF Pairs from Non-SRDF Devices” for a description of this feature.

A full establish (`symrdf establish –full`) is typically done when an SRDF pair is initially configured from SRDF devices and connected via the RDF links. Otherwise, you can perform an incremental establish, where the R1 device copies to the R2 device only the new data that was updated while the SRDF pair was split or suspended. To initiate an establish operation on all SRDF pairs in a device group or composite group, all pairs must be in the Split or Suspended state. Use the `symrdf query` command to check the state of SRDF pairs in a group.

Figure 2 illustrates establishing an SRDF pair. When you initiate the establish operation, the system write-disables the R2 device to its host and performs other tasks. When the establish operation is complete, the SRDF pair is in the Synchronized state. The R1 device and R2 device contain identical data.

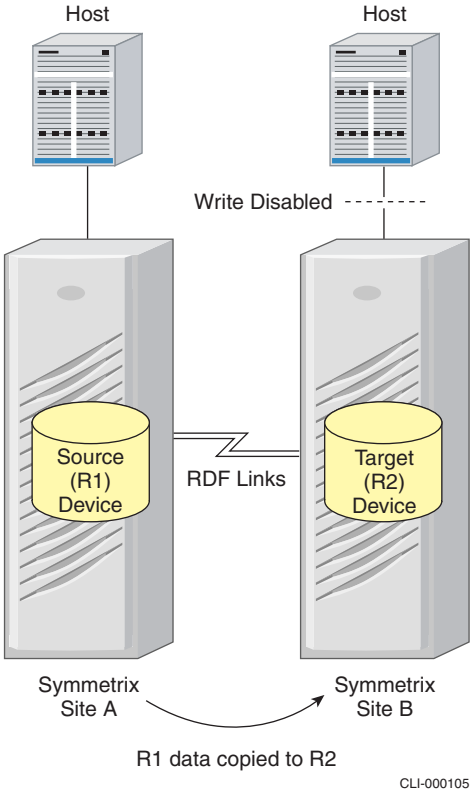

Figure 2. Establishing an SRDF Pair

You can initiate the establish operation from either host, provided that a host has the appropriate device group or composite group definition. The following command initiates an incremental establish operation for all SRDF pairs in the device group named Rdf1Grp:

```
symrdf –g Rdf1Grp establish
```

Splitting an SRDF Pair

When you need to have read/write access to a target (R2) device, you can split the SRDF pair. When the split completes, the target host can access the R2 device. The R2 device contains valid data and is available for business continuance tasks or restoring data to the R1 device if there is a loss of data on that device.

While an SRDF pair is in the Split state, local I/O to the R1 device can still occur. While these updates are not sent to the R2 device immediately, the system does propagate these updates to the R2 side once you initiate a resumption of normal SRDF mirroring activity.

To initiate a split, an SRDF pair must already be in one of the following states:

- Synchronized
- Suspended
- R1 Updated
- SyncInProg and the `–symforce` option was specified[4]

You can initiate the split operation from either host. The following command initiates a split operation on all SRDF pairs in the device group named Rdf1Grp:

```
symrdf –g Rdf1Grp split
```

The `symrdf split` command provides exactly the same functionality as the `symrdf suspend` and `rw enabled R2` commands together. Moreover, the split operation and the suspend operation have exactly the same consistency characteristics for device groups (which use a single RA group for a single Symmetrix unit) and RDF composite groups (which can have multiple RA groups that can span multiple Symmetrix units).

When SRDF pairs are in a device group on a single Symmetrix unit, you can split the SRDF pairs as shown above and have copies on the R2 devices. If the R2 devices need to be consistent (that is, a restartable copy on the R2 side), include the SRDF pairs in a composite group and enable the group for consistency protection.

When a set of SRDF pairs spans multiple Symmetrix units, you can include the SRDF pairs in a composite group and split the group. If consistency is required, enable the composite group for consistency protection. For information on RDF composite groups and consistency, refer to the white paper *Using SYMCLI to Implement RDF Consistency Protection with SRDF Family Products* (P/N 300-000-284).

Note: It is possible after a split operation that one or more R1 devices may not be mapped to an SA. If so, and if you do not intend to restore R2 data changes to the R1 devices, you should perform an establish operation on these unmapped R1 devices and *not* a failback operation.

[4] To accept `–symforce` with an SRDF command, SYMCLI must recognize that you have enabled `-symforce` in the options file. It is recommended that you do not enable this option except for an emergency.

Restoring an SRDF Pair

When you need to copy data from the target (R2) device back to the source (R1) device, you can restore the SRDF pair. After an SRDF pair is split, the R2 device contains valid data and is available for business continuance tasks (such as running a new application) or restoring data to the R1 device if there is a loss of data on that device. Moreover, if the results of running a new application on the R2 device are good, you may want to move the changed data and new application to the R1 device.

You can perform a *full restore* or *incremental restore*. A full restore operation copies the entire contents of the R2 device to the R1 device. An incremental restore operation is much quicker because it copies only new data that was updated on the R2 device while the SRDF pair was split. If any tracks on the R1 device changed while the SRDF pair was split, those tracks are refreshed from the corresponding tracks on the R2 device.

To initiate a restore, an SRDF pair must already be in the Split state.

You can initiate the restore operation from either host. The following command initiates an incremental restore operation on all SRDF pairs in the device group named Rdf1Grp (add the `–full` option if you need a full restore):

```
symrdf –g Rdf1Grp restore
```

The restore operation is complete when the R1 and R2 devices contain identical data. The SRDF pair is then in a Synchronized state.

Failover and Failback

Having a synchronized SRDF pair allows you to switch data processing operations from the source site to the target site if operations at the source site are disrupted or if you need to schedule downtime for maintenance. This switchover from source to target is called *failover*. When operations at the source site are back to normal, you can use a *failback* operation to re-establish I/O communications links between source and target, resynchronize the data between the sites, and resume normal operation.

In a period of scheduled downtime for maintenance, or after a serious system problem that renders either the source (R1) host or Symmetrix unreachable, no read/write operations can occur on the R1 device. In this case, you can initiate a failover operation to make the paired R2 device read/write enabled to its host.

You can initiate the failover operation from either host. However, if the R1 host is down, the only alternative is to initiate failover from the R2 host. The following command initiates a failover on all SRDF pairs in the Rdf1Grp device group:

```
symrdf –g Rdf1Grp failover
```

To initiate a failover, the SRDF pair must already be in one of the following states:

- Synchronized
- Suspended
- R1 Updated
- Partitioned (when you are invoking this operation from the target side)

Figure 3 illustrates the failover operation. If the R1 device is operational, the RDF links are suspended. If the source side is operational, the R1 device is write disabled to its host. The R2 device is then read/write enabled to its host.

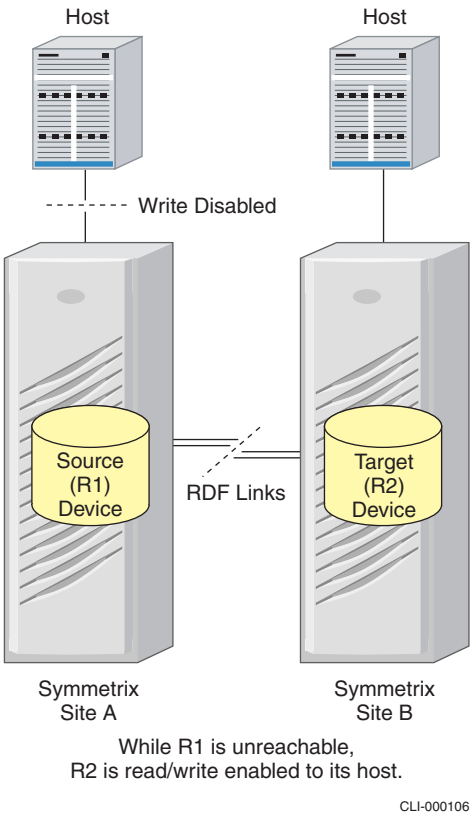

Figure 3. Failover from the Source System to the Target System

Failback

When you are ready to resume normal SRDF operations, you can initiate a failback (R1 device takeover). This means starting read/write operations on the R1 device, and stopping read/write operations on the R2 device. The R2 becomes read-only to its host, while the R1 becomes read/write enabled to its host.

```
symrdf –g Rdf1Grp failback
```

To initiate a failback, the SRDF pair must already be in one of the following states:

- Failed Over
- Suspended and Write Disabled at the source
- Suspended and Not Ready at the source
- R1 Updated
- R1 UpdInProg

Failover with a Fast Swap and Establish

Beginning with Solutions Enabler version 5.4 and Enginuity version 5567, SRDF provides a failover option for *dynamic RDF devices* that allows you to failover, swap R1/R2 personalities, and establish the dynamic R1 and R2 devices. This operation is useful if you want to transfer data processing to the remote site but continue to replicate this data at the local site. For example:

```
symrdf –g Rdf1Grp failover -establish
```

If the device pairs have the same data, the "fast swap" feature performs the swap and makes the R1/R2 devices read/write (RW) on the link without having to perform the time-consuming merge operation.

If you want to return to the original setup, perform the `failover –establish` operation again from the local host (or from the remote host if device group Rdf1Grp is also defined on the remote host).

Updating the R1 Device

You usually perform the *R1 update* action after a failover, when data processing on the target side has created large numbers of changed tracks on the target devices since the point of the failover. An update operation can bring the R1 device closer to synchronization with the R2 before a failback is initiated. This close-to-synchronized state helps to minimize downtime during the failback process.

You can use the `symrdf update` command with or without the `–until` option. If you omit the `-until` option, the system takes a snapshot of whatever tracks have changed on the R2 side (R1 invalid tracks). These tracks are marked to be copied over to the R1 side. If there are new changes to the tracks on the R2 side while the marked tracks are being copied, those changes accumulate as R1 invalid tracks on the R2 side but are not marked for copying. Once the original set of invalid tracks has been copied, the update operation stops.

If you use the `–until` *`track_threshold`* option, the system examines the number of R1 invalid tracks on the R2 side once the initial set of tracks has been copied. If the R1 invalid tracks on the R2 side are *under* the threshold, the update command exits with a successful completion. If the R1 invalid tracks on the R2 side are *equal to or greater than* the threshold, the update process begins again with a fresh snapshot. This process repeats until the R1 invalid tracks on the R2 side are under the threshold when the R1 update completes.

To initiate an R1 update, the SRDF pair must already be in one of the following states:

- R1 Updated
- Failed Over
- Suspended and Write Disabled at the source
- Suspended and Not Ready at the source

Caution: If you perform an update while the SRDF pair is in the Suspended/Not Ready state, the SRDF pair enters an Invalid state as the update completes. To put the SRDF pair in a Synchronized state, you can perform the `symrdf rw_enable r1` control operation to write-enable the R1 device to its host.

The following command initiates an update on all R1 devices in the Rdf1Grp device group:

```
symrdf –g Rdf1Grp update
```

Using `symrdf update –until #` identifies the number of R1 invalid tracks that can accumulate on the R2 side before another update (R2-to-R1 copy) is re-triggered. For example, to update the R1 device until the number of R1 invalid tracks on the R2 side is less than 1000, enter the following command:

```
symrdf –g Rdf1Grp update –until 1000
```

An update sequence begins with an immediate update once the command is initiated. With this operation, a new R1 update cycle will occur every time the previous batch of invalid tracks that was updated has been fully copied to the R1 side and the count of R1 invalid tracks that have accumulated in the interim is equal to or greater than 1000. When the R1 invalid track count at the end of an update cycle is under 1000, no more update cycles are performed. The R1 update is complete.

Using Concurrent RDF Devices

You can have two identical remote copies available at any point in time by having two target R2 devices configured as concurrent remote mirrors of one source R1 device. Using a concurrent SRDF pair like this allows you to easily generate two copies of the same data at remote locations. When you split the two R2 devices from their source R1 device, each target site's host can access its own data at the time of the split.

Concurrent RDF requires two different RDF (RA) groups to achieve the connection between the local R1 device and its two remote R2 mirrors. It is recommended that the RDF groups be on two different RA adapter interfaces, but this is not necessary if you are using Fibre RA adapters. As shown in Figure 4, each RDF (RA) group represents an established link between two Symmetrix units.

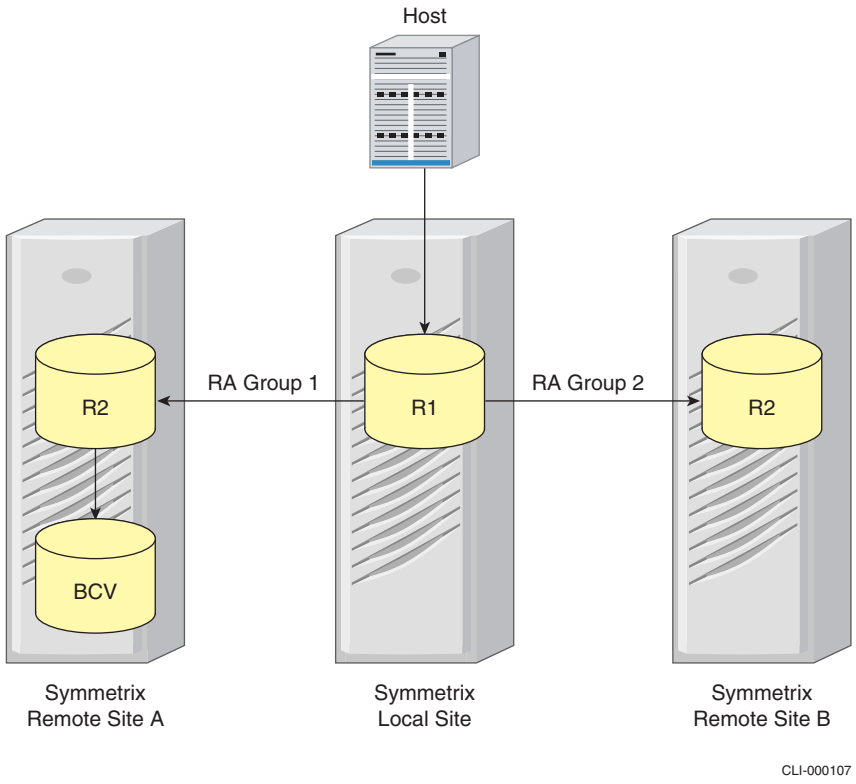

Figure 4. Using RDF Links to Copy the Same Data Concurrently to Two Different Remote Sites

Each of the two remote mirrors can operate with the same method of replication (Synchronous, Semi-Synchronous, Asynchronous, or Adaptive Copy) or in different methods with one exception: concurrent RDF cannot have one remote mirror with Synchronous replication and the other with Semi-Synchronous replication.

When using concurrent RDF, you can build a device group containing concurrent RDF standard devices that belong only to the two RDF groups representing the concurrent links. Your device group can also include BCVs and RDF standard devices that are not concurrent RDF devices. However, within the context of the device group, you can remotely associate a BCV with only one of the concurrent R2 mirrors (as the illustration shows), but not with both[5].

[5] If you disassociate the BCV at Site A from the device group, you can then remotely associate a BCV from Site B and create a BCV pair with the concurrent R2 mirror there. However, the BCV pair at Site A is no longer under the control of the device group, even though that BCV pair remains synchronized if the pair was in this state when disassociated from the device group.

To determine which devices on a particular Symmetrix system have been configured as concurrent RDF devices, use `symrdf list` with the `–concurrent` option. For example, on Symmetrix 000000003265:

```
symrdf list –sid 3265 –concurrent
```

You can establish a concurrent SRDF pair simultaneously with one command. For example, to incrementally establish the concurrent SRDF pair:

```
symrdf -g Rdf1Grp establish –RDFG ALL
```

The `–RDFG ALL` option tells SYMCLI to establish the R1-to-R2 links for both RDF groups at the same time. If you want to establish one mirror of the SRDF pair first and then the other, use two establish operations:

```
symrdf -g Rdf1Grp establish –RDFG 1
symrdf -g Rdf1Grp establish –RDFG 2
```

The example assumes that one link of the concurrent SRDF pair is represented by RDF group 1, and the other by RDF group 2.

You can use the `symrdf verify` command to verify the state of one or both mirrors of the concurrent SRDF pair. For example, to verify the Synchronized state of both concurrent mirrors:

```
symrdf -g Rdf1Grp verify –RDFG ALL
```

You can also split the concurrent SRDF pair either simultaneously (`-RDFG ALL`) or one at a time (`-RDFG #`). The following command splits the concurrent mirror that is represented by RDF group 2:

```
symrdf -g Rdf1Grp split –RDFG 2
```

The following `symrdf query` command displays the status of both mirrors of the concurrent SRDF pair:

```
symrdf -g Rdf1Grp query –RDFG ALL
```

If you split only one of the concurrent mirrors, the link for the split mirror will go to the Not Ready status, and the link for the still-synchronized mirror will remain in the Ready status.

If you want to restore data from the target (R2) devices to the source (R1) device, only one of the concurrent R2 mirrors does the restoring at any given time. (This rule applies to failback and R1 update operations also.) For example, assuming that both concurrent R2 mirrors are split from their R1 standard device, the following command restores the R1 standard device from the R2 mirror represented by RDF group 2:

```
symrdf -g Rdf1Grp restore –RDFG 2
```

After the restore operation, the R2 mirror associated with RDF group 2 is synchronized with the source (R1) device, and the R2 device associated with RDF group 1 is still in the Split state. If you want the split R2 device to mirror the standard device again, you can simply re-establish them explicitly.

However, if you have written new data to the split R2 device in the interim and you want the split mirror's data to become the resychronized data, you can *restore* again from the split mirror. In this case, however, you need to include the `–remote` option on the command line. For example:

```
symrdf -g Rdf1Grp restore –RDFG 1 -remote
```

This operation copies data from the split R2 device to the source (R1) device, and from the R1 device to the R2 mirror that was previously used to restore the R1 device.

For all restore, R1 update, and failback operations where one of the concurrent mirrors is synchronized with its R1 device, using the `–remote` option tells SYMCLI that you intend to copy data from the unsynchronized concurrent mirror to both the R1 device and the other (synchronized) concurrent R2 mirror. If you inadvertently omit the `–remote` option in this case, SYMCLI returns an error message.

Swapping R1 Devices with R2 Devices

In a device group with SRDF pairs, you can swap the R1/R2 personality of all standard RDF or BCV RDF pairs in the device group. Source R1 devices become target R2 devices, and vice versa. Swapping SRDF devices allows the R2 side to take over operations while retaining a remote mirror on the R1 side. Swapping is especially useful after failing over an application from the R1 side to the R2 side.

Before you can perform an R1/R2 swap, all SRDF pairs in the device group must be in one of the following states: Failed Over, R1 Updated, or Suspended. Table 3 illustrates pre- and post-swap states.

Table 3. Pre- and Post-Swap States

SRDF Pair State before the Swap	SRDF Pair State after the Swap	State of New R1 after the Swap	State of New R2 after the Swap	Operation Needed to Resume Mirroring[6]
Failed Over	Suspended	Read/Write Enabled	Write Disabled	Incremental Establish; or Resume
R1 Updated	Suspended	Read/Write Enabled	Write Disabled	Incremental Establish; or Resume
Suspended with R1 Write Disabled	Suspended	Write Disabled	Write Disabled	Incremental Establish; or Resume and rw_enable R1

After the swap operation is complete, an SRDF pair is in the Suspended state. If the swap followed a failover or R1 update operation, I/O to the new R1 device is not interrupted; the new R1's state is now that of the pre-swap R2 (read/write enabled). If the swap follows a Suspend operation, the state of the new R1 device after the swap is write disabled. I/O to the new R1 cannot begin until you perform an incremental establish operation.

After the swap, the new R2 device is write disabled, and it can begin functioning as the remote mirror. To begin propagating data from the new R1 to the new R2, perform a `symrdf establish` operation.

The `symrdf swap` command can swap all the SRDF devices in the device group, both RDF standard devices and RDF BCVs, as long as the RDF BCV devices belong to the same RDF group as the RDF standard devices. For example, after a failover operation:

```
symrdf –g Rdf1Grp swap –all
```

Omitting the `–all` option swaps just the standard devices, and substituting `–bcv` for `–all` swaps just the RDF BCV devices.

The `symrdf establish` command re-synchronizes the SRDF pairs incrementally and resumes mirroring between them, albeit in the opposite direction from before the swap operation:

```
symrdf –g Rdf1Grp establish
```

If you use the `–refresh R1` option with the swap operation, SYMCLI marks any modified tracks on the pre-swap R1 device to refresh from data on the R2 device. The `–refresh R2` option does the opposite, but using this option is possible only when the SRDF pair is in the Suspended state prior to the swap.

[6] The use of `symrdf resume` after an R1/R2 swap requires that you either include the `–force` option or that you issue the `symrdf merge` command prior to the resume operation. SYMCLI uses the `–force` option to initiate the merge operation if Enginuity requests it, and then the resume operation is performed.

The following command swaps the SRDF standard devices in the device group and marks any modified tracks on the pre-swap R1 device to refresh from data on the R2 device:

```
symrdf –g Rdf1Grp swap –refresh R1 –i 30 –c 10
```

To execute the swap operation successfully, SYMCLI needs to acquire an exclusive lock on the Symmetrix host database, the local Symmetrix, and the remote Symmetrix systems. If some other operation currently holds this exclusive lock, the previous command tells SYMCLI to retry the swap operation every 30 seconds until the operation succeeds or until it has tried 10 times without success.

With Enginuity version 5568 and the introduction of *dynamic* SRDF pairs, the *static* SRDF pairs can no longer be configured for dynamic RDF swap. With version 5568 and higher, the only devices capable of dynamic RDF swap are *dynamic* SRDF pairs.[7] You can create dynamic SRDF pairs using the `symrdf createpair` command (see the section "Creating SRDF Pairs from Non-SRDF Devices").

Dynamic RDF swap is not supported for *concurrent* dynamic RDF.

Data Mobility Replication

Data Mobility replication is an SRDF configuration that restricts SRDF devices to operating only in Adaptive Copy replication. There are two types of Adaptive Copy replication:

- Adaptive Copy Disk (AD) replication (acp_disk)
- Adaptive Copy Write-Pending replication (acp_wp)

Adaptive Copy replication facilitates data sharing and migration. These methods of replication allow the source and target devices to be more than one I/O out of synchronization. Both methods allow write tasks to accumulate on the local side before being copied to the remote side.

With Adaptive Copy Disk replication, write tasks are stored as invalid tracks on the source device of the SRDF pair. With Adaptive Copy Write-Pending replication, write tasks accumulate in a local cache. A background process moves (or destages) the write-pending tasks to the source device and its corresponding target device. The advantage of this method is that it is typically faster to read data from cache than from disk. The disadvantage is that cache is temporarily consumed by the data until it moves to disk.

At configuration time, EMC can configure SRDF devices for Adaptive Copy replication. By also configuring these devices for Data Mobility replication, EMC applies the restriction. If you attempt to take a restricted device out of Adaptive Copy replication, SYMCLI returns a message to the effect that switching out of Adaptive Copy replication is not possible with data mobility enabled.

To determine if your Symmetrix system has data mobility enabled, you can use the `symcfg list –v` command as shown in the "Examples" section of this paper.

With data mobility enabled, SYMCLI does not require you to use the `–force` option with a split or failover operation as you would normally need to do when data mobility is not enabled.

[7] Devices intended for dynamic RDF swap must be configured with the `dyn_rdf` attribute, which makes a device capable of being both a dynamic R1 device and a dynamic R2 device (refer to the white paper *Using the SYMCLI Configuration Manager*, P/N 300-000-475).

SRDF Asynchronous (SRDF/A) Replication

If a Symmetrix unit is configured to operate with SRDF/A replication[8], you can create a device group containing R1 devices that copy data to R2 devices using an all-or-nothing paradigm similar to composite groups. Beginning with Solutions Enabler version 5.3 and Enginuity version 5670, you can use an *SRDF/A device group* (one containing devices capable of operating with SRDF/A replication) to transfer R1 source data in cycled chunks to the target R2 system. When a complete chunk of data is received on the R2 side, a copy cycle is committed. If the RDF links are lost during data transfer, any partial chunk is discarded, preserving consistency on the R2 with data that is two cycles or less behind the R1.

You can set up and enable SRDF/A such that SRDF devices of an SRDF/A device group act in unison to maintain the integrity of a database being remotely mirrored on a target Symmetrix unit. If a source R1 device in the device group cannot propagate data to its corresponding target R2 device, SRDF/A suspends data propagation from all R1 devices in the device group, halting all data flow to the R2 targets. This suspension of all data propagation, called *tripping the device group*, ensures a consistent R2 copy of the database up to the point in time that the group was tripped. Tripping an SRDF/A device group can occur either automatically or manually.

An automatic trip occurs when one or more R1 source devices in an enabled SRDF/A device group cannot propagate data to their corresponding R2 target devices. For example:

- All RDF links between the R1 and R2 might go down for an extended period of time.
- The RDF directors on the R1 side or R2 side might fail.

A manual trip occurs when you invoke the `symrdf suspend, split,` or `failover` commands for the SRDF/A device group. Tripping the device group creates a consistent point-in-time R2 image from the N-2 cycle (the cycle that is two cycles behind the R1 side). The current minimum cycle time is 30 seconds.

There are two choices with a manual trip: the default or the `–immediate` option. By default, the SRDF/A session is dropped at the end of the current "in-process" cycle, which may cause execution time of this command to be longer but results in no invalid tracks on the R2 side. By dropping on a cycle boundary, there is no need to resolve invalid tracks when SRDF/A is resumed.

Dropping the SRDF/A session immediately may result in remote invalid tracks on both the R1 and R2 sides. The SRDF/A devices go NR on the link, and write-pending tracks are converted to invalid tracks on both the R1 and R2 sides. However, dropping a session immediately does not compromise the consistency of the data on the R2 side; SRDF/A always provides a consistent image of the data at the remote site at all times. It is only during a resynchronization operation that data consistency is not guaranteed. (It is recommended that you use BCV devices to preserve a copy of the R2 devices prior to resynchronization.)

The following command drops the SRDF/A session for the device group AsyncGrp. The `–force` option is required here to ensure that you want to stop SRDF/A operation and end consistency protection.

```
symrdf –g AsyncGrp suspend -force
```

[8] SRDF/A replication can be enabled on only one RA (RDF) group in the Symmetrix unit, either using the Symmetrix configuration server or the Configuration Manager (refer to the white paper *Using the SYMCLI Configuration Manager*, P/N 300-000-475). All R1 or R2 devices belonging to the RA group enabled for SRDF/A replication become devices capable of participating in SRDF/A operations.

SRDF/A session status can be one of the following:

- Active with consistency protection enabled

  The R2 side can be either consistent or inconsistent. If there are any invalid tracks to be copied, the R2 side is not consistent until the N-2 cycle containing the last invalid track is fully on the R2 side.

- Active with consistency protection disabled

  The SRDF/A devices are ready or NR on the RDF link and running with SRDF/A replication.

- Inactive

  The SRDF/A devices are ready on the RDF link and are working with their basic methods of replication (Synchronous, Semi-Synchronous, or Adaptive Copy).

To ensure consistency protection, Asynchronous replication must be set and consistency protection enabled on the device group containing the SRDF/A devices. For example:

```
symrdf –g AsyncGrp set mode async
symrdf –g AsyncGrp enable
```

Disabling the SRDF/A-backed device group causes consistency protection to be disabled. For example:

```
symrdf –g AsyncGrp disable
```

SRDF/A devices remain ready on the RDF link and operating with their last primary method of replication. Data continues to flow.

For restrictions regarding the use of SRDF/A replication, refer to *EMC Solutions Enabler Symmetrix SRDF CLI Product Guide* and to the product release notes.

Using BCVs to Preserve a Copy of the Remote SRDF/A Data

Although SRDF/A replication does not require TimeFinder/Mirror software during normal operation, you may find it useful to mirror the R2 data on TimeFinder BCVs at the remote site to preserve a consistent image of the data on the R2 devices before resynchronization operations. SRDF/A allows you to perform a consistent split on R2-side BCVs without dropping the RDF links, allowing you to preserve point-in-time copies of the R2 data without disrupting the SRDF/A operation between the R1 and R2 devices.

You can control the R2-side BCVs through the RDF1 type device group defined on the local host or an RDF2 type device group defined on the remote host. Controlling remotely-associated BCVs from the R1 side requires using the `–rdf` option. Controlling from the R2 side omits the option.

From the R1-side host, the `symmir split –rdf` command with the `–consistent` option splits the R2-side BCV pairs, making the BCV data available to the R2-side host. For example:

```
symmir -g AsyncGrp split –rdf -consistent
```

For more information on consistent split, refer to the white paper *Using SYMCLI to Perform Consistent Splits with TimeFinder Family Products* (P/N 300-000-283).

Setting Up SRDF Asynchronous Operation

The following steps outline the procedure for performing SRDF control operations with SRDF Asynchronous (SRDF/A) replication (for outputs to these commands, refer to Example 5):

1. List SRDF/A devices on the source Symmetrix unit. The RDF devices displayed belong to the RDF (RA) group that has been configured for SRDF/A operation. The devices displayed will be all R1 devices.

   ```
   symrdf list -rdfa
   ```

2. Create an RDF1 type device group. For example an group named AsyncGrp1:

   ```
   symdg create AsyncGrp1 –type rdf1
   ```

3. Add to the device group all devices from the RDF (RA) group configured for SRDF/A operation. For example, if the RDF group displayed in the `symrdf list` display is group number 1, then all devices in this RDF group must be managed for SRDF/A operation.

   ```
   symld –g AsyncGrp1 addall –rdfg 1
   ```

4. Query the device group to display the R1-to-R2 setup and the state of the SRDF/A device pairs:

   ```
   symrdf –g AsyncGrp1 query -rdfa
   ```

5. Set the device group to Asynchronous replication:

   ```
   symrdf –g AsyncGrp1 set mode async
   ```

6. If the SRDF pairs are *not* in the Consistent state at this time (for example, the Split or Suspended state with invalid tracks on the R1 side), you can initiate SRDF copying of R1 data to the R2 side. The device state will be SyncInProg until the Consistent state is reached.

   ```
   symrdf -g AsyncGrp1 establish
   ```

7. Enable consistency protection for the SRDF/A devices in the device group:

   ```
   symrdf –g AsyncGrp1 enable
   ```

8. If you need to manually trip the device group, you can suspend the RDF links for the device group or split the SRDF pairs. By default, the SRDF/A session is dropped at the next switch in copy cycles. The SRDF/A session becomes inactive. The `–force` option is required to ensure that you actually want to stop SRDF/A operation and end consistency protection.

   ```
   symrdf –g AsyncGrp1 suspend -force
   ```

9. If there are writes to the R1 source devices while the SRDF/A session is inactive, invalid R1 tracks will accumulate on the R1 side. These are tracks that are owed to the R2 side. If the devices are made ready on the link, you can resume SRDF operation, copying these tracks to the R2 side. The SRDF/A session is automatically activated.

   ```
   symrdf –g AsyncGrp1 resume
   ```

10. At this point, the SRDF/A devices are ready on the RDF link and operating with Asynchronous replication. However, consistency protection is not automatically enabled upon resumption of the link. You must explicitly enable consistency protection again.

    ```
    symrdf –g AsyncGrp1 enable
    ```

Performing SRDF Control Operations in Parallel

Some SRDF operations require a considerable amount of time to complete, especially establish operations that need to mark and merge track tables between the R1 and R2 devices. If an application such as batch replication to copy data warehouse loads requires establishing device groups across multiple Symmetrix units, doing so sequentially can take a considerable amount of time.

SYMCLI provides a configuration database on your host for SYMAPI access, which by default is set to EXCLUSIVE access to prevent the database from being changed while an operation is in progress. Under this access setting, one establish operation must complete before another begins.

However, if you know that there are no Symmetrix configuration changes currently being made to your host's SYMAPI database, you can set access to the database to PARALLEL. With parallel access set, SYMCLI can process multiple SRDF operations such as `symrdf establish` simultaneously. For example, if you need to establish SRDF pairs on three Symmetrix units, you can build a device group for each Symmetrix unit, set the `SYMCLI_CTL_ACCESS` environment variable to `PARALLEL`, and issue three `symrdf establish` commands simultaneously:

```
Export SYMCLI_CTL_ACCESS=PARALLEL
symrdf –g group1 establish –noprompt &
symrdf –g group2 establish –noprompt &
symrdf –g group3 establish –noprompt &
wait
```

Beginning with the SRDF component of Solutions Enabler version 5.2 running on Symmetrix units using Enginuity version 5667, parallel processing is controlled at the device level rather than at the RDF group level. You can perform up to 64 parallel processing operations on different devices but no more than one operation on any one device at a time. Beginning with Enginuity version 5669, you can perform *unlimited* parallel processing operations on different devices. Moreover, these operations can span RDF groups.

However, if your Symmetrix unit is still running Solutions Enabler version 5.1 (or lower) with Enginuity version 5568 (or lower) and you want to perform SRDF parallel processing for different RDF groups within a *single* Symmetrix unit, you need to set the following parameter in the SYMAPI options file:

```
SYMAPI_PARALLEL_RA_GROUPS = ENABLE
```

The options file in the SYMAPI configuration directory contains behavior parameters that you can set to change the default behavior of SYMCLI operations. The default for `PARALLEL_RA_GROUPS` is `DISABLE`. In the example above, if the three device groups contain devices from three different RDF groups, then these devices can belong to the same Symmetrix unit and can be established using the same parallel processing syntax.

Creating Dynamic SRDF Pairs

When a Symmetrix unit is configured, some devices are usually configured as SRDF devices (an R1 paired with an R2), and others as non-SRDF devices. Dynamic RDF technology allows you to create additional SRDF pairs from non-SRDF devices that have been configured as RDF-capable devices, provided that the Symmetrix unit itself has been enabled for dynamic RDF mode of operation[9]. You synchronize and control dynamic SRDF pairs the same way that you manage configured SRDF pairs.

Prior to Enginuity version 5568 (as described in Appendix A), source and target SRDF device pairing was limited to those static SRDF pairs set at configuration time. Dynamic RDF provides you with the flexibility to create and delete new SRDF pairs while the Symmetrix unit is in operation.

You can use the `symdev list –dynamic` command to display non-SRDF devices that have been configured as dynamic volumes (see "Example 3: Creating Dynamic SRDF Devices"). Non-SRDF devices can be configured with the capability to be R1 devices, R2 devices, or both. Once you determine which dynamic devices on the source Symmetrix unit you want to pair with which dynamic devices on the target Symmetrix unit, you need to create a device file and list your device pairs in the file. For example, a list of device pairs in a device file called pairsfile:

```
09C  054
09D  055
09E  056
```

Each SRDF pair must be on a separate line in the file (for example, device 09C paired with device 054).

When issuing the `symrdf createpair` command for this file, specify the device type of first-column devices (R1 or R2 type) and the Symmetrix unit on which the first-column devices reside. Specify also the name of the device file and the RDF (RA) group on the Symmetrix unit that you wish to use as the RDF link between the R1 and R2 devices.

If your initial operation is to establish the pairs, include the `–establish` option or `–invalidate R2` option. The `–establish` option invalidates the R2 devices, merges the track tables for the pair from both devices, brings up the RDF links, and initiates data copying from the R1 to the R2 devices. If your initial operation is a restore, include the `–restore` option or `–invalidate R1` option. Including the `–restore` option invalidates the R1 devices, merges the track tables for the pair from both devices, brings up the RDF links, and initiates data copying from the R2 to the R1 devices. The `–invalidate` option allows the creation of dynamic SRDF pairs without bringing up the RDF links and initiating the copying of data.

```
symrdf createpair –file pairsfile –sid 77 –rdfg 2 –type rdf1 –invalidate r2
```

The example executes a file called pairsfile and uses the `–type rdf1` option to identify the first-column devices as R1 type devices residing on source Symmetrix 000185400077 (`–sid 77`). Therefore, the second-column devices are R2 devices that reside on the target Symmetrix unit. The SRDF pairs can communicate using RDF group number 2 (`–rdfg 2`) from source Symmetrix 000185400077. This pairing information is added to your host's SYMAPI database file. The `–invalidate R2` option invalidates the R2 devices in preparation for a subsequent establish operation.

Performing SRDF operations on members of the device file allows you to synchronize new SRDF pairs in the file and query or verify (or both) the progress of the establish or restore operation. For example:

```
symrdf –f pairsfile establish –sid 77 –rdfg 2
symrdf –f pairsfile query –sid 77
```

[9] The `symcfg list –v` command displays Symmetrix characteristics, including whether "Dynamic RDF Configuration State" is set to "enabled."

To discontinue using these dynamic SRDF pairs, your pairs need to be in a state in which the RDF link state is Not Ready (NR): Suspended, Split, or Failed Over. For example, to suspend the RDF links using a `symrdf suspend` command, and then perform a `symrdf deletepair` command:

```
symrdf –f pairsfile suspend –sid 77 –rdfg 2
symrdf deletepair –file pairsfile –sid 77 –rdfg 2
```

The `symrdf deletepair` command cancels the dynamic SRDF pairings by removing the pairing information from your host's SYMAPI database file.

Beginning with Solutions Enabler version 5.4, if concurrent dynamic RDF is enabled on the Symmetrix unit, you can create *concurrent* dynamic SRDF pairs by creating a second pairs file that lists the same R1 devices with a different set of R2 devices. For example, a second pairs file called conpairs:

```
09C  030
09D  031
09E  032
```

This new set of R2's can be either on the same remote Symmetrix unit as the initial set of R2 devices or on a second remote Symmetrix unit, but you must choose a different RDF group than the one used to connect the initial set of R2 devices. Once you have defined the second pairs file (conpairs), create the concurrent pairs:

```
symrdf createpair –file conpairs –sid 77 –rdfg 3 –type rdf1 –invalidate r2
```

The concurrent SRDF pairs defined in the conpairs file can communicate using RDF group number 3 from the source Symmetrix (sid 77).

Once you have created dynamic SRDF pairs, you can display all, or subsets of, these pairs by omitting or including various options (`–both`, `–r1`, or `–r2`) with the `symrdf list –dynamic` command. Including the `–both` option displays dynamic SRDF pairs in which the paired devices can be either R1 or R2 devices (a requirement for dynamic R1/R2 swap). For example:

```
symrdf list –dynamic -both
```

Including the `–r1` option displays only dynamic SRDF pairs in which R1 devices cannot become R2 devices. Including the `–r2` option displays only pairs in which R2 devices cannot become R1 devices. Omitting all three options displays all dynamic SRDF pairs, regardless of their device configurations.

SRDF allows you to perform SRDF control operations on dynamic SRDF pairs within the more-often-used context of a *device group* instead of a device file by specifying `–g <GroupName>` on a `createpair` command line. Thereafter all other SRDF control commands involving the dynamic SRDF pairs can be executed within the context of a device *group* (for example, DynaGrp). Once the SRDF pairs' RDF link state is Not Ready (NR), for instance, you can cancel those dynamic SRDF pairings as follows:

```
symrdf deletepair -g DynaGrp
```

This operation changes the type of the device group from RDF1 to Regular. Devices in the device group are changed from R1 devices to standard devices. A `symrdf query` on the device group returns a message stating that the device group is not an RDF group. A `symld list` command on the device group (see "Example 3") shows that the device group type has changed to Regular and that the same devices that were created as dynamic R1 devices have returned to being RDF-capable standard devices[10].

[10]If you added other devices to the device group (such as pre-configured R1 devices), using `symrdf deletepair` to cancel the dynamic SRDF pairings in the group may result in an invalid device group. That is, you cannot have R1 devices in a Regular type device group.

Creating Dynamic RDF Groups

An RDF group (RA group) represents a logical connection between two Symmetrix units. Historically, RDF groups were limited to those static RDF groups defined at configuration time. Beginning with Solutions Enabler version 5.2 and Enginuity version 5669, you now have the flexibility to create, modify, and delete RDF groups while the Symmetrix unit is in operation. Currently, the only environment that supports dynamic RDF groups is Switched Fabric SRDF.

An RDF group is a collection of paths (links) between two Symmetrix units. When you create an SRDF pair, that pair is associated with an RDF group to define the communication paths that will be used to synchronize data between the R1 device and R2 device.

A static RDF group is defined using the configuration server and loaded when the Symmetrix unit is configured. This static definition cannot change without using the configuration server to perform the change and load a new configuration.

Creating a dynamic RDF group requires setting the Symmetrix `dynamic_rdf` parameter in the configuration. The relationship between the pair link and the RDF group is not defined in the configuration but is performed dynamically using host software, allowing you to change these relationships on the fly.

SRDF commands allow you to add, modify (such as adding or deleting RA directors), or delete a dynamic RDF group. Adding a dynamic RDF group creates an empty group. Once the group is created, you can then add dynamic SRDF pairs to it.

The following example creates dynamic RDF group 2 on the local Symmetrix unit and RDF group 3 on the remote Symmetrix unit. (Beginning with Solutions Enabler version 5.2 running on Symmetrix units using Enginuity version 5669, the maximum number of RDF groups is 64.) The command requires that you specify a group label that can be used when modifying or deleting the group.

```
symrdf addgrp –label dynagrp –rdfg 2 –sid 3264 –dir 3A,3B \
              -remote_rdfg 3 –remote_sid 3265 –remote_dir 14A,14B
```

Creation of RDF group 2 includes directors 3A and 3B from local Symmetrix 3264, and RDF group 3 includes directors 14A and 14B from remote Symmetrix 3265. Before specifying directors, make sure that the physical connections between the local RA and remote RA directors are valid and operational.

The following command adds dynamic SRDF pairs (as defined in the file named pairsfile) to the new dynamic RDF group 2 (as specified by the `-rdfg` option).

```
symrdf createpair –file pairsfile –sid 3264 –rdfg 2 –type rdf1 –invalidate r2
```

For more information on creating and establishing dynamic SRDF pairs, refer to the previous section.

You can issue the `symcfg list –ra all –switched` command to display all RDF groups on the local Symmetrix and its remotely-connected Symmetrix units. The display indicates whether an RDF group is static or dynamic (refer to “Example 4: Creating a Dynamic RDF Group”).

A dynamic RDF group must be empty to delete it. The following commands delete the SRDF pairs from the group and remove the local and remote dynamic RDF groups that were created using the label dynagrp.

```
symrdf deletepair –file pairsfile –sid 3264 –rdfg 2
symrdf removegrp –sid 3264 –label dynagrp
```

For information about command options and modifying a dynamic RDF group, refer to the *EMC Solutions Enabler Symmetrix SRDF CLI Product Guide*.

Using SRDF and TimeFinder/Mirror in Multi-Hop Configurations

Using SRDF and TimeFinder/Mirror (also referred to as “TimeFinder” in this document) to mirror data from a source Symmetrix system to a remotely-associated BCV located across the hall or across the globe creates a “global” BCV concept. That is, production data can be available on *remotely-associated* BCVs for data mining, e-business content delivery, application development, testing, backups, and numerous other uses. Moreover, an *SRDF multi-hop configuration* (multi-hop mirroring to a third site) provides a recovery solution for component or site failures between remotely-mirrored devices. SRDF reduces backup and recovery costs and significantly reduces recovery time after a disaster.

SRDF data propagation allows you to mirror to a Symmetrix system in a third location only the data that has changed since the last update. By copying only the changed tracks, you consume less bandwidth and enhance performance. You can perform Symmetrix-to-Symmetrix transmissions synchronously in the local or campus area, or use delayed synchronization methods of replication for long-distance transmission. Multi-hop mirroring to a third site can take place during off-peak times or over lower cost transmission lines or via IP-based Virtual Private Networks.

By using SRDF/AR, you can set up remote mirroring on an automatic basis according to your own predefined copy cycle (every hour, for example). For more information about SRDF/AR, refer to the white paper *Using SYMCLI to Perform SRDF/AR* (P/N 300-000-078).

Establishing a BCV Pair with an RDF BCV

With both SRDF and TimeFinder/Mirror installed, EMC can configure an SRDF device either as an RDF standard device or an RDF BCV device. An *RDF BCV* device is a BCV that is configured in a one-to-one relationship with a remote target (R2) device that mirrors the BCV, forming an SRDF pair as shown in the illustration. You can also establish an RDF BCV device as part of a BCV pair (shown in Figure 5 on the Site A Symmetrix system) but, in doing so, the RDF BCV is suspended from copying data to its target (R2) device.

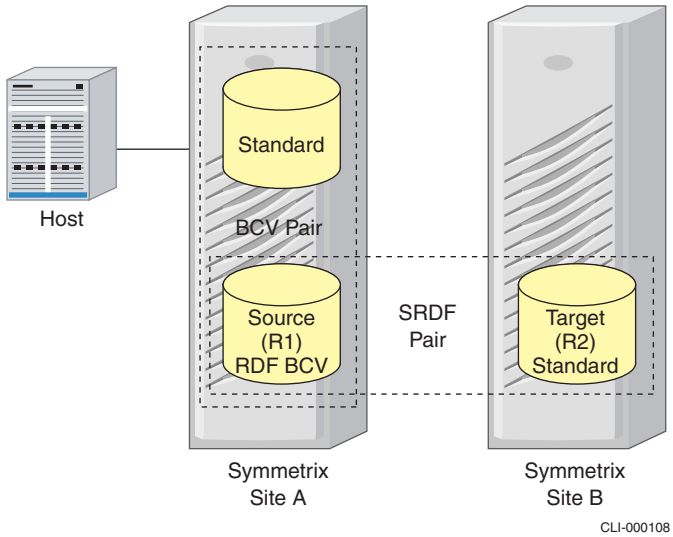

Figure 5. Distinguishing between an SRDF Pair and an RDF BCV Pair

To copy data from the standard device to the RDF BCV device at Site A, you can build a Regular type device group (RegGrp, for example) on the local host that includes these two devices. For example:

```
symdg create RegGrp –type regular
symld –g RegGrp add dev 020
symbcv –g RegGrp associate dev 090
```

Use the `symmir establish` command to initiate their synchronization:

```
symmir –g RegGrp establish -full
```

When established as a BCV pair, the RDF BCV is suspended from copying data to its remote (R2) target. To resume mirroring between them, you can perform a normal split and re-establish the RDF BCV with its remote (R2) target device as shown in the following example:

```
symmir –g RegGrp split
symrdf –g RegGrp establish –bcv
```

The `symrdf establish` command with the `–bcv` option resumes the RDF links and initiates the propagation of data from the source RDF BCV device to its remote (R2) target device.

If later you want to re-establish the BCV pair incrementally, perform a `symmir establish`. If, instead, you want to resynchronize the BCV pair but restore the standard device with data from the BCV, use a restore operation. In either case, the RDF BCV is again suspended from copying data to its remote (R2) target.

```
symmir –g RegGrp restore
```

Copying Data to a Remotely-Associated BCV

A BCV pair is a relationship that you create using the `symmir` command, matching a device configured as a BCV with a specific standard device. Using SRDF technology and the TimeFinder `-rdf` option, you can establish a BCV pair on the remote Symmetrix system (Site B in Figure 6). By doing this, the remotely-associated BCV is synchronized with the SRDF remote (R2) mirror until such time that you may decide to split the BCV pair.

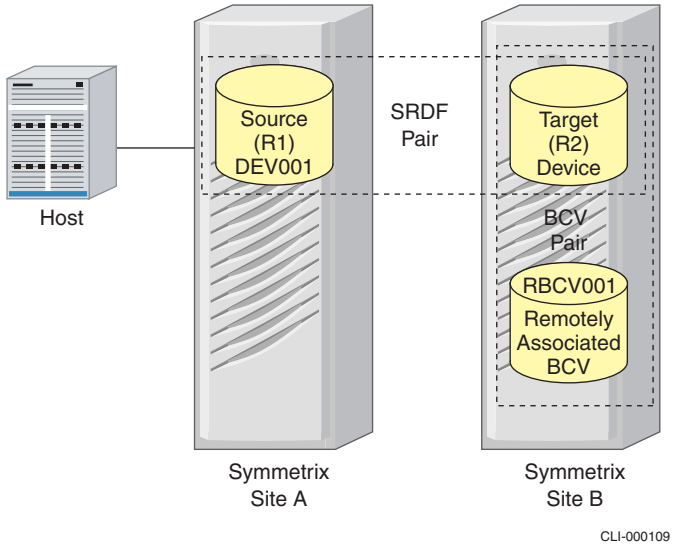

Figure 6. Copying Data across RDF Links to a Remotely-Associated BCV

To copy data from the source (R1) standard device to the remotely-associated BCV device, you can build an RDF1 device group (Rdf1Grp, for example) on the local host that includes these two devices (for example, standard device 27C from Site A and BCV device 2E9 from the Site B). The `–rdf` option is required when targeting the remotely-associated BCV for inclusion in the device group (if the source device is an R1 BCV instead of a standard device (as in Figure 4), include the `–bcv` option also with the `symbcv associate` command):

```
symdg create Rdf1Grp –type rdf1
symld –g Rdf1Grp add dev 27C
symbcv –g Rdf1Grp associate dev 2E9 -rdf
```

When establishing the remote BCV pair, assume that the source (R1) standard device and its remote (R2) target device are already synchronized, as is usually the case. A `symmir establish –rdf` command from the local host establishes the remote BCV pair, copying data from the standard's remote R2 mirror to the remotely-associated BCV:

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

Once the establish operation is complete, any new data written to the source (R1) standard device is copied to its remote (R2) mirror and subsequently to the remotely-associated BCV.

The `symmir split –rdf` command splits the remote BCV pair, making the remotely-associated BCV available to its host for business continuance tasks or a restore operation, if needed:

```
symmir -g Rdf1Grp split –rdf
```

If you want to re-establish the remote BCV pair later, perform an incremental establish by issuing the `symmir establish –rdf` command without the `–full` option. If, instead, you need to restore data from the remotely-associated BCV back to the local R1 standard device, the process involves two steps:

1. Copy the data from the remotely-associated BCV to its paired R2 standard device. For example:

   ```
   symmir -g Rdf1Grp restore –rdf
   ```

2. Then copy that data from the R2 standard device back to the R1 standard device. For example:

   ```
   symrdf -g Rdf1Grp restore
   ```

Once the data transfer initiates successfully in step 1, you can begin the data copy to the R1 standard in parallel, even though the BCV copy to its R2 standard is still in progress. Once the *restore* operation is complete, any new data written to the R1 standard is copied to the remotely-associated BCV device while this remote BCV pair exists. *This two-step process is recommended instead of the one-step process that uses `symmir restore –remote`. The two-step process is safer and less error prone.*

Copying Data in a Complex Multi-Hop Environment

TimeFinder and SRDF components of Solutions Enabler allow you to manage complex remote configurations from your local host. One of the basic remote configurations is to have a remote site (Hop 1) serve as a remote mirror to the standard devices on a local Symmetrix unit. Multi-hop configurations include having a third site (Hop 2) serve as a remote mirror to the remotely-associated BCV devices at Hop 1.

Targeting commands to various multi-hop devices and links in an SRDF multi-hop environment requires an understanding of several `symrdf` and `symmir` commands and their options. The target of an SRDF or TimeFinder operation is determined by the option or options that you choose with these commands.

Keep in mind that `symrdf` commands perform control operations on SRDF pairs. TimeFinder `symmir` commands perform control operations on BCV pairs, regardless of whether the BCV pair is local or remote. Also, you create the device group on the host attached to the local site Symmetrix.

Targeting Remote Devices When the Data Source Is the Local Standard RDF Device

The following commands issued from the local host show the use of various options to copy data across RDF links to targets associated with the local standard device. Steps 1, 2, and 3 in Figure 7 involve copying data from the local Symmetrix to the Hop 2 Symmetrix. Steps 4, 5, and 6 involve restoring data from the Hop 2 Symmetrix to the local Symmetrix. An RDF1 device group named Rdf1Grp has been created and the appropriate devices added to it (see the section “Copying Data to a Remotely-Associated BCV”).

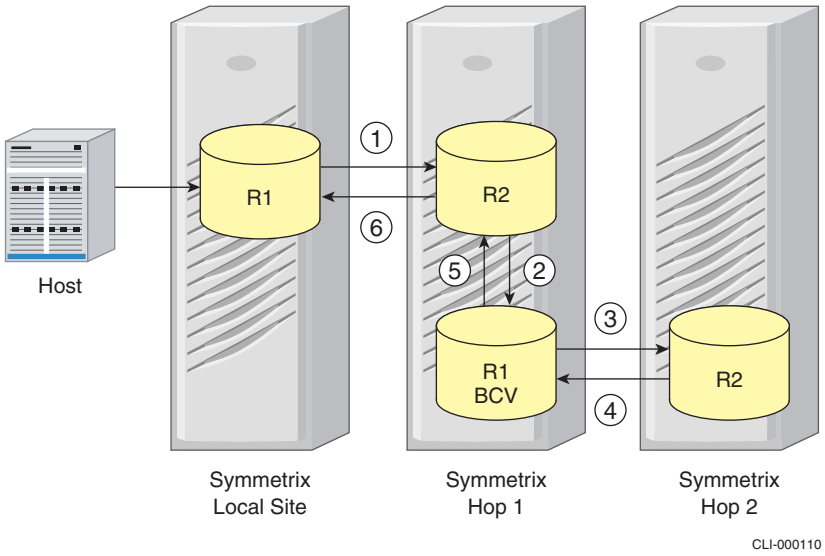

Figure 7. Copying Data to Remote Sites from a Local Standard Device

1. When EMC installs an SRDF configuration, the installers usually establish SRDF pairs at that time. Assume here that the local R1 standard device is synchronized with its remote R2 mirror. To prepare for a restore operation later, the `symrdf split` command splits the local SRDF pair.

   ```
   symrdf -g Rdf1Grp split
   ```

2. The `symmir establish` command copies data from the R2 mirror to its paired BCV. Using the `–rdf` option in the command targets the remotely-associated BCV. The `symmir verify –i` command checks every 30 seconds until the BCV pair is synchronized. The `symmir split` command splits the Hop 1 BCV pair.

   ```
   symmir –g Rdf1Grp establish –rdf
   symmir –g Rdf1Grp verify –rdf –i 30
   symmir -g Rdf1Grp split -rdf
   ```

3. While part of a BCV pair, the Hop 1 BCV is temporarily suspended from copying data to its remote R2 target. To resume mirroring between the two, you can split the Hop 1 BCV pair (as was done in step 2) and re-establish the Hop 1 BCV now with its remote R2 mirror. The `–rbcv` option targets the Hop 2 device. The `symrdf verify –i` command checks every 30 seconds until the remote SRDF pair is synchronized. The `symrdf split` command splits that SRDF pair. The copy on Hop 2 can now be used for business continuance tasks or a restore operation, if needed.

   ```
   symrdf -g Rdf1Grp establish –rbcv
   symrdf –g Rdf1Grp verify –rbcv –i 30
   symrdf -g Rdf1Grp split –rbcv
   ```

4. The `symrdf restore` command restores the Hop 1 BCV from its Hop 2 remote mirror. The next `symrdf split` command splits that SRDF pair, protecting the integrity of the Hop 2 copy.

```
symrdf -g Rdf1Grp restore –rbcv
symrdf –g Rdf1Grp verify –rbcv –i 30
symrdf -g Rdf1Grp split –rbcv
```

5. The `symmir restore` command restores the Hop 1 R2 mirror from the Hop 1 BCV.

```
symmir -g Rdf1Grp restore –rdf
```

6. The `symrdf restore` command restores a data copy from the target R2 side to the source R1 side. You can initiate this operation while the BCV pair at Hop 1 is still synchronizing as a result of the action in step 5 (that is, the BCV pair state is RestInProg).

```
symrdf -g Rdf1Grp restore
```

Targeting Remote Devices When Remote Copying Is from a Local R1 BCV

Once a local standard device has copied data to a local RDF BCV, the BCV can be used to copy that data to remote sites. In Figure 8, a local R1 BCV is configured as the data source in a one-to-one relationship with a remote R2 mirror. The following commands issued from the local host show the use of various options to copy data across RDF links to targets associated with the local R1 BCV. A Regular type device group named RegGrp has been created and the appropriate devices added to it. The BCV on Hop 1 is remotely associated with the device group using `symbcv –g RegGrp –rdf –bcv` (see the section "Establishing a BCV Pair With an RDF BCV").

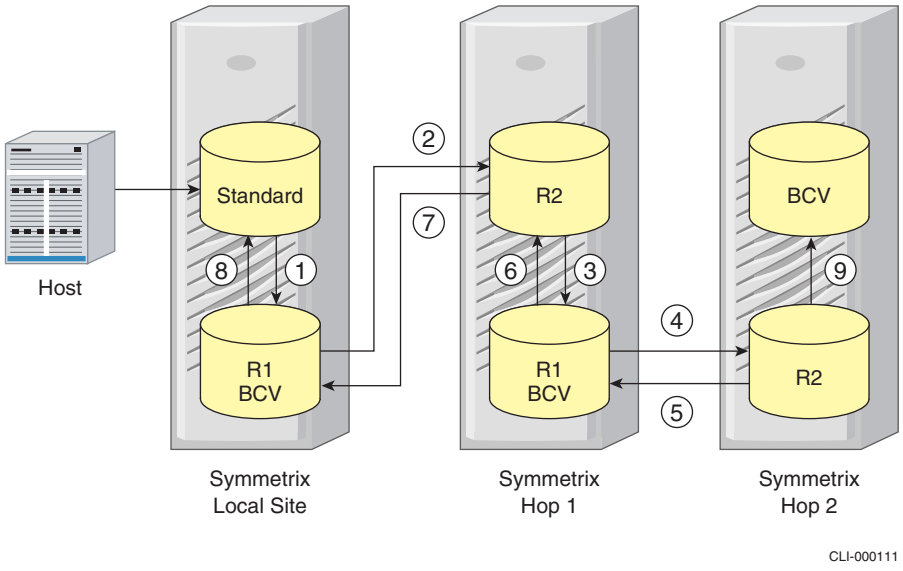

Figure 8. Copying Data to Remote Sites from a Local R1 BCV

1. The `symmir establish` command copies data from the local standard device to the local RDF BCV (a BCV with a remote R2 mirror), forming a BCV pair. The `symmir split` command splits the BCV pair.

```
symmir -g RegGrp establish
symmir -g RegGrp verify –i 30
symmir -g RegGrp split
```

2. When established as a BCV pair, the local R1 BCV is suspended from copying data to its remote (R2) target. To resume mirroring between the two, you can split the BCV pair (as in step 1) and re-establish the SRDF pair. The `symrdf establish` command copies data from the local R1 BCV to its remote R2 mirror. The `–bcv` option signifies that the local BCV should be the source. The `symrdf split` command then splits the local BCV from its remote R2 mirror.

```
symrdf -g RegGrp establish –bcv
symrdf -g RegGrp verify –bcv –i 30
symrdf -g RegGrp split –bcv
```

3. The `symmir establish` command copies data from the Hop 1 R2 mirror to the Hop 1 BCV. The `–rdf` and `–bcv` options together in the command target the BCV that is remotely associated via the local RDF BCV device. The `symmir split` command splits the Hop 1 BCV pair.

```
symmir -g RegGrp establish -rdf –bcv
symmir -g RegGrp verify –rdf –bcv –i 30
symmir -g RegGrp split -rdf -bcv
```

4. When established as a BCV pair, the Hop 1 BCV is suspended from copying data to its remote R2 target. To resume mirroring between the two, split the BCV pair (as was done in step 3) and establish the Hop 1 BCV with its remote R2 mirror, using the `–brbcv` option to target the Hop 2 device. The `symrdf split` command splits that SRDF pair. The Hop 2 copy is now accessible to its host.

```
symrdf -g RegGrp establish –brbcv
symrdf -g RegGrp verify –brbcv –i 30
symrdf -g RegGrp split –brbcv
```

5. If you need to restore data from the Hop 2 copy, the `symrdf restore` command restores the Hop 1 BCV from its Hop 2 remote mirror. The `symrdf split` command splits the SRDF pair.

```
symrdf -g RegGrp restore –brbcv
symrdf -g RegGrp verify –brbcv –i 30
symrdf -g RegGrp split –brbcv
```

6. The `symmir restore` command restores the Hop 1 mirror from the Hop 1 BCV copy.

```
symmir -g RegGrp restore -rdf –bcv
```

7. The `symrdf restore` command restores the local BCV from its Hop 1 mirror.

```
symrdf -g RegGrp restore –bcv
```

8. The `symmir restore` command restores the local standard device from the local BCV copy, re-establishing the local BCV pair and suspending the propagation of data from the local BCV to its remote R2 mirror. You can initiate this operation while the SRDF pair is still synchronizing as a result of the action in step 7 (that is, the SRDF pair state is SyncInProg).

```
symmir -g RegGrp restore
```

9. The `symmir establish` command with the `–rrbcv` option creates a BCV copy at the Hop 2 site.

```
symmir –g RegGrp establish –rrbcv
```

Using a Composite Group to Control a Set of Devices That Spans Multiple Symmetrix Units

Beginning with Solutions Enabler version 5.4, you can create a *composite group* to control a set of SRDF pairs and BCV pairs that spans multiple Symmetrix units. If you need to enable an RDF composite group for consistency protection, PowerPath® support is required (refer to the white paper *Using SYMCLI to ImplementRDF Consistency Protection with SRDF Family Products*, P/N 300-000-284).

When an RDF composite group is enabled for consistency protection, it is known as an RDF *consistency group*, the term used prior to Solutions Enabler version 5.4. Now you can use a composite group (with or without consistency protection) to control a set of SRDF pairs *and BCV pairs* that spans multiple Symmetrix units. Prior to version 5.4, you could use a consistency group to control SRDF pairs only.

Although SRDF control operations might normally be performed from a single controlling host (as shown in Figure 9) because the composite group is defined there in its SYMAPI database, there are methods that would allow you to initiate operations from other locally connected hosts. One way is to copy the composite group definition from one host to another host. A more efficient method is to enable Group Name Services (GNS), which automatically propagates the composite group definition to the Symmetrix units and other locally attached hosts that are running the GNS daemon (refer to the white paper *Using SYMCLI and GNS to Propagate Group Definitions to Multiple Hosts*, P/N 300-001-384)

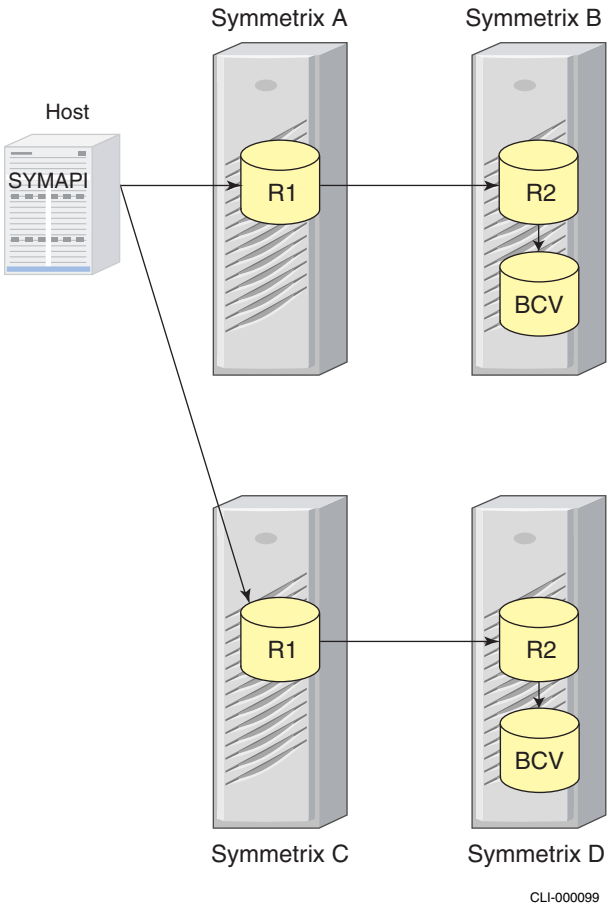

Figure 9. Using a Composite Group When Source Devices Exist on Two Symmetrix Units

Figure 9 shows a configuration in which you can control the R1 devices on two source Symmetrix units (A and C) and BCV devices on two remote Symmetrix units by including these devices in a composite group. By applying `symrdf` commands to the composite group, you control the SRDF pairs as a group. By applying `symmir` commands using the `–rdf` option, you control the remote BCV pairs as a group.

The following steps outline how to use a composite group to control a set of devices that spans multiple Symmetrix units as shown in Figure 9. Note that when adding devices to a composite group, you always specify the Symmetrix IDs (`–sid`) of the local Symmetrix units, regardless of whether the devices being added are local or remote.

1. From the controlling host, create an RDF1 type composite group (for example, MyGrp). You need to include the `–ppath` option only if you have PowerPath support and you intend to enable the group for consistency protection.

   ```
   symcg create MyGrp –type rdf1
   ```

2. Add to the composite group those standard devices on Symmetrix A (3087) and Symmetrix C (3143) that hold the source data.

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

3. Associate the remote BCV devices on Symmetrix units B and D. If there is more than one RDF group on a local Symmetrix unit (in this case, 3087 and 3143), you must include the RDF group number of the local source devices (the group number of R1 devices 0076 and 0091).

   ```
   symbcv –cg MyGrp –sid 3087 associate dev 2E5 –rdf –rdfg 2
   symbcv –cg MyGrp –sid 3143 associate dev 041 –rdf –rdfg 1
   ```

4. If the SRDF pairs are not already synchronized, establish the SRDF pairs (the local R1 devices with their paired R2 devices).

   ```
   symrdf –cg MyGrp establish
   ```

5. When the SRDF pairs are synchronized, establish the remote BCV pairs. The following command copies data from the R2 devices on Symmetrix units B and D to the BCV devices there. The `–rdf` option signifies that the targets are the remote BCVs.

   ```
   symmir –cg MyGrp establish –full –rdf
   ```

6. When the BCV pairs are synchronized, split the BCV pairs.

   ```
   symmir –cg MyGrp split –rdf -consistent
   ```

   When the split is complete, the data is now accessible on the remote BCVs.

A composite group provides greater flexibility than a device group, which can define devices only on a single Symmetrix unit. However, unlike the device group, the composite group cannot currently operate on specific pairs within the group but must perform an operation on the entire group.

Example 1: Basic SRDF Control Operations

The hardware setup for the following examples consists of two Solaris hosts, one connected to a source Symmetrix and the other connected to a target Symmetrix. To identify commands on each host, the command prompts were set to `src` and `tgt`. Display outputs may vary slightly according to the version of Solutions Enabler that you are using.

The `symrdf list` command displays information about local (R1) and remote (R2) SRDF devices. Entries in the RDF Typ:G column identify the device as either an R1 or R2 device and the RDF (RA) group number after the colon. The ellipsis (.....) represents truncated output.

```
src# symrdf list

Symmetrix ID: 000000003264

                                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  MDA   Tracks    Tracks Dev RDev Pair
---- ----   ------ --------- ----- -------  ------- --- ---- -------------
.........................................................................................
0045 0045    R2:2  RW WD NR   S..         0     49500 WD  RW   Suspended
0046 0046    R2:2  ?? WD NR   S..         0     33000 WD  RW   Suspended
0047 0047    R2:2  ?? WD NR   S..         0         0 WD  RW   Suspended
009C 0054    R1:2  RW RW RW   S..         0         0 RW  NR   Synchronized
009D 0055    R1:2  RW RW RW   S..         0         0 RW  NR   Synchronized
009E 0056    R1:2  RW RW RW   S..         0         0 RW  NR   Synchronized
009F 0057    R1:2  RW RW RW   S..         0         0 RW  NR   Synchronized
00A0 0058    R1:2  RW RW RW   A.W         0         0 RW  NR   Synchronized
00A1 0059    R1:2  RW RW RW   A.W         0         0 RW  NR   Synchronized
00A2 005A    R1:2  RW RW RW   A.W         0         0 RW  NR   Synchronized
00A3 005B    R1:2  RW RW RW   A.W         0         0 RW  NR   Synchronized
.........................................................................................
```

The `symdev list` command with the `–r1` option displays all R1 devices. Those R1 devices that are not already part of a device group are displayed as "N/Grp'd," which means they are available to be added to a new RDF1 device group.

```
src# symdev list –r1

Symmetrix ID: 000000003264

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

009C /dev/rdsk/emcpower84c   16B:1 01A:C0 RDF1          N/Grp'd      RW     516
009D /dev/rdsk/emcpower85c   16B:1 02B:D3 RDF1          N/Grp'd      RW     516
009E /dev/rdsk/emcpower90c   16B:1 02A:C0 RDF1          N/Grp'd      RW     516
009F /dev/rdsk/emcpower91c   16B:1 01B:D3 RDF1          N/Grp'd      RW     516
00A0 /dev/rdsk/emcpower92c   16B:1 01B:C0 RDF1          N/Grp'd      RW     516
00A1 /dev/rdsk/emcpower93c   16B:1 02A:D3 RDF1          Grp'd        RW     516
00A2 /dev/rdsk/emcpower94c   16B:1 02B:C0 RDF1          N/Grp'd      RW     516
```

Creating a device group and adding devices to it are prerequisites for performing SRDF operations. The `symdg create` command creates a device group (Rdf1Grp). The `symld add` commands add standard devices to the group, using either a device's physical device (pd) name or, as shown below, its Symmetrix device (dev) name. In the `symdg show` display, "Device Group RDF Information" refers to information that is applicable to all RDF standard devices in the group.

```
src# symdg create Rdf1Grp -type rdf1
src# symld -g Rdf1Grp –sid 3264 add dev 9C
src# symld -g Rdf1Grp –sid 3264 add dev 9D
src# symdg show Rdf1Grp

Group Name:  Rdf1Grp

    Group Type                                    : RDF1
    Device Group in GNS                           : Yes
    Valid                                         : Yes
    Symmetrix ID                                  : 000000003264
    Group Creation Time                           : Tue Jan  6 12:08:17 2004
    Vendor ID                                     : EMC Corp
    Application ID                                : SYMCLI

    Number of STD Devices in Group                :    2
    Number of Associated GK's                     :    0
    Number of Locally-associated BCV's            :    0
    Number of Locally-associated VDEV's           :    0
    Number of Remotely-associated BCV's (STD RDF):    0
    Number of Remotely-associated BCV's (BCV RDF):    0
    Number of Remotely-assoc'd RBCV's (RBCV RDF) :    0

    Standard (STD) Devices (2):
        {
        --------------------------------------------------------------------
                                                  Sym               Cap
        LdevName              PdevName            Dev  Att. Sts     (MB)
        --------------------------------------------------------------------
        DEV001                /dev/rdsk/c2t6d3s2  009C      RW       516
        DEV002                /dev/rdsk/c2t6d4s2  009D      RW       516
        }

    Device Group RDF Information
        {
        RDF Type                            : R1
        RDF (RA) Group Number               : 2              (01)

        Remote Symmetrix ID                 : 000000003265

        R2 Device Is Larger Than The R1 Device : False

        RDF Mode                            : Synchronous
        RDF Adaptive Copy                   : Disabled
        RDF Adaptive Copy Write Pending State  : N/A
        RDF Adaptive Copy Skew (Tracks)     : 65535

        RDF Device Domino                   : Disabled
```

```
        RDF Link Configuration                  : Fibre
        RDF Link Domino                         : Disabled
        Prevent Automatic RDF Link Recovery     : Disabled
        Prevent RAs Online Upon Power ON        : Enabled

        Device RDF Status                       : Ready          (RW)

        Device RA Status                        : Ready          (RW)
        Device Link Status                      : Ready          (RW)

        Device Suspend State                    : N/A
        Device Consistency State                : Disabled
        RDF R2 Not Ready If Invalid             : Enabled

        Device RDF State                        : Ready          (RW)
        Remote Device RDF State                 : Not Ready      (NR)

        RDF Pair State (  R1 <===> R2 )         : Synchronized

        Number of R1 Invalid Tracks             : 0
        Number of R2 Invalid Tracks             : 0
        }
```

When EMC installs an SRDF configuration, the installers usually establish static SRDF pairs at that time. The `symrdf query` command demonstrates the state of the SRDF devices and their RDF links. Under normal circumstances, the SRDF pair is synchronized (as shown below). The R1 devices are read-writeable and the RDF links are read-writeable. However, the R2 devices, which are acting as mirrors to the R1 devices, are write disabled (WD) and cannot be written to by the target-side host at this time. The link is operating with Synchronous replication (indicated by an S in the M column).

```
src# symrdf -g Rdf1Grp query

Device Group (DG) Name: Rdf1Grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000003264

       Source (R1) View                 Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
            ST                  LI      ST
Standard     A                   N       A
Logical      T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev  E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  009C RW       0        0 RW 0054 WD       0        0 S..   Synchronized
DEV002  009D RW       0        0 RW 0055 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

While the identity of the remote SRDF devices of each pair is known (9C is paired with 54, and 9D with 55), the configuration of remote Symmetrix units connected to the local Symmetrix may not be known. You can usually determine the identity of these remote Symmetrix units using the `symcfg list` command shown below. However, to identify the remote Symmetrix that contains a specific R2 device, you need to issue a `symdev show <device>` command on its paired R1.

```
src# symcfg list

                                S Y M M E T R I X

                                          Mcode    Cache      Num Phys  Num Symm
    SymmID       Attachment  Model        Version  Size (MB)  Devices   Devices

    000000003264 Local       DMX2000P     5669       20480     100       396
    000000003263 Remote      DMX2000P     5669       20480       0       534
    000000003265 Remote      8230         5568       16384       0       504
```

When two or more remote Symmetrix units are present, `symdev show` for a specific R1 device (9C) identifies its configured R2 device (54) and the remote Symmetrix on which it resides (000000003265).

```
src# symdev show 9C

Symmetrix ID: 000000003264

    Device Physical Name     : /dev/rdsk/c2t6d3s2

    Device Symmetrix Name    : 009C
    Device Serial ID         : 6409C321
    Symmetrix ID             : 000000003264

    Device Group Name        : Rdf1Grp
    Device Logical Name      : DEV001

    Attached BCV Device      : N/A

    Vendor ID                : EMC
    Product ID               : SYMMETRIX
    Product Revision         : 5669

    Device Emulation Type    : FBA
    Device Defined Label Type: N/A
    Device Defined Label     : N/A

    Device Block Size        : 512

    Device Capacity
        {
        Cylinders            :       1100
        Tracks               :      16500
        512-byte Blocks      :    1056000
        MegaBytes            :        516
        KiloBytes            :     528000
        }

    Device Configuration     : RDF1
```

```
    RDF Information
        {
        Device Symmetrix Name                  : 009C
        RDF Type                               : R1
        RDF (RA) Group Number                  : 2              (01)

        Remote Device Symmetrix Name           : 0054
        Remote Symmetrix ID                    : 000000003265
```

Another useful command to examine Symmetrix connections is `symcfg list –ra all`. This command reaches all Symmetrix units (one or two hops away) accessible through RDF links and displays the Remote Link Director information. Information in the Remote Symm ID column below shows that both Symmetrix 3264 and 3263 are connected to 3265, but 3263 and 3264 are not connected to each other. Refer to "Example 4: Creating a Dynamic RDF Group" for more information about this display.

```
src# symcfg list -ra all

Symmetrix ID: 000000003264 (Local)

                S Y M M E T R I X    R D F    D I R E C T O R S

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

  RF-3A   03A     3     3  RDF-BI-DIR   -   000000003265  2 (01)  2 (B)  Online
  RF-3B   03B    19     3  RDF-BI-DIR   -   000000003265  2 (01)  2 (B)  Online

Symmetrix ID: 000000003263 (Remote)

                S Y M M E T R I X    R D F    D I R E C T O R S

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

  RF-3A   03A     3     3  RDF-BI-DIR   -   000000003265  1 (00)  1 (A)  Online
  RF-3B   03B    19     3  RDF-BI-DIR   -   000000003265  1 (00)  1 (A)  Online

Symmetrix ID: 000000003265 (Remote)

                S Y M M E T R I X    R D F    D I R E C T O R S

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

  RF-3A   03A     3     3  RDF-BI-DIR   -   000000003263  1 (A)   1 (00) Online
  RF-14A  14A    14    14  RDF-BI-DIR   -   000000003264  2 (B)   2 (01) Online
  RF-3B   03B    19     3  RDF-BI-DIR   -   000000003263  1 (A)   1 (00) Online
  RF-3B   03B    30    14  RDF-BI-DIR   -   000000003264  2 (B)   2 (01) Online
```

The view from the target host and target Symmetrix 3265 reflects the view from the source. Symmetrix 3265 shows up as local, whereas 3264 shows up as remote.

```
tgt# symcfg list

                                S Y M M E T R I X

                                                 Mcode    Cache      Num Phys  Num Symm
    SymmID       Attachment  Model     Version  Size (MB)  Devices   Devices

    000000003265 Local       8230      5568       16384      58        504
    000000003263 Remote      DMX2000P  5669       20480       0        534
    000000003264 Remote      DMX2000P  5669       20480       0        396
```

The `symrdf list` command issued from the target host shows 54 and 55 as local, and 9C and 9D as remote. Note the RDF Typ:G column for SRDF device 47. The B- indicates an RDF BCV device, and R1 indicates an RDF1 type device. The G column value indicates that the device belongs to RDF group number 2.

```
tgt# symrdf list

Symmetrix ID: 000000003265

                              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  MDA   Tracks   Tracks Dev RDev Pair
---- ----  ------ --------- ----- -------  ------- --- ---- -------------
.........................................................................
0047 0047 B-R1:2  ?? RW NR   S..        0        0 RW  WD   Suspended
0054 009C   R2:2  RW NR RW   S..        0        0 NR  RW   Synchronized
0055 009D   R2:2  RW NR RW   S..        0        0 NR  RW   Synchronized
0056 009E   R2:2  WD NR RW   S..        0        0 NR  RW   Synchronized
0057 009F   R2:2  WD NR RW   S..        0        0 NR  RW   Synchronized
.........................................................................
```

To issue the same SRDF commands from the target-side host as from the source-side host, it is necessary to build an RDF2 target-side device group that has the same definitions as the RDF1 source-side device group. The `symdg export` command creates a text file (Rdf1Grp.txt) that contains the RDF1 group definitions. You then use `rcp` (or `ftp`) to transfer that file to the target host.

```
src# symdg export Rdf1Grp -f Rdf1Grp.txt -rdf
src# rcp Rdf1Grp.txt api28:/.
```

On the target host, the `symdg import` command builds the RDF2 device group using the definitions from the text file.

```
tgt# symdg import Rdf2Grp -f Rdf1Grp.txt

Adding standard device 0054 as DEV001...
Adding standard device 0055 as DEV002...
```

The `symld list` command displays the new RDF2 device group.

```
tgt# symld -g Rdf2Grp list

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

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

DEV001       /dev/rdsk/c1t3d0s2 0054  15A:0 01A:C0 RDF2              NR     516
DEV002       /dev/rdsk/c1t3d1s2 0055  15A:0 02B:D3 RDF2              NR     516
```

The following query from the target host displays the status of device group Rdf2Grp, and this information is the same as the previous query from the source host. The link is operating with Synchronous replication, and the state of the R2 devices is Write Disabled (WD).

```
tgt# symrdf -g Rdf2Grp query

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------

DEV001  0054 WD       0        0 RW 009C RW       0        0 S..   Synchronized
DEV002  0055 WD       0        0 RW 009D RW       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The following `newfs` commands prepare the R1 devices for writing by creating a new file system on each. The physical device names for the R1 devices are c2t6d3s2 and c2t6d4s2 (refer back to the section where Rdf1Grp was created).

```
src# newfs /dev/rdsk/c2t6d3s2
newfs: construct a new file system /dev/rdsk/c2t6d3s2: (y/n)? y
/dev/rdsk/c2t6d3s2:     1054080 sectors in 1098 cylinders of 15 tracks, 64 sectors
        514.7MB in 69 cyl groups (16 c/g, 7.50MB/g, 3584 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 15456, 30880, 46304, 61728, 77152, 92576, 108000, 123424, 138848, 154272,
 169696, 185120, 200544, 215968, 231392, 245792, 261216, 276640, 292064,
 307488, 322912, 338336, 353760, 369184, 384608, 400032, 415456, 430880,
 446304, 461728, 477152, 491552, 506976, 522400, 537824, 553248, 568672,
 584096, 599520, 614944, 630368, 645792, 661216, 676640, 692064, 707488,
 722912, 737312, 752736, 768160, 783584, 799008, 814432, 829856, 845280,
 860704, 876128, 891552, 906976, 922400, 937824, 953248, 968672, 983072,
 998496, 1013920, 1029344, 1044768,

src# newfs /dev/rdsk/c2t6d4s2
newfs: construct a new file system /dev/rdsk/c2t6d4s2: (y/n)? y
/dev/rdsk/c2t6d4s2:     1054080 sectors in 1098 cylinders of 15 tracks, 64 sectors
        514.7MB in 69 cyl groups (16 c/g, 7.50MB/g, 3584 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 15456, 30880, 46304, 61728, 77152, 92576, 108000, 123424, 138848, 154272,
 169696, 185120, 200544, 215968, 231392, 245792, 261216, 276640, 292064,
 307488, 322912, 338336, 353760, 369184, 384608, 400032, 415456, 430880,
 446304, 461728, 477152, 491552, 506976, 522400, 537824, 553248, 568672,
 584096, 599520, 614944, 630368, 645792, 661216, 676640, 692064, 707488,
 722912, 737312, 752736, 768160, 783584, 799008, 814432, 829856, 845280,
 860704, 876128, 891552, 906976, 922400, 937824, 953248, 968672, 983072,
 998496, 1013920, 1029344, 1044768,
```

The following commands create two mount points for the two volumes, mount the first one, and create a file on it called firstfile.

```
src# mkdir /R1-1 /R1-2

src# mount /dev/rdsk/c2t6d3s2 /R1-1

src# touch /R1-1/firstfile

src# ls -l /R1-1/firstfile
-rw-r--r--   1 root     other          0 Apr 16 13:18 /R1-1/firstfile

src# umount /R1-1
```

The following command splits the SRDF pairs in the device group. As part of the `symrdf split` command, the individual operations suspend and rw_enable r2 are performed. When the split is complete, a query will reveal the altered state of the links and the R2 devices.

```
src# symrdf -g Rdf1Grp -noprompt split

An RDF 'Split' operation execution is in progress for device group 'Rdf1Grp'.
Please wait...

    Suspend RDF link(s).....................................Done.
    Read/Write Enable device(s) on RA at target (R2).........Done.

The RDF 'Split' operation successfully executed for device group 'Rdf1Grp'.
```

A query from the source host reveals that the links have been logically set to NR (not ready) and the state of the R2 devices has been changed from WD to RW.

```
src# symrdf –g Rdf1Grp query

Device Group (DG) Name: Rdf1Grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000003264

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  009C RW       0        0 NR 0054 RW       0        0 S..   Split
DEV002  009D RW       0        0 NR 0055 RW       0        0 S..   Split

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The following commands create two mount points on the target and examine the contents of device 54 (c1t3d0s2) to confirm the existence of the file called firstfile.

```
tgt# mkdir /R2-1 /R2-2

tgt# mount /dev/rdsk/c1t3d0s2 /R2-1

tgt# ls -l /R2-1
total 16
-rw-r--r--   1 root     other          0 Apr 16 13:18 firstfile
drwx------   2 root     root        8192 Apr 16 13:13 lost+found
```

While the source and target are split, both R1 and R2 devices are accessible for reads and writes. The following commands change the contents of the R2 device by deleting firstfile on the target side and replacing it with a file called secondfile.

```
tgt# rm /R2-1/firstfile; touch /R2-1/secondfile

tgt# ls -l /R2-1
total 16
drwx------   2 root     root        8192 Apr 16 13:13 lost+found
-rw-r--r--   1 root     other          0 Apr 16 14:17 secondfile

tgt# umount /dev/rdsk/c1t3d0s2
```

The `symrdf query` displays the results of changing the contents of the R2 device — that there are now local (R1) invalid tracks on the target (R2) side.

```
tgt# symrdf -g Rdf2Grp query

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW       3        0 NR 009C RW       0        0 S..   Split
DEV002  0055 RW       0        0 NR 009D RW       0        0 S..   Split

Total          -------- --------           -------- --------
  Track(s)            3        0                  0        0
  MB(s)             0.0      0.0                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The following command on the target side performs an incremental establish for the SRDF pairs in device group Rdf2Grp, copying to the R2 device any changes that have been made to the R1 device while the devices were split. Like all RDF control operations, you can initiate the establish action from either the source or target side with the same results. The individual operations that combine to create an establish action are logged as they occur. For a more detailed report, you can examine the log file in /var/symapi/log/symapi-yyyymmdd.log.

```
tgt# symrdf -g Rdf2Grp -noprompt establish

An RDF 'Incremental Establish' operation execution is in progress for
device group 'Rdf2Grp'. Please wait...

    Write Disable device(s) on RA at target (R2).............Done.
    Suspend RDF link(s)......................................Done.
    Mark target (R2) devices to refresh from source (R1)......Started.
    Device: 0054 ............................................ Marked.
    Mark target (R2) devices to refresh from source (R1)......Done.
    Suspend RDF link(s)......................................Done.
    Merge device track tables between source and target.......Started.
    Device: 009C ............................................ Merged.
    Merge device track tables between source and target.......Done.
    Resume RDF link(s).......................................Done.

The RDF 'Incremental Establish' operation successfully initiated for
device group 'Rdf2Grp'.
```

The `symrdf verify` command confirms that the SRDF pairs are completely synchronized.

```
tgt#> symrdf -g Rdf2Grp verify

All devices in the RDF group 'Rdf2Grp' are in the 'Synchronized' state.
```

A `symrdf split` command followed by an examination of device c1t3d0s2 confirms that the recently created secondfile on the R2 device has been removed and firstfile has been restored as a result of re-establishing the SRDF device pair.

```
tgt# symrdf -g Rdf2Grp -noprompt split

An RDF 'Split' operation execution is in progress for device group 'Rdf2Grp'.
Please wait...

    Suspend RDF link(s)......................................Done.
    Read/Write Enable device(s) on RA at target (R2).........Done.

The RDF 'Split' operation successfully executed for device group 'Rdf2Grp'.

tgt# mount /dev/rdsk/c1t3d0s2 /R2-1

tgt# ls -l  /R2-1
total 16
-rw-r--r--   1 root     other          0 Apr 16 13:18 firstfile
drwx------   2 root     root        8192 Apr 16 13:13 lost+found
```

In preparation for demonstrating a restore operation, the following commands replace firstfile on the R2 device with a file called thirdfile.

```
tgt# rm /R2-1/firstfile; touch /R2-1/thirdfile

tgt# ls -l /R2-1
total 16
drwx------   2 root     root        8192 Apr 16 13:13 lost+found
-rw-r--r--   1 root     other          0 Apr 16 14:56 thirdfile

tgt# umount /R2-1
```

The `symrdf query` displays again the results of changing the contents of the R2 device — that there are now local (R1) invalid tracks on the target (R2) side.

```
src# symrdf -g Rdf1Grp query

Device Group (DG) Name: Rdf1Grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000003264

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  009C RW       0        0 NR 0054 RW       3        0 S..   Split
DEV002  009D RW       0        0 NR 0055 RW       0        0 S..   Split

Total          -------- --------           -------- --------
  Track(s)            0        0                  3        0
  MB(s)             0.0      0.0                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The following `symrdf restore` command is issued from the host on the source side. Because the `–full` option is omitted from the command line, the system performs an incremental restore, copying tracks that changed on the R2 device to the R1 device. In the process, any tracks on the R1 side that changed while the SRDF pairs were split are overwritten with data from corresponding tracks on the R2 side. When the restore is complete, the R1 device will contain the same data as the R2 device.

```
src# symrdf -g Rdf1Grp -noprompt restore

An RDF 'Incremental Restore' operation execution is in progress for device
group 'Rdf1Grp'. Please wait...

    Write Disable device(s) on SA at source (R1).............Done.
    Write Disable device(s) on RA at target (R2).............Done.
    Suspend RDF link(s).....................................Done.
```

```
    Merge device track tables between source and target.......Started.
    Devices: 009C-009D ..................................... Merged.
    Merge device track tables between source and target.......Done.
    Resume RDF link(s)........................................Done.
    Read/Write Enable device(s) on SA at source (R1)..........Done.

The RDF 'Incremental Restore' operation successfully initiated for device
group 'Rdf1Grp'.
```

The following commands mount the c2t6d3s2 device (an R1 device) and examine its contents. The directory listing below confirms that the restore operation copied thirdfile from the R2 device to the R1 device.

```
src# mount /dev/rdsk/c2t6d3s2 /R1-1

src# ls -l /R1-1
total 16
drwx------   2 root     root        8192 Apr 16 13:13 lost+found
-rw-r--r--   1 root     other          0 Apr 16 14:56 thirdfile
```

The following query illustrates that the SRDF pairs are now in the in the Synchronized state. (Note that the same restore operation with TimeFinder/Mirror software places the standard device in the Restored state. However, SRDF does not use the Restored state and places SRDF pairs in the Synchronized state after either an establish or restore operation.)

```
src# symrdf -g Rdf1Grp query

Device Group (DG) Name: Rdf1Grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000003264

       Source (R1) View                 Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  009C RW       0        0 RW 0054 WD       0        0 S..   Synchronized
DEV002  009D RW       0        0 RW 0055 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

A failover operation is similar to a split operation. However, because a failover is usually executed when a disaster on the source side necessitates moving data processing to the target side, a failover will write disable the R1 devices.

```
src# symrdf -g Rdf1Grp -noprompt failover

An RDF 'Failover' operation execution is in progress for device group 'Rdf1Grp'.
Please wait...

    Write Disable device(s) on SA at source (R1).............Done.
    Suspend RDF link(s).....................................Done.
    Read/Write Enable device(s) on RA at target (R2).........Done.

The RDF 'Failover' operation successfully executed for device group 'Rdf1Grp'.
```

The following query shows that the R1 devices are write disabled (WD), the RDF links have been suspended, and the R2 devices are read/write enabled (RW).

```
src# symrdf -g Rdf1Grp query

Device Group (DG) Name: Rdf1Grp
DG's Type             : RDF1
DG's Symmetrix ID     : 000000003264

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  009C WD       0        0 NR 0054 RW       0        0 S..   Failed Over
DEV002  009D WD       0        0 NR 0055 RW       0        0 S..   Failed Over

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

While data processing continues on the target (R2) side, Symmetrix keeps a record of the tracks on the R2 side that have changed since the failover. The remote (R1) invalid tracks on the target (R2) side are those tracks that must be copied from the R2 device to the R1 device when the RDF links are re-established and a failback is performed. For the example, a C-Shell interactive script is run to continually rewrite the data on the R2 devices. The subsequent query illustrates that there is a continuous accumulation of remote (R1) invalid tracks on the target (R2) side.

```
tgt# while (1)
? dd if=/dev/rdsk/c1t3d0s2 of=/dev/rdsk/c1t3d0s2 bs=1024k count=512
? dd if=/dev/rdsk/c1t3d1s2 of=/dev/rdsk/c1t3d1s2 bs=1024k count=512
? end

tgt# symrdf -g Rdf2Grp query

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW   16385        0 NR 009C WD       0        0 S..   Failed Over
DEV002  0055 RW   16385        0 NR 009D WD       0        0 S..   Failed Over

Total          -------- --------           -------- --------
  Track(s)        32770        0                  0        0
  MB(s)          1024.0      0.0                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

While the R2 side remains accessible for reads and writes, the `symrdf update` command takes a one-time snapshot of the remote (R1)invalid tracks on the target (R2) side for each device in the group (16385 in each case) and copies those tracks to the R1 side. The function of the update operation is to minimize downtime when issuing a failback command, which write disables the R2.

```
src# symrdf -g Rdf1Grp -noprompt update

An RDF 'Update R1' operation execution is in progress for device group 'Rdf1Grp'.
Please wait...

    Suspend RDF link(s).....................................Done.
    Merge device track tables between source and target.....Started.
    Device: 009C ........................................... Merged.
    Device: 009D ........................................... Merged.
    Merge device track tables between source and target.....Done.
    Resume RDF link(s)......................................Done.

The RDF 'Update R1' operation successfully initiated for device group 'Rdf1Grp'.
```

As an update session begins, the source Symmetrix invalidates tracks (16385) on the source (R1) that need updating.

```
tgt# symrdf -g Rdf2Grp query –i 5

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW   16385        0 RW 009C WD   16385        0 S..   R1 UpdInProg
DEV002  0055 RW   16385        0 RW 009D WD   16385        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)        32770        0              32770        0
  MB(s)          1024.0      0.0             1024.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

As the update progresses, the number of local (R1) invalid tracks as viewed on the source (R1) side keep decreasing because the tracks are being counted down from the original snapshot taken at the beginning of the update process. Meanwhile, the remote (R1) invalid tracks on the target (R2) side continue to be incremented as new writes are executed there.

```
Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW   15842        0 RW 009C WD   15125        0 S..   R1 UpdInProg
DEV002  0055 RW   15533        0 RW 009D WD   14891        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)        31375        0              30016        0
  MB(s)           980.0      0.0              938.0      0.0

Synchronization rate           :     17.2 MB/S
Estimated time to completion   : 00:00:54


Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW   13040        0 RW 009C WD   12406        0 S..   R1 UpdInProg
DEV002  0055 RW   15819        0 RW 009D WD   12479        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)        28859        0              24885        0
  MB(s)           901.0      0.0              777.0      0.0

Synchronization rate           :     32.1 MB/S
Estimated time to completion   : 00:00:24
```

Once the initial 16385 tracks have been updated, the local (R1) invalid tracks on the source (R1) side reach zero, signifying the end of the update operation. During this time, any newly written tracks on the R2 side continue being marked as remote (R1) invalid tracks on the target (R2) side.

```
Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
            ST                   LI      ST
Standard     A                    N       A
Logical      T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev  E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW    9650        0 RW 009C WD       0        0 S..   R1 Updated
DEV002  0055 RW    8574        0 RW 009D WD       0        0 S..   R1 Updated

Total          -------- --------           -------- --------
  Track(s)        18224        0                  0        0
  MB(s)           569.0      0.0                0.0      0.0
```

To demonstrate the `update –until` option, the example keeps running continuous I/O to the R2 devices and employs two windows: one to provide query displays as the update cycles progress, and one to follow the continuing output from the `symrdf update –until` command.

In the query window below, the `symrdf query` command displays the initial status of the RDF pairs and will redisplay every five seconds. Recall that remote (R1) invalid tracks on the target (R2) side represent continuous I/O to the R2 devices. The local (R1) invalid tracks on the source (R1) side represent the number of tracks that still need to be copied from the target (R2) side (currently zero until the update begins). Only a sampling of the many query displays is shown here, not every one.

```
tgt# symrdf -g Rdf2Grp query -i 5

Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
            ST                   LI      ST
Standard     A                    N       A
Logical      T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev  E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW   12381        0 RW 009C WD       0        0 S..   R1 Updated
DEV002  0055 RW   12371        0 RW 009D WD       0        0 S..   R1 Updated

Total          -------- --------           -------- --------
  Track(s)        14752        0                  0        0
  MB(s)           459.0      0.0                0.0      0.0
```

The update window below illustrates the `symrdf update` command with the `–until` option track threshold of 100 tracks. While the target (R2) side remains accessible for reads and writes, SYMCLI takes a one-time snapshot of the invalid tracks for each device in the group on the target (R2) side and requests SRDF to begin copying those tracks to the source (R1) side. If SRDF finishes fully copying the snapshot batch of updates to the R1 side and there are still 100 or more R1 (modified) invalid tracks on the target (R2) side, SYMCLI takes another snapshot and requests SRDF to begin copying another batch of tracks to the R1 side. The window displays the series of operations that initiate this first update cycle.

```
tgt# symrdf -g Rdf2Grp -noprompt update -until 100

An RDF 'Update R1' operation execution is in progress for device group 'Rdf2Grp'.
Please wait...

    Suspend RDF link(s).....................................Done.
    Merge device track tables between source and target.....Started.
    Device: 009C ........................................... Merged.
    Device: 009D ........................................... Merged.
    Merge device track tables between source and target.....Done.
    Resume RDF link(s)......................................Done.

The RDF 'Update R1' operation successfully initiated for device group 'Rdf2Grp'.
```

The query window below indicates the progression of the first update cycle. As the update progresses, the number of R1 invalid tracks as viewed on the R1 side will continue to decrease as the tracks copied to the R1 device are subtracted from the original snapshot taken at the beginning of the update process. In this update cycle, there are 7379 tracks that remain to be copied from DEV001 on the target (R2) side, and 6767 tracks still to be copied from DEV002 on the target (R2) side. Meanwhile, the R1 (modified) invalid tracks on the R2 side continue to be incremented as new I/O continues there.

```
Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW    7658        0 RW 009C WD    7379        0 S..   R1 UpdInProg
DEV002  0055 RW    7026        0 RW 009D WD    6767        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)        14684        0              14146        0
  MB(s)           458.0      0.0              442.0      0.0

Synchronization rate           :     16.1 MB/S
Estimated time to completion   : 00:00:27
```

The query window below indicates that the first batch of updates has been fully copied to the R1 side. The local (R1) invalid track count on the R1 side is zero. Because of continuous I/O on the R2 side during the update cycle, the R1 (modified) invalid track count there is 1436 and still over the 100-track threshold. Therefore, SYMCLI will automatically initiate another update cycle. Note, however, that I/O to the R2 side was turned off before the end of this update cycle, which means that this batch of invalid tracks (1436) will be the last batch copied before termination.

```
       Target (R2) View                Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv   K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks   S Dev   E  Tracks   Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW     757        0 RW 009C WD       0        0 S..   R1 UpdInProg
DEV002  0055 RW     679        0 RW 009D WD       0        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)         1436        0                  0        0
  MB(s)            44.0      0.0                0.0      0.0
```

The update window below indicates the beginning of the second update cycle.

```
An RDF 'Update R1' operation execution is in progress for device group 'Rdf2Grp'.
Please wait...

    Suspend RDF link(s).....................................Done.
    Merge device track tables between source and target.....Started.
    Devices: 009C-009D ..................................... Merged.
    Merge device track tables between source and target.....Done.
    Resume RDF link(s)......................................Done.

The RDF 'Update R1' operation successfully initiated for device group 'Rdf2Grp'.
```

The query window below confirms that continuous I/O to the R2 side has stopped. The number of R1 invalid tracks on both the target side and source side is exactly the same. Recall that when I/O to the R2 side was continuing, the R1 invalid track count there continued to increase and was always greater than the R1 invalid tracks on the R1 side. However, this last batch of updates has not yet been fully copied to the R1 side. Therefore, the RDF pair state still reads R1 UpdInProg.

```
Device Group (DG) Name: Rdf2Grp
DG's Type             : RDF2
DG's Symmetrix ID     : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW     292        0 RW 009C WD     292        0 S..   R1 UpdInProg
DEV002  0055 RW     167        0 RW 009D WD     167        0 S..   R1 UpdInProg

Total          -------- --------           -------- --------
  Track(s)          459        0                459        0
  MB(s)            14.0      0.0               14.0      0.0
```

The final query window below shows that the update is complete. The zero count of R1 invalid tracks on the R1 side indicates that this batch was fully copied. The RDF pair state is R1 Updated. The zero count on the total of R1 (modified) invalid tracks on the R2 side indicates a number lower than the 100-track threshold that defined the limit of this update operation.

```
Device Group (DG) Name : Rdf2Grp
DG's Type              : RDF2
DG's Symmetrix ID      : 000000003265

       Target (R2) View                 Source (R1) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0054 RW       0        0 RW 009C WD       0        0 S..   R1 Updated
DEV002  0055 RW       0        0 RW 009D WD       0        0 S..   R1 Updated

Total          -------- --------           -------- --------
  Track(s)            0        0                  0        0
  MB(s)             0.0      0.0                0.0      0.0
```

Example 2: Concurrent RDF

The hardware configuration for the following concurrent RDF example consists of:

- Local Source Symmetrix (sid 77): R1 standard devices 28 and 29
- Remote Target Symmetrix (sid 123): R2 concurrent devices 00 (with 28) and 01 (with 29)
- Remote Target Symmetrix (sid 124): R2 concurrent devices 50 (with 28) and 51 (with 29)

Display outputs may vary slightly according to the version of Solutions Enabler that you are using.

All commands are issued from the source-side host. The `symcfg list -v` command displays the characteristics of these Symmetrix systems in detail. Note that each Symmetrix system in the configuration must have its "Concurrent RDF Configuration State" set to Enabled, which is a prerequisite for establishing the concurrent SRDF pairs.

```
# symcfg -v list

Symmetrix ID: 000185400077 (Local)

    Product Model                          : DMX800
    Symmetrix ID                           : 000185400077

    Microcode Version (Number)             : 5670 (16260000)
    Microcode Date                         : 01.05.2004

    Microcode Patch Date                   : 01.05.2004
    Microcode Patch Level                  : 64

    Cache Size                             :    6144 (MB)
    # of Available Cache Slots             :  107946
    Max # of System Write Pending Slots    :   86441
    Max # of DA Write Pending Slots        :   43220
    Max # of Device Write Pending Slots    :    1330

    Symmetrix Total Operating Time         : 62 days, 22:23:35
    Symmetrix Power ON Time                : Tue Nov  4 13:44:30 2003
    Symmetrix Last IPL Time (Cold)         : Mon Dec 15 14:38:35 2003
    Symmetrix Last Fast IPL Time (Hot)     : Mon Jan  5 16:35:51 2004

    Host DB Sync Time                      : Tue Jan  6 11:14:43 2004
    Symmetrix CLI (SYMCLI) Version         : V5.4.0.0 (Edit Level: 516)
    Built with SYMAPI Version              : V5.4.0.0 (Edit Level: 516)
    SYMAPI Run Time Version                : V5.4.0.0 (Edit Level: 516)

    Number of Configured (Sym) Devices     : 938
    Number of Visible (Host) Devices       :   2
    Number of Configured Actual Disks      :  60
    Number of Configured Hot Spares        :   0
    Number of Unconfigured Disks           :   0
    Maximum number of hypers per disk      :  32

    Number of Powerpath Devices            :   2
    Powerpath Run Time Version             : 4.1.0

    SDDF Configuration State               : Enabled
    Configuration Change State             : Enabled
    WORM Configuration Level               : WORM_3
    WORM Characteristics                   : MANUAL_LOCK
```

```
    Symmetrix Configuration Checksum     : 2E43F0
    Switched RDF Configuration State     : Disabled
    Concurrent RDF Configuration State   : Enabled
    Dynamic RDF Configuration State      : Enabled
    Concurrent Dynamic RDF Configuration : Enabled
    RDF Data Mobility Configuration State: Disabled
    Access Control Configuration State   : Enabled
    Device Masking (VCM) Config State    : Disabled
    VCMdb Access Restricted State        : Disabled
    Multi LRU Device Assignment          : BY_NUMBER
    Disk Group Assignments               : Not in Use

    Parity Raid Configuration            : RAID-S (7+1)
    Raid-5 Configuration                 : RAID-5 (7+1)

Symmetrix ID: 000185400123 (Remote)

    Product Model                        : DMX800
    Symmetrix ID                         : 000185400123

    Microcode Version (Number)           : 5670 (16260000)
    Microcode Date                       : 01.05.2004

    Microcode Patch Date                 : 01.05.2004
    Microcode Patch Level                : 64

    Cache Size                           :    6144 (MB)
    # of Available Cache Slots           :  112671
    # of PermaCache Slots In Use         :    3276
    Max # of System Write Pending Slots  :   90224
    Max # of DA Write Pending Slots      :   45112
    Max # of Device Write Pending Slots  :    1680

    Symmetrix Total Operating Time       : 67 days, 21:40:33
    Symmetrix Power ON Time              : Thu Oct 30 14:27:32 2003
    Symmetrix Last IPL Time (Cold)       : Fri Oct 31 09:21:58 2003
    Symmetrix Last Fast IPL Time (Hot)   : Mon Jan  5 16:32:19 2004

    Host DB Sync Time                    : Tue Jan  6 11:14:43 2004
    Symmetrix CLI (SYMCLI) Version       : V5.4.0.0 (Edit Level: 516)
    Built with SYMAPI Version            : V5.4.0.0 (Edit Level: 516)
    SYMAPI Run Time Version              : V5.4.0.0 (Edit Level: 516)

    Number of Configured (Sym) Devices   : 763
    Number of Visible (Host) Devices     :   0
    Number of Configured Actual Disks    :  60
    Number of Configured Hot Spares      :   0
    Number of Unconfigured Disks         :   0
    Maximum number of hypers per disk    :  32

    Number of Powerpath Devices          :   0
    Powerpath Run Time Version           : 4.1.0

    SDDF Configuration State             : Enabled
    Configuration Change State           : Enabled
    WORM Configuration Level             : WORM_3
```

```
    WORM Characteristics                 : MANUAL_LOCK

    Symmetrix Configuration Checksum     : 2622AC
    Switched RDF Configuration State     : Disabled
    Concurrent RDF Configuration State   : Enabled
    Dynamic RDF Configuration State      : Enabled
    Concurrent Dynamic RDF Configuration : Enabled
    RDF Data Mobility Configuration State: Disabled
    Access Control Configuration State   : Enabled
    Device Masking (VCM) Config State    : Disabled
    VCMdb Access Restricted State        : Disabled
    Multi LRU Device Assignment          : BY_NUMBER
    Disk Group Assignments               : Not in Use

    Parity Raid Configuration            : RAID-S (7+1)
    Raid-5 Configuration                 : RAID-5 (7+1)

Symmetrix ID: 000185400124 (Remote)

    Product Model                        : DMX800
    Symmetrix ID                         : 000185400124

    Microcode Version (Number)           : 5670 (16260000)
    Microcode Date                       : 01.05.2004

    Microcode Patch Date                 : 01.05.2004
    Microcode Patch Level                : 64

    Cache Size                           :    6144 (MB)
    # of Available Cache Slots           :  112671
    # of PermaCache Slots In Use         :    3276
    Max # of System Write Pending Slots  :   90224
    Max # of DA Write Pending Slots      :   45112
    Max # of Device Write Pending Slots  :    1680

    Symmetrix Total Operating Time       : 67 days, 21:40:33
    Symmetrix Power ON Time              : Thu Oct 30 14:27:32 2003
    Symmetrix Last IPL Time (Cold)       : Fri Oct 31 09:21:58 2003
    Symmetrix Last Fast IPL Time (Hot)   : Mon Jan  5 16:32:19 2004

    Host DB Sync Time                    : Tue Jan  6 11:14:43 2004
    Symmetrix CLI (SYMCLI) Version       : V5.4.0.0 (Edit Level: 516)
    Built with SYMAPI Version            : V5.4.0.0 (Edit Level: 516)
    SYMAPI Run Time Version              : V5.4.0.0 (Edit Level: 516)

    Number of Configured (Sym) Devices   : 763
    Number of Visible (Host) Devices     :   0
    Number of Configured Actual Disks    :  60
    Number of Configured Hot Spares      :   0
    Number of Unconfigured Disks         :   0
    Maximum number of hypers per disk    :  32

    Number of Powerpath Devices          :   0
    Powerpath Run Time Version           : 4.1.0

    SDDF Configuration State             : Enabled
    Configuration Change State           : Enabled
```

```
    WORM Configuration Level             : WORM_3
    WORM Characteristics                 : MANUAL_LOCK

    Symmetrix Configuration Checksum     : 2622AC
    Switched RDF Configuration State     : Disabled
    Concurrent RDF Configuration State   : Enabled
    Dynamic RDF Configuration State      : Enabled
    Concurrent Dynamic RDF Configuration : Enabled
    RDF Data Mobility Configuration State: Disabled
    Access Control Configuration State   : Enabled
    Device Masking (VCM) Config State    : Disabled
    VCMdb Access Restricted State        : Disabled
    Multi LRU Device Assignment          : BY_NUMBER
    Disk Group Assignments               : Not in Use

    Parity Raid Configuration            : RAID-S (7+1)
    Raid-5 Configuration                 : RAID-5 (7+1)
```

The `symrdf list` command with the `–concurrent` option shows devices on the local Symmetrix (sid 77) that are configured as concurrent RDF devices. Note that each of the two concurrent devices of an SRDF concurrent pair belongs to a different RDF group ("RDF Typ:G" 1 and 2). The ellipsis (.....) represents truncated output.

```
# symrdf list -sid 77 -concurrent

Symmetrix ID: 000185400077

                              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  MDA   Tracks   Tracks Dev RDev Pair
---- ----  ------ --------- ----- -------  ------- --- ---- -------------

0028 0050   R1:1  RW RW RW   S..        0        0 RW  WD   Synchronized
     0000   R1:2  RW RW RW   S..        0        0 RW  WD   Synchronized
0029 0051   R1:1  RW RW RW   S..        0        0 RW  WD   Synchronized
     0001   R1:2  RW RW RW   S..        0        0 RW  WD   Synchronized
002A 0052   R1:1  RW RW RW   S..        0        0 RW  WD   Synchronized
     0002   R1:2  RW RW RW   S..        0        0 RW  WD   Synchronized
```

The `sympd list` command displays all Symmetrix devices that are visible to the local host. The display below has been edited to show only those devices that will be used in the example. The N/Grp'd attribute means that these devices are not already part of a device group and are free to be added to a device group.

```
# sympd list -sid 77

Symmetrix ID: 000185400077

        Device Name          Directors                  Device
-------------------------- ------------ -------------------------------------
                                                                          Cap
Physical                Sym  SA :P DA :IT Config        Attribute    Sts    (MB)
-------------------------- ------------ -------------------------------------
/dev/rdsk/c1t0d32s2    0028 16A:0 02A:C4 RDF1          N/Grp'd      RW      480
/dev/rdsk/c1t0d33s2    0029 16A:0 01B:D1 RDF1          N/Grp'd      RW      480
```

Creating a device group and adding devices to it are prerequisites for performing SRDF and TimeFinder operations. The `symdg create` command creates a device group named conrdf. The `symld` commands add devices 28 and 29 to it.

```
# symdg create conrdf –type rdf1
# symld -g conrdf add dev 28
# symld -g conrdf add dev 29
```

The `symrdf query` command displays the status of the SRDF pairs in the device group. The `–rdfg all` option ensures that the display shows the status of both links of a concurrent SRDF pair.

```
# symrdf -g conrdf query -rdfg all

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124
RDF (RA) Group Number :  1 (00)
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW       0        0 RW 0050 WD       0        0 S..   Synchronized
             RW       0        0 RW 0000 WD       0        0 S..   Synchronized
DEV002  0029 RW       0        0 RW 0051 WD       0        0 S..   Synchronized
             RW       0        0 RW 0001 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf split` command splits the SRDF pairs in the device group. You can split a concurrent SRDF pair either simultaneously or sequentially. The `–rdfg all` option causes both concurrent devices of an SRDF concurrent pair to be split at the same time.

```
# symrdf -g conrdf split -rdfg all -noprompt

An RDF 'Split' operation execution is in progress for device group 'conrdf'.
Please wait...

    Suspend RDF link(s) ..........................................Done.
    Read/Write Enable device(s) in (0077,01) on RA at target (R2)...Done.
    Read/Write Enable device(s) in (0077,02) on RA at target (R2)...Done.

The RDF 'Split' operation successfully executed for device group 'conrdf'.
```

The `symrdf establish` command performs an incremental establish on the SRDF pairs in the device group. You can establish a concurrent SRDF pair either simultaneously or sequentially. The `–rdfg all` option causes both concurrent devices of an SRDF concurrent pair to be established simultaneously.

```
# symrdf -g conrdf establish -rdfg all -noprompt

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

    Write Disable device(s) in (0077,01) on RA at target (R2).......Done.
    Write Disable device(s) in (0077,02) on RA at target (R2).......Done.
    Suspend RDF link(s) for device(s) in (0077,01)..................Done.
    Suspend RDF link(s) for device(s) in (0077,02)..................Done.
    Mark target device(s) in (0077,01) for incremental copy from source..Started.
    Device: 0028 ................................................ Marked.
    Device: 0029 ................................................ Marked.
    Mark target device(s) in (0077,01) for incremental copy from source..Done.
    Mark target device(s) in (0077,02) for incremental copy from source..Started.
    Device: 0028 ................................................ Marked.
    Device: 0029 ................................................ Marked.
    Mark target device(s) in (0077,02) for incremental copy from source..Done.
    Merge track tables between source and target in (0077,01).......Started.
    Device: 0028 ................................................ Merged.
    Device: 0029 ................................................ Merged.
    Merge track tables between source and target in (0077,01).......Done.
    Merge track tables between source and target in (0077,02).......Started.
    Device: 0028 ................................................ Merged.
    Device: 0029 ................................................ Merged.
    Merge track tables between source and target in (0077,02).......Done.
    Resume RDF link(s) for device(s) in (0077,01)...................Done.
    Resume RDF link(s) for device(s) in (0077,02)...................Done.

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

The following query shows that the concurrent SRDF pairs are in the process of synchronizing (state is SyncInProg).

```
# symrdf -g conrdf query -rdfg all

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124
RDF (RA) Group Number :  1 (00)
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW    5832        0 RW 0050 WD       0        0 S..   SyncInProg
             RW    5832        0 RW 0000 WD       0        0 S..   SyncInProg
DEV002  0029 RW    8426        0 RW 0051 WD       0        0 S..   SyncInProg
             RW    8426        0 RW 0001 WD       0        0 S..   SyncInProg

Total          -------- --------           -------- --------
  Track(s)        28516        0                  0        0
  MB(s)           891.0      0.0                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf verify` command with the `–rdfg all` option displays a message every 30 seconds until both concurrent mirrors of each SRDF pair are synchronized.

```
# symrdf -g conrdf verify -rdfg all –i 30 -synchronized

Not all devices in the RDF group 'conrdf' are in the 'Synchronized' state.

Not all devices in the RDF group 'conrdf' are in the 'Synchronized' state.

All devices in the RDF group 'conrdf' are in the 'Synchronized' state.
```

The `symrdf query` command confirms that both concurrent SRDF pairs are in the Synchronized state.

```
# symrdf -g conrdf query -rdfg all

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124
RDF (RA) Group Number :  1 (00)
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW       0        0 RW 0050 WD       0        0 S..   Synchronized
             RW       0        0 RW 0000 WD       0        0 S..   Synchronized
DEV002  0029 RW       0        0 RW 0051 WD       0        0 S..   Synchronized
             RW       0        0 RW 0001 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf split` command splits the SRDF pairs in the device group. The `–rdfg all` option causes both concurrent devices of an SRDF concurrent pair to be split at the same time.

```
# symrdf -g conrdf split -rdfg all -noprompt

An RDF 'Split' operation execution is in progress for device group 'conrdf'.
Please wait...

    Suspend RDF link(s) ........................................Done.
    Read/Write Enable device(s) in (0077,01) on RA at target (R2)...Done.
    Read/Write Enable device(s) in (0077,02) on RA at target (R2)...Done.

The RDF 'Split' operation successfully executed for device group 'conrdf'.
```

If you want to restore data from the concurrent target (R2) devices to their respective source (R1) device, you can restore from one of the concurrent R2 mirrors at any given time. The following `symrdf restore` command with the `–rdfg 2` option causes a restore operation from the concurrent R2 mirror whose link is represented by RDF group 2. (An earlier `symrdf list –concurrent` command displayed which concurrent R2 mirrors belong to which RDF group.)

```
# symrdf -g conrdf restore -rdfg 2 -noprompt

An RDF 'Incremental Restore' operation execution is in progress for device
group 'conrdf'. Please wait...

    Write Disable device(s) in (0077,02) on SA at source (R1).......Done.
    Write Disable device(s) in (0077,02) on RA at target (R2).......Done.
    Suspend RDF link(s) for device(s) in (0077,02)..................Done.
    Merge track tables between source and target in (0077,02).......Started.
    Devices: 0028-0029 ............................................ Merged.
    Merge track tables between source and target in (0077,02).......Done.
    Resume RDF link(s) for device(s) in (0077,02)...................Done.
    Read/Write Enable device(s) in (0077,02) on SA at source (R1)...Done.

The RDF 'Incremental Restore' operation successfully initiated for device
group 'conrdf'.
```

The following query with the `–rdfg 2` option shows the status of each concurrent R2 mirror whose link is represented by RDF group 2. These devices (0000 and 0001) are the concurrent mirrors from which the R1 devices were just restored. The state of the R1s and these R2s is now Synchronized.

```
# symrdf -g conrdf query -rdfg 2 -noprompt

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                 Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW    5231        0 RW 0000 WD    5122        0 S..   SyncInProg
DEV002  0029 RW    7809        0 RW 0001 WD    7754        0 S..   SyncInProg

Total          -------- --------           -------- --------
  Track(s)        13040        0              12876        0
  MB(s)           407.5      0.0              402.3      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The following query shows the status of *all* links of each concurrent SRDF pair. The concurrent R2 mirror from which the restore occurred is now synchronized with its R1 device (state is Synchronized). The other concurrent mirror is still in the Split state.

```
# symrdf -g conrdf query -rdfg all

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124
RDF (RA) Group Number :  1 (00)
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW       0        0 NR 0050 RW       0        0 S..   Split
             RW       0        0 RW 0000 WD       0        0 S..   Synchronized
DEV002  0029 RW       0        0 NR 0051 RW       0        0 S..   Split
             RW       0        0 RW 0001 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

If you have written new data to the concurrent R2 mirror that is still in the Split state and you want this data to become the resynchronized data, you can restore again from the split mirror. In this case, however, include the `–remote` option on the `symrdf restore` command line to indicate that you intend to copy data from the split concurrent mirror to both the R1 device and the other (synchronized) concurrent mirror. The `–rdfg 1` option causes the restore operation to occur now from the concurrent mirror whose link is represented by RDF group 1.

```
# symrdf -g conrdf restore -rdfg 1 -remote -noprompt

An RDF 'Incremental Restore' operation execution is in progress for device
group 'conrdf'. Please wait...

    Write Disable device(s) in (0077,01) on SA at source (R1).......Done.
    Write Disable device(s) in (0077,01) on RA at target (R2).......Done.
    Suspend RDF link(s) for device(s) in (0077,01)..................Done.
    Merge track tables between source and target in (0077,01).......Started.
    Devices: 0028-0029 ............................................ Merged.
    Merge track tables between source and target in (0077,01).......Done.
    Resume RDF link(s) for device(s) in (0077,01)...................Done.
```

```
    Read/Write Enable device(s) in (0077,01) on SA at source (R1)...Done.

The RDF 'Incremental Restore' operation successfully initiated for device
group 'conrdf'.
```

The `symrdf verify` command with the `–rdfg 1` option displays a message every 30 seconds until each concurrent R2 mirror represented by RDF group 1 is synchronized with its R1 device.

```
# symrdf -g conrdf verify -rdfg 1 –i 30 -synchronized

Not all devices in the RDF group 'conrdf' are in the 'Synchronized' state.

Not all devices in the RDF group 'conrdf' are in the 'Synchronized' state.

All devices in the RDF group 'conrdf' are in the 'Synchronized' state.
```

The `symrdf query` command verifies that both links of the concurrent SRDF pairs are now in the Synchronized state.

```
# symrdf -g conrdf query -rdfg all

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124
RDF (RA) Group Number :  1 (00)
Remote Symmetrix ID   : 000185400123
RDF (RA) Group Number :  2 (01)

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  0028 RW       0        0 RW 0050 WD       0        0 S..   Synchronized
             RW       0        0 RW 0000 WD       0        0 S..   Synchronized
DEV002  0029 RW       0        0 RW 0051 WD       0        0 S..   Synchronized
             RW       0        0 RW 0001 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

Currently, in the context of the device group, you can associate a remote BCV with one of the R2 mirrors of a concurrent SRDF pair, but not with both mirrors. Consequently, your device group can include a BCV that belongs to one of the RDF groups, but not both. The following `symbcv associate` command includes in the device group a remotely-associated (`-rdf`) BCV device that belongs to RDF group 1.

```
# symbcv -g conrdf -rdfg 1 associate dev 14 -rdf
```

The `symmir establish` command fully establishes standard device DEV001 with the remotely-associated BCV. When there are more standard devices in a device group than BCVs, specify which standard device you want to establish.

```
# symmir -g conrdf -full establish -rdf DEV001 -noprompt -v

Remote 'Full Establish' operation execution is in progress for device 'DEV001'
in device group 'conrdf'. Please wait...

PAIRING of Standard and BCV devices:

  Devices: 0050(S) - 0014(B) [PAIRED]

STARTING a BCV 'ESTABLISH' operation.

The BCV 'ESTABLISH' operation SUCCEEDED.

Remote 'Full Establish' operation successfully initiated for device 'DEV001'
in device group 'conrdf'.
```

The `symmir query` command with the `–rdf` option shows that RBCV001 (device 14) is now synchronized as a BCV pair with the DEV001 remote R2 mirror (device 50).

```
# symmir -g conrdf query -rdf -noprompt

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400124

                    R E M O T E     S Y M M E T R I X

     Standard Device                   BCV Device                 State
------------------------- ------------------------------------ ------------
                    Inv.                                  Inv.
Logical        Sym  Tracks Logical               Sym      Tracks STD <=> BCV
------------------------- ------------------------------------ ------------

DEV001         0050      0 RBCV001              0014 *         0 Synchronized

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

Legend:

(*): The paired BCV device is associated with this group.
```

The following `symbcv disassociate` command disassociates BCV device 14 from the device group. The BCV pair remains synchronized even though it is no longer under the control of the device group (see footnote on page 19).

```
# symbcv -g conrdf -rdfg 1 disassociate dev 14 -rdf
```

The following `symbcv associate` command includes in the device group a remotely-associated BCV device that belongs to RDF group 2 (that is, this BCV resides on the other remote Symmetrix system).

```
# symbcv -g conrdf -rdfg 2 associate dev 61 -rdf
```

The `symmir establish` command fully establishes standard device DEV001 with remotely-associated BCV 61. When there are more standard devices in a device group than BCVs, specify which standard device you want to establish. This BCV is now the only BCV device under the control of the device group.

```
# symmir -g conrdf -full establish -rdf DEV001 -noprompt -v

Remote 'Full Establish' operation execution is in progress for device 'DEV001'
in device group 'conrdf'. Please wait...

PAIRING of Standard and BCV devices:

  Devices: 0000(S) - 0061(B) [PAIRED]

STARTING a BCV 'ESTABLISH' operation.

The BCV 'ESTABLISH' operation SUCCEEDED.

Remote 'Full Establish' operation successfully initiated for device 'DEV001'
in device group 'conrdf'.
```

The `symmir query` command with the `–rdf` option shows that RBCV001 (device 0061) is now synchronized as a BCV pair with DEV001's other remote R2 mirror (device 0000).

```
# symmir -g conrdf query -rdf -noprompt

Device Group (DG) Name: conrdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185400077
Remote Symmetrix ID   : 000185400123

                    R E M O T E      S Y M M E T R I X

     Standard Device                   BCV Device                 State
------------------------- ----------------------------------- ------------
                    Inv.                                 Inv.
Logical        Sym  Tracks Logical               Sym      Tracks STD <=> BCV
------------------------- ----------------------------------- ------------

DEV001         0000      0 RBCV001             0061 *         0 Synchronized

Total                ------                                -------
  Track(s)              0                                      0
  MB(s)               0.0                                    0.0
```

Example 3: Creating Dynamic SRDF Pairs

This example uses the following Symmetrix systems to create dynamic SRDF pairs from non-SRDF devices that are configured for dynamic SRDF capability:

- Local Source Symmetrix (sid 810): RDF-capable standard devices 106, 10A, and 10F
- Remote Target Symmetrix (sid 506): RDF-capable standard devices B7, BF, and C5

With Enginuity version 5568 and higher, the `symdev list` command with the `–dynamic` option displays devices configured for dynamic RDF capability. This command displays devices that have been created as, or are capable of being created as, dynamic SRDF pairs using the `symrdf createpair` command. When combined with the `–r1` option, `symdev list -dynamic` displays devices configured for dynamic R1/R2[11] and R1-only; when combined with the `–r2` option, the command displays devices configured for dynamic R1/R2 and R2-only. "RDF1+Mir" in the display indicates devices that have already been created as dynamic RDF devices.

You can use this same command with the `–dynamic` and `–r2` options to list RDF-capable devices on the remote Symmetrix (sid 506) and choose devices there that can be paired as dynamic R2 devices.

```
# symdev list -dynamic -sid 810 –r1

Symmetrix ID: 000185500810

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

00F2 /dev/rdsk/emcpower224c  04A:0 15B:C4 Unprotected   N/Grp'd      RW    2063
00F5 /dev/rdsk/emcpower227c  04A:0 01A:D1 Unprotected   N/Grp'd      RW    2063
00FA /dev/rdsk/emcpower239c  04A:0 15A:C4 Unprotected   N/Grp'd      RW    2063
0106 /dev/rdsk/emcpower235c  04A:0 01B:D3 Unprotected   N/Grp'd      RW    2063
010A /dev/rdsk/emcpower237c  04A:0 15A:D3 Unprotected   N/Grp'd      RW    2063
010F /dev/rdsk/emcpower240c  04A:0 02B:C2 Unprotected   N/Grp'd      RW    2063
0145 Not Visible             04A:0 01A:C1 RDF1+Mir      N/Grp'd      RW    2063
0146 Not Visible             04A:0 15A:C1 RDF1+Mir      N/Grp'd      RW    2063
0147 Not Visible             04A:0 02A:C1 RDF1+Mir      N/Grp'd      RW    2063
0148 Not Visible             04A:0 15A:D1 RDF1+Mir      N/Grp'd      RW    2063
0149 Not Visible             04A:0 02A:D1 3-Way Mir     N/Grp'd      RW    2063
```

The following command illustrates the use of the vi text editor to create a text file named "devices." As was done here, you can enter into the file those Symmetrix device names that will constitute the dynamic SRDF pairs. The R1 devices are listed in the first column, and the R2 devices (B7, BF, and C5) chosen from the remote Symmetrix are listed in the second column on the same line as their respective R1 source.

```
# vi devices

10A  B7
10F  BF
106  C5
```

[11] Devices intended for dynamic RDF swap must be configured with the `dyn_rdf` attribute, which makes a device capable of being both a dynamic R1 device and a dynamic R2 device (refer to the white paper *Using the SYMCLI Configuration Manager*, P/N 300-000-475).

The `symrdf createpair` command executes the file called "devices" that defines the dynamic SRDF pairs and specifies that the column-1 devices in the file are RDF1 type devices on the local Symmetrix (sid 810). Communication is via RDF group 2. The `–invalidate r2` option invalidates all tracks on the R2 devices in preparation for a subsequent establish operation. The `–g` option creates a device group named "drdf" and adds the dynamic SRDF pairs to the group.

```
# symrdf createpair -file devices -sid 810 -rdfg 2 -invalidate r2 -noprompt \
                                                          -type rdf1 –g drdf

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

    Create RDF Pair.....................................................Done.
    Mark target device(s) in (0810,02) to refresh from source.......Started.
    Device: 00B7 .......................................................Marked.
    Device: 00BF .......................................................Marked.
    Device: 00C5 .......................................................Marked.
    Mark target device(s) in (0810,02) to refresh from source.......Done.
    Mark target device(s) in (0810,02) for full copy from source....Started.
    Device: 0106 .......................................................Marked.
    Device: 010A .......................................................Marked.
    Device: 010F .......................................................Marked.
    Mark target device(s) in (0810,02) for full copy from source....Done.

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

The `symrdf query` command shows the status of the dynamic SRDF pairs in the device group (drdf). All three pairs are in the Suspended state.

```
# symrdf query -g drdf

Device Group (DG) Name: drdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185500810

       Source (R1) View                 Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                  LI      ST
Standard      A                   N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  010A WD       0    66000 NR 00B7 WD       0        0 S..   Suspended
DEV002  010F WD       0    66000 NR 00BF WD       0        0 S..   Suspended
DEV003  0106 WD       0    66000 NR 00C5 WD       0        0 S..   Suspended

Total           ------   ------             ------   ------
  Track(s)           0   198000                  0        0
  MB(s)            0.0   6187.0                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

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

```
# symrdf establish -g drdf -noprompt

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

    Suspend RDF link(s).....................................Done.
    Read/Write Enable device(s) on SA at source (R1).......Done.
    Resume RDF link(s)......................................Done.

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

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

```
# symrdf query -g drdf

Device Group (DG) Name: drdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185500810

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
            ST                  LI      ST
Standard     A                   N       A
Logical      T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev  E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
-------------------------------- -- ------------------------ ----- ------------

DEV001  010A RW       0    59491 RW 00B7 WD       0        0 S..   SyncInProg
DEV002  010F RW       0    61732 RW 00BF WD       0        0 S..   SyncInProg
DEV003  0106 RW       0    64059 RW 00C5 WD       0        0 S..   SyncInProg

Total           ------   ------             ------   ------
  Track(s)           0   185282                  0        0
  MB(s)            0.0   5782.7                0.0      0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf verify` command verifies when the dynamic SRDF pairs have reached the Synchronized state. The ellipsis (.....) represents repetitive output that was omitted.

```
# symrdf verify -g drdf -i 5 -synchronized

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

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

All devices in the RDF group 'drdf' are in the 'Synchronized' state.
```

Another query confirms that the SRDF pairs are now in the Synchronized state.

```
# symrdf query -g drdf

Device Group (DG) Name: drdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185500810

       Source (R1) View                 Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------

DEV001  010A RW       0        0 RW 00B7 WD       0        0 S..   Synchronized
DEV002  010F RW       0        0 RW 00BF WD       0        0 S..   Synchronized
DEV003  0106 RW       0        0 RW 00C5 WD       0        0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf split` command splits all dynamic SRDF pairs in the device group.

```
# symrdf split -g drdf -noprompt

An RDF 'Split' operation execution is in progress for device group 'drdf'.
Please wait...

    Suspend RDF link(s).......................................Done.
    Read/Write Enable device(s) on RA at target (R2)..........Done.

The RDF 'Split' operation successfully executed for device group 'drdf'.
```

The `symrdf query` command confirms that the SRDF pairs are in the Split state.

```
# symrdf query -g drdf

Device Group (DG) Name: drdf
DG's Type             : RDF1
DG's Symmetrix ID     : 000185500810

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
            ST                    LI      ST
Standard     A                     N       A
Logical      T  R1 Inv   R2 Inv  K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev  E  Tracks   Tracks  S Dev   E  Tracks   Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------

DEV001  010A RW       0        0 NR 00B7 RW       0        0 S..   Split
DEV002  010F RW       0        0 NR 00BF RW       0        0 S..   Split
DEV003  0106 RW       0        0 NR 00C5 RW       0        0 S..   Split

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

Once the pairs' link state is Not Ready (NR), you can use the `symrdf deletepair` command to cancel the dynamic SRDF pairings defined in the device group and delete this pairing information from the host's SYMAPI database file. This operation also changes the type of the device group from RDF1 to REGULAR; the devices in the device group are changed from R1 devices to standard devices.

```
# symrdf deletepair -g drdf -noprompt

An RDF 'Delete Pair' operation execution is in progress for device group 'drdf'.
Please wait...

    Delete RDF Pair.............................................Done.

The RDF 'Delete Pair' operation successfully executed for device group 'drdf'.
```

Attempting to perform a `symrdf query` on the device group results in an output confirming that this device group is no longer an RDF1 type group as a result of the `symrdf deletepair` command.

```
# symrdf query -g drdf

Device group 'drdf' is not an RDF group.
```

The `symld list` command on the device group shows that the device group type was changed to REGULAR and that the same devices that had been created as dynamic R1 type devices have returned to being standard devices (although still configured as RDF-capable). These devices lost their SRDF characteristics as a result of the `symrdf deletepair` command.

```
# symld list -g drdf

Device Group (DG) Name: drdf
DG's Type             : REGULAR
DG's Symmetrix ID     : 000185500810

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

DEV001      emcpower237c       010A   04A:0 15A:D3 Unprotected       WD    2063
DEV002      emcpower240c       010F   04A:0 02B:C2 Unprotected       WD    2063
DEV003      emcpower235c       0106   04A:0 01B:D3 Unprotected       WD    2063
```

Example 4: Creating a Dynamic RDF Group

The hardware setup consists of two Symmetrix units (sid 6190 and sid 0257) that are connected to each other and to two remote Symmetrix units (sid 6202 and sid 0254).

The `symcfg list` command displays those Symmetrix units that are visible to this host. Note that two Symmetrix units are running Enginuity version 5568, and two are running version 5669. Creating a dynamic RDF group is possible only for Symmetrix units running version 5669 or higher.

```
# symcfg list

                                S Y M M E T R I X

                                          Mcode    Cache      Num Phys  Num Symm
SymmID          Attachment  Model         Version  Size (MB)  Devices   Devices

000000006190    Local       DMX2000P      5669       20480     100       396
000184600257    Local       8230          5568       16384      79       483
000000006202    Remote      DMX2000P      5669       20480       0       534
000184600254    Remote      8230          5568       16384       0       504
```

The `symcfg list –ra all` command displays the RDF (RA) groups of all connected Symmetrix units (one or two hops away) that are accessible through RDF links. The `–switched` option displays whether the RDF group type is dynamic (Enginuity version 5669 or higher) or static. If you query a Symmetrix running Enginuity version 5569 or higher, a group's label name is displayed (the default is RDFDVGROUP). Symmetrix 6190 has multiple links to remote Symmetrix 6202 and a single link to local Symmetrix 0257.

```
# symcfg list -ra all -switched

Symmetrix ID: 000000006190 (Local)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local                Group                    Remote
    -------------------  ------------------  ---------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID        Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-14A 14A  60 (3B)  Dynamic DYNGRP60    000000006202 RF-14A 14A  60 (3B)
                55 (36)  Static  DYNGRP55    000000006202 RF-14A 14A   4 (03)
                 1 (00)  Dynamic DYNGRP1     000000006202 RF-14A 14A   1 (00)

    RF-14B 14B   8 (07)  Static  HOUSTON     000000006202 RF-14C 14C   9 (08)
                21 (14)  Static  RDFDVGROUP  000000006202 RF-14C 14C  18 (11)

    RF-14C 14C   5 (04)  Static  RDFDVGROUP  000000006202 RF-14D 14D  49 (30)

    RF-14D 14D   6 (05)  Static  HOPKINTON   000184600257 RF-16A 16A   4 (D)
```

```
Symmetrix ID: 000184600257 (Local)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-3A  03A   2 (B)   Static  -           000000006202 RF-14A 14A   3 (02)

    RF-16A 16A   4 (D)   Static  -           000000006190 RF-14D 14D   6 (05)

    RF-3B  03B   5 (E)   Static  -           000184600254 RF-16B 16B   8 (H)
                 7 (G)   Static  -           000184600254 RF-16B 16B   3 (C)

    RF-16B 16B   -       Static  -           -            -      -     -

Symmetrix ID: 000000006202 (Remote)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-14A 14A  60 (3B)  Dynamic DYNGRP60    000000006190 RF-14A 14A  60 (3B)
                 3 (02)  Static  RDFDVGROUP  000184600257 RF-3A  03A   2 (B)
                 4 (03)  Static  RDFDVGROUP  000000006190 RF-14A 14A  55 (36)
                 1 (00)  Dynamic DYNGRP1     000000006190 RF-14A 14A   1 (00)

    RF-14B 14B  10 (09)  Static  RDFDVGROUP  000184600254 RF-16A 16A   2 (B)
                 5 (04)  Static  RDFDVGROUP  000184600254 RF-16A 16A  10 (J)

    RF-14C 14C   9 (08)  Static  RDFDVGROUP  000000006190 RF-14B 14B   8 (07)
                18 (11)  Static  RDFDVGROUP  000000006190 RF-14B 14B  21 (14)

    RF-14D 14D  49 (30)  Static  RDFDVGROUP  000000006190 RF-14C 14C   5 (04)

Symmetrix ID: 000184600254 (Remote)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-16A 16A   2 (B)   Static  -           000000006202 RF-14B 14B  10 (09)
                10 (J)   Static  -           000000006202 RF-14B 14B   5 (04)

    RF-16B 16B   8 (H)   Static  -           000184600257 RF-3B  03B   5 (E)
                 3 (C)   Static  -           000184600257 RF-3B  03B   7 (G)
```

The following `symrdf addgrp` command creates a dynamic RDF group that represents another RDF link between Symmetrix 6190 and Symmetrix 6202. It adds dynamic RDF group 63 on the local Symmetrix 6190, and RDF group 63 on the remote Symmetrix 6202. The command requires that you specify a group label (DYNGRP63 in this case) that can be used when modifying or deleting the group. Creation of the local and remote RDF groups includes director 14A from both the local and remote Symmetrix unit. It is not necessary that the RDF group number or the director on the local and remote Symmetrix units be the same.

However, it is important to be aware of your network topology when creating dynamic RDF groups between two Symmetrix units. To create a dynamic RDF link (a connection) between directors, the director end points must be able to see each other through the Fibre Channel fabric. For example, a dynamic RDF link can be created between director 14A on Symmetrix 6190 and director 14D on Symmetrix 6202 only if the Fibre Channel zoning is set up so that the two directors can see each other through the fabric.

```
# symrdf addgrp -label DYNGRP63 -rdfg 63 -sid 90 -dir 14a \
                -remote_rdfg 63 -remote_sid 02 -remote_dir 14a

  Successfully Added Dynamic RDF Group 'DYNGRP63' for Symm: 000000006190
```

Another `symcfg list –ra all` command with the `–switched` option verifies that RDF group 63 (DYNGRP63) has been added to both the local and remote Symmetrix units. A Symmetrix unit running Enginuity version 5669 or higher can have up to 64 RDF groups, each group having its hexadecimal value[12] in parenthesis. Symmetrix 0257 and 0254, which are running Enginuity version 5568, always display Group Type as “Static” because dynamic RDF groups are valid only for version 5669 and higher.

```
# symcfg list -ra all -switched

Symmetrix ID: 000000006190 (Local)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    -------------------  ------------------  ---------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-14A 14A  60 (3B)  Dynamic DYNGRP60    000000006202 RF-14A 14A  60 (3B)
                55 (36)  Static  DYNGRP55    000000006202 RF-14A 14A   4 (03)
                 1 (00)  Dynamic DYNGRP1     000000006202 RF-14A 14A   1 (00)
                63 (3E)  Dynamic DYNGRP63    000000006202 RF-14A 14A  63 (3E)

    RF-14B 14B   8 (07)  Static  HOUSTON     000000006202 RF-14C 14C   9 (08)
                21 (14)  Static  RDFDVGROUP  000000006202 RF-14C 14C  18 (11)

    RF-14C 14C   5 (04)  Static  RDFDVGROUP  000000006202 RF-14D 14D  49 (30)

    RF-14D 14D   6 (05)  Static  HOPKINTON   000184600257 RF-16A 16A   4 (D)
```

[12] Prior to Enginuity version 5669, the maximum number of RDF groups was 16, and groups created under those versions are displayed as letters A through P. For version 5669 and higher, the maximum number is 64, and each group is displayed as a hex value that is one less than its decimal value (internal to the Symmetrix, RDF groups are 0-based; from the SYMCLI point of view, they are 1-based).

```
Symmetrix ID: 000184600257 (Local)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-3A  03A   2 (B)   Static  -           000000006202 RF-14A 14A   3 (02)

    RF-16A 16A   4 (D)   Static  -           000000006190 RF-14D 14D   6 (05)

    RF-3B  03B   5 (E)   Static  -           000184600254 RF-16B 16B   8 (H)
                 7 (G)   Static  -           000184600254 RF-16B 16B   3 (C)

    RF-16B 16B   -       Static  -           -            -      -     -

Symmetrix ID: 000000006202 (Remote)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-14A 14A  60 (3B)  Dynamic DYNGRP60    000000006190 RF-14A 14A  60 (3B)
                 3 (02)  Static  RDFDVGROUP  000184600257 RF-3A  03A   2 (B)
                 4 (03)  Static  RDFDVGROUP  000000006190 RF-14A 14A  55 (36)
                 1 (00)  Dynamic DYNGRP1     000000006190 RF-14A 14A   1 (00)
                63 (3E)  Dynamic DYNGRP63    000000006190 RF-14A 14A  63 (3E)

    RF-14B 14B  10 (09)  Static  RDFDVGROUP  000184600254 RF-16A 16A   2 (B)
                 5 (04)  Static  RDFDVGROUP  000184600254 RF-16A 16A  10 (J)

    RF-14C 14C   9 (08)  Static  RDFDVGROUP  000000006190 RF-14B 14B   8 (07)
                18 (11)  Static  RDFDVGROUP  000000006190 RF-14B 14B  21 (14)

    RF-14D 14D  49 (30)  Static  RDFDVGROUP  000000006190 RF-14C 14C   5 (04)

Symmetrix ID: 000184600254 (Remote)

            S Y M M E T R I X    R D F    D I R E C T O R S

          Local              Group                    Remote
    ------------------  ------------------  --------------------------------
    Ident  Symb RA Grp   Type    Name        SymmID       Ident  Symb RA Grp
    ------ ---- -------  ------- ----------  ------------ ------ ---- -------

    RF-16A 16A   2 (B)   Static  -           000000006202 RF-14B 14B  10 (09)
                10 (J)   Static  -           000000006202 RF-14B 14B   5 (04)

    RF-16B 16B   8 (H)   Static  -           000184600257 RF-3B  03B   5 (E)
                 3 (C)   Static  -           000184600257 RF-3B  03B   7 (G)
```

Example 5: Operating with SRDF Asynchronous Replication

This example is performed using Solutions Enabler version 5.3. The hardware setup consists of a host connected to a source Symmetrix (sid 6163) running Enginuity version 5670 and remotely connected via RDF links to a target Symmetrix (sid 6201) that is also running version 5670. RDF (RA) group number 3 has been configured to provide SRDF/A operations.

The `symrdf list` command with the `–rdfa` option displays all devices that are configured for SRDF/A operation. The "G" column indicates that RDF group number 3 is the SRDF/A-configured group. Devices in this type of RDF group have to be either all R1 devices or all R2 devices.

```
# symrdf list -rdfa

Symmetrix ID: 000000006163

                                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  MDA   Tracks   Tracks Dev RDev Pair
---- ----  ------ --------- ----- -------  ------- --- ---- -------------
00F2 00E6   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F3 00E7   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F4 00E8   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F5 00E9   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F6 00EA   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F7 00EB   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F8 00EC   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00F9 00ED   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FA 00EE   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FB 00EF   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FC 0104   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FD 0105   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FE 0106   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
00FF 0107   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
0100 0108   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized
0101 0109   R1:3  RW RW RW   S..        0        0 RW  WD   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symdg` command creates an RDF1 type device group named AsyncGrp1. The `symld` command adds all devices from RDF group 3 to the device group. All devices in RDF group 3 must be managed together using async replication; no subset of this group can be managed using async replication.

```
# symdg create AsyncGrp1 -type rdf1
# symld -g AsyncGrp1 addall -rdfg 3
```

The `symrdf query` command displays the status of the SRDF pairs in the device group. Currently the pairs are in the Synchronized state and running with Synchronous (S) replication. As is shown later in this example, you can include the `–rdfa` option to display SRDF/A information such as the session number, cycle number, and session status (which is currently inactive).

```
# symrdf -g AsyncGrp1 query

Device Group (DG) Name             : AsyncGrp1
DG's Type                          : RDF1
DG's Symmetrix ID                  : 000000006163

       Source (R1) View                Target (R2) View     MODES
--------------------------------    ------------------------ ----- ------------
             ST                   LI      ST
Standard      A                    N       A
Logical       T  R1 Inv   R2 Inv   K       T  R1 Inv   R2 Inv       RDF Pair
Device  Dev   E  Tracks    Tracks   S Dev   E  Tracks    Tracks MDA   STATE
--------------------------------- -- ------------------------ ----- ------------
DEV001  00F2 RW       0         0 RW 00E6 WD       0         0 S..   Synchronized
DEV002  00F3 RW       0         0 RW 00E7 WD       0         0 S..   Synchronized
DEV003  00F4 RW       0         0 RW 00E8 WD       0         0 S..   Synchronized
DEV004  00F5 RW       0         0 RW 00E9 WD       0         0 S..   Synchronized
DEV005  00F6 RW       0         0 RW 00EA WD       0         0 S..   Synchronized
DEV006  00F7 RW       0         0 RW 00EB WD       0         0 S..   Synchronized
DEV007  00F8 RW       0         0 RW 00EC WD       0         0 S..   Synchronized
DEV008  00F9 RW       0         0 RW 00ED WD       0         0 S..   Synchronized
DEV009  00FA RW       0         0 RW 00EE WD       0         0 S..   Synchronized
DEV010  00FB RW       0         0 RW 00EF WD       0         0 S..   Synchronized
DEV011  00FC RW       0         0 RW 0104 WD       0         0 S..   Synchronized
DEV012  00FD RW       0         0 RW 0105 WD       0         0 S..   Synchronized
DEV013  00FE RW       0         0 RW 0106 WD       0         0 S..   Synchronized
DEV014  00FF RW       0         0 RW 0107 WD       0         0 S..   Synchronized
DEV015  0100 RW       0         0 RW 0108 WD       0         0 S..   Synchronized
DEV016  0101 RW       0         0 RW 0109 WD       0         0 S..   Synchronized

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

The `symrdf set mode async` command sets the method of replication to Asynchronous for the SRDF/A devices in the device group. At this point, however, consistency protection is still disabled.

```
# symrdf -g AsyncGrp1 set mode async -noprompt

An RDF Set 'Asynchronous Mode' operation execution is in progress for device
group 'AsyncGrp1'. Please wait...

The RDF Set 'Asynchronous Mode' operation successfully executed for device
group 'AsyncGrp1'.
```

The `symrdf enable` command enables consistency protection for the SRDF/A devices in the device group.

```
# symrdf -g AsyncGrp1 enable -noprompt

An RDF 'Enable' operation execution is in progress for device group 'AsyncGrp1'.
Please wait...

The RDF 'Enable' operation successfully executed for device group 'AsyncGrp1'.
```

The `symdg show` display verifies in its “RDFA Information” section that SRDF/A session is active and that the consistency state is enabled.

```
# symdg show AsyncGrp1

Group Name:  AsyncGrp1

    Group Type                                     : RDF1     (RDFA)
    Valid                                          : Yes
    Symmetrix ID                                   : 000000006163
    Group Creation Time                            : Mon Jun 30 14:02:12 2003
    Vendor ID                                      : EMC Corp
    Application ID                                 : SYMCLI

    Number of STD Devices in Group                 :   16
    Number of Associated GK's                      :    0
    Number of Locally-associated BCV's             :    0
    Number of Locally-associated VDEV's            :    0
    Number of Remotely-associated BCV's (STD RDF):    0
    Number of Remotely-associated BCV's (BCV RDF):    0
    Number of Remotely-assoc'd RBCV's (RBCV RDF) :    0
```

```
Standard (STD) Devices (16):
    {
    --------------------------------------------------------------------
                                                    Sym                 Cap
    LdevName              PdevName                  Dev  Att. Sts       (MB)
    --------------------------------------------------------------------
    DEV001                /dev/rdsk/emcpower99c     00F2       RW       1031
    DEV002                /dev/rdsk/emcpower100c    00F3       RW       1031
    DEV003                /dev/rdsk/emcpower101c    00F4       RW       1031
    DEV004                /dev/rdsk/emcpower102c    00F5       RW       1031
    DEV005                /dev/rdsk/emcpower103c    00F6       RW       1031
    DEV006                /dev/rdsk/emcpower104c    00F7       RW       1031
    DEV007                /dev/rdsk/emcpower105c    00F8       RW       1031
    DEV008                /dev/rdsk/emcpower106c    00F9       RW       1031
    DEV009                /dev/rdsk/emcpower107c    00FA       RW       1031
    DEV010                /dev/rdsk/emcpower108c    00FB       RW       1031
    DEV011                /dev/rdsk/emcpower109c    00FC       RW       1031
    DEV012                /dev/rdsk/emcpower110c    00FD       RW       1031
    DEV013                /dev/rdsk/emcpower111c    00FE       RW       1031
    DEV014                /dev/rdsk/emcpower112c    00FF       RW       1031
    DEV015                /dev/rdsk/emcpower113c    0100       RW       1031
    DEV016                /dev/rdsk/emcpower114c    0101       RW       1031
    }

Device Group RDF Information
    {
    RDF Type                                : R1
    RDF (RA) Group Number                   : 3              (02)

    Remote Symmetrix ID                     : 000000006201

    R2 Device Is Larger Than The R1 Device : False

    RDF Mode                                : Asynchronous
    RDF Adaptive Copy                       : Disabled
    RDF Adaptive Copy Write Pending State   : N/A
    RDF Adaptive Copy Skew (Tracks)         : 65535

    RDF Device Domino                       : Disabled

    RDF Link Configuration                  : Fibre
    RDF Link Domino                         : Disabled
    Prevent Automatic RDF Link Recovery     : Disabled
    Prevent RAs Online Upon Power ON        : Enabled

    Device RDF Status                       : Ready          (RW)

    Device RA Status                        : Ready          (RW)
    Device Link Status                      : Ready          (RW)

    Device Suspend State                    : N/A
    Device Consistency State                : Disabled
    RDF R2 Not Ready If Invalid             : Enabled

    Device RDF State                        : Ready          (RW)
    Remote Device RDF State                 : Write Disabled (WD)
```

```
RDF Pair State (  R1 <===> R2 )        : Consistent

Number of R1 Invalid Tracks            : 0
Number of R2 Invalid Tracks            : 0

RDFA Information:
    {
    Session Number                     : 0
    Cycle Number                       : 5
    Number of Devices in the Session   : 16
    Session Status                     : Active

    Session Consistency State          : Enabled
    Tracks not Committed to the R2 Side: 0
    Average Cycle Time                 : 00:00:30
    Time that R2 is behind R1          : 00:00:46
    }
}
```

In the RDFA information display above, “Tracks not Committed to the R2 Side” indicates all R1 tracks owed to the R2 side that have not been committed to the R2 side yet. The “Average Cycle Time” is 30 seconds. The “Time that R2 is behind R1” indicates that data on the R2 side is currently 46 seconds behind the R1 side.

The `symrdf verify` command checks the state of the SRDF pairs and verifies that they are in the Consistent state.

```
# symrdf -g AsyncGrp1 verify -consistent

All devices in the RDF group 'AsyncGrp1' are in the 'Consistent' state.
```

The `symrdf suspend` command with the `–force` option trips the device group, making the devices NR on the link and disabling SRDF/A consistency protection. Suspending is useful if you need to trip the device group but also maintain the consistency of the R2 database copy with the production copy on the R1 side. The `–force` option is required here to ensure that you really want to stop SRDF/A operation and end consistency protection.

```
# symrdf -g AsyncGrp1 suspend -noprompt -force

An RDF 'Suspend' operation execution is in progress for device group 'AsyncGrp1'.
Please wait...

    Suspend RDF link(s).....................................Done.

The RDF 'Suspend' operation successfully executed for device group 'AsyncGrp1'.
```

The `symrdf query` command with the `–rdfa` option shows that the SRDF/A session status is now inactive and that the SRDF pairs are in the Suspended state. Normally there would be invalid tracks on the R1 side to indicate continuing I/O on the R1 side, but currently there is no I/O.

```
# symrdf -g AsyncGrp1 query -rdfa

Device Group (DG) Name             : AsyncGrp1
DG's Type                          : RDF1
DG's Symmetrix ID                  : 000000006163
RDFA Session Number                : 0
RDFA Cycle Number                  : 0
RDFA Session Status                : Inactive
RDFA Avg Cycle Time                : 00:00:00
Time that R2 is behind R1          : 00:00:00

       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 MDAC  STATE
--------------------------------- -- ------------------------ ----- ------------
DEV001  00F2 RW       0        0 NR 00E6 WD       0        0 A..-  Suspended
DEV002  00F3 RW       0        0 NR 00E7 WD       0        0 A..-  Suspended
DEV003  00F4 RW       0        0 NR 00E8 WD       0        0 A..-  Suspended
DEV004  00F5 RW       0        0 NR 00E9 WD       0        0 A..-  Suspended
DEV005  00F6 RW       0        0 NR 00EA WD       0        0 A..-  Suspended
DEV006  00F7 RW       0        0 NR 00EB WD       0        0 A..-  Suspended
DEV007  00F8 RW       0        0 NR 00EC WD       0        0 A..-  Suspended
DEV008  00F9 RW       0        0 NR 00ED WD       0        0 A..-  Suspended
DEV009  00FA RW       0        0 NR 00EE WD       0        0 A..-  Suspended
DEV010  00FB RW       0        0 NR 00EF WD       0        0 A..-  Suspended
DEV011  00FC RW       0        0 NR 0104 WD       0        0 A..-  Suspended
DEV012  00FD RW       0        0 NR 0105 WD       0        0 A..-  Suspended
DEV013  00FE RW       0        0 NR 0106 WD       0        0 A..-  Suspended
DEV014  00FF RW       0        0 NR 0107 WD       0        0 A..-  Suspended
DEV015  0100 RW       0        0 NR 0108 WD       0        0 A..-  Suspended
DEV016  0101 RW       0        0 NR 0109 WD       0        0 A..-  Suspended

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
 C(onsistency State): X = Enabled, . = Disabled, - = N/A
```

The `symrdf resume` command resumes the RDF links between the SRDF pairs in the device group and I/O traffic between the R1 devices and their paired R2 devices. The SRDF/A session is automatically activated again.

```
# symrdf -g AsyncGrp1 resume -noprompt

An RDF 'Resume' operation execution is in progress for device group 'AsyncGrp1'.
Please wait...

    Resume RDF link(s).....................................Done.

The RDF 'Resume' operation successfully executed for device group 'AsyncGrp1'.
```

The `symrdf enable` command enables consistency protection again for the SRDF/A devices in the device group.

```
# symrdf -g AsyncGrp1 enable -noprompt

An RDF 'Enable' operation execution is in progress for device group 'AsyncGrp1'.
Please wait...

The RDF 'Enable' operation successfully executed for device group 'AsyncGrp1'.
```

The `symrdf query` command verifies that the SRDF/A session is active again and that the devices are in the Consistent state. Using the `–rdfa` option results in the SRDF/A information being displayed.

```
# symrdf -g AsyncGrp1 query -rdfa

Device Group (DG) Name             : AsyncGrp1
DG's Type                          : RDF1
DG's Symmetrix ID                  : 000000006163
RDFA Session Number                : 0
RDFA Cycle Number                  : 6
RDFA Session Status                : Active
RDFA Avg Cycle Time                : 00:00:30
Time that R2 is behind R1          : 00:00:38

       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 MDAC  STATE
-------------------------------- -- ------------------------ ----- ------------
DEV001  00F2 RW       0        0 RW 00E6 WD       0        0 A..X  Consistent
DEV002  00F3 RW       0        0 RW 00E7 WD       0        0 A..X  Consistent
DEV003  00F4 RW       0        0 RW 00E8 WD       0        0 A..X  Consistent
DEV004  00F5 RW       0        0 RW 00E9 WD       0        0 A..X  Consistent
DEV005  00F6 RW       0        0 RW 00EA WD       0        0 A..X  Consistent
DEV006  00F7 RW       0        0 RW 00EB WD       0        0 A..X  Consistent
DEV007  00F8 RW       0        0 RW 00EC WD       0        0 A..X  Consistent
DEV008  00F9 RW       0        0 RW 00ED WD       0        0 A..X  Consistent
DEV009  00FA RW       0        0 RW 00EE WD       0        0 A..X  Consistent
DEV010  00FB RW       0        0 RW 00EF WD       0        0 A..X  Consistent
DEV011  00FC RW       0        0 RW 0104 WD       0        0 A..X  Consistent
DEV012  00FD RW       0        0 RW 0105 WD       0        0 A..X  Consistent
```

```
DEV013  00FE RW       0        0 RW 0106 WD       0        0 A..X  Consistent
DEV014  00FF RW       0        0 RW 0107 WD       0        0 A..X  Consistent
DEV015  0100 RW       0        0 RW 0108 WD       0        0 A..X  Consistent
DEV016  0101 RW       0        0 RW 0109 WD       0        0 A..X  Consistent

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

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
 C(onsistency State): X = Enabled, . = Disabled, - = N/A
```

Example 6: Using a Composite Group to Contol SRDF Pairs

This example is performed using Solutions Enabler version 5.4. The hardware setup consists of a Solaris host connected to two source Symmetrix units (Symmetrix 000187900035 and Symmetrix 000000003143). The example builds a composite group with source R1 devices from both Symmetrix units. It then demonstrates how to enable consistency protection for the composite group and perform a suspend operation on the group. For more examples using SRDF consistency protection, refer to the white paper *Using SYMCLI to Implement RDF Consistency Protection with SRDF Family Products* (P/N 300-000-284).

The `symcg create` command creates an RDF1 type composite group named SRDF on this host. If you intend to enable the group for consistency protection and have not set the SYMAPI_RDF_CG_TO_PPATH variable to ENABLE, you must include the `–ppath` option so that the group is added to PowerPath.

```
# symcg create SRDF -type rdf1 -ppath
```

The following `symcg addall` command adds to the composite group a range of PowerPath standard devices from Symmetrix 000187900035.

```
# symcg -cg SRDF addall dev -range 137:14F -sid 35
```

The following `symcg addall` command adds to the composite group a range of PowerPath standard devices from Symmetrix 000000003143.

```
# symcg -cg SRDF addall dev -range F7:10F -sid 43
```

The `symrdf query` command checks the state of the SRDF pairs. Note that SRDF pairs from one Symmetrix unit are in the Suspended state, while the other Symmetrix unit has synchronized SRDF pairs.

```
# symrdf -cg SRDF query

Composite Group Name    :  SRDF
Composite Group Type    :  RDF1
Number of Symmetrix Units :  2
Number of RDF (RA) Groups :  2

Symmetrix ID               : 000000003143    (Microcode Version: 5267)
Remote Symmetrix ID        : 000000003156    (Microcode Version: 5267)
RDF (RA) Group Number      :  1 (A)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                LI      ST                       C  S
      A                 N       A                       o  u
      T  R1 Inv  R2 Inv  K       T  R1 Inv  R2 Inv       n  s   RDF Pair
Dev   E  Tracks Tracks   S Dev   E  Tracks  Tracks MDA   s  p   STATE
----------------------- -- ----------------------- ----- ------ ------------
00F7 RW       0      46 NR 0062 NR       0       0 S..   .  -   Suspended
00FA RW       0      46 NR 0065 NR       0       0 S..   .  -   Suspended
00FC RW       0      46 NR 0067 NR       0       0 S..   .  -   Suspended
00FD RW       0      46 NR 0068 NR       0       0 S..   .  -   Suspended
00FE RW       0      46 NR 0069 NR       0       0 S..   .  -   Suspended
00FF RW       0      46 NR 006A NR       0       0 S..   .  -   Suspended
0100 RW       0      46 NR 006B NR       0       0 S..   .  -   Suspended
0101 RW       0      46 NR 006C NR       0       0 S..   .  -   Suspended
0102 RW       0      46 NR 006D NR       0       0 S..   .  -   Suspended
```

```
0103 RW       0      46 NR 006E NR       0       0 S..   .  -   Suspended
0104 RW       0      46 NR 006F NR       0       0 S..   .  -   Suspended
0105 RW       0      46 NR 0070 NR       0       0 S..   .  -   Suspended
0106 RW       0      46 NR 0071 NR       0       0 S..   .  -   Suspended
0107 RW       0      46 NR 0072 NR       0       0 S..   .  -   Suspended
0108 RW       0      46 NR 0073 NR       0       0 S..   .  -   Suspended
0109 RW       0      46 NR 0074 NR       0       0 S..   .  -   Suspended
010A RW       0      46 NR 0075 NR       0       0 S..   .  -   Suspended
010B RW       0      46 NR 0076 NR       0       0 S..   .  -   Suspended
010C RW       0      46 NR 0077 NR       0       0 S..   .  -   Suspended
010D RW       0      46 NR 0078 NR       0       0 S..   .  -   Suspended
010E RW       0      46 NR 0079 NR       0       0 S..   .  -   Suspended
010F RW       0      46 NR 007A NR       0       0 S..   .  -   Suspended

Symmetrix ID               : 000187900035    (Microcode Version: 5670)
Remote Symmetrix ID        : 000187900041    (Microcode Version: 5670)
RDF (RA) Group Number      :  1 (00)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                 LI      ST                     C  S
      A                  N       A                     o  u
      T  R1 Inv  R2 Inv  K       T  R1 Inv  R2 Inv      n  s   RDF Pair
Dev   E  Tracks Tracks   S Dev   E  Tracks  Tracks MDA   s  p   STATE
----------------------- -- ------------------------ ----- ------ ------------
0137 RW       0       0 RW 0056 NR       0       0 S..   .  -   Synchronized
013A RW       0       0 RW 0059 NR       0       0 S..   .  -   Synchronized
013C RW       0       0 RW 005B NR       0       0 S..   .  -   Synchronized
013D RW       0       0 RW 005C NR       0       0 S..   .  -   Synchronized
013E RW       0       0 RW 005D NR       0       0 S..   .  -   Synchronized
013F RW       0       0 RW 005E NR       0       0 S..   .  -   Synchronized
0140 RW       0       0 RW 005F NR       0       0 S..   .  -   Synchronized
0141 RW       0       0 RW 0060 NR       0       0 S..   .  -   Synchronized
0142 RW       0       0 RW 0061 NR       0       0 S..   .  -   Synchronized
0143 RW       0       0 RW 0062 NR       0       0 S..   .  -   Synchronized
0144 RW       0       0 RW 0063 NR       0       0 S..   .  -   Synchronized
0145 RW       0       0 RW 0064 NR       0       0 S..   .  -   Synchronized
0146 RW       0       0 RW 0065 NR       0       0 S..   .  -   Synchronized
0147 RW       0       0 RW 0066 NR       0       0 S..   .  -   Synchronized
0148 RW       0       0 RW 0067 NR       0       0 S..   .  -   Synchronized
0149 RW       0       0 RW 0068 NR       0       0 S..   .  -   Synchronized
014A RW       0       0 RW 0069 NR       0       0 S..   .  -   Synchronized
014B RW       0       0 RW 006A NR       0       0 S..   .  -   Synchronized
014C RW       0       0 RW 006B NR       0       0 S..   .  -   Synchronized
014D RW       0       0 RW 006C NR       0       0 S..   .  -   Synchronized
014E RW       0       0 RW 006D NR       0       0 S..   .  -   Synchronized
014F RW       0       0 RW 006E NR       0       0 S..   .  -   Synchronized

Total   ------- -------            ------- -------
Trks          0    1012                  0       0
MBs         0.0    31.6                0.0     0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

```
Legend for STATES:

 Cons(istency State): X = Enabled, M = Mixed, . = Disabled, - = N/A
 Susp(end State)    : X = Online, . = Offline, P = Offline Pending, - = N/A
```

The `symrdf establish` command initiates an incremental establish operation on SRDF pairs in the composite group that are not synchronized (that is, the suspended pairs on Symmetrix 3143).

```
# symrdf -cg SRDF establish -noprompt

An RDF 'Incremental Establish' operation execution is in progress for composite
group 'SRDF'. Please wait...

    Suspend RDF link(s) for device(s) in (3143,01).................Done.
    Resume RDF link(s) for device(s) in (3143,01)..................Not Done.
    Merge track tables between source and target in (3143,01).......Started.
    Devices: 00F7-00F8 ............................................ Merged.
    Device: 00FA .................................................. Merged.
    Devices: 00FC-0101 ............................................ Merged.
    Devices: 0102-0107 ............................................ Merged.
    Devices: 0108-010D ............................................ Merged.
    Devices: 010E-010F ............................................ Merged.
    Merge track tables between source and target in (3143,01).......Done.
    Resume RDF link(s) for device(s) in (3143,01)..................Done.

The RDF 'Incremental Establish' operation successfully initiated for composite
group 'SRDF'.
```

Another `symrdf query` command checks the state of the SRDF pairs and shows that the previously suspended pairs are now in the process of synchronizing (SyncInProg).

```
# symrdf -cg SRDF query

Composite Group Name    :  SRDF
Composite Group Type    :  RDF1
Number of Symmetrix Units :  2
Number of RDF (RA) Groups :  2

Symmetrix ID              : 000000003143    (Microcode Version: 5267)
Remote Symmetrix ID       : 000000003156    (Microcode Version: 5267)
RDF (RA) Group Number     :  1 (A)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                LI      ST                       C  S
      A                 N       A                       o  u
      T  R1 Inv  R2 Inv  K       T  R1 Inv  R2 Inv       n  s   RDF Pair
Dev   E  Tracks Tracks   S Dev   E  Tracks  Tracks MDA   s  p   STATE
----------------------- -- ----------------------- ----- ------ ------------
00F7 RW       0       0 RW 0062 NR       0       0 S..   .  -   Synchronized
00FA RW       0      45 RW 0065 NR       0       0 S..   .  -   SyncInProg
00FC RW       0      46 RW 0067 NR       0       0 S..   .  -   SyncInProg
00FD RW       0      46 RW 0068 NR       0       0 S..   .  -   SyncInProg
00FE RW       0       1 RW 0069 NR       0       0 S..   .  -   SyncInProg
00FF RW       0       1 RW 006A NR       0       0 S..   .  -   SyncInProg
0100 RW       0       1 RW 006B NR       0       0 S..   .  -   SyncInProg
```

```
0101 RW       0       1 RW 006C NR       0       0 S..   .  -   SyncInProg
0102 RW       0      46 RW 006D NR       0       0 S..   .  -   SyncInProg
0103 RW       0      46 RW 006E NR       0       0 S..   .  -   SyncInProg
0104 RW       0       1 RW 006F NR       0       0 S..   .  -   SyncInProg
0105 RW       0      46 RW 0070 NR       0       0 S..   .  -   SyncInProg
0106 RW       0       1 RW 0071 NR       0       0 S..   .  -   SyncInProg
0107 RW       0       1 RW 0072 NR       0       0 S..   .  -   SyncInProg
0108 RW       0      46 RW 0073 NR       0       0 S..   .  -   SyncInProg
0109 RW       0       1 RW 0074 NR       0       0 S..   .  -   SyncInProg
010A RW       0       1 RW 0075 NR       0       0 S..   .  -   SyncInProg
010B RW       0      46 RW 0076 NR       0       0 S..   .  -   SyncInProg
010C RW       0       1 RW 0077 NR       0       0 S..   .  -   SyncInProg
010D RW       0       1 RW 0078 NR       0       0 S..   .  -   SyncInProg
010E RW       0       1 RW 0079 NR       0       0 S..   .  -   SyncInProg
010F RW       0       1 RW 007A NR       0       0 S..   .  -   SyncInProg

Symmetrix ID              : 000187900035    (Microcode Version: 5670)
Remote Symmetrix ID       : 000187900041    (Microcode Version: 5670)
RDF (RA) Group Number     :  1 (00)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                LI      ST                       C  S
      A                 N       A                       o  u
      T  R1 Inv  R2 Inv  K       T  R1 Inv  R2 Inv       n  s   RDF Pair
Dev   E  Tracks Tracks   S Dev   E  Tracks  Tracks MDA   s  p   STATE
----------------------- -- ------------------------ ----- ------ ------------
0137 RW       0       0 RW 0056 NR       0       0 S..   .  -   Synchronized
013A RW       0       0 RW 0059 NR       0       0 S..   .  -   Synchronized
013C RW       0       0 RW 005B NR       0       0 S..   .  -   Synchronized
013D RW       0       0 RW 005C NR       0       0 S..   .  -   Synchronized
013E RW       0       0 RW 005D NR       0       0 S..   .  -   Synchronized
013F RW       0       0 RW 005E NR       0       0 S..   .  -   Synchronized
0140 RW       0       0 RW 005F NR       0       0 S..   .  -   Synchronized
0141 RW       0       0 RW 0060 NR       0       0 S..   .  -   Synchronized
0142 RW       0       0 RW 0061 NR       0       0 S..   .  -   Synchronized
0143 RW       0       0 RW 0062 NR       0       0 S..   .  -   Synchronized
0144 RW       0       0 RW 0063 NR       0       0 S..   .  -   Synchronized
0145 RW       0       0 RW 0064 NR       0       0 S..   .  -   Synchronized
0146 RW       0       0 RW 0065 NR       0       0 S..   .  -   Synchronized
0147 RW       0       0 RW 0066 NR       0       0 S..   .  -   Synchronized
0148 RW       0       0 RW 0067 NR       0       0 S..   .  -   Synchronized
0149 RW       0       0 RW 0068 NR       0       0 S..   .  -   Synchronized
014A RW       0       0 RW 0069 NR       0       0 S..   .  -   Synchronized
014B RW       0       0 RW 006A NR       0       0 S..   .  -   Synchronized
014C RW       0       0 RW 006B NR       0       0 S..   .  -   Synchronized
014D RW       0       0 RW 006C NR       0       0 S..   .  -   Synchronized
014E RW       0       0 RW 006D NR       0       0 S..   .  -   Synchronized
014F RW       0       0 RW 006E NR       0       0 S..   .  -   Synchronized

Total   ------- -------            ------- -------
Trks          0     380                  0       0
MBs         0.0    11.9                0.0     0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
```

```
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off

Legend for STATES:

 Cons(istency State): X = Enabled, M = Mixed, . = Disabled, - = N/A
 Susp(end State)    : X = Online, . = Offline, P = Offline Pending, - = N/A
```

The `symcg show` command displays that the consistency state of the devices is currently Disabled.

```
# symcg show SRDF

Composite Group Name:  SRDF

  Composite Group Type                                : RDF1
  Valid                                               : Yes
  CG in PowerPath                                     : Yes
  CG in GNS                                           : No

  Number of RDF (RA) Groups                           :    2
  Number of STD Devices                               :   44
  Number of BCV's (Locally-associated)                :    0
  Number of VDEV's (Locally-associated)               :    0
  Number of RBCV's (Remotely-associated STD-RDF)      :    0
  Number of BRBCV's (Remotely-associated BCV-RDF)     :    0
  Number of RRBCV's (Remotely-associated RBCV)        :    0

  Number of Symmetrix Units (2):
     {

     1) Symmetrix ID                                    : 000000003143
        Microcode Version                               : 5267
        Number of STD Devices                           :   22
        Number of BCV's (Locally-associated)            :    0
        Number of VDEV's (Locally-associated)           :    0
        Number of RBCV's (Remotely-associated STD_RDF)  :    0
        Number of BRBCV's (Remotely-associated BCV-RDF):    0
        Number of RRBCV's (Remotely-associated RBCV)    :    0

        Number of RDF (RA) Groups (1):
           {

           1) RDF (RA) Group Number  :  1           (A)
              Remote Symmetrix ID    : 000000003156
              Microcode Version      : 5267

              STD Devices (22):
                 {
                 ------------------------------------------------------------
                                          Sym  Device        Consistency   Cap
                 PdevName                 Dev  Config        State         (MB)
                 ------------------------------------------------------------
                 /dev/vx/rdmp/c15t1d24s2 00F7 RDF1          Disabled     12946
                 /dev/vx/rdmp/c15t1d25s2 00FA RDF1          Disabled      8631
                 /dev/vx/rdmp/c15t1d26s2 00FC RDF1          Disabled      4315
                 /dev/vx/rdmp/c15t1d27s2 00FD RDF1          Disabled      4315
                 /dev/vx/rdmp/c15t1d28s2 00FE RDF1          Disabled      4315
```

```
            /dev/vx/rdmp/c15t1d29s2 00FF RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d30s2 0100 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d31s2 0101 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d32s2 0102 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d33s2 0103 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d34s2 0104 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d35s2 0105 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d36s2 0106 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d37s2 0107 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d38s2 0108 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d39s2 0109 RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d40s2 010A RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d41s2 010B RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d42s2 010C RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d43s2 010D RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d44s2 010E RDF1          Disabled      4315
            /dev/vx/rdmp/c15t1d45s2 010F RDF1          Disabled      4315
            }
      }

   2) Symmetrix ID                                  : 000187900035
      Microcode Version                             : 5670
      Number of STD Devices                         :   22
      Number of BCV's (Locally-associated)          :    0
      Number of VDEV's (Locally-associated)         :    0
      Number of RBCV's (Remotely-associated STD_RDF) :    0
      Number of BRBCV's (Remotely-associated BCV-RDF):    0
      Number of RRBCV's (Remotely-associated RBCV)   :    0

      Number of RDF (RA) Groups (1):
         {

         1) RDF (RA) Group Number  :  1           (00)
            Remote Symmetrix ID    : 000187900041
            Microcode Version      : 5670

            STD Devices (22):
               {
               ------------------------------------------------------------
                                       Sym  Device        Consistency   Cap
               PdevName                Dev  Config        State         (MB)
               ------------------------------------------------------------
               /dev/vx/rdmp/c15t2d24s2 0137 RDF1+Mir      Disabled     12946
               /dev/vx/rdmp/c15t2d25s2 013A RDF1+Mir      Disabled      8631
               /dev/vx/rdmp/c15t2d26s2 013C RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d27s2 013D RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d28s2 013E RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d29s2 013F RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d30s2 0140 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d31s2 0141 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d32s2 0142 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d33s2 0143 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d34s2 0144 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d35s2 0145 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d36s2 0146 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d37s2 0147 RDF1+Mir      Disabled      4315
               /dev/vx/rdmp/c15t2d38s2 0148 RDF1+Mir      Disabled      4315
```

```
                /dev/vx/rdmp/c15t2d39s2 0149 RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d40s2 014A RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d41s2 014B RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d42s2 014C RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d43s2 014D RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d44s2 014E RDF1+Mir      Disabled      4315
                /dev/vx/rdmp/c15t2d45s2 014F RDF1+Mir      Disabled      4315
                }
        }
    }
```

The `symcg enable` command enables consistency protection for device pairs in the composite group.

```
# symcg -cg SRDF enable -noprompt

A consistency 'Enable' operation execution is in progress for composite group
'SRDF'. Please wait...

The consistency 'Enable' operation successfully executed for composite group
'SRDF'.
```

Another `symrdf query` command displays all pairs in the Synchronized state and an X in the “Cons” column to indicate that all pairs are now enabled for consistency protection.

```
# symcg -cg SRDF query

Composite Group Name    :  SRDF
Composite Group Type    :  RDF1
Number of Symmetrix Units :  2
Number of RDF (RA) Groups :  2

Symmetrix ID              : 000000003143    (Microcode Version: 5267)
Remote Symmetrix ID       : 000000003156    (Microcode Version: 5267)
RDF (RA) Group Number     :  1 (A)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                 LI      ST                       C  S
      A                  N       A                       o  u
      T  R1 Inv  R2 Inv  K       T  R1 Inv  R2 Inv       n  s   RDF Pair
Dev   E  Tracks Tracks   S Dev   E  Tracks  Tracks MDA   s  p   STATE
----------------------- -- ----------------------- ----- ------ ------------
00F7 RW       0       0 RW 0062 NR       0       0 S..   X  -   Synchronized
00FA RW       0       0 RW 0065 NR       0       0 S..   X  -   Synchronized
00FC RW       0       0 RW 0067 NR       0       0 S..   X  -   Synchronized
00FD RW       0       0 RW 0068 NR       0       0 S..   X  -   Synchronized
00FE RW       0       0 RW 0069 NR       0       0 S..   X  -   Synchronized
00FF RW       0       0 RW 006A NR       0       0 S..   X  -   Synchronized
0100 RW       0       0 RW 006B NR       0       0 S..   X  -   Synchronized
0101 RW       0       0 RW 006C NR       0       0 S..   X  -   Synchronized
0102 RW       0       0 RW 006D NR       0       0 S..   X  -   Synchronized
0103 RW       0       0 RW 006E NR       0       0 S..   X  -   Synchronized
0104 RW       0       0 RW 006F NR       0       0 S..   X  -   Synchronized
0105 RW       0       0 RW 0070 NR       0       0 S..   X  -   Synchronized
0106 RW       0       0 RW 0071 NR       0       0 S..   X  -   Synchronized
0107 RW       0       0 RW 0072 NR       0       0 S..   X  -   Synchronized
0108 RW       0       0 RW 0073 NR       0       0 S..   X  -   Synchronized
```

```
0109 RW       0       0 RW 0074 NR       0       0 S..   X  -   Synchronized
010A RW       0       0 RW 0075 NR       0       0 S..   X  -   Synchronized
010B RW       0       0 RW 0076 NR       0       0 S..   X  -   Synchronized
010C RW       0       0 RW 0077 NR       0       0 S..   X  -   Synchronized
010D RW       0       0 RW 0078 NR       0       0 S..   X  -   Synchronized
010E RW       0       0 RW 0079 NR       0       0 S..   X  -   Synchronized
010F RW       0       0 RW 007A NR       0       0 S..   X  -   Synchronized

Symmetrix ID               : 000187900035    (Microcode Version: 5670)
Remote Symmetrix ID        : 000187900041    (Microcode Version: 5670)
RDF (RA) Group Number      :  1 (00)

   Source (R1) View         Target (R2) View       MODES STATES
----------------------  ------------------------ ----- ------ ------------
     ST                  LI      ST                       C  S
      A                   N       A                       o  u
      T  R1 Inv  R2 Inv   K       T  R1 Inv  R2 Inv        n  s   RDF Pair
Dev   E  Tracks Tracks    S Dev   E  Tracks  Tracks MDA    s  p   STATE
----------------------- -- ------------------------ ----- ------ ------------
0137 RW       0       0 RW 0056 NR       0       0 S..   X  -   Synchronized
013A RW       0       0 RW 0059 NR       0       0 S..   X  -   Synchronized
013C RW       0       0 RW 005B NR       0       0 S..   X  -   Synchronized
013D RW       0       0 RW 005C NR       0       0 S..   X  -   Synchronized
013E RW       0       0 RW 005D NR       0       0 S..   X  -   Synchronized
013F RW       0       0 RW 005E NR       0       0 S..   X  -   Synchronized
0140 RW       0       0 RW 005F NR       0       0 S..   X  -   Synchronized
0141 RW       0       0 RW 0060 NR       0       0 S..   X  -   Synchronized
0142 RW       0       0 RW 0061 NR       0       0 S..   X  -   Synchronized
0143 RW       0       0 RW 0062 NR       0       0 S..   X  -   Synchronized
0144 RW       0       0 RW 0063 NR       0       0 S..   X  -   Synchronized
0145 RW       0       0 RW 0064 NR       0       0 S..   X  -   Synchronized
0146 RW       0       0 RW 0065 NR       0       0 S..   X  -   Synchronized
0147 RW       0       0 RW 0066 NR       0       0 S..   X  -   Synchronized
0148 RW       0       0 RW 0067 NR       0       0 S..   X  -   Synchronized
0149 RW       0       0 RW 0068 NR       0       0 S..   X  -   Synchronized
014A RW       0       0 RW 0069 NR       0       0 S..   X  -   Synchronized
014B RW       0       0 RW 006A NR       0       0 S..   X  -   Synchronized
014C RW       0       0 RW 006B NR       0       0 S..   X  -   Synchronized
014D RW       0       0 RW 006C NR       0       0 S..   X  -   Synchronized
014E RW       0       0 RW 006D NR       0       0 S..   X  -   Synchronized
014F RW       0       0 RW 006E NR       0       0 S..   X  -   Synchronized

Total   ------- -------            ------- -------
Trks          0       0                  0       0
MBs         0.0     0.0                0.0     0.0

Legend for MODES:

 M(ode of Operation): A = Async, S = Sync, E = Semi-sync, C = Adaptive Copy
 D(omino)           : X = Enabled, . = Disabled
 A(daptive Copy)    : D = Disk Mode, W = WP Mode, . = ACp off
```

```
Legend for STATES:

 Cons(istency State): X = Enabled, M = Mixed, . = Disabled, - = N/A
 Susp(end State)    : X = Online, . = Offline, P = Offline Pending, - = N/A
```

Another `symcg show` command also displays that the consistency state of the devices is now Enabled.

```
# symcg show SRDF

Composite Group Name:  SRDF

  Composite Group Type                              : RDF1
  Valid                                             : Yes
  CG in PowerPath                                   : Yes
  CG in GNS                                         : No

  Number of RDF (RA) Groups                         :    2
  Number of STD Devices                             :   44
  Number of BCV's (Locally-associated)              :    0
  Number of VDEV's (Locally-associated)             :    0
  Number of RBCV's (Remotely-associated STD-RDF)    :    0
  Number of BRBCV's (Remotely-associated BCV-RDF)   :    0
  Number of RRBCV's (Remotely-associated RBCV)      :    0

  Number of Symmetrix Units (2):
     {

     1) Symmetrix ID                                  : 000000003143
        Microcode Version                             : 5267
        Number of STD Devices                         :   22
        Number of BCV's (Locally-associated)          :    0
        Number of VDEV's (Locally-associated)         :    0
        Number of RBCV's (Remotely-associated STD_RDF) :    0
        Number of BRBCV's (Remotely-associated BCV-RDF):    0
        Number of RRBCV's (Remotely-associated RBCV)  :    0

        Number of RDF (RA) Groups (1):
           {

           1) RDF (RA) Group Number  :  1            (A)
              Remote Symmetrix ID    : 000000003156
              Microcode Version      : 5267

              STD Devices (22):
                 {
                 ------------------------------------------------------------
                                          Sym  Device        Consistency   Cap
                 PdevName                 Dev  Config        State         (MB)
                 ------------------------------------------------------------
                 /dev/vx/rdmp/c15t1d24s2 00F7 RDF1          Enabled      12946
                 /dev/vx/rdmp/c15t1d25s2 00FA RDF1          Enabled       8631
                 /dev/vx/rdmp/c15t1d26s2 00FC RDF1          Enabled       4315
                 /dev/vx/rdmp/c15t1d27s2 00FD RDF1          Enabled       4315
                 /dev/vx/rdmp/c15t1d28s2 00FE RDF1          Enabled       4315
                 /dev/vx/rdmp/c15t1d29s2 00FF RDF1          Enabled       4315
                 /dev/vx/rdmp/c15t1d30s2 0100 RDF1          Enabled       4315
                 /dev/vx/rdmp/c15t1d31s2 0101 RDF1          Enabled       4315
```

```
            /dev/vx/rdmp/c15t1d32s2 0102 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d33s2 0103 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d34s2 0104 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d35s2 0105 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d36s2 0106 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d37s2 0107 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d38s2 0108 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d39s2 0109 RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d40s2 010A RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d41s2 010B RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d42s2 010C RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d43s2 010D RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d44s2 010E RDF1          Enabled       4315
            /dev/vx/rdmp/c15t1d45s2 010F RDF1          Enabled       4315
            }
      }

2) Symmetrix ID                                  : 000187900035
   Microcode Version                             : 5670
   Number of STD Devices                         :   22
   Number of BCV's (Locally-associated)          :    0
   Number of VDEV's (Locally-associated)         :    0
   Number of RBCV's (Remotely-associated STD_RDF) :    0
   Number of BRBCV's (Remotely-associated BCV-RDF):    0
   Number of RRBCV's (Remotely-associated RBCV)   :    0

   Number of RDF (RA) Groups (1):
      {

      1) RDF (RA) Group Number  :  1           (00)
         Remote Symmetrix ID    : 000187900041
         Microcode Version      : 5670

         STD Devices (22):
            {
            ------------------------------------------------------------
                                    Sym  Device        Consistency   Cap
            PdevName                Dev  Config        State         (MB)
            ------------------------------------------------------------
            /dev/vx/rdmp/c15t2d24s2 0137 RDF1+Mir      Enabled      12946
            /dev/vx/rdmp/c15t2d25s2 013A RDF1+Mir      Enabled       8631
            /dev/vx/rdmp/c15t2d26s2 013C RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d27s2 013D RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d28s2 013E RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d29s2 013F RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d30s2 0140 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d31s2 0141 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d32s2 0142 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d33s2 0143 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d34s2 0144 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d35s2 0145 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d36s2 0146 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d37s2 0147 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d38s2 0148 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d39s2 0149 RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d40s2 014A RDF1+Mir      Enabled       4315
            /dev/vx/rdmp/c15t2d41s2 014B RDF1+Mir      Enabled       4315
```

```
                /dev/vx/rdmp/c15t2d42s2 014C RDF1+Mir      Enabled       4315
                /dev/vx/rdmp/c15t2d43s2 014D RDF1+Mir      Enabled       4315
                /dev/vx/rdmp/c15t2d44s2 014E RDF1+Mir      Enabled       4315
                /dev/vx/rdmp/c15t2d45s2 014F RDF1+Mir      Enabled       4315
                }
        }
    }
```

The `symrdf suspend` command attempts to suspend SRDF pairs in the composite group. The message about using the "force flag" is meant to ensure that you really want to stop the SRDF mirroring operation and end consistency protection.

```
# symrdf -cg SRDF suspend -noprompt

An RDF 'Suspend' operation execution is in progress for composite group 'SRDF'.
Please wait...

Cannot proceed in the current RDF Consistency state except if the force flag is
used
```

The `symrdf suspend` command with the the `–force` option successfully suspends the SRDF pairs in the composite group.

```
# symrdf -cg SRDF suspend -force -noprompt

An RDF 'Suspend' operation execution is in progress for composite group 'SRDF'.
Please wait...

    Pend I/O on RDF link(s) for device(s) in (0035,01).............Done.
    Pend I/O on RDF link(s) for device(s) in (3143,01).............Done.
    Suspend RDF link(s) for devices in (0035).....................Done.
    Suspend RDF link(s) for devices in (3143).....................Done.

The RDF 'Suspend' operation successfully executed for composite group 'SRDF'.
```

Appendix A: Dynamic RDF with Enginuity Versions 5x67 and Higher

Dynamic SRDF allows you to create SRDF pairs from non-SRDF devices and subsequently cancel these dynamic SRDF pairings if they are no longer needed. Historically, source and target SRDF device pairing has been static once these devices have been configured as RDF1 and RDF2 type devices, and this is still true of devices that you configure this way.

However, beginning with the SRDF component of EMC Solutions Enabler version 5.0 running on Symmetrix units using Enginuity version 5568, you can create non-SRDF type devices that acquire the capability of being R1 or R2 devices when you use the Configuration Manager to set these devices and the Symmetrix unit for dynamic RDF.

Table 3 shows which dynamic RDF operations are valid for the Enginuity version that you are using.

Table 3. Using Dynamic RDF with Various Enginuity Versions

Enginuity Version	Description
5x66 and 5x67	You can create static SRDF devices using the Configuration Manager.
5x67 only	Dynamic RDF needs to be enabled in the Symmetrix unit, and RDF devices need to be marked as *dynamic* RDF devices. Only the Symmetrix service processor can create this configuration. The only dynamic RDF operation that you can perform from the host is `symrdf swap`.
5568 or higher	You can use the Configuration Manager to enable the dynamic RDF feature in the Symmetrix unit and set existing non-RDF devices to be capable of being dynamic RDF (R1 or R2, R1 only, or R2 only).
	If the dynamic RDF feature is enabled in the Symmetrix unit, and if these devices are marked as dynamic-RDF-capable devices, you can use these dynamic RDF devices to create RDF pairs. Use either the Configuration Manager syntax (requires a Configuration Manager license) or the `symrdf createpair` command (requires an RDF license only).
	If the dynamic RDF feature is *not* enabled in the Symmetrix, but *all* devices in the request are capable of being dynamic RDF devices, the Configuration Manager creates these devices as *static* RDF devices (as Enginuity versions 5x66 and 5x67). If the devices in the request are mixed (some capable of being dynamic RDF devices and some are not), the request is rejected.