Data protection with the CLI

ONTAP 9

NetApp
May 02, 2022

Table of Contents

Data protection with the CLI . . . 1
- Data protection overview with the CLI . . . 1
- Manage local Snapshot copies . . . 1
- SnapMirror volume replication . . . 12
- Manage SnapMirror volume replication . . . 33
- About SnapMirror SVM replication . . . 69
- Manage SnapMirror SVM replication . . . 76
- Manage SnapMirror root volume replication . . . 101
- SnapMirror technical details . . . 105

Data protection with the CLI

Data protection overview with the CLI

You can use CLI commands to manage Snapshot copies on a local ONTAP system and to replicate Snapshot copies to a remote system using SnapMirror. You can replicate Snapshot copies for disaster recovery or long-term retention.

Use these procedures under the following circumstances:

- You want to understand the range of ONTAP backup and recovery capabilities.
- You want to use the command-line interface (CLI), not System Manager, an automated scripting tool, or a SnapCenter product.
- You have already created peer relationships between the source and destination clusters and the source and destination SVMs.

  Cluster and SVM peering

- You are backing up volumes or SVMs from AFF or FAS storage systems to AFF or FAS storage systems.
  - If you are replicating Element volumes to ONTAP, or ONTAP LUNs to an Element system, see the NetApp Element software documentation.

    Replication between NetApp element software and ONTAP

  - Beginning with ONTAP 9.10.1, you can create data protection relationships between S3 buckets using S3 SnapMirror. For more information, see S3 SnapMirror overview.
- You want to provide data protection using online methods, not tape.

Other ways to do this in ONTAP

To perform these tasks with…	Refer to…
The redesigned System Manager (available with ONTAP 9.7 and later)	Prepare for mirroring and vaulting
System Manager Classic (available with ONTAP 9.7 and earlier)	Volume disaster recovery preparation overview

Manage local Snapshot copies

Manage local Snapshot copies overview

A *Snapshot copy* is a read-only, point-in-time image of a volume. The image consumes minimal storage space and incurs negligible performance overhead because it records only changes to files since the last Snapshot copy.

You can use a Snapshot copy to restore the entire contents of a volume, or to recover individual files or LUNs. Snapshot copies are stored in the directory `.snapshot` on the volume.

In ONTAP 9.3 and earlier, a volume can contain up to 255 Snapshot copies. In ONTAP 9.4 and later, a FlexVol volume can contain up to 1023 Snapshot copies.

Beginning with ONTAP 9.8, FlexGroup volumes can contain 1023 Snapshot copies. For more information, see Protect FlexGroup volumes using Snapshot copies.

Configure custom Snapshot policies

Configure custom Snapshot policies overview

A *Snapshot policy* defines how the system creates Snapshot copies. The policy specifies when to create Snapshot copies, how many copies to retain, and how to name them. For example, a system might create one Snapshot copy every day at 12:10 a.m., retain the two most recent copies, and name the copies "daily.`timestamp`."

The default policy for a volume automatically creates Snapshot copies on the following schedule, with the oldest Snapshot copies deleted to make room for newer copies:

- A maximum of six hourly Snapshot copies taken five minutes past the hour.
- A maximum of two daily Snapshot copies taken Monday through Saturday at 10 minutes after midnight.
- A maximum of two weekly Snapshot copies taken every Sunday at 15 minutes after midnight.

Unless you specify a Snapshot policy when you create a volume, the volume inherits the Snapshot policy associated with its containing storage virtual machine (SVM).

When to configure a custom Snapshot policy

If the default Snapshot policy is not appropriate for a volume, you can configure a custom policy that modifies the frequency, retention, and name of Snapshot copies. The schedule will be dictated mainly by the rate of change of the active file system.

You might back up a heavily used file system like a database every hour, while you back up rarely used files once a day. Even for a database, you will typically run a full backup once or twice a day, while backing up transaction logs every hour.

Other factors are the importance of the files to your organization, your Service Level Agreement (SLA), your Recovery Point Objective (RPO), and your Recovery Time Objective (RTO). Generally speaking, you should retain only as many Snapshot copies as necessary.

Create a Snapshot job schedule

A Snapshot policy requires at least one Snapshot copy job schedule. You can use the `job schedule cron create` command to create a job schedule.

About this task

By default, ONTAP forms the names of Snapshot copies by appending a timestamp to the job schedule name.

If you specify values for both day of the month and day of the week, the values are considered independently. For example, a cron schedule with the day specification `Friday` and the day of the month specification `13` runs every Friday and on the 13th day of each month, not just on every Friday the 13th.

Step

1. Create a job schedule:

   `job schedule cron create -name job_name -month month -dayofweek day_of_week -day day_of_month -hour hour -minute minute`

   For `-month`, `-dayofweek`, and `-hour`, you can specify `all` to run the job every month, day of the week, and hour, respectively.

   Beginning with ONTAP 9.10.1, you can include the Vserver for your job schedule:

   `job schedule cron create -name job_name -vserver Vserver_name -month month -dayofweek day_of_week -day day_of_month -hour hour -minute minute`

   The following example creates a job schedule named `myweekly` that runs on Saturdays at 3:00 a.m.:

   ```
   cluster1::> job schedule cron create -name myweekly -dayofweek
   "Saturday" -hour 3 -minute 0
   ```

   The following example creates a schedule named `myweeklymulti` that specifies multiple days, hours and minutes:

   ```
   job schedule cron create -name myweeklymulti -dayofweek
   "Monday,Wednesday,Sunday" -hour 3,9,12 -minute 0,20,50
   ```

Create a Snapshot policy

A Snapshot policy specifies when to create Snapshot copies, how many copies to retain, and how to name them. For example, a system might create one Snapshot copy every day at 12:10 a.m., retain the two most recent copies, and name them “daily.*timestamp*.” A Snapshot policy can contain up to five job schedules.

About this task

By default, ONTAP forms the names of Snapshot copies by appending a timestamp to the job schedule name:

```
daily.2017-05-14_0013/              hourly.2017-05-15_1106/
daily.2017-05-15_0012/              hourly.2017-05-15_1206/
hourly.2017-05-15_1006/             hourly.2017-05-15_1306/
```

You can substitute a prefix for the job schedule name if you prefer.

The `snapmirror-label` option is for SnapMirror replication. For more information, see Defining a rule for a policy.

Step

1. Create a Snapshot policy:

```
volume snapshot policy create -vserver SVM -policy policy_name -enabled
true|false -schedule1 schedule1_name -count1 copies_to_retain -prefix1
snapshot_prefix -snapmirror-label1 snapshot_label … -schedule1 schedule5_name
-count5 copies_to_retain-prefix5 snapshot_prefix -snapmirror-label5
snapshot_label
```

The following example creates a Snapshot policy named `snap_policy_daily` that runs on a `daily` schedule. The policy has a maximum of five Snapshot copies, each with the name `daily.timestamp` and the SnapMirror label `daily`:

```
cluster1::> volume snapshot policy create -vserver vs0 -policy
snap_policy_daily -schedule1 daily -count1 5 -snapmirror-label1 daily
```

Manage the Snapshot copy reserve

Manage the Snapshot copy reserve overview

The *Snapshot copy reserve* sets aside a percentage of disk space for Snapshot copies, five percent by default. Because Snapshot copies use space in the active file system when the Snapshot copy reserve is exhausted, you might want to increase the Snapshot copy reserve as needed. Alternatively, you can autodelete Snapshot copies when the reserve is full.

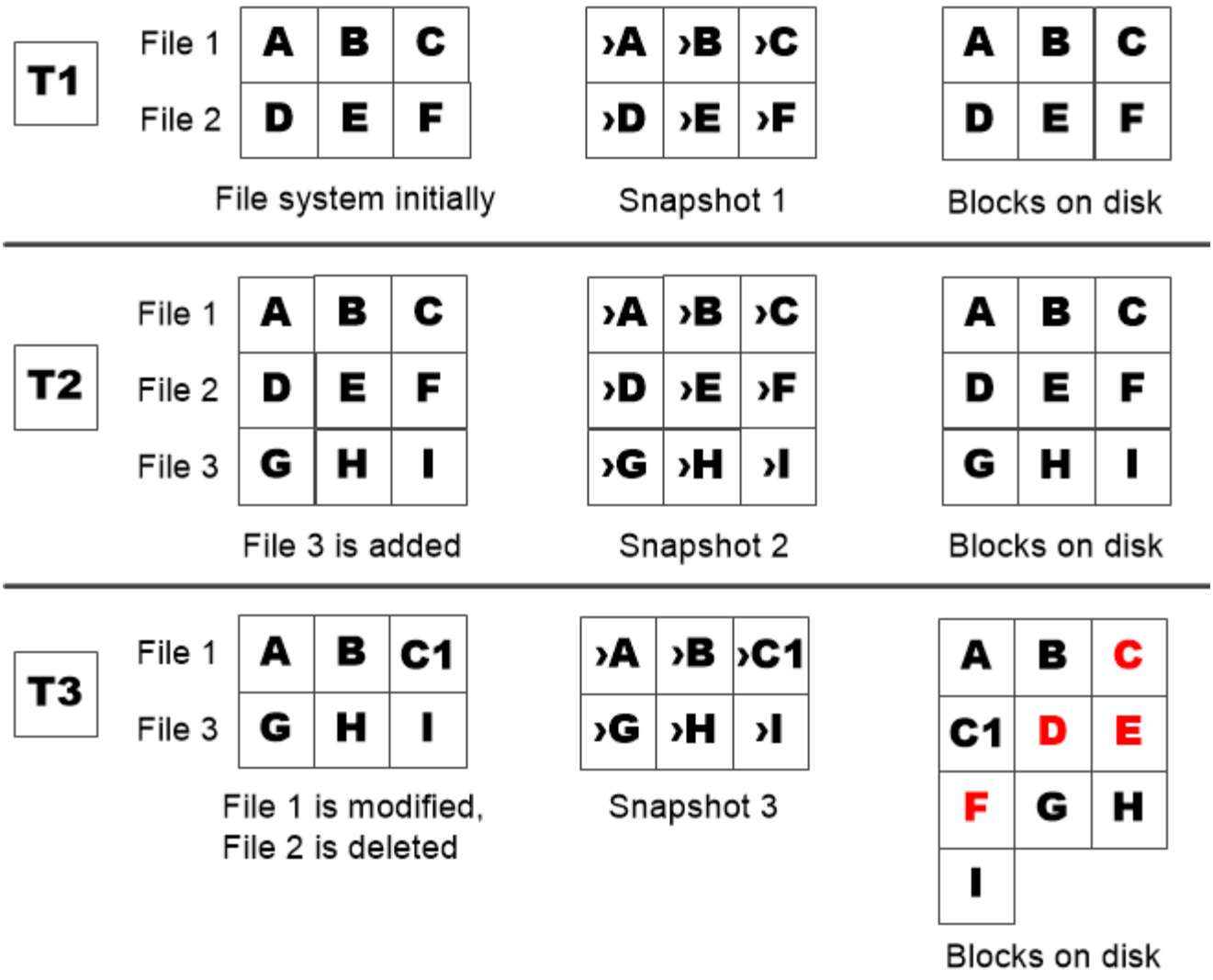
When to increase the Snapshot copy reserve

In deciding whether to increase the Snapshot reserve, it's important to remember that a Snapshot copy records only changes to files since the last Snapshot copy was made. It consumes disk space only when blocks in the active file system are modified or deleted.

This means that the rate of change of the file system is the key factor in determining the amount of disk space used by Snapshot copies. No matter how many Snapshot copies you create, they will not consume disk space if the active file system has not changed.

A FlexVol volume containing database transaction logs, for example, might have a Snapshot copy reserve as large as 20% to account for its greater rate of change. Not only will you want to create more Snapshot copies to capture the more frequent updates to the database, you will also want to have a larger Snapshot copy reserve to handle the additional disk space the Snapshot copies consume.

A Snapshot copy consists of pointers to blocks rather than copies of blocks. You can think of a pointer as a "claim" on a block: ONTAP "holds" the block until the Snapshot copy is deleted.



A Snapshot copy consumes disk space only when blocks in the active file system are modified or deleted.

How deleting protected files can lead to less file space than expected

A Snapshot copy points to a block even after you delete the file that used the block. This explains why an exhausted Snapshot copy reserve might lead to the counter-intuitive result in which deleting an entire file system results in less space being available than the file system occupied.

Consider the following example. Before deleting any files, the `df` command output is as follows:

```
Filesystem          kbytes  used    avail  capacity
/vol/vol0/          3000000 3000000 0       100%
/vol/vol0/.snapshot 1000000 500000  500000   50%
```

After deleting the entire file system and making a Snapshot copy of the volume, the `df` command generates the following output:

```
Filesystem          kbytes  used    avail  capacity
/vol/vol0/          3000000 2500000 500000   83%
/vol/vol0/.snapshot 1000000 3500000 0       350%
```

As the output shows, the entire 3 GB formerly used by the active file system is now being used by Snapshot copies, in addition to the 0.5 GB used before the deletion.

Because the disk space used by the Snapshot copies now exceeds the Snapshot copy reserve, the overflow of 2.5 GB “spills” into the space reserved for active files, leaving you with 0.5 GB free space for files where you might reasonably have expected 3 GB.

Monitor Snapshot copy disk consumption

You can monitor Snapshot copy disk consumption using the `df` command. The command displays the amount of free space in the active file system and the Snapshot copy reserve.

Step

1. Display Snapshot copy disk consumption: `df`

   The following example shows Snapshot copy disk consumption:

   ```
   cluster1::> df
   Filesystem          kbytes  used    avail  capacity
   /vol/vol0/          3000000 3000000 0       100%
   /vol/vol0/.snapshot 1000000 500000  500000   50%
   ```

Check available Snapshot copy reserve on a volume

You might want to check how much Snapshot copy reserve is available on a volume by using the `snapshot-reserve-available` parameter with the `volume show` command.

Step

1. Check the Snapshot copy reserve available on a volume:

   `vol show -vserver` *`SVM`* `-volume` *`volume`* `-fields snapshot-reserve-available`

   For complete command syntax, see the man page.

   The following example displays the available Snapshot copy reserve for `vol1`:

```
cluster1::> vol show -vserver vs0 -volume vol1 -fields snapshot-reserve-
available

vserver volume snapshot-reserve-available
------- ------ --------------------------
vs0     vol1   4.84GB
```

Modify the Snapshot copy reserve

You might want to configure a larger Snapshot copy reserve to prevent Snapshot copies from using space reserved for the active file system. You can decrease the Snapshot copy reserve when you no longer need as much space for Snapshot copies.

Step

1. Modify the Snapshot copy reserve:

   `volume modify -vserver` *`SVM`* `-volume` *`volume`* `-percent-snapshot-space` *`snap_reserve`*

   For complete command syntax, see the man page.

   The following example sets the Snapshot copy reserve for `vol1` to 10 percent:

   ```
   cluster1::> volume modify -vserver vs0 -volume vol1 -percent-snapshot
   -space 10
   ```

Autodelete Snapshot copies

You can use the `volume snapshot autodelete modify` command to trigger automatic deletion of Snapshot copies when the Snapshot reserve is exceeded. By default, the oldest Snapshot copies are deleted first.

About this task

LUN and file clones are deleted when there are no more Snapshot copies to be deleted.

Step

1. Autodelete Snapshot copies:

   `volume snapshot autodelete modify -vserver` *`SVM`* `-volume` *`volume`* `-enabled true|false -trigger volume|snap_reserve`

   For complete command syntax, see the man page.

   The following example autodeletes Snapshot copies for `vol1` when the Snapshot copy reserve is exhausted:

```
cluster1::> volume snapshot autodelete modify -vserver vs0 -volume vol1
-enabled true -trigger snap_reserve
```

Restore files from Snapshot copies

Restore a file from a Snapshot copy on an NFS or SMB client

A user on an NFS or SMB client can restore a file directly from a Snapshot copy without the intervention of a storage system administrator.

Every directory in the file system contains a subdirectory named `.snapshot` accessible to NFS and SMB users. The `.snapshot` subdirectory contains subdirectories corresponding to the Snapshot copies of the volume:

```
$ ls .snapshot
daily.2017-05-14_0013/              hourly.2017-05-15_1106/
daily.2017-05-15_0012/              hourly.2017-05-15_1206/
hourly.2017-05-15_1006/             hourly.2017-05-15_1306/
```

Each subdirectory contains the files referenced by the Snapshot copy. If users accidentally delete or overwrite a file, they can restore the file to the parent read-write directory by copying the file from the Snapshot subdirectory to the read-write directory:

```
$ ls my.txt
ls: my.txt: No such file or directory
$ ls .snapshot
daily.2017-05-14_0013/              hourly.2017-05-15_1106/
daily.2017-05-15_0012/              hourly.2017-05-15_1206/
hourly.2017-05-15_1006/             hourly.2017-05-15_1306/
$ ls .snapshot/hourly.2017-05-15_1306/my.txt
my.txt
$ cp .snapshot/hourly.2017-05-15_1306/my.txt .
$ ls my.txt
my.txt
```

Enable and disable NFS and SMB client access to Snapshot copy directory

To determine whether the Snapshot copy directory is visible to NFS and SMB clients to restore a file or LUN from a Snapshot copy, you can enable and disable access to the Snapshot copy directory using the `-snapdir-access` option of the `volume modify` command.

Steps

1. Check the Snapshot directory access status:

```
`volume show -vserver _SVM_name_ -volume _vol_name_ -fields snapdir-
access`
```

Example:

```
clus1::> volume show -vserver vs0 -volume vol1 -fields snapdir-access
vserver volume snapdir-access
------- ------ --------------
vs0     vol1   false
```

2. Enable or disable the Snapshot copy directory access:

   `volume modify -volume` *`vol_name`* `-snapdir-access true|false`

   The following example enables Snapshot copy directory access on vol1:

```
clus1::> volume modify -volume vol1 -snapdir-access true
Volume modify successful on volume vol1 of Vserver vs0.
```

Restore a single file from a Snapshot copy

You can use the `volume snapshot restore-file` command to restore a single file or LUN from a Snapshot copy. You can restore the file to a different location in the parent read-write volume if you do not want to replace an existing file.

About this task

If you are restoring an existing LUN, a LUN clone is created and backed up in the form of a Snapshot copy. During the restore operation, you can read to and write from the LUN.

Files with streams are restored by default.

Steps

1. List the Snapshot copies in a volume:

   `volume snapshot show -vserver` *`SVM`* `-volume` *`volume`*

   For complete command syntax, see the man page.

   The following example shows the Snapshot copies in `vol1`:

```
clus1::> volume snapshot show -vserver vs1 -volume vol1

Vserver Volume Snapshot                State    Size  Total% Used%
------- ------ ---------- ----------- ------   -----  ------ -----
vs1  vol1   hourly.2013-01-25_0005  valid   224KB     0%    0%
               daily.2013-01-25_0010   valid   92KB      0%    0%
               hourly.2013-01-25_0105  valid   228KB     0%    0%
               hourly.2013-01-25_0205  valid   236KB     0%    0%
               hourly.2013-01-25_0305  valid   244KB     0%    0%
               hourly.2013-01-25_0405  valid   244KB     0%    0%
               hourly.2013-01-25_0505  valid   244KB     0%    0%

7 entries were displayed.
```

2. Restore a file from a Snapshot copy:

   `volume snapshot restore-file -vserver` *`SVM`* `-volume` *`volume`* `-snapshot` *`snapshot`* `-path` *`file_path`* `-restore-path` *`destination_path`*

   For complete command syntax, see the man page.

   The following example restores the file `myfile.txt`:

```
cluster1::> volume snapshot restore-file -vserver vs0 -volume vol1
-snapshot daily.2013-01-25_0010 -path /myfile.txt
```

Restore part of a file from a Snapshot copy

You can use the `volume snapshot partial-restore-file` command to restore a range of data from a Snapshot copy to a LUN or to an NFS or SMB container file, assuming you know the starting byte offset of the data and the byte count. You might use this command to restore one of the databases on a host that stores multiple databases in the same LUN.

Steps

1. List the Snapshot copies in a volume:

   `volume snapshot show -vserver` *`SVM`* `-volume` *`volume`*

   For complete command syntax, see the man page.

   The following example shows the Snapshot copies in `vol1`:

```
clus1::> volume snapshot show -vserver vs1 -volume vol1

Vserver Volume Snapshot                State    Size  Total% Used%
------- ------ ---------- ----------- ------   -----  ------ -----
vs1  vol1   hourly.2013-01-25_0005  valid   224KB     0%    0%
               daily.2013-01-25_0010   valid   92KB      0%    0%
               hourly.2013-01-25_0105  valid   228KB     0%    0%
               hourly.2013-01-25_0205  valid   236KB     0%    0%
               hourly.2013-01-25_0305  valid   244KB     0%    0%
               hourly.2013-01-25_0405  valid   244KB     0%    0%
               hourly.2013-01-25_0505  valid   244KB     0%    0%

7 entries were displayed.
```

2. Restore part of a file from a Snapshot copy:

   `volume snapshot partial-restore-file -vserver` *`SVM`* `-volume` *`volume`* `-snapshot` *`snapshot`* `-path` *`file_path`* `-start-byte` *`starting_byte`* `-byte-count` *`byte_count`*

   The starting byte offset and byte count must be multiples of 4,096.

   The following example restores the first 4,096 bytes of the file `myfile.txt`:

```
cluster1::> volume snapshot partial-restore-file -vserver vs0 -volume
vol1 -snapshot daily.2013-01-25_0010 -path /myfile.txt -start-byte 0
-byte-count 4096
```

Restore the contents of a volume from a Snapshot copy

You can use the `volume snapshot restore` command to restore the contents of a volume from a Snapshot copy.

About this task

If the volume has SnapMirror relationships, manually replicate all mirror copies of the volume immediately after you restore from a Snapshot copy. Not doing so can result in unusable mirror copies that must be deleted and recreated.

Steps

1. List the Snapshot copies in a volume:

   `volume snapshot show -vserver` *`SVM`* `-volume` *`volume`*

   For complete command syntax, see the man page.

   The following example shows the Snapshot copies in `vol1`:

```
clus1::> volume snapshot show -vserver vs1 -volume vol1

Vserver Volume Snapshot                State    Size  Total% Used%
------- ------ ---------- ----------- ------   -----  ------ -----
vs1  vol1   hourly.2013-01-25_0005  valid   224KB     0%    0%
               daily.2013-01-25_0010   valid   92KB      0%    0%
               hourly.2013-01-25_0105  valid   228KB     0%    0%
               hourly.2013-01-25_0205  valid   236KB     0%    0%
               hourly.2013-01-25_0305  valid   244KB     0%    0%
               hourly.2013-01-25_0405  valid   244KB     0%    0%
               hourly.2013-01-25_0505  valid   244KB     0%    0%

7 entries were displayed.
```

2. Restore the contents of a volume from a Snapshot copy:

   `volume snapshot restore -vserver` *`SVM`* `-volume` *`volume`* `-snapshot` *`snapshot`*

   For complete command syntax, see the man page.

   The following example restores the contents of `vol1`:

```
cluster1::> volume snapshot restore -vserver vs0 -volume vol1 -snapshot
daily.2013-01-25_0010
```

SnapMirror volume replication

About SnapMirror volume replication

Traditionally, ONTAP replication technologies served the need for disaster recovery (DR) and data archiving. In ONTAP 9.3, these technologies were combined in a way that allows you to configure disaster recovery and archiving on the same destination volume.

Asynchronous SnapMirror disaster recovery basics

SnapMirror is disaster recovery technology, designed for failover from primary storage to secondary storage at a geographically remote site. As its name implies, SnapMirror creates a replica, or *mirror,* of your working data in secondary storage from which you can continue to serve data in the event of a catastrophe at the primary site.

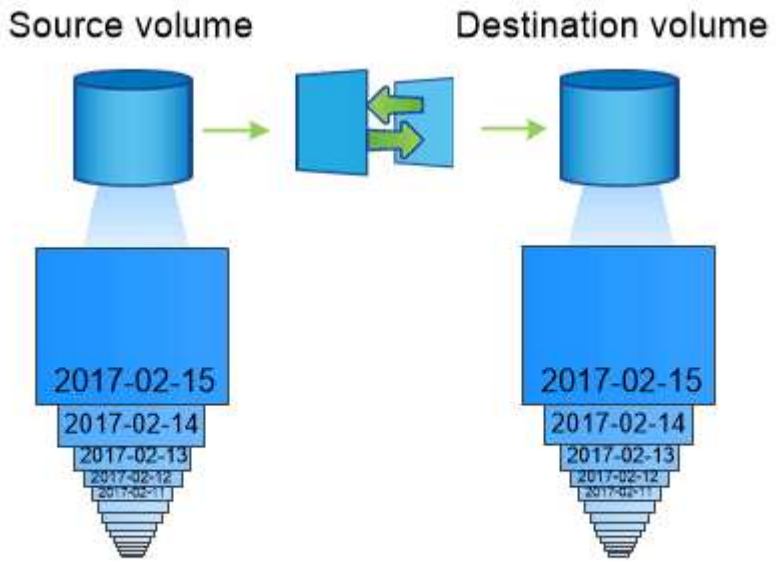
If the primary site is still available to serve data, you can simply transfer any needed data back to it, and not serve clients from the mirror at all. As the failover use case implies, the controllers on the secondary system should be equivalent or nearly equivalent to the controllers on the primary system to serve data efficiently from mirrored storage.

Data protection relationships

Data is mirrored at the volume level. The relationship between the source volume in primary storage and the destination volume in secondary storage is called a *data protection relationship.* The clusters in which the volumes reside and the SVMs that serve data from the volumes must be *peered.* A peer relationship enables clusters and SVMs to exchange data securely.

Cluster and SVM peering

The figure below illustrates SnapMirror data protection relationships.



A SnapMirror data protection relationship typically mirrors the Snapshot copies available on the source volume.

Scope of data protection relationships

You can create a data protection relationship directly between volumes or between the SVMs that own the volumes. In an *SVM data protection relationship,* all or part of the SVM configuration, from NFS exports and SMB shares to RBAC, is replicated, as well as the data in the volumes that the SVM owns.

You can also use SnapMirror for two special data protection applications:

- A *load-sharing mirror* copy of the SVM root volume ensures that data remains accessible in the event of a node outage or failover.
- A data protection relationship between *SnapLock volumes* lets you replicate WORM files to secondary storage.

  Archive and compliance using SnapLock technology

How SnapMirror data protection relationships are initialized

The first time you invoke SnapMirror, it performs a *baseline transfer* from the source volume to the destination volume. The *SnapMirror policy* for the relationship defines the contents of the baseline and any updates.

A baseline transfer under the default SnapMirror policy `MirrorAllSnapshots` involves the following steps:

- Make a Snapshot copy of the source volume.
- Transfer the Snapshot copy and all the data blocks it references to the destination volume.
- Transfer the remaining, less recent Snapshot copies on the source volume to the destination volume for use in case the “active” mirror is corrupted.

How SnapMirror data protection relationships are updated

Updates are asynchronous, following the schedule you configure. Retention mirrors the Snapshot policy on the source.

At each update under the `MirrorAllSnapshots` policy, SnapMirror creates a Snapshot copy of the source volume and transfers that Snapshot copy and any Snapshot copies that have been made since the last update. In the following output from the `snapmirror policy show` command for the `MirrorAllSnapshots` policy, note the following:

- `Create Snapshot` is “true”, indicating that `MirrorAllSnapshots` creates a Snapshot copy when SnapMirror updates the relationship.
- `MirrorAllSnapshots` has rules “sm_created” and “all_source_snapshots”, indicating that both the Snapshot copy created by SnapMirror and any Snapshot copies that have been made since the last update are transferred when SnapMirror updates the relationship.

```
cluster_dst::> snapmirror policy show -policy MirrorAllSnapshots -instance

                    Vserver: vs0
      SnapMirror Policy Name: MirrorAllSnapshots
      SnapMirror Policy Type: async-mirror
                Policy Owner: cluster-admin
                 Tries Limit: 8
           Transfer Priority: normal
   Ignore accesstime Enabled: false
     Transfer Restartability: always
 Network Compression Enabled: false
             Create Snapshot: true
                     Comment: Asynchronous SnapMirror policy for mirroring
all snapshots
                              and the latest active file system.
       Total Number of Rules: 2
                  Total Keep: 2
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              sm_created              1  false       0 -
-
                              all_source_snapshots    1  false       0 -
-
```

MirrorLatest policy

The preconfigured `MirrorLatest` policy works exactly the same way as `MirrorAllSnapshots`, except that only the Snapshot copy created by SnapMirror is transferred at initialization and update.

```
                      Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                             ----------------     ----  -------- ----
-------- ------
                             sm_created              1  false       0 -
-
```

SnapMirror Synchronous disaster recovery basics

Beginning with ONTAP 9.5, SnapMirror Synchronous (SM-S) technology is supported on all FAS and AFF platforms that have at least 16 GB of memory and on all ONTAP Select platforms. SnapMirror Synchronous technology is a per-node, licensed feature that provides synchronous data replication at the volume level.

This functionality addresses the regulatory and national mandates for synchronous replication in financial, healthcare, and other regulated industries where zero data loss is required.

The limit on the number of SnapMirror Synchronous replication operations per HA pair depends on the controller model.

Platform	Number of SnapMirror Synchronous operations that are allowed per HA pair in releases earlier than ONTAP 9.9.1	Number of SnapMirror Synchronous operations that are allowed per HA pair in ONTAP 9.9.1	Number of SnapMirror Synchronous operations that are allowed per HA pair in ONTAP 9.10.1
AFF	80	160	200
FAS	40	80	80
ONTAP Select	20	40	40

Supported features

The following feature is supported for SnapMirror Synchronous technology in ONTAP 9.10.1; provided all nodes in the source and destination cluster are running ONTAP 9.10.1:

- NFSv4.2

In ONTAP 9.5 and later, SnapMirror Synchronous technology supports the NFSv3, FC, and iSCSI protocols over all networks for which the latency does not exceed 10ms.

The following features are supported for SnapMirror Synchronous technology in ONTAP 9.7:

- Replication of application-created Snapshot copies

  Only the Snapshot copies with the SnapMirror label that match the rule associated with Sync or Strict Sync policy. Scheduled Snapshot copies created using a Snapshot policy are not replicated.

- FC-NVMe
- LUN clones and NVMe namespace clones

  LUN clones backed by application-created Snapshot copies are also supported.

The following features are supported for SnapMirror Synchronous technology in ONTAP 9.6; provided all nodes in the source and destination cluster are running ONTAP 9.6:

- SVM DR
  - A SnapMirror Synchronous source can also be a SVM DR source, for example, a fan-out configuration with SM-S as one leg and SVM DR as the other.
  - A SnapMirror Synchronous source cannot be an SVM DR destination because SM-S does not support cascading a DP source.

    You must release the synchronous relationship before performing an SVM DR flip resync in the destination cluster.

  - A SnapMirror Synchronous destination cannot be an SVM DR source because SVM DR does not support replication of DP volumes.

    A flip resync of the synchronous source would result in the SVM DR excluding the DP volume in the destination cluster.

- NFSv4.0 and NFSv4.1
- SMB 2.0 or later
- Mixed protocol access(NFSv3 and SMB)
- Antivirus on the primary volume of the SnapMirror Synchronous relationship
- Hard or soft quotas on the primary volume of the SnapMirror Synchronous relationship

  The quota rules are not replicated to the destination; therefore, the quota database is not replicated to the destination.

- FPolicy on the primary volume of the SnapMirror Synchronous relationship
- SnapMirror Synchronous mirror-mirror cascade

  The relationship from the destination volume of the SnapMirror Synchronous relationship must be an asynchronous SnapMirror relationship.

- Timestamp parity between source and destination volumes for NAS

  If you have upgraded from ONTAP 9.5 to ONTAP 9.6, the timestamp is replicated only for any new and modified files in the source volume. The timestamp of existing files in the source volume is not synchronized.

- Removal of high metadata operation frequency limitation

- Security for sensitive data in-transit using TLS 1.2 encryption
- Clone autodelete

Unsupported features

The following features are not supported with Synchronous SnapMirror relationships:

- MCC
- SFMoD
- SFCoD
- VVol
- Mixed SAN and NAS access

  The primary volume of a SnapMirror Synchronous relationship can either serve NAS data or SAN data. Both SAN and NAS access from the primary volume of a SnapMirror Synchronous relationship is not supported.

- Mixed SAN and NVMe access

  LUNs and NVMe namespaces are not supported on the same volume or SVM.

- SnapLock volumes
- FlexGroup volumes
- FlexCache volumes
- SnapRestore
- DP_Optimized (DPO) systems
- Tape backup or restore using dump and SMTape on the destination volume
- Tape based restore to the source volume
- Throughput floor (QoS Min) for source volumes
- In a fan-out configuration, only one relationship can be a SnapMirror Synchronous relationship; all the other relationships from the source volume must be asynchronous SnapMirror relationships.
- Global throttling

Modes of operation

SnapMirror Synchronous has two modes of operation based on the type of the SnapMirror policy used:

- **Sync mode**

  In Sync mode, an I/O to primary storage is first replicated to secondary storage. Then the I/O is written to primary storage, and acknowledgment is sent to the application that issued the I/O. If the write to the secondary storage is not completed for any reason, the application is allowed to continue writing to the primary storage. When the error condition is corrected, SnapMirror Synchronous technology automatically resynchronizes with the secondary storage and resumes replicating from primary storage to secondary storage in Synchronous mode.

  In Sync mode, RPO=0 and RTO is very low until a secondary replication failure occurs at which time RPO and RTO become indeterminate, but equal the time to repair the issue that caused secondary replication to

fail and for the resync to complete.

- **StrictSync mode**

  SnapMirror Synchronous can optionally operate in StrictSync mode. If the write to the secondary storage is not completed for any reason, the application I/O fails, thereby ensuring that the primary and secondary storage are identical. Application I/O to the primary resumes only after the SnapMirror relationship returns to the `InSync` status. If the primary storage fails, application I/O can be resumed on the secondary storage, after failover, with no loss of data.

  In StrictSync mode RPO is always zero, and RTO is very low.

Relationship status

The status of a SnapMirror Synchronous relationship is always in the `InSync` status during normal operation. If the SnapMirror transfer fails for any reason, the destination is not in sync with the source and can go to the `OutofSync` status.

For SnapMirror Synchronous relationships, the system automatically checks the relationship status (`InSync` or `OutofSync`) at a fixed interval. If the relationship status is `OutofSync`, ONTAP automatically triggers the auto resync process to bring back the relationship to the `InSync` status. Auto resync is triggered only if the transfer fails due to any operation, such as unplanned storage failover at source or destination or a network outage. User-initiated operations such as `snapmirror quiesce` and `snapmirror break` do not trigger auto resync.

If the relationship status becomes `OutofSync` for a SnapMirror Synchronous relationship in the StrictSync mode, all I/O operations to the primary volume are stopped. The `OutofSync` state for SnapMirror Synchronous relationship in the Sync mode is not disruptive to the primary and I/O operations are allowed on the primary volume.

Related information

NetApp Technical Report 4733: SnapMirror Synchronous for ONTAP 9.6

About workloads supported by StrictSync and Sync policies

StrictSync and Sync policies support all LUN-based applications with FC, iSCSI, and FC-NVMe protocols, as well as NFSv3 and NFSv4 protocols for enterprise applications such as databases, VMWare, quota, SMB, and so on. Beginning with ONTAP 9.6, SnapMirror Synchronous can be used for enterprise file services such as electronic design automation (EDA), home directories, and software build workloads.

In ONTAP 9.5, for a Sync policy, you need to consider a few important aspects while selecting the NFSv3 or NFSv4 workloads. The amount of data read or write operations by workloads is not a consideration, as Sync policy can handle high read or write IO workloads. In ONTAP 9.5, workloads that have excessive file creation, directory creation, file permission changes, or directory permission changes may not be suitable (these are referred to as high-metadata workloads). A typical example of a high-metadata workload is a DevOps workload in which you create multiple test files, run automation, and delete the files. Another example is parallel build workload that generate multiple temporary files during compilation. The impact of a high rate of write metadata activity is that it can cause synchronization between mirrors to temporarily break which stalls the read and write IOs from the client.

Beginning with ONTAP 9.6, these limitations are removed and SnapMirror Synchronous can be used for

enterprise file services workloads that include multiuser environments, such as home directories and software build workloads.

Related information

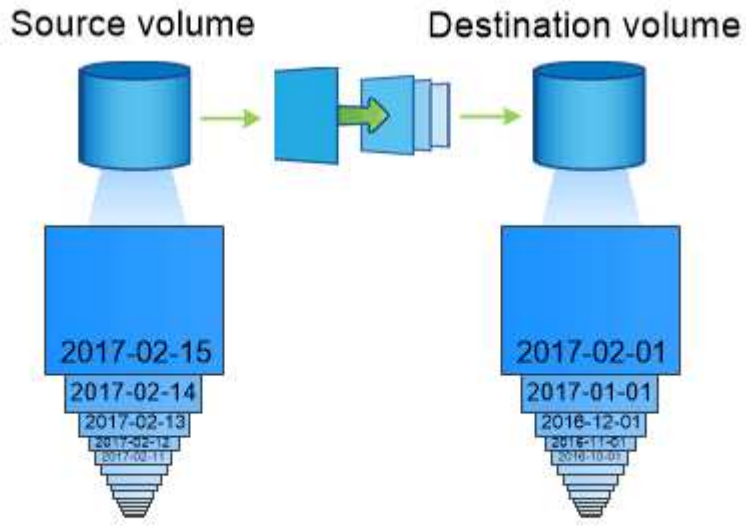
SnapMirror Synchronous Configuration and Best Practices

Vault archiving using SnapMirror technology

SnapMirror vault policies replace SnapVault technology in ONTAP 9.3 and later. You use a SnapMirror vault policy for disk-to-disk Snapshot copy replication for standards compliance and other governance-related purposes. In contrast to a SnapMirror relationship, in which the destination usually contains only the Snapshot copies currently in the source volume, a vault destination typically retains point-in-time Snapshot copies created over a much longer period.

You might want to keep monthly Snapshot copies of your data over a 20-year span, for example, to comply with government accounting regulations for your business. Since there is no requirement to serve data from vault storage, you can use slower, less expensive disks on the destination system.

The figure below illustrates SnapMirror vault data protection relationships.



A SnapVault data protection relationship typically retains point-in-time Snapshot copies created over a longer period than the Snapshot copies on the source volume.

How vault data protection relationships are initialized

The SnapMirror policy for the relationship defines the contents of the baseline and any updates.

A baseline transfer under the default vault policy `XDPDefault` makes a Snapshot copy of the source volume, then transfers that copy and the data blocks it references to the destination volume. Unlike SnapMirror relationships, a vault backup does not include older Snapshot copies in the baseline.

How vault data protection relationships are updated

Updates are asynchronous, following the schedule you configure. The rules you define in the policy for the relationship identify which new Snapshot copies to include in updates and how many copies to retain. The labels defined in the policy ("monthly," for example) must match one or more labels defined in the Snapshot policy on the source. Otherwise, replication fails.

At each update under the `XDPDefault` policy, SnapMirror transfers Snapshot copies that have been made since the last update, provided they have labels matching the labels defined in the policy rules. In the following output from the `snapmirror policy show` command for the `XDPDefault` policy, note the following:

- `Create Snapshot` is "false", indicating that `XDPDefault` does not create a Snapshot copy when SnapMirror updates the relationship.
- `XDPDefault` has rules "daily" and "weekly", indicating that all Snapshot copies with matching labels on the source are transferred when SnapMirror updates the relationship.

```
cluster_dst::> snapmirror policy show -policy XDPDefault -instance

                    Vserver: vs0
      SnapMirror Policy Name: XDPDefault
      SnapMirror Policy Type: vault
                Policy Owner: cluster-admin
                 Tries Limit: 8
           Transfer Priority: normal
   Ignore accesstime Enabled: false
     Transfer Restartability: always
 Network Compression Enabled: false
             Create Snapshot: false
                     Comment: Default policy for XDP relationships with
daily and weekly
                              rules.
       Total Number of Rules: 2
                  Total Keep: 59
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              daily                   7  false       0 -
-
                              weekly                 52  false       0 -
-
```

SnapMirror unified replication basics

SnapMirror *unified replication* allows you to configure disaster recovery and archiving on the same destination volume. When unified replication is appropriate, it offers benefits in reducing the amount of secondary storage you need, limiting the number of baseline

transfers, and decreasing network traffic.

How unified data protection relationships are initialized

As with SnapMirror, unified data protection performs a baseline transfer the first time you invoke it. The SnapMirror policy for the relationship defines the contents of the baseline and any updates.

A baseline transfer under the default unified data protection policy `MirrorAndVault` makes a Snapshot copy of the source volume, then transfers that copy and the data blocks it references to the destination volume. Like vault archiving, unified data protection does not include older Snapshot copies in the baseline.

How unified data protection relationships are updated

At each update under the `MirrorAndVault` policy, SnapMirror creates a Snapshot copy of the source volume and transfers that Snapshot copy and any Snapshot copies that have been made since the last update, provided they have labels matching the labels defined in the Snapshot policy rules. In the following output from the `snapmirror policy show` command for the `MirrorAndVault` policy, note the following:

- `Create Snapshot` is “true”, indicating that `MirrorAndVault` creates a Snapshot copy when SnapMirror updates the relationship.
- `MirrorAndVault` has rules “sm_created”, “daily”, and “weekly”, indicating that both the Snapshot copy created by SnapMirror and the Snapshot copies with matching labels on the source are transferred when SnapMirror updates the relationship.

```
cluster_dst::> snapmirror policy show -policy MirrorAndVault -instance

                    Vserver: vs0
      SnapMirror Policy Name: MirrorAndVault
      SnapMirror Policy Type: mirror-vault
                Policy Owner: cluster-admin
                 Tries Limit: 8
           Transfer Priority: normal
   Ignore accesstime Enabled: false
     Transfer Restartability: always
 Network Compression Enabled: false
             Create Snapshot: true
                     Comment: A unified Synchronous SnapMirror and
SnapVault policy for
                              mirroring the latest file system and daily
and weekly snapshots.
       Total Number of Rules: 3
                  Total Keep: 59
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              sm_created              1  false       0 -
-
                              daily                   7  false       0 -
-
                              weekly                 52  false       0 -
-
```

Unified7year policy

The preconfigured `Unified7year` policy works exactly the same way as `MirrorAndVault`, except that a fourth rule transfers monthly Snapshot copies and retains them for seven years.

```
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              sm_created              1  false       0 -
-
                              daily                   7  false       0 -
-
                              weekly                 52  false       0 -
-
                              monthly                84  false       0 -
-
```

Protect against possible data corruption

Unified replication limits the contents of the baseline transfer to the Snapshot copy created by SnapMirror at initialization. At each update, SnapMirror creates another Snapshot copy of the source and transfers that Snapshot copy and any new Snapshot copies that have labels matching the labels defined in the Snapshot policy rules.

You can protect against the possibility that an updated Snapshot copy is corrupted by creating a copy of the last transferred Snapshot copy on the destination. This “local copy” is retained regardless of the retention rules on the source, so that even if the Snapshot originally transferred by SnapMirror is no longer available on the source, a copy of it will be available on the destination.

When to use unified data replication

You need to weigh the benefit of maintaining a full mirror against the advantages that unified replication offers in reducing the amount of secondary storage, limiting the number of baseline transfers, and decreasing network traffic.

The key factor in determining the appropriateness of unified replication is the rate of change of the active file system. A traditional mirror might be better suited to a volume holding hourly Snapshot copies of database transaction logs, for example.

XDP replaces DP as the SnapMirror default

Beginning with ONTAP 9.3, SnapMirror extended data protection (XDP) mode replaces SnapMirror data protection (DP) mode as the SnapMirror default.

Until ONTAP 9.3, SnapMirror invoked in DP mode and SnapMirror invoked in XDP mode used different replication engines, with different approaches to version-dependence:

- SnapMirror invoked in DP mode used a *version-dependent* replication engine in which the ONTAP version was required to be the same on primary and secondary storage:

```
cluster_dst::>  snapmirror create -type DP -source-path ... -destination
-path ...
```

- SnapMirror invoked in XDP mode used a *version-flexible* replication engine that supported different ONTAP versions on primary and secondary storage:

```
cluster_dst::>  snapmirror create -type XDP -source-path ...
-destination-path ...
```

With improvements in performance, the significant benefits of version-flexible SnapMirror outweigh the slight advantage in replication throughput obtained with version-dependent mode. For this reason, beginning with ONTAP 9.3, XDP mode has been made the new default, and any invocations of DP mode on the command line or in new or existing scripts are automatically converted to XDP mode.

Existing relationships are not affected. If a relationship is already of type DP, it will continue to be of type DP. Beginning with ONTAP 9.5, MirrorAndVault is the new default policy when no data protection mode is specified or when XDP mode is specified as the relationship type. The table below shows the behavior you can expect.

If you specify…	**The type is…**	**The default policy (if you do not specify a policy) is…**
DP	XDP	MirrorAllSnapshots (SnapMirror DR)
Nothing	XDP	MirrorAndVault (unified replication)
XDP	XDP	MirrorAndVault (unified replication)

As the table shows, the default policies assigned to XDP in different circumstances ensure that the conversion maintains the functional equivalence of the old types. Of course, you can use different policies as needed, including policies for unified replication:

If you specify…	**And the policy is…**	**The result is…**
DP	MirrorAllSnapshots	SnapMirror DR
	XDPDefault	SnapVault
	MirrorAndVault	Unified replication
XDP	MirrorAllSnapshots	SnapMirror DR
	XDPDefault	SnapVault
	MirrorAndVault	Unified replication

The only exceptions to conversion are as follows:

- SVM data protection relationships continue to default to DP mode in ONTAP 9.3 and earlier.

  Beginning with ONTAP 9.4, SVM data protection relationships default to XDP mode.

- Root volume load-sharing data protection relationships continue to default to DP mode.
- SnapLock data protection relationships continue to default to DP mode in ONTAP 9.4 and earlier.

  Beginning with ONTAP 9.5, SnapLock data protection relationships default to XDP mode.

- Explicit invocations of DP continue to default to DP mode if you set the following cluster-wide option:

  ```
  options replication.create_data_protection_rels.enable on
  ```

  This option is ignored if you do not explicitly invoke DP.

When a destination volume grows automatically

During a data protection mirror transfer, the destination volume grows automatically in size if the source volume has grown, provided there is available space in the aggregate that contains the volume.

This behavior occurs irrespective of any automatic growth setting on the destination. You cannot limit the volume's growth or prevent ONTAP from growing it.

By default, data protection volumes are set to the `grow_shrink` autosize mode, which enables the volume to grow or shrink in response to the amount of used space. The max-autosize for data protection volumes is equal to the maximum FlexVol size and is platform dependent. For example:

- FAS2220, default DP volume max-autosize = 60TB
- FAS6220, default DP volume max-autosize = 70TB
- FAS8200, default DP volume max-autosize = 100TB

For more information, see NetApp Hardware Universe.

Fan-out and cascade data protection deployments

You can use a *fan-out* deployment to extend data protection to multiple secondary systems. You can use a *cascade* deployment to extend data protection to tertiary systems.

Both fan-out and cascade deployments support any combination of SnapMirror DR, SnapVault, or unified replication; however, SnapMirror Synchronous relationships (supported beginning with ONTAP 9.5) support only fan-out deployments with one or more asynchronous SnapMirror relationships and do not support cascade deployments. Only one relationship in the fan-out configuration can be a SnapMirror Synchronous relationship, all the other relationships from the source volume must be asynchronous SnapMirror relationships. SnapMirror Business Continuity (supported beginning with ONTAP 9.8) also supports fan-out configurations.

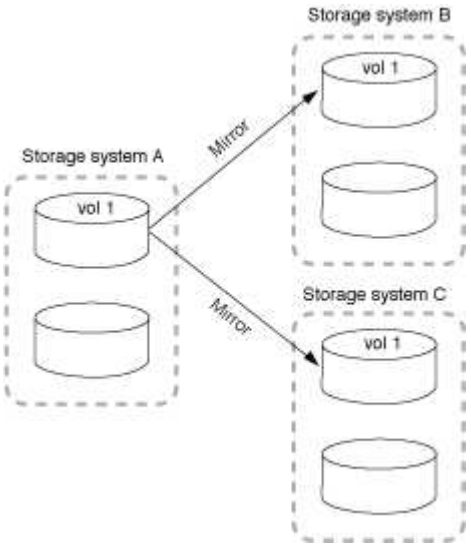
You can use a *fan-in* deployment to create data protection relationships between multiple primary systems and a single secondary system. Each relationship must use a different volume on the secondary system.

You should be aware that volumes that are part of a fan-out or cascade configuration can take longer to resynchronize. It is not uncommon to see the SnapMirror relationship reporting the status "preparing" for an extended time period.

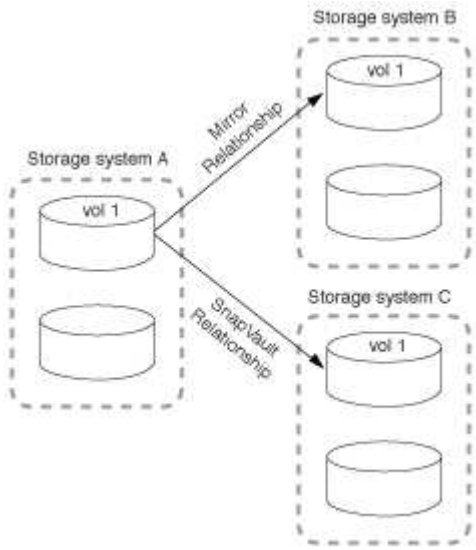
How fan-out deployments work

SnapMirror supports *multiple-mirrors* and *mirror-vault* fan-out deployments.

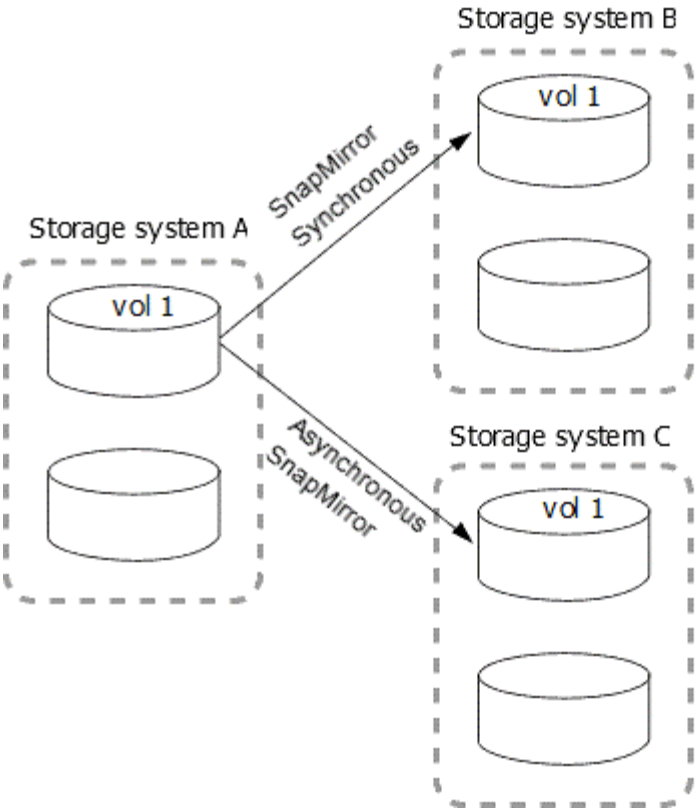
A multiple-mirrors fan-out deployment consists of a source volume that has a mirror relationship to multiple secondary volumes.



A mirror-vault fan-out deployment consists of a source volume that has a mirror relationship to a secondary volume and a SnapVault relationship to a different secondary volume.



Beginning with ONTAP 9.5, you can have fan-out deployments with SnapMirror Synchronous relationships; however, only one relationship in the fan-out configuration can be a SnapMirror Synchronous relationship, all the other relationships from the source volume must be asynchronous SnapMirror relationships.

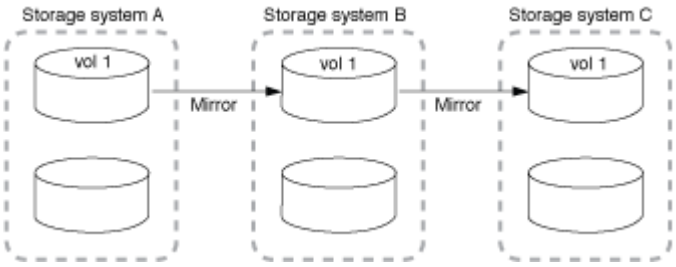


How cascade deployments work

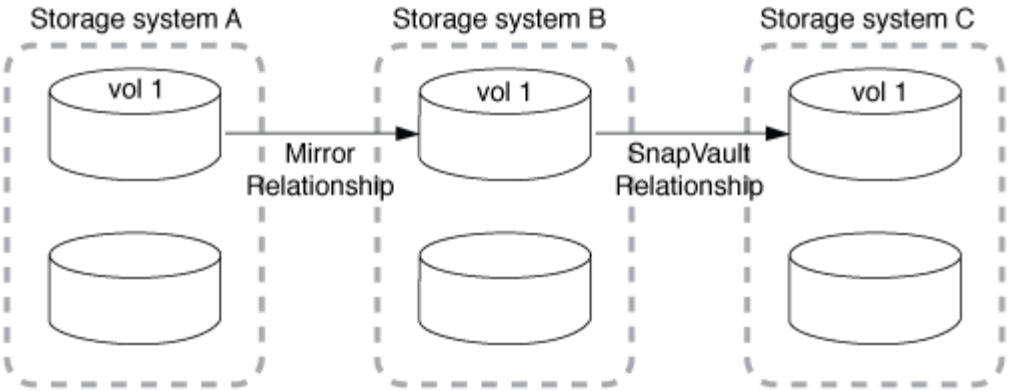
SnapMirror supports *mirror-mirror*, *mirror-vault*, *vault-mirror*, and *vault-vault* cascade deployments.

A mirror-mirror cascade deployment consists of a chain of relationships in which a source volume is mirrored to a secondary volume, and the secondary volume is mirrored to a tertiary volume. If the secondary volume becomes unavailable, you can synchronize the relationship between the primary and tertiary volumes without performing a new baseline transfer.

Beginning with ONTAP 9.6, SnapMirror Synchronous relationships are supported in a mirror-mirror cascade deployment. Only the primary and secondary volumes can be in a SnapMirror Synchronous relationship. The relationship between the secondary volumes and tertiary volumes must be asynchronous.



A mirror-vault cascade deployment consists of a chain of relationships in which a source volume is mirrored to a secondary volume, and the secondary volume is vaulted to a tertiary volume.



Vault-mirror and, beginning with ONTAP 9.2, vault-vault cascade deployments are also supported:

- A vault-mirror cascade deployment consists of a chain of relationships in which a source volume is vaulted to a secondary volume, and the secondary volume is mirrored to a tertiary volume.
- (Beginning with ONTAP 9.2) A vault-vault cascade deployment consists of a chain of relationships in which a source volume is vaulted to a secondary volume, and the secondary volume is vaulted to a tertiary volume.

Further Reading

- Resume protection in a fan-out configuration with SM-BC

SnapMirror licensing

SnapMirror licensing overview

Beginning with ONTAP 9.3, licensing has been simplified for replicating between ONTAP instances. In ONTAP 9 releases, the SnapMirror license supports both vault and mirror relationships. Users can now purchase a SnapMirror license to support ONTAP replication for both backup and disaster recovery use cases.

Prior to the ONTAP 9.3 release, two licenses were available to support different replication use cases. A SnapVault license was needed to configure *vault* relationships between ONTAP instances, where the DP instance could retain a higher number of Snapshot copies to support backup use cases where retention times are longer. A SnapMirror license was needed to configure *mirror* relationships between ONTAP instances, where each ONTAP instance would maintain the same number of snapshot copies (that is, a *mirror* image) to support disaster recovery use cases where cluster failovers would be possible. Both SnapMirror and SnapVault licenses can continue to be used and supported for ONTAP 8.x and 9.x releases.

SnapVault licenses continue to function and are supported for both ONTAP 8.x and 9.x releases, but they are no longer being sold. The SnapMirror license continues to be available and can be used in place of SnapVault and can be used for both mirror and vault configurations.

For ONTAP asynchronous replication, beginning with ONTAP 9.3 a single unified replication engine is used to configure extended data protection mode (XDP) policies, where the SnapMirror license can be configured for a mirror policy, a vault policy, or a mirror-vault policy. A SnapMirror license is required on both the source and destination clusters. A SnapVault license is not required if a SnapMirror license is already installed.

Data protection configuration limits are determined using several factors, including your ONTAP version, hardware platform, and the licenses installed. For more information, see Hardware Universe.

SnapMirror Synchronous license

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships are supported. You require the following licenses for creating a SnapMirror Synchronous relationship:

- The SnapMirror Synchronous license is required on both the source cluster and the destination cluster.

  The SnapMirror Synchronous license is enabled with either the Premium bundle or the Data Protection bundle.

  If your system was purchased before June 2019 with a Premium or Flash Bundle, you can download a NetApp master key to get the required SnapMirror Synchronous license from the NetApp Support Site: Master License Keys

- The SnapMirror license is required on both the source cluster and the destination cluster.

SnapMirror Cloud license

Beginning with ONTAP 9.8, the SnapMirror Cloud license provides asynchronous replication of Snapshot copies from ONTAP instances to object storage endpoints. Replication targets can be configured using both on-premises object stores as well as S3 and S3-compatible public cloud object storage services. SnapMirror Cloud relationships are supported from ONTAP systems to pre-qualified object storage targets. ONTAP 9.8 approved object storage targets include ONTAP S3, StorageGRID, AWS S3 Standard, S3 Standard-IA, and S3 One Zone-IA, Microsoft Azure Blob Premium, Hot and Cool, and GCP Standard and Nearline storage.

SnapMirror Cloud is not available as a standalone license and is available only with purchase of the Hybrid Cloud Bundle. Beginning with ONTAP 9.8, the Hybrid Cloud Bundle includes licenses for both SnapMirror Cloud and FabricPool. Similarly, the SnapMirror license is not available as a standalone license and is available only with purchase of the Data Protection Bundle.

You require the following licenses for creating a SnapMirror Cloud relationship:

- Both a SnapMirror license (purchased through Data Protection Bundle, or through Premium Bundle) and a SnapMirror Cloud license (purchased through Hybrid Cloud Bundle) is replicating directly to the object store endpoint.
- When configuring a multi-policy replication workflow (for example, Disk-to-Disk-to-Cloud), a SnapMirror license is required on all ONTAP instances, while the SnapMirror Cloud license is only required for the source cluster which is replicating directly to the object storage endpoint.

SnapMirror Cloud is an end user license which can be purchased from NetApp or from an approved NetApp reseller partner. The SnapMirror Cloud license provides end user entitlement but does not enable asynchronous ONTAP to object storage replication. To invoke ONTAP APIs for SnapMirror Cloud, a unique API key from an authorized application is required. Authorized and licensed applications used to orchestrate SnapMirror Cloud replication include System Manager, and are also available from multiple third-party application providers. These authorized applications will embed the unique API key to invoke ONTAP APIs. A combination of the SnapMirror Cloud end user license and an authorized third-party backup application is required to orchestrate and enable SnapMirror Cloud replication.

Beginning with ONTAP 9.9.1, you can use System Manager for SnapMirror Cloud replication. For more information, see Back up to the cloud.

A list of authorized SnapMirror Cloud third-party applications is published on the NetApp web site.

Data Protection Bundle

Beginning with ONTAP 9.1, new ONTAP data protection features were packaged with the FAS8200 as part of a solution called the Data Protection Bundle. This new hardware and software bundle included a new DP_Optimized (DPO) license that provided unique ONTAP features for secondary workloads. With the introduction of ONTAP 9.3 the DPO license increased the number of volumes per node from 1,000 to 1,500. Also introduced with ONTAP 9.3 were new configurations of the Data Protection Bundle based on configurations of FAS2620.

The DPO license was specifically designed for ONTAP clusters that were to be dedicated as secondary targets for SnapMirror replication. In addition to increasing the maximum volumes per node on the DPO controller, the DPO license also modified controller QoS settings to support greater replication traffic at the expense of application I/O. For this reason, the DPO license should never be installed on a cluster that supports application I/O, as application performance would be impacted. Later, Data Protection Bundles based on the FAS8200 and FAS2620 were offered as a solution and included programmatic free licenses based on the customer environment. When purchasing the solution bundles, free SnapMirror licenses would be provided for select older clusters which replicated to the DPO secondary. While the DPO license is needed on the Data Protection solution cluster, primary clusters from the following platform list would be provided free SnapMirror licenses. Primary clusters not included in this list would require purchase of SnapMirror licenses.

- FAS2200 Series
- FAS3000 Series
- FAS6000 Series
- FAS8000 Series

Data Protection Optimized (DPO) License

Data Protection hardware and software solution bundles introduced with ONTAP 9.1 and 9.3 were based on FAS8200 and FAS2620 only. As these platforms matured and new platforms were introduced new requests to support ONTAP features for secondary replication use cases increased. As a result, a new standalone DPO license was introduced in November 2018 with ONTAP 9.5 release.

The standalone DPO license was supported on both FAS and AFF platforms and could be purchased pre-configured with new clusters or added to deployed clusters as a software upgrade in the field. Because these new DPO licenses were not part of a hardware and software solution bundle they carried a lower price and free SnapMirror licenses for primary clusters were not provided. Secondary clusters configured with the a la carte DPO license must also purchase a SnapMirror license, and all primary clusters replicating to the DPO secondary cluster must purchase a SnapMirror license.

Additional ONTAP features were delivered with the DPO across multiple ONTAP releases.

Feature	**9.3**	**9.4**	**9.5**	**9.6**	**9.7+**
Max vols/node	1500	1500	1500	1500/2500	1500/2500
Max concurrent repl sessions	100	200	200	200	200
Workload bias*	client apps	Apps/SM	SnapMirror	SnapMirror	SnapMirror

Cross volume aggregate deduplication for HDD	No	Yes	Yes	Yes	Yes

- Details about priority for the SnapMirror backoff (workload bias) feature:
- Client: cluster I/O priority is set to client workloads (production apps), not SnapMirror traffic.
- Equality: SnapMirror replication requests have equal priority to I/O for production apps.
- SnapMirror: all SnapMirror I/O requests have higher priority that I/O for production apps.

Table 1: Max FlexVolumes per node across ONTAP releases

	9.3—9.5 Without DPO	**9.3—9.5 With DPO**	**9.6 Without DPO**	**9.6 With DPO**	**9.7—9.9.1 Without DPO**	**9.7—9..9.1 With DPO**
FAS2620	1000	1500	1000	1500	1000	1500
FAS2650	1000	1500	1000	1500	1000	1500
FAS2720	1000	1500	1000	1500	1000	1500
FAS2750	1000	1500	1000	1500	1000	1500
A200	1000	1500	1000	1500	1000	1500
A220	1000	1500	1000	1500	1000	1500
FAS8200/8300	1000	1500	1000	2500	1000	2500
A300	1000	1500	1000	2500	2500	2500
A400	1000	1500	1000	2500	2500	2500
FAS8700/9000	1000	1500	1000	2500	1000	2500
A700	1000	1500	1000	2500	2500	2500
A700s	1000	1500	1000	2500	2500	2500
A800	1000	1500	1000	2500	2500	2500

For the latest maximum FlexVol volume support for your configuration, see Hardware Universe.

Considerations for all new DPO installations

- After it is enabled, the DPO license feature cannot be disabled or undone.
- Installation of the DPO license requires a re-boot of ONTAP or failover to enable.
- The DPO solution is intended for secondary storage workloads; application workload performance on DPO clusters may be impacted
- The DPO license is supported on a select list of NetApp storage platform models.
- DPO features vary by ONTAP release. Refer to the compatibility table for reference.

Install a SnapMirror Cloud license

Beginning with ONTAP 9.8, SnapMirror Cloud provides asynchronous snapshot replication from ONTAP to object storage endpoints. SnapMirror Cloud relationships can only be configured using pre-qualified third-party backup applications. To configure ONTAP to object storage replication, both SnapMirror and SnapMirror Cloud licenses are required on the ONTAP source cluster configured for replication to the object store endpoint.

About this task

The SnapMirror Cloud license is a single-instance cluster-wide license, which means it does not need to be installed on every node in the cluster. It is a term-based license where both term and backup capacity are enforced. In addition to this end user license, SnapMirror Cloud requires an authorized and approved backup application to configure and invoke ONTAP APIs for replication. Both SnapMirror Cloud end user license and authorized app are necessary to utilize SnapMirror Cloud replication.

SnapMirror Cloud licenses are acquired through purchase of the Hybrid Cloud Bundle, which can be purchased with 1 or 3 year terms in 1 TB increments. The Hybrid Cloud Bundle includes a license for SnapMirror Cloud. Each license has a unique serial number. Purchases of the Hybrid Cloud Bundle are based on capacity, where the purchased capacity of the Hybrid Cloud Bundle is applied to the SnapMirror Cloud license.

The SnapMirror Cloud license can be installed on the cluster using the ONTAP command line or System Manager.

Steps

1. Download two NetApp License File (NLF) for SnapMirror Cloud from the NetApp Support Site.

   NetApp Support

2. Use System Manager to upload the SnapMirror Cloud NLF file to the cluster:
   a. Click **Configuration** > **Licenses**.
   b. In the **Cluster Settings** pane, click **Licenses**.
   c. In the **Packages** window, click **Add**.
   d. In the **Add License Packages** dialog box, click **Choose Files** to select the NLF you downloaded, and then click **Add** to upload the file to the cluster.

Related information

NetApp Software License Search

DPO systems feature enhancements

Beginning with ONTAP 9.6, the maximum number of FlexVol volumes supported increases when the DP_Optimized (DPO) license is installed. Beginning with ONTAP 9.4, systems with the DPO license support SnapMirror backoff, cross-volume background deduplication, cross-volume inline deduplication, use of Snapshot blocks as donors, and compaction.

Beginning with ONTAP 9.6, the maximum supported number of FlexVol volumes on secondary or data protection systems has increased, enabling you to scale up to 2,500 FlexVol volumes per node, or up to 5,000 in failover mode. The increase in FlexVol volumes is enabled with the DP_Optimized (DPO) license. A SnapMirror license is still required on both the source and destination nodes.

Beginning with ONTAP 9.4, the following feature enhancements are made to DPO systems:

- SnapMirror backoff: In DPO systems, replication traffic is given the same priority that client workloads are given.

  SnapMirror backoff is disabled by default on DPO systems.

- Volume background deduplication and cross-volume background deduplication: Volume background deduplication and cross-volume background deduplication are enabled in DPO systems.

  You can run the `storage aggregate efficiency cross-volume-dedupe start -aggregate` *`aggregate_name`* `-scan-old-data true` command to deduplicate the existing data. The best practice is to run the command during off-peak hours to reduce the impact on performance.

- Increased savings by using Snapshot blocks as donors: The data blocks that are not available in the active file system but are trapped in Snapshot copies are used as donors for volume deduplication.

  The new data can be deduplicated with the data that was trapped in Snapshot copies, effectively sharing the Snapshot blocks as well. The increased donor space provides more savings, especially when the volume has a large number of Snapshot copies.

- Compaction: Data compaction is enabled by default on DPO volumes.

Manage SnapMirror volume replication

SnapMirror replication workflow

SnapMirror offers three types of data protection relationship: SnapMirror DR, archive (previously known as SnapVault), and unified replication. You can follow the same basic workflow to configure each type of relationship.

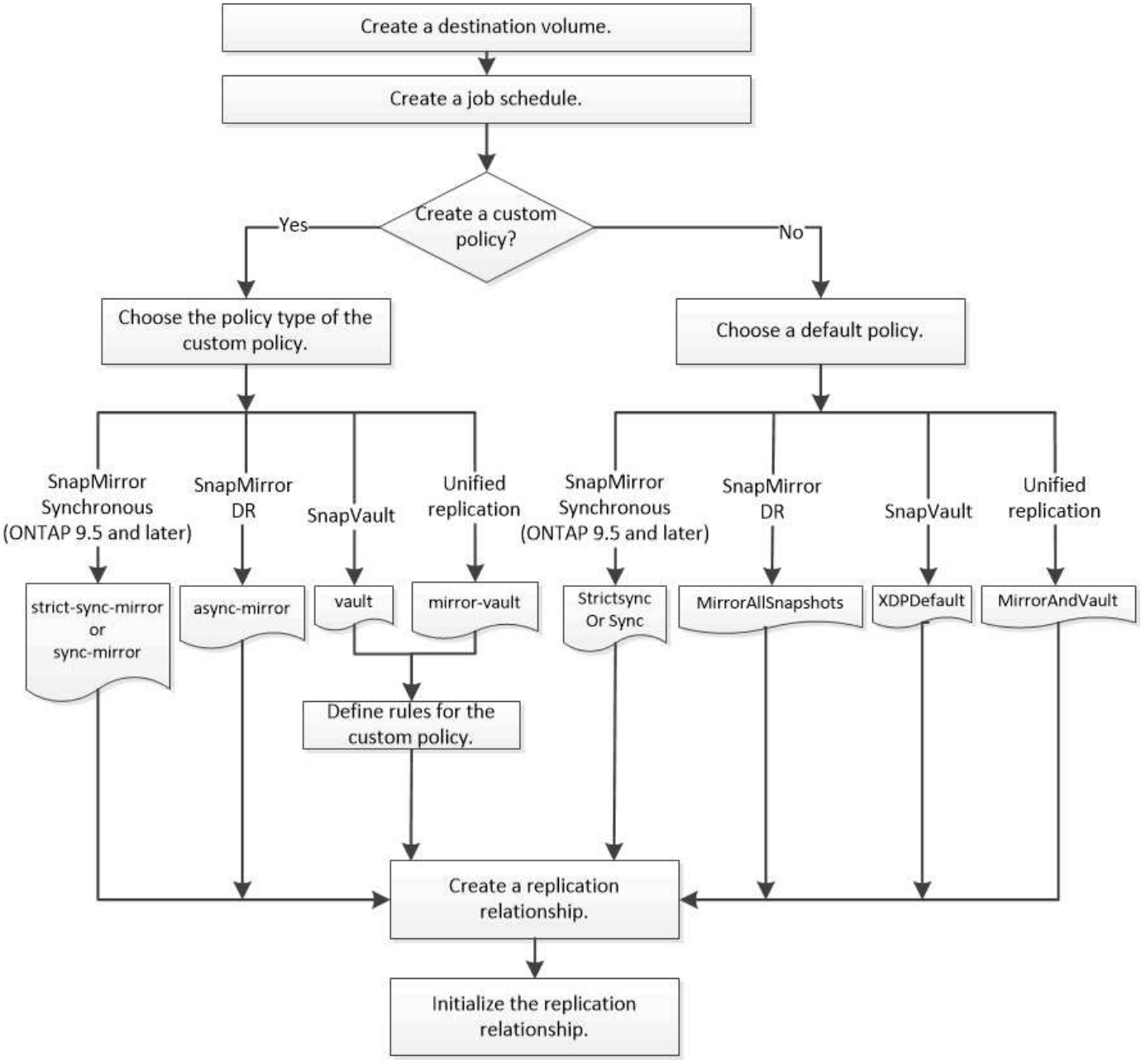
Beginning with general availability in ONTAP 9.9.1, SnapMirror Business Continuity (SM-BC) provides Zero Recovery Time Objective (Zero RTO) or Transparent Application Failover (TAF) to enable automatic failover of business-critical applications in SAN environments. SM-BC is supported in a configuration of either two AFF clusters or two All SAN Array (ASA) clusters.

NetApp Documentation: SnapMirror Business Continuity

For each type of SnapMirror data protection relationship, the workflow is the same: create a destination

volume, create a job schedule, specify a policy, create and initialize the relationship.

Beginning with ONTAP 9.3, you can use the `snapmirror protect` command to configure a data protection relationship in a single step. Even if you use `snapmirror protect`, you need to understand each step in the workflow.



Configure a replication relationship in one step

Beginning with ONTAP 9.3, you can use the `snapmirror protect` command to configure a data protection relationship in a single step. You specify a list of volumes to be replicated, an SVM on the destination cluster, a job schedule, and a SnapMirror policy. `snapmirror protect` does the rest.

What you'll need

- The source and destination clusters and SVMs must be peered.

- The language on the destination volume must be the same as the language on the source volume.

About this task

The `snapmirror protect` command chooses an aggregate associated with the specified SVM. If no aggregate is associated with the SVM, it chooses from all the aggregates in the cluster. The choice of aggregate is based on the amount of free space and the number of volumes on the aggregate.

The `snapmirror protect` command then performs the following steps:

- Creates a destination volume with an appropriate type and amount of reserved space for each volume in the list of volumes to be replicated.
- Configures a replication relationship appropriate for the policy you specify.
- Initializes the relationship.

The name of the destination volume is of the form *`source_volume_name_dst`*. In case of a conflict with an existing name, the command appends a number to the volume name. You can specify a prefix and/or suffix in the command options. The suffix replaces the system-supplied `dst` suffix.

In ONTAP 9.3 and earlier, a destination volume can contain up to 251 Snapshot copies. In ONTAP 9.4 and later, a destination volume can contain up to 1019 Snapshot copies.

Initialization can be time-consuming. `snapmirror protect` does not wait for initialization to complete before the job finishes. For this reason, you should use the `snapmirror show` command rather than the `job show` command to determine when initialization is complete.

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships can be created by using the `snapmirror protect` command.

Step

1. Create and initialize a replication relationship in one step:

   `snapmirror protect -path-list` *`SVM:volume|cluster://SVM/volume`*`, … -destination -vserver` *`destination_SVM`* `-policy` *`policy`* `-schedule` *`schedule`* `-auto-initialize true|false -destination-volume-prefix` *`prefix`* `-destination-volume-suffix` *`suffix`*

   You must run this command from the destination SVM or the destination cluster. The `-auto -initialize` option defaults to “true”.

   The following example creates and initializes a SnapMirror DR relationship using the default `MirrorAllSnapshots` policy:

```
cluster_dst::> snapmirror protect -path-list svm1:volA, svm1:volB
-destination-vserver svm_backup -policy MirrorAllSnapshots -schedule
replication_daily
```

You can use a custom policy if you prefer. For more information, see Creating a custom replication policy.

The following example creates and initializes a SnapVault relationship using the default `XDPDefault` policy:

```
cluster_dst::> snapmirror protect -path-list svm1:volA, svm1:volB
-destination-vserver svm_backup -policy XDPDefault -schedule
replication_daily
```

The following example creates and initializes a unified replication relationship using the default `MirrorAndVault` policy:

```
cluster_dst::> snapmirror protect -path-list svm1:volA, svm1:volB
-destination-vserver svm_backup -policy MirrorAndVault
```

The following example creates and initializes a SnapMirror Synchronous relationship using the default `Sync` policy:

```
cluster_dst::> snapmirror protect -path-list svm1:volA, svm1:volB
-destination-vserver svm_sync -policy Sync
```

For SnapVault and unified replication policies, you might find it useful to define a schedule for creating a copy of the last transferred Snapshot copy on the destination. For more information, see Defining a schedule for creating a local copy on the destination.

After you finish

Use the `snapmirror show` command to verify that the SnapMirror relationship was created. For complete command syntax, see the man page.

Configure a replication relationship one step at a time

Create a destination volume

You can use the `volume create` command on the destination to create a destination volume. The destination volume should be the same or greater in size than the source volume.

Step

1. Create a destination volume:

   `volume create -vserver` *`SVM`* `-volume volume -aggregate` *`aggregate`* `-type DP -size` *`size`*

   For complete command syntax, see the man page.

The following example creates a 2-GB destination volume named `volA_dst`:

```
cluster_dst::> volume create -vserver SVM_backup -volume volA_dst
-aggregate node01_aggr -type DP -size 2GB
```

Create a replication job schedule

You can use the `job schedule cron create` command to create a replication job schedule. The job schedule determines when SnapMirror automatically updates the data protection relationship to which the schedule is assigned.

About this task

You assign a job schedule when you create a data protection relationship. If you do not assign a job schedule, you must update the relationship manually.

Step

1. Create a job schedule:

   `job schedule cron create -name` *`job_name`* `-month` *`month`* `-dayofweek` *`day_of_week`* `-day` *`day_of_month`* `-hour` *`hour`* `-minute` *`minute`*

   For `-month`, `-dayofweek`, and `-hour`, you can specify `all` to run the job every month, day of the week, and hour, respectively.

   Beginning with ONTAP 9.10.1, you can include the Vserver for your job schedule:

   `job schedule cron create -name` *`job_name`* `-vserver` *`Vserver_name`* `-month` *`month`* `-dayofweek` *`day_of_week`* `-day` *`day_of_month`* `-hour` *`hour`* `-minute` *`minute`*

   The following example creates a job schedule named `my_weekly` that runs on Saturdays at 3:00 a.m.:

   ```
   cluster_dst::> job schedule cron create -name my_weekly -dayofweek
   "Saturday" -hour 3 -minute 0
   ```

Customize a replication policy

Create a custom replication policy

You can create a custom replication policy if the default policy for a relationship is not suitable. You might want to compress data in a network transfer, for example, or modify the number of attempts SnapMirror makes to transfer Snapshot copies.

You can use a default or custom policy when you create a replication relationship. For a custom archive (formerly SnapVault) or unified replication policy, you must define one or more *rules* that determine which Snapshot copies are transferred during initialization and update.You might also want to define a schedule for creating local Snapshot copies on the destination.

The *policy type* of the replication policy determines the type of relationship it supports. The table below shows the available policy types.

Policy type	Relationship type
async-mirror	SnapMirror DR
vault	SnapVault
mirror-vault	Unified replication
strict-sync-mirror	SnapMirror Synchronous in the StrictSync mode (supported beginning with ONTAP 9.5)
sync-mirror	SnapMirror Synchronous in the Sync mode (supported beginning with ONTAP 9.5)

When you create a custom replication policy, it is a good idea to model the policy after a default policy.

Step

1. Create a custom replication policy:

   `snapmirror policy create -vserver` *`SVM`* `-policy` *`policy`* `-type async-mirror|vault|mirror-vault|strict-sync-mirror|sync-mirror -comment` *`comment`* `-tries` *`transfer_tries`* `-transfer-priority low|normal -is-network-compression -enabled true|false`

   For complete command syntax, see the man page.

   Beginning with ONTAP 9.5, you can specify the schedule for creating a common Snapshot copy schedule for SnapMirror Synchronous relationships by using the `-common-snapshot-schedule` parameter. By default, the common Snapshot copy schedule for SnapMirror Synchronous relationships is one hour. You can specify a value from 30 minutes to two hours for the Snapshot copy schedule for SnapMirror Synchronous relationships.

   The following example creates a custom replication policy for SnapMirror DR that enables network compression for data transfers:

   ```
   cluster_dst::> snapmirror policy create -vserver svm1 -policy
   DR_compressed -type async-mirror -comment "DR with network compression
   enabled" -is-network-compression-enabled true
   ```

   The following example creates a custom replication policy for SnapVault:

   ```
   cluster_dst::> snapmirror policy create -vserver svm1 -policy
   my_snapvault -type vault
   ```

The following example creates a custom replication policy for unified replication:

```
cluster_dst::> snapmirror policy create -vserver svm1 -policy my_unified
-type mirror-vault
```

The following example creates a custom replication policy for SnapMirror Synchronous relationship in the StrictSync mode:

```
cluster_dst::> snapmirror policy create -vserver svm1 -policy
my_strictsync -type strict-sync-mirror -common-snapshot-schedule
my_sync_schedule
```

After you finish

For “vault” and “mirror-vault” policy types, you must define rules that determine which Snapshot copies are transferred during initialization and update.

Use the `snapmirror policy show` command to verify that the SnapMirror policy was created. For complete command syntax, see the man page.

Define a rule for a policy

For custom policies with the “vault” or “mirror-vault” policy type, you must define at least one rule that determines which Snapshot copies are transferred during initialization and update. You can also define rules for default policies with the “vault” or"`mirror-vault`" policy type.

About this task

Every policy with the “vault” or “mirror-vault” policy type must have a rule that specifies which Snapshot copies to replicate. The rule “bi-monthly”, for example, indicates that only Snapshot copies assigned the SnapMirror label “bi-monthly” should be replicated. You specify the SnapMirror label when you configure the Snapshot policy on the source.

Each policy type is associated with one or more system-defined rules. These rules are automatically assigned to a policy when you specify its policy type. The table below shows the system-defined rules.

System-defined rule	Used in policy types	Result
sm_created	async-mirror, mirror-vault, Sync, StrictSync	A Snapshot copy created by SnapMirror is transferred on initialization and update.
all_source_snapshots	async-mirror	New Snapshot copies on the source are transferred on initialization and update.

daily	vault,mirror-vault	New Snapshot copies on the source with the SnapMirror label “daily” are transferred on initialization and update.
weekly	vault,mirror-vault	New Snapshot copies on the source with the SnapMirror label “weekly” are transferred on initialization and update.
monthly	mirror-vault	New Snapshot copies on the source with the SnapMirror label “monthly” are transferred on initialization and update.
app_consistent	Sync, StrictSync	Snapshot copies with the SnapMirror label “app_consistent” on source are synchronously replicated to the destination. Supported Beginning with ONTAP 9.7.

Except for the “async-mirror” policy type, you can specify additional rules as needed, for default or custom policies. For example:

- For the default `MirrorAndVault` policy, you might create a rule called “bi-monthly” to match Snapshot copies on the source with the “bi-monthly” SnapMirror label.
- For a custom policy with the “mirror-vault” policy type, you might create a rule called “bi-weekly” to match Snapshot copies on the source with the “bi-weekly” SnapMirror label.

Step

1. Define a rule for a policy:

   `snapmirror policy add-rule -vserver` *`SVM`* `-policy` *`policy_for_rule`* `-snapmirror -label` *`snapmirror-label`* `-keep` *`retention_count`*

   For complete command syntax, see the man page.

   The following example adds a rule with the SnapMirror label `bi-monthly` to the default `MirrorAndVault` policy:

   ```
   cluster_dst::> snapmirror policy add-rule -vserver svm1 -policy
   MirrorAndVault -snapmirror-label bi-monthly -keep 6
   ```

   The following example adds a rule with the SnapMirror label `bi-weekly` to the custom `my_snapvault` policy:

```
cluster_dst::> snapmirror policy add-rule -vserver svm1 -policy
my_snapvault -snapmirror-label bi-weekly -keep 26
```

The following example adds a rule with the SnapMirror label `app_consistent` to the custom `Sync` policy:

```
cluster_dst::> snapmirror policy add-rule -vserver svm1 -policy Sync
-snapmirror-label app_consistent -keep 1
```

You can then replicate Snapshot copies from the source cluster that match this SnapMirror label:

```
cluster_src::> snapshot create -vserver vs1 -volume vol1 -snapshot
snapshot1 -snapmirror-label app_consistent
```

Define a schedule for creating a local copy on the destination

For SnapVault and unified replication relationships, you can protect against the possibility that an updated Snapshot copy is corrupted by creating a copy of the last transferred Snapshot copy on the destination. This “local copy” is retained regardless of the retention rules on the source, so that even if the Snapshot originally transferred by SnapMirror is no longer available on the source, a copy of it will be available on the destination.

About this task

You specify the schedule for creating a local copy in the `-schedule` option of the `snapmirror policy add-rule` command.

Step

1. Define a schedule for creating a local copy on the destination:

   `snapmirror policy add-rule -vserver` *`SVM`* `-policy` *`policy_for_rule`* `-snapmirror -label` *`snapmirror-label`* `-schedule` *`schedule`*

   For complete command syntax, see the man page. For an example of how to create a job schedule, see Creating a replication job schedule.

   The following example adds a schedule for creating a local copy to the default `MirrorAndVault` policy:

   ```
   cluster_dst::> snapmirror policy add-rule -vserver svm1 -policy
   MirrorAndVault -snapmirror-label my_monthly -schedule my_monthly
   ```

   The following example adds a schedule for creating a local copy to the custom `my_unified` policy:

```
cluster_dst::> snapmirror policy add-rule -vserver svm1 -policy
my_unified -snapmirror-label my_monthly -schedule my_monthly
```

Create a replication relationship

The relationship between the source volume in primary storage and the destination volume in secondary storage is called a *data protection relationship.* You can use the `snapmirror create` command to create SnapMirror DR, SnapVault, or unified replication data protection relationships.

What you'll need

- The source and destination clusters and SVMs must be peered.

  Cluster and SVM peering

- The language on the destination volume must be the same as the language on the source volume.

About this task

Until ONTAP 9.3, SnapMirror invoked in DP mode and SnapMirror invoked in XDP mode used different replication engines, with different approaches to version-dependence:

- SnapMirror invoked in DP mode used a *version-dependent* replication engine in which the ONTAP version was required to be the same on primary and secondary storage:

  ```
  cluster_dst::>  snapmirror create -type DP -source-path ... -destination
  -path ...
  ```

- SnapMirror invoked in XDP mode used a *version-flexible* replication engine that supported different ONTAP versions on primary and secondary storage:

  ```
  cluster_dst::>  snapmirror create -type XDP -source-path ...
  -destination-path ...
  ```

With improvements in performance, the significant benefits of version-flexible SnapMirror outweigh the slight advantage in replication throughput obtained with version-dependent mode. For this reason, beginning with ONTAP 9.3, XDP mode has been made the new default, and any invocations of DP mode on the command line or in new or existing scripts are automatically converted to XDP mode.

Existing relationships are not affected. If a relationship is already of type DP, it will continue to be of type DP. The table below shows the behavior you can expect.

If you specify…	The type is…	The default policy (if you do not specify a policy) is…
DP	XDP	MirrorAllSnapshots (SnapMirror DR)

Nothing	XDP	MirrorAllSnapshots (SnapMirror DR)
XDP	XDP	XDPDefault (SnapVault)

See also the examples in the procedure below.

The only exceptions to conversion are as follows:

- SVM data protection relationships continue to default to DP mode.

  Specify XDP explicitly to obtain XDP mode with the default `MirrorAllSnapshots` policy.

- Load-sharing data protection relationships continue to default to DP mode.
- SnapLock data protection relationships continue to default to DP mode.
- Explicit invocations of DP continue to default to DP mode if you set the following cluster-wide option:

  ```
  options replication.create_data_protection_rels.enable on
  ```

  This option is ignored if you do not explicitly invoke DP.

In ONTAP 9.3 and earlier, a destination volume can contain up to 251 Snapshot copies. In ONTAP 9.4 and later, a destination volume can contain up to 1019 Snapshot copies.

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships are supported.

Step

1. From the destination cluster, create a replication relationship:

   `snapmirror create -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, … -type DP|XDP -schedule` *`schedule`* `-policy` *`policy`*

   For complete command syntax, see the man page.

   > The `schedule` parameter is not applicable when creating SnapMirror Synchronous relationships.

   The following example creates a SnapMirror DR relationship using the default `MirrorLatest` policy:

   ```
   cluster_dst::> snapmirror create -source-path svm1:volA -destination
   -path svm_backup:volA_dst -type XDP -schedule my_daily -policy
   MirrorLatest
   ```

   The following example creates a SnapVault relationship using the default `XDPDefault` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -schedule my_daily -policy
XDPDefault
```

The following example creates a unified replication relationship using the default `MirrorAndVault` policy:

```
cluster_dst:> snapmirror create -source-path svm1:volA -destination-path
svm_backup:volA_dst -type XDP -schedule my_daily -policy MirrorAndVault
```

The following example creates a unified replication relationship using the custom `my_unified` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -schedule my_daily -policy
my_unified
```

The following example creates a SnapMirror Synchronous relationship using the default `Sync` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -policy Sync
```

The following example creates a SnapMirror Synchronous relationship using the default `StrictSync` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -policy StrictSync
```

The following example creates a SnapMirror DR relationship. With the DP type automatically converted to XDP and with no policy specified, the policy defaults to the `MirrorAllSnapshots` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type DP -schedule my_daily
```

The following example creates a SnapMirror DR relationship. With no type or policy specified, the policy defaults to the `MirrorAllSnapshots` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -schedule my_daily
```

The following example creates a SnapMirror DR relationship. With no policy specified, the policy defaults to the `XDPDefault` policy:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -schedule my_daily
```

The following example creates a SnapMirror Synchronous relationship with the predefined policy `SnapCenterSync`:

```
cluster_dst::> snapmirror create -source-path svm1:volA -destination
-path svm_backup:volA_dst -type XDP -policy SnapCenterSync
```

The predefined policy `SnapCenterSync` is of type `Sync`. This policy replicates any Snapshot copy that is created with the `snapmirror-label` of "app_consistent".

After you finish

Use the `snapmirror show` command to verify that the SnapMirror relationship was created. For complete command syntax, see the man page.

Other ways to do this in ONTAP

To perform these tasks with…	See this content…
The redesigned System Manager (available with ONTAP 9.7 and later)	Configure mirrors and vaults
System Manager Classic (available with ONTAP 9.7 and earlier)	Volume backup using SnapVault overview

Initialize a replication relationship

For all relationship types, initialization performs a *baseline transfer*: it makes a Snapshot copy of the source volume, then transfers that copy and all the data blocks it references to the destination volume. Otherwise, the contents of the transfer depend on the policy.

What you'll need

The source and destination clusters and SVMs must be peered.

Cluster and SVM peering

About this task

Initialization can be time-consuming. You might want to run the baseline transfer in off-peak hours.

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships are supported.

Step

1. Initialize a replication relationship:

```
snapmirror initialize -source-path SVM:volume|cluster://SVM/volume, …
-destination-path SVM:volume|cluster://SVM/volume, …
```

For complete command syntax, see the man page.

> You must run this command from the destination SVM or the destination cluster.

The following example initializes the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

```
cluster_dst::> snapmirror initialize -source-path svm1:volA -destination
-path svm_backup:volA_dst
```

Example: Configure a vault-vault cascade

An example will show in concrete terms how you can configure replication relationships one step at a time. You can use the vault-vault cascade deployment configured in the example to retain more than 251 Snapshot copies labeled “my-weekly”.

What you’ll need

- The source and destination clusters and SVMs must be peered.
- You must be running ONTAP 9.2 or later. Vault-vault cascades are not supported in earlier ONTAP releases.

About this task

The example assumes the following:

- You have configured Snapshot copies on the source cluster with the SnapMirror labels “my-daily”, “my-weekly”, and “my-monthly”.
- You have configured destination volumes named “volA” on the secondary and tertiary destination clusters.
- You have configured replication job schedules named “my_snapvault” on the secondary and tertiary destination clusters.

The example shows how to create replication relationships based on two custom policies:

- The “snapvault_secondary” policy retains 7 daily, 52 weekly, and 180 monthly Snapshot copies on the secondary destination cluster.
- The “snapvault_tertiary policy” retains 250 weekly Snapshot copies on the tertiary destination cluster.

Steps

1. On the secondary destination cluster, create the “snapvault_secondary” policy:

   ```
   cluster_secondary::> snapmirror policy create -policy snapvault_secondary
   -type vault -comment “Policy on secondary for vault to vault cascade” -vserver
   svm_secondary
   ```

2. On the secondary destination cluster, define the “my-daily” rule for the policy:

   ```
   cluster_secondary::> snapmirror policy add-rule -policy snapvault_secondary
   -snapmirror-label my-daily -keep 7 -vserver svm_secondary
   ```

3. On the secondary destination cluster, define the “my-weekly” rule for the policy:

```
cluster_secondary::> snapmirror policy add-rule -policy snapvault_secondary
-snapmirror-label my-weekly -keep 52 -vserver svm_secondary
```

4. On the secondary destination cluster, define the “my-monthly” rule for the policy:

```
cluster_secondary::> snapmirror policy add-rule -policy snapvault_secondary
-snapmirror-label my-monthly -keep 180 -vserver svm_secondary
```

5. On the secondary destination cluster, verify the policy:

```
cluster_secondary::> snapmirror policy show snapvault_secondary -instance
```

```
                      Vserver: svm_secondary
      SnapMirror Policy Name: snapvault_secondary
      SnapMirror Policy Type: vault
                Policy Owner: cluster-admin
                 Tries Limit: 8
           Transfer Priority: normal
   Ignore accesstime Enabled: false
     Transfer Restartability: always
 Network Compression Enabled: false
             Create Snapshot: false
                     Comment: Policy on secondary for vault to vault
cascade
       Total Number of Rules: 3
                  Total Keep: 239
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              my-daily                7  false       0 -
-
                              my-weekly              52  false       0 -
-
                              my-monthly            180  false       0 -
-
```

6. On the secondary destination cluster, create the relationship with the source cluster:

```
cluster_secondary::> snapmirror create -source-path svm_primary:volA
-destination-path svm_secondary:volA -type XDP -schedule my_snapvault -policy
snapvault_secondary
```

7. On the secondary destination cluster, initialize the relationship with the source cluster:

```
cluster_secondary::> snapmirror initialize -source-path svm_primary:volA
```

```
-destination-path svm_secondary:volA
```

8. On the tertiary destination cluster, create the “snapvault_tertiary” policy:

```
cluster_tertiary::> snapmirror policy create -policy snapvault_tertiary -type
vault -comment “Policy on tertiary for vault to vault cascade” -vserver
svm_tertiary
```

9. On the tertiary destination cluster, define the “my-weekly” rule for the policy:

```
cluster_tertiary::> snapmirror policy add-rule -policy snapvault_tertiary
-snapmirror-label my-weekly -keep 250 -vserver svm_tertiary
```

10. On the tertiary destination cluster, verify the policy:

```
cluster_tertiary::> snapmirror policy show snapvault_tertiary -instance
```

```
                    Vserver: svm_tertiary
      SnapMirror Policy Name: snapvault_tertiary
      SnapMirror Policy Type: vault
                Policy Owner: cluster-admin
                 Tries Limit: 8
            Transfer Priority: normal
   Ignore accesstime Enabled: false
     Transfer Restartability: always
 Network Compression Enabled: false
             Create Snapshot: false
                     Comment: Policy on tertiary for vault to vault
cascade
       Total Number of Rules: 1
                  Total Keep: 250
                       Rules: SnapMirror Label     Keep  Preserve Warn
Schedule Prefix
                              ----------------     ----  -------- ----
-------- ------
                              my-weekly             250  false       0 -
-
```

11. On the tertiary destination cluster, create the relationship with the secondary cluster:

```
cluster_tertiary::> snapmirror create -source-path svm_secondary:volA
-destination-path svm_tertiary:volA -type XDP -schedule my_snapvault -policy
snapvault_tertiary
```

12. On the tertiary destination cluster, initialize the relationship with the secondary cluster:

```
cluster_tertiary::> snapmirror initialize -source-path svm_secondary:volA
-destination-path svm_tertiary:volA
```

Convert an existing DP-type relationship to XDP

You can easily convert an existing DP-type relationship to XDP to take advantage of version-flexible SnapMirror.

About this task

- SnapMirror does not automatically convert existing DP-type relationships to XDP. To convert the relationship, you need to break and delete the existing relationship, create a new XDP relationship, and resync the relationship. For background information, see XDP replaces DP as the SnapMirror default.
- When planning your conversion, you should be aware that background preparation and the data warehousing phase of an XDP SnapMirror relationship can take a long time. It is not uncommon to see the SnapMirror relationship reporting the status "preparing" for an extended time period.

> After you convert a SnapMirror relationship type from DP to XDP, space-related settings, such as autosize and space guarantee are no longer replicated to the destination.

Steps

1. Quiesce the existing DP-type relationship:

   `snapmirror quiesce -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following example quiesces the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror quiesce -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

2. Break the existing DP-type relationship:

   `snapmirror break -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following example breaks the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror break -source-path svm1:volA -destination-path
   svm_backup:volA_dst
   ```

3. Disable automatic deletion of Snapshot copies on the destination volume:

   `volume snapshot autodelete modify -vserver SVM -volume volume -enabled false`

   The following example disables Snapshot copy autodelete on the destination volume `volA_dst`:

   ```
   cluster_dst::> volume snapshot autodelete modify -vserver svm_backup
   -volume volA_dst -enabled false
   ```

4. Delete the existing DP-type relationship:

   `snapmirror delete -source-path SVM:volume|cluster://SVM/volume, … -destination -path SVM:volume|cluster://SVM/volume, …`

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster.

   The following example deletes the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror delete -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

5. Create the new XDP-type relationship:

   `snapmirror create -source-path SVM:volume|cluster://SVM/volume, … -destination -path SVM:volume|cluster://SVM/volume, … -type XDP -schedule schedule -policy policy`

   The new relationship must use the same source and destination volume. For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster.

   The following example creates a SnapMirror DR relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup` using the default `MirrorAllSnapshots` policy:

   ```
   cluster_dst::> snapmirror create -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   -type XDP -schedule my_daily -policy MirrorAllSnapshots
   ```

6. Resync the source and destination volumes:

   `snapmirror resync -source-path SVM:volume|cluster://SVM/volume, … -destination -path SVM:volume|cluster://SVM/volume, …`

For complete command syntax, see the man page.

> You must run this command from the destination SVM or the destination cluster. Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

The following example resyncs the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

```
cluster_dst::> snapmirror resync -source-path svm1:volA -destination
-path svm_backup:volA_dst
```

After you finish

Use the `snapmirror show` command to verify that the SnapMirror relationship was created. For complete command syntax, see the man page.

Convert the type of a SnapMirror relationship

Beginning with ONTAP 9.5, SnapMirror Synchronous is supported. You can convert an asynchronous SnapMirror relationship to a SnapMirror Synchronous relationship or vice versa without performing a baseline transfer.

About this task

You cannot convert an asynchronous SnapMirror relationship to a SnapMirror Synchronous relationship or vice versa by changing the SnapMirror policy

Steps

- **Converting an asynchronous SnapMirror relationship to a SnapMirror Synchronous relationship**
  a. From the destination cluster, delete the asynchronous SnapMirror relationship:

     `snapmirror delete -destination-path` *`SVM:volume`*

     ```
     cluster2::>snapmirror delete -destination-path vs1_dr:vol1
     ```

  b. From the source cluster, release the SnapMirror relationship without deleting the common Snapshot copies:

     `snapmirror release -relationship-info-only true -destination-path` *`dest_SVM:dest_volume`*

     ```
     cluster1::>snapmirror release -relationship-info-only true
     -destination-path vs1_dr:vol1
     ```

  c. From the destination cluster, create a SnapMirror Synchronous relationship:

```
snapmirror create -source-path src_SVM:src_volume -destination-path
dest_SVM:dest_volume -policy sync-mirror
```

```
cluster2::>snapmirror create -source-path vs1:vol1 -destination-path
vs1_dr:vol1 -policy sync
```

d. Resynchronize the SnapMirror Synchronous relationship:

```
snapmirror resync -destination-path dest_SVM:dest_volume
```

```
cluster2::>snapmirror resync -destination-path vs1_dr:vol1
```

- **Converting a SnapMirror Synchronous relationship to an asynchronous SnapMirror relationship**

a. From the destination cluster, quiesce the existing SnapMirror Synchronous relationship:

```
snapmirror quiesce -destination-path dest_SVM:dest_volume
```

```
cluster2::> snapmirror quiesce -destination-path vs1_dr:vol1
```

b. From the destination cluster, delete the asynchronous SnapMirror relationship:

```
snapmirror delete -destination-path SVM:volume
```

```
cluster2::>snapmirror delete -destination-path vs1_dr:vol1
```

c. From the source cluster, release the SnapMirror relationship without deleting the common Snapshot copies:

```
snapmirror release -relationship-info-only true -destination-path
dest_SVM:dest_volume
```

```
cluster1::>snapmirror release -relationship-info-only true
-destination-path vs1_dr:vol1
```

d. From the destination cluster, create an asynchronous SnapMirror relationship:

```
snapmirror create -source-path src_SVM:src_volume -destination-path
dest_SVM:dest_volume -policy MirrorAllSnapshots
```

```
cluster2::>snapmirror create -source-path vs1:vol1 -destination-path
vs1_dr:vol1 -policy sync
```

e. Resynchronize the SnapMirror Synchronous relationship:

`snapmirror resync -destination-path dest_SVM:dest_volume`

```
cluster2::>snapmirror resync -destination-path vs1_dr:vol1
```

Convert the mode of a SnapMirror Synchronous relationship

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships are supported. You can convert the mode of a SnapMirror Synchronous relationship from StrictSync to Sync or vice versa.

About this task

You cannot modify the policy of a Snapmirror Synchronous relationship to convert its mode.

Steps

1. From the destination cluster, quiesce the existing SnapMirror Synchronous relationship:

   `snapmirror quiesce -destination-path dest_SVM:dest_volume`

   ```
   cluster2::> snapmirror quiesce -destination-path vs1_dr:vol1
   ```

2. From the destination cluster, delete the existing SnapMirror Synchronous relationship:

   `snapmirror delete -destination-path dest_SVM:dest_volume`

   ```
   cluster2::> snapmirror delete -destination-path vs1_dr:vol1
   ```

3. From the source cluster, release the SnapMirror relationship without deleting the common Snapshot copies:

   `snapmirror release -relationship-info-only true -destination-path dest_SVM:dest_volume`

   ```
   cluster1::> snapmirror release -relationship-info-only true -destination
   -path vs1_dr:vol1
   ```

4. From the destination cluster, create a SnapMirror Synchronous relationship by specifying the mode to which you want to convert the SnapMirror Synchronous relationship:

   `snapmirror create -source-path vs1:vol1 -destination-path dest_SVM:dest_volume -policy Sync|StrictSync`

```
cluster2::> snapmirror create -source-path vs1:vol1 -destination-path
vs1_dr:vol1 -policy Sync
```

5. From the destination cluster, resynchronize the SnapMirror relationship:

   `snapmirror resync -destination-path dest_SVM:dest_volume`

```
cluster2::> snapmirror resync -destination-path vs1_dr:vol1
```

Serve data from a SnapMirror DR destination volume

Make the destination volume writeable

You need to make the destination volume writeable before you can serve data from the volume to clients. You can use the `snapmirror quiesce` command to stop scheduled transfers to the destination, the `snapmirror abort` command to stop ongoing transfers, and the `snapmirror break` command to make the destination writeable.

About this task

You must perform this task from the destination SVM or the destination cluster.

Steps

1. Stop scheduled transfers to the destination:

   `snapmirror quiesce -source-path` *`SVM:volume|cluster://SVM/volume`*`, …`
   `-destination-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   The following example stops scheduled transfers between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

```
cluster_dst::> snapmirror quiesce -source-path svm1:volA -destination
-path svm_backup:volA_dst
```

2. Stop ongoing transfers to the destination:

   `snapmirror abort -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination`
   `-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   This step is not required for SnapMirror Synchronous relationships (supported beginning with ONTAP 9.5).

The following example stops ongoing transfers between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

```
cluster_dst::> snapmirror abort -source-path svm1:volA -destination-path
svm_backup:volA_dst
```

3. Break the SnapMirror DR relationship:

   `snapmirror break -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   The following example breaks the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror break -source-path svm1:volA -destination-path
   svm_backup:volA_dst
   ```

Other ways to do this in ONTAP

To perform these tasks with…	See this content…
The redesigned System Manager (available with ONTAP 9.7 and later)	Serve data from a SnapMirror destination
System Manager Classic (available with ONTAP 9.7 and earlier)	Volume disaster recovery overview

Configure the destination volume for data access

After making the destination volume writeable, you must configure the volume for data access. NAS clients, NVMe subsystem, andSAN hosts can access the data from the destination volume until the source volume is reactivated.

NAS environment:

1. Mount the NAS volume to the namespace using the same junction path that the source volume was mounted to in the source SVM.
2. Apply the appropriate ACLs to the SMB shares at the destination volume.
3. Assign the NFS export policies to the destination volume.
4. Apply the quota rules to the destination volume.
5. Redirect clients to the destination volume.
6. Remount the NFS and SMB shares on the clients.

SAN environment:

1. Map the LUNs in the volume to the appropriate initiator group.
2. For iSCSI, create iSCSI sessions from the SAN host initiators to the SAN LIFs.
3. On the SAN client, perform a storage re-scan to detect the connected LUNs.

For information about NVMe environment, see SAN administration.

Reactivate the original source volume

You can reestablish the original data protection relationship between the source and destination volumes when you no longer need to serve data from the destination.

About this task

- The procedure below assumes that the baseline in the original source volume is intact. If the baseline is not intact, you must create and initialize the relationship between the volume you are serving data from and the original source volume before performing the procedure.
- Background preparation and the data warehousing phase of an XDP SnapMirror relationship can take a long time. It is not uncommon to see the SnapMirror relationship reporting the status "preparing" for an extended time period.

Steps

1. Delete the original data protection relationship:

   `snapmirror delete -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster.

   The following example deletes the relationship between the original source volume, `volA` on `svm1`, and the volume you are serving data from, `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror delete -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

2. Reverse the original data protection relationship:

   `snapmirror resync -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster. Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

   The following example reverses the relationship between the original source volume, `volA` on `svm1`, and the volume you are serving data from, `volA_dst` on `svm_backup`:

```
cluster_dst::> snapmirror resync -source-path svm_backup:volA_dst
-destination-path svm1:volA
```

3. Stop the source SVM for the reversed relationship:

   `vserver stop -vserver` *SVM*

   For complete command syntax, see the man page.

   The following example stops the source SVM for the reversed relationship:

```
cluster_src::> vserver stop svm_backup
```

4. Update the reversed relationship:

   `snapmirror update -source-path` *SVM:volume|cluster://SVM/volume*`, … -destination -path` *SVM:volume|cluster://SVM/volume*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.The command fails if a common Snapshot copy does not exist on the source and destination. Use `snapmirror initialize` to re-initialize the relationship.

   The following example updates the relationship between the volume you are serving data from, `volA_dst` on `svm_backup`, and the original source volume, `volA` on `svm1`:

```
cluster_dst::> snapmirror update -source-path svm_backup:volA_dst
-destination-path svm1:volA
```

5. Stop scheduled transfers for the reversed relationship:

   `snapmirror quiesce -source-path` *SVM:volume|cluster://SVM/volume*`, …`
   `-destination-path` *SVM:volume|cluster://SVM/volume*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following example stops scheduled transfers between the volume you are serving data from, `volA_dst` on `svm_backup`, and the original source volume, `volA` on `svm1`:

```
cluster_dst::> snapmirror quiesce -source-path svm_backup:volA_dst
-destination-path svm1:volA
```

6. Stop ongoing transfers for the reversed relationship:

   `snapmirror abort -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following example stops ongoing transfers between the volume you are serving data from, `volA_dst` on `svm_backup`, and the original source volume, `volA` on `svm1`:

   ```
   cluster_dst::> snapmirror abort -source-path svm_backup:volA_dst
   -destination-path svm1:volA
   ```

7. Break the reversed relationship:

   `snapmirror break -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following example breaks the relationship between the volume you are serving data from, `volA_dst` on `svm_backup`, and the original source volume, `volA` on `svm1`:

   ```
   cluster_dst::> snapmirror break -source-path svm_backup:volA_dst
   -destination-path svm1:volA
   ```

8. Start the original source SVM:

   `vserver start -vserver` *`SVM`*

   For complete command syntax, see the man page.

   The following example starts the original source SVM:

   ```
   cluster_dst::> vserver start svm1
   ```

9. Delete the reversed data protection relationship:

   `snapmirror delete -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

You must run this command from the source SVM or the source cluster for the reversed relationship.

The following example deletes the reversed relationship between the original source volume, `volA` on `svm1`, and the volume you are serving data from, `volA_dst` on `svm_backup`:

```
cluster_src::> snapmirror delete -source-path svm_backup:volA_dst
-destination-path svm1:volA
```

10. Reestablish the original data protection relationship:

    `snapmirror resync -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, …`

    For complete command syntax, see the man page.

    The following example reestablishes the relationship between the original source volume, `volA` on `svm1`, and the original destination volume, `volA_dst` on `svm_backup`:

    ```
    cluster_dst::> snapmirror resync -source-path svm1:volA -destination
    -path svm_backup:volA_dst
    ```

After you finish

Use the `snapmirror show` command to verify that the SnapMirror relationship was created. For complete command syntax, see the man page.

Restore files from a SnapMirror destination volume

Restore a single file, LUN, or NVMe namespace from a SnapMirror destination

You can restore a single file, LUN, a set of files or LUNs from a Snapshot copy, or an NVMe namespace from a SnapMirror destination volume. Beginning with ONTAP 9.7, you can also restore NVMe namespaces from a SnapMirror Synchronous destination. You can restore files to the original source volume or to a different volume.

What you'll need

To restore a file or LUN from a SnapMirror Synchronous destination (supported beginning with ONTAP 9.5), you must first delete and release the relationship.

About this task

The volume to which you are restoring files or LUNs (the destination volume) must be a read-write volume:

- SnapMirror performs an *incremental restore* if the source and destination volumes have a common Snapshot copy (as is typically the case when you are restoring to the original source volume).
- Otherwise, SnapMirror performs a *baseline restore*, in which the specified Snapshot copy and all the data blocks it references are transferred to the destination volume.

Steps

1. List the Snapshot copies in the destination volume:

   `volume snapshot show -vserver SVM -volume volume`

   For complete command syntax, see the man page.

   The following example shows the Snapshot copies on the `vserverB:secondary1` destination:

```
cluster_dst::> volume snapshot show -vserver vserverB -volume secondary1

Vserver     Volume      Snapshot                State    Size  Total%
Used%
-------     ------      ---------- ----------- ------   -----  ------
-----
vserverB    secondary1  hourly.2013-01-25_0005  valid   224KB     0%
0%
                        daily.2013-01-25_0010   valid   92KB      0%
0%
                        hourly.2013-01-25_0105  valid   228KB     0%
0%
                        hourly.2013-01-25_0205  valid   236KB     0%
0%
                        hourly.2013-01-25_0305  valid   244KB     0%
0%
                        hourly.2013-01-25_0405  valid   244KB     0%
0%
                        hourly.2013-01-25_0505  valid   244KB     0%
0%

7 entries were displayed.
```

2. Restore a single file or LUN or a set of files or LUNs from a Snapshot copy in a SnapMirror destination volume:

   `snapmirror restore -source-path SVM:volume|cluster://SVM/volume, … -destination-path SVM:volume|cluster://SVM/volume, … -source-snapshot snapshot -file-list source_file_path,@destination_file_path`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster.

   The following command restores the files `file1` and `file2` from the Snapshot copy `daily.2013-01-25_0010` in the original destination volume `secondary1`, to the same location in the active file system of the original source volume `primary1`:

```
cluster_dst::> snapmirror restore -source-path vserverB:secondary1
-destination-path vserverA:primary1 -source-snapshot daily.2013-01-
25_0010 -file-list /dir1/file1,/dir2/file2

[Job 3479] Job is queued: snapmirror restore for the relationship with
destination vserverA:primary1
```

The following command restores the files `file1` and `file2` from the Snapshot copy `daily.2013-01-25_0010` in the original destination volume `secondary1`, to a different location in the active file system of the original source volume `primary1`.

The destination file path begins with the @ symbol followed by the path of the file from the root of the original source volume. In this example, `file1` is restored to `/dir1/file1.new` and file2 is restored to `/dir2.new/file2` on `primary1`:

```
cluster_dst::> snapmirror restore -source-path vserverB:secondary1
-destination-path vserverA:primary1 -source-snapshot daily.2013-01-
25_0010 -file-list
/dir/file1,@/dir1/file1.new,/dir2/file2,@/dir2.new/file2

[Job 3479] Job is queued: snapmirror restore for the relationship with
destination vserverA:primary1
```

The following command restores the files `file1` and `file3` from the Snapshot copy `daily.2013-01-25_0010` in the original destination volume `secondary1`, to different locations in the active file system of the original source volume `primary1`, and restores `file2` from `snap1` to the same location in the active file system of `primary1`.

In this example, the file `file1` is restored to `/dir1/file1.new` and `file3` is restored to `/dir3.new/file3`:

```
cluster_dst::> snapmirror restore -source-path vserverB:secondary1
-destination-path vserverA:primary1 -source-snapshot daily.2013-01-
25_0010 -file-list
/dir/file1,@/dir1/file1.new,/dir2/file2,/dir3/file3,@/dir3.new/file3

[Job 3479] Job is queued: snapmirror restore for the relationship with
destination vserverA:primary1
```

Restore the contents of a volume from a SnapMirror destination

You can restore the contents of an entire volume from a Snapshot copy in a SnapMirror destination volume. You can restore the volume's contents to the original source volume or to a different volume.

About this task

The destination volume for the restore operation must be one of the following:

- A read-write volume, in which case SnapMirror performs an *incremental restore*, provided that the source and destination volumes have a common Snapshot copy (as is typically the case when you are restoring to the original source volume).

  The command fails if there is not a common Snapshot copy. You cannot restore the contents of a volume to an empty read-write volume.

- An empty data protection volume, in which case SnapMirror performs a *baseline restore*, in which the specified Snapshot copy and all the data blocks it references are transferred to the source volume.

Restoring the contents of a volume is a disruptive operation. SMB traffic must not be running on the SnapVault primary volume when a restore operation is running.

If the destination volume for the restore operation has compression enabled, and the source volume does not have compression enabled, disable compression on the destination volume. You need to re-enable compression after the restore operation is complete.

Any quota rules defined for the destination volume are deactivated before the restore is performed. You can use the `volume quota modify` command to reactivate quota rules after the restore operation is complete.

Steps

1. List the Snapshot copies in the destination volume:

   `volume snapshot show -vserver` *`SVM`* `-volume` *`volume`*

   For complete command syntax, see the man page.

   The following example shows the Snapshot copies on the `vserverB:secondary1` destination:

```
cluster_dst::> volume snapshot show -vserver vserverB -volume secondary1

Vserver     Volume      Snapshot                State    Size  Total%
Used%
-------     ------      ---------- ----------- ------   -----  ------
-----
vserverB    secondary1  hourly.2013-01-25_0005  valid   224KB     0%
0%
                        daily.2013-01-25_0010   valid   92KB      0%
0%
                        hourly.2013-01-25_0105  valid   228KB     0%
0%
                        hourly.2013-01-25_0205  valid   236KB     0%
0%
                        hourly.2013-01-25_0305  valid   244KB     0%
0%
                        hourly.2013-01-25_0405  valid   244KB     0%
0%
                        hourly.2013-01-25_0505  valid   244KB     0%
0%

7 entries were displayed.
```

2. Restore the contents of a volume from a Snapshot copy in a SnapMirror destination volume:

   `snapmirror restore -source-path` *`SVM:volume|cluster://SVM/volume`*`, …`
   `-destination-path` *`SVM:volume|cluster://SVM/volume`*`, … -source-snapshot` *`snapshot`*

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster.

   The following command restores the contents of the original source volume `primary1` from the Snapshot copy `daily.2013-01-25_0010` in the original destination volume `secondary1`:

```
cluster_dst::> snapmirror restore -source-path vserverB:secondary1
-destination-path vserverA:primary1 -source-snapshot daily.2013-01-
25_0010

Warning: All data newer than Snapshot copy daily.2013-01-25_0010 on
volume vserverA:primary1 will be deleted.

Do you want to continue? {y|n}: y

[Job 34] Job is queued: snapmirror restore from source
vserverB:secondary1 for the snapshot daily.2013-01-25_0010.
```

3. Remount the restored volume and restart all applications that use the volume.

Other ways to do this in ONTAP

To perform these tasks with…	See this content…
The redesigned System Manager (available with ONTAP 9.7 and later)	Restore a volume from an earlier Snapshot copy
System Manager Classic (available with ONTAP 9.7 and earlier)	Volume restore using SnapVault overview

Update a replication relationship manually

You might need to update a replication relationship manually if an update fails because the source volume has been moved.

About this task

SnapMirror aborts any transfers from a moved source volume until you update the replication relationship manually.

Beginning with ONTAP 9.5, SnapMirror Synchronous relationships are supported. Although the source and destination volumes are in sync at all times in these relationships, the view from the secondary cluster is synchronized with the primary only on an hourly basis. If you want to view the point-in-time data at the destination, you should perform a manual update by running the `snapmirror update` command.

Step

1. Update a replication relationship manually:

   `snapmirror update -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination-path` *`SVM:volume|cluster://SVM/volume`*`, …`

   For complete command syntax, see the man page.

   > You must run this command from the destination SVM or the destination cluster. The command fails if a common Snapshot copy does not exist on the source and destination. Use `snapmirror initialize` to re-initialize the relationship.

The following example updates the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

```
cluster_src::> snapmirror update -source-path svm1:volA -destination
-path svm_backup:volA_dst
```

Resynchronize a replication relationship

You need to resynchronize a replication relationship after you make a destination volume writeable, after an update fails because a common Snapshot copy does not exist on the source and destination volumes, or if you want to change the replication policy for the relationship.

About this task

- Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.
- Volumes that are part of a fan-out or cascade configuration can take longer to resynchronize. It is not uncommon to see the SnapMirror relationship reporting the status "preparing" for an extended time period.

Step

1. Resync the source and destination volumes:

   `snapmirror resync -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, … -type DP|XDP -schedule` *`schedule`* `-policy` *`policy`*

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster.

   The following example resyncs the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror resync -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

Delete a volume replication relationship

You can use the `snapmirror delete` and `snapmirror release` commands to delete a volume replication relationship. You can then delete unneeded destination volumes manually.

About this task

The `snapmirror release` command deletes any SnapMirror-created Snapshot copies from the source. You can use the `-relationship-info-only` option to preserve the Snapshot copies.

Steps

1. Quiesce the replication relationship:

   `snapmirror quiesce -destination-path SVM:volume|cluster://SVM/volume`

   ```
   cluster_dst::> snapmirror quiesce -destination-path svm_backup:volA_dst
   ```

2. Break the replication relationship:

   `snapmirror break -source-path SVM:volume|cluster://SVM/volume, … -destination -path SVM:volume|cluster://SVM/volume, …`

   ```
   cluster_dst::> snapmirror break -source-path svm1:volA -destination-path
   svm_backup:volA_dst
   ```

3. Delete the replication relationship:

   `snapmirror delete -source-path SVM:volume|cluster://SVM/volume, … -destination -path SVM:volume|cluster://SVM/volume, …`

   For complete command syntax, see the man page.

   You must run this command from the destination cluster or destination SVM.

   The following example deletes the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`:

   ```
   cluster_dst::> snapmirror delete -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

4. Release replication relationship information from the source SVM:

   `snapmirror release -source-path SVM:volume|cluster://SVM/volume, … -destination-path SVM:volume|cluster://SVM/volume, …`

   For complete command syntax, see the man page.

   You must run this command from the source cluster or source SVM.

   The following example releases information for the specified replication relationship from the source SVM `svm1`:

   ```
   cluster_src::> snapmirror release -source-path svm1:volA -destination
   -path svm_backup:volA_dst
   ```

Manage storage efficiency

SnapMirror preserves storage efficiency on the source and destination volumes, with one exception, when postprocess data compression is enabled on the destination. In that case, all storage efficiency is lost on the destination. To correct this issue, you need to disable postprocess compression on the destination, update the relationship manually, and re-enable storage efficiency.

What you'll need

- The source and destination clusters and SVMs must be peered.

  Cluster and SVM peering

- You must disable postprocess compression on the destination.

About this task

You can use the `volume efficiency show` command to determine whether efficiency is enabled on a volume. For more information, see the man pages.

You can check if SnapMirror is maintaining storage efficiency by viewing the SnapMirror audit logs and locating the transfer description. If the transfer description displays `transfer_desc=Logical Transfer`, SnapMirror is not maintaining storage efficiency. If the transfer description displays `transfer_desc=Logical Transfer with Storage Efficiency`, SnapMirror is maintaining storage efficiency. For example:

```
Fri May 22 02:13:02 CDT 2020 ScheduledUpdate[May 22 02:12:00]:cc0fbc29-
b665-11e5-a626-00a09860c273 Operation-Uuid=39fbcf48-550a-4282-a906-
df35632c73a1 Group=none Operation-Cookie=0 action=End source=<sourcepath>
destination=<destpath> status=Success bytes_transferred=117080571
network_compression_ratio=1.0:1 transfer_desc=Logical Transfer - Optimized
Directory Mode
```

Logical Transfer with storage

Beginning with ONTAP 9.3, manual update is no longer required to re-enable storage efficiency. If SnapMirror detects that postprocess compression has been disabled, it automatically re-enables storage efficiency at the next scheduled update. Both the source and the destination must be running ONTAP 9.3.

Beginning with ONTAP 9.3, AFF systems manage storage efficiency settings differently from FAS systems after a destination volume is made writeable:

- After you make a destination volume writeable using the `snapmirror break` command, the caching policy on the volume is automatically set to "auto" (the default).

  This behavior is applicable to FlexVol volumes, only, and it does not apply to FlexGroup volumes.

- On resync, the caching policy is automatically set to "none", and deduplication and inline compression are automatically disabled, regardless of your original settings. You must modify the settings manually as needed.

Manual updates with storage efficiency enabled can be time-consuming. You might want to run the operation in off-peak hours.

Step

1. Update a replication relationship and re-enable storage efficiency:

   `snapmirror update -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, … -enable-storage-efficiency true`

   For complete command syntax, see the man page.

   You must run this command from the destination SVM or the destination cluster. The command fails if a common Snapshot copy does not exist on the source and destination. Use `snapmirror initialize` to re-initialize the relationship.

   The following example updates the relationship between the source volume `volA` on `svm1` and the destination volume `volA_dst` on `svm_backup`, and re-enables storage efficiency:

```
cluster_dst::> snapmirror update -source-path svm1:volA -destination
-path svm_backup:volA_dst -enable-storage-efficiency true
```

Use SnapMirror global throttling

Global network throttling is available for all SnapMirror and SnapVault transfers at a per-node level.

About this task

SnapMirror global throttling restricts the bandwidth used by incoming and/or outgoing SnapMirror and SnapVault transfers. The restriction is enforced cluster wide on all nodes in the cluster.

For example, if the outgoing throttle is set to 100 Mbps, each node in the cluster will have the outgoing bandwidth set to 100 Mbps. If global throttling is disabled, it is disabled on all nodes.

The throttle has no effect on `volume move` transfers or load-sharing mirror transfers. Although data transfer rates are often expressed in bits per second (bps), the throttle values must be entered in kilobytes per second (KBps).

Global throttling works with the per-relationship throttle feature for SnapMirror and SnapVault transfers. The per-relationship throttle is enforced until the combined bandwidth of per-relationship transfers exceeds the value of the global throttle, after which the global throttle is enforced. A throttle value `0` implies that global throttling is disabled.

SnapMirror global throttling has no effect on SnapMirror Synchronous relationships when they are In-Sync. However, the throttle does effect SnapMirror Synchronous relationships when they perform an asynchronous transfer phase such as an initialization operation or after an Out Of Sync event. For this reason, enabling global throttling with SnapMirror Synchronous relationships is not recommended.

Steps

1. Enable global throttling:

   `options -option-name replication.throttle.enable on|off`

   The following example shows how to enable SnapMirror global throttling on `cluster_dst`:

   ```
   cluster_dst::> options -option-name replication.throttle.enable on
   ```

2. Specify the maximum total bandwidth used by incoming transfers:

   `options -option-name replication.throttle.incoming.max_kbs` *KBps*

   The recommended minimum throttle bandwidth is 4 KBps and the maximum is up to 2 TBps. The default value for this option is `unlimited`, which means there is no limit on total bandwidth used.

   The following example shows how to set the maximum total bandwidth used by incoming transfers to 100 Mbps:

   ```
   cluster_dst::> options -option-name
   replication.throttle.incoming.max_kbs 12500
   ```

   100 Mbps = 12500 KBps

3. Specify the maximum total bandwidth used by outgoing transfers:

   `options -option-name replication.throttle.outgoing.max_kbs` *KBps*

   KBps is the maximum transfer rate in kilobytes per second. Valid transfer rate values are 1 to 125000. The default value for this option is `unlimited`, which means there is no limit on total bandwidth used.

   The following example shows how to set the maximum total bandwidth used by outgoing transfers to 100 Mbps:

   ```
   cluster_dst::> options -option-name
   replication.throttle.outgoing.max_kbs 12500
   ```

About SnapMirror SVM replication

You can use SnapMirror to create a data protection relationship between SVM. In this type of data protection relationship, all or part of the SVM's configuration, from NFS exports and SMB shares to RBAC, is replicated, as well as the data in the volumes that the SVM owns.

Supported relationship types

Only data-serving SVM can be replicated. The following data protection relationship types are supported:

- *SnapMirror DR,* in which the destination typically contains only the Snapshot copies currently on the source.

  Beginning with ONTAP 9.9.1, this behavior changes when you are using the mirror-vault policy. Beginning with ONTAP 9.9.1, you can create different Snapshot policies on the source and destination, and the Snapshot copies on the destination are not overwritten by Snapshot copies on the source:

  - They are not overwritten from the source to the destination during normal scheduled operations, updates and resync
  - They are not deleted during break operations.
  - They are not deleted during flip-resync operations. When you configure an SVM DR relationship using the mirror-vault policy using ONTAP 9.9.1 and later, the policy behaves as follows:
  - User-defined Snapshot copy policies at the source are not copied to the destination.
  - System-defined Snapshot copy policies are not copied to the destination.
  - Volume association with user and system defined Snapshot policies are not copied to the destination. SVM.
- Beginning with ONTAP 9.2, *SnapMirror unified replication,* in which the destination is configured for both DR and long-term retention.

Details about these relationship types can be found here: Understanding SnapMirror volume replication.

The *policy type* of the replication policy determines the type of relationship it supports. The following table shows the available policy types.

Policy type	Relationship type
async-mirror	SnapMirror DR
mirror-vault	Unified replication

XDP replaces DP as the SVM replication default in ONTAP 9.4

Beginning with ONTAP 9.4, SVM data protection relationships default to XDP mode. SVM data protection relationships continue to default to DP mode in ONTAP 9.3 and earlier.

Existing relationships are not affected by the new default. If a relationship is already of type DP, it will continue to be of type DP. The following table shows the behavior you can expect.

If you specify…	The type is…	The default policy (if you do not specify a policy) is…
DP	XDP	MirrorAllSnapshots (SnapMirror DR)

Nothing	XDP	MirrorAllSnapshots (SnapMirror DR)
XDP	XDP	MirrorAndVault (Unified replication)

Details about the changes in the default can be found here: XDP replaces DP as the SnapMirror default.

Version-independence is not supported for SVM replication.

Compatible ONTAP versions for SnapMirror relationships

How SVM configurations are replicated

The content of an SVM replication relationship is determined by the interaction of the following fields:

- The `-identity-preserve true` option of the `snapmirror create` command replicates the entire SVM configuration.

  The `-identity-preserve false` option replicates only the volumes and authentication and authorization configurations of the SVM, and the protocol and name service settings listed in Configurations replicated in SVM DR relationships.

- The `-discard-configs network` option of the `snapmirror policy create` command excludes LIFs and related network settings from SVM replication, for use in cases where the source and destination SVMs are in different subnets.
- The `-vserver-dr-protection unprotected` option of the `volume modify` command excludes the specified volume from SVM replication.

Otherwise, SVM replication is almost identical to volume replication. You can use virtually the same workflow for SVM replication as you use for volume replication.

Support details

The following table shows support details for SnapMirror SVM replication.

Resource or feature	Support details
Relationship types	• SnapMirror DR • Beginning with ONTAP 9.2, SnapMirror unified replication
Replication scope	Intercluster only. You cannot replicate SVMs in the same cluster.
Version-independence	Not supported.

Deployment types	• Single source to single destination • Beginning with ONTAP 9.4, fan-out. You can fan-out to two destinations only. By default, only one -identity-preserve true relationship is allowed per source SVM.
Volume encryption	• Encrypted volumes on the source are encrypted on the destination. • Onboard Key Manager or KMIP servers must be configured on the destination. • New encryption keys are generated at the destination. • If the destination does not contain a node that supports volume .encryption, replication succeeds, but the destination volumes are not encrypted.
FabricPool	Beginning with ONTAP 9.6, SnapMirror SVM replication is supported with FabricPools.

MetroCluster	Beginning with ONTAP 9.5, SnapMirror SVM replication is supported on MetroCluster configurations. • A MetroCluster configuration cannot be the destination of an SVM DR relationship. • Only an active SVM within a MetroCluster configuration can be the source of an SVM DR relationship. A source can be a sync-source SVM before switchover or a sync-destination SVM after switchover. • When a MetroCluster configuration is in a steady state, the MetroCluster sync-destination SVM cannot be the source of an SVM DR relationship, since the volumes are not online. • When the sync-source SVM is the source of an SVM DR relationship, the source SVM DR relationship information is replicated to the MetroCluster partner. • During the switchover and switchback processes, replication to the SVM DR destination might fail. However, after the switchover or switchback process completes, the next SVM DR scheduled updates will succeed.
SnapMirror Synchronous	Not supported with SVM DR.

Configurations replicated in SVM DR relationships

The following table shows the interaction of the `snapmirror create `-identity-preserve` option and the `snapmirror policy create` -discard-configs network` option:

Configuration replicated		**`-identity-preserve true`**		**`-identity-preserve false`**
		Policy without `-discard -configs network` set	**Policy with `-discard -configs network` set**	

Network	NAS LIFs	Yes	No	No
	LIF Kerberos configuration	Yes	No	No
	SAN LIFs	No	No	No
	Firewall policies	Yes	Yes	No
	Routes	Yes	No	No
	Broadcast domain	No	No	No
	Subnet	No	No	No
	IPspace	No	No	No
SMB	SMB server	Yes	Yes	No
	Local groups and local user	Yes	Yes	Yes
	Privilege	Yes	Yes	Yes
	Shadow copy	Yes	Yes	Yes
	BranchCache	Yes	Yes	Yes
	Server options	Yes	Yes	Yes
	Server security	Yes	Yes	No
	Home directory, share	Yes	Yes	Yes
	Symlink	Yes	Yes	Yes
	Fpolicy policy, Fsecurity policy, and Fsecurity NTFS	Yes	Yes	Yes
	Name mapping and group mapping	Yes	Yes	Yes
	Audit information	Yes	Yes	Yes

NFS	Export policies	Yes	Yes	No
	Export policy rules	Yes	Yes	No
	NFS server	Yes	Yes	No
RBAC	Security certificates	Yes	Yes	No
	Login user, public key, role, and role configuration	Yes	Yes	Yes
	SSL	Yes	Yes	No
Name services	DNS and DNS hosts	Yes	Yes	No
	UNIX user and UNIX group	Yes	Yes	Yes
	Kerberos realm and Kerberos keyblocks	Yes	Yes	No
	LDAP and LDAP client	Yes	Yes	No
	Netgroup	Yes	Yes	No
	NIS	Yes	Yes	No
	Web and web access	Yes	Yes	No
Volume	Object	Yes	Yes	Yes
	Snapshot copies, Snapshot policy, and autodelete policy	Yes	Yes	Yes
	Efficiency policy	Yes	Yes	Yes
	Quota policy and quota policy rule	Yes	Yes	Yes
	Recovery queue	Yes	Yes	Yes

Root volume	Namespace	Yes	Yes	Yes
	User data	No	No	No
	Qtrees	No	No	No
	Quotas	No	No	No
	File-level QoS	No	No	No
	Attributes: state of the root volume, space guarantee, size, autosize, and total number of files	No	No	No
Storage QoS	QoS policy group	Yes	Yes	Yes
Fibre Channel (FC)		No	No	No
iSCSI		No	No	No
LUNs	Object	Yes	Yes	Yes
	igroups	No	No	No
	portsets	No	No	No
	Serial numbers	No	No	No
SNMP	v3 users	Yes	Yes	No

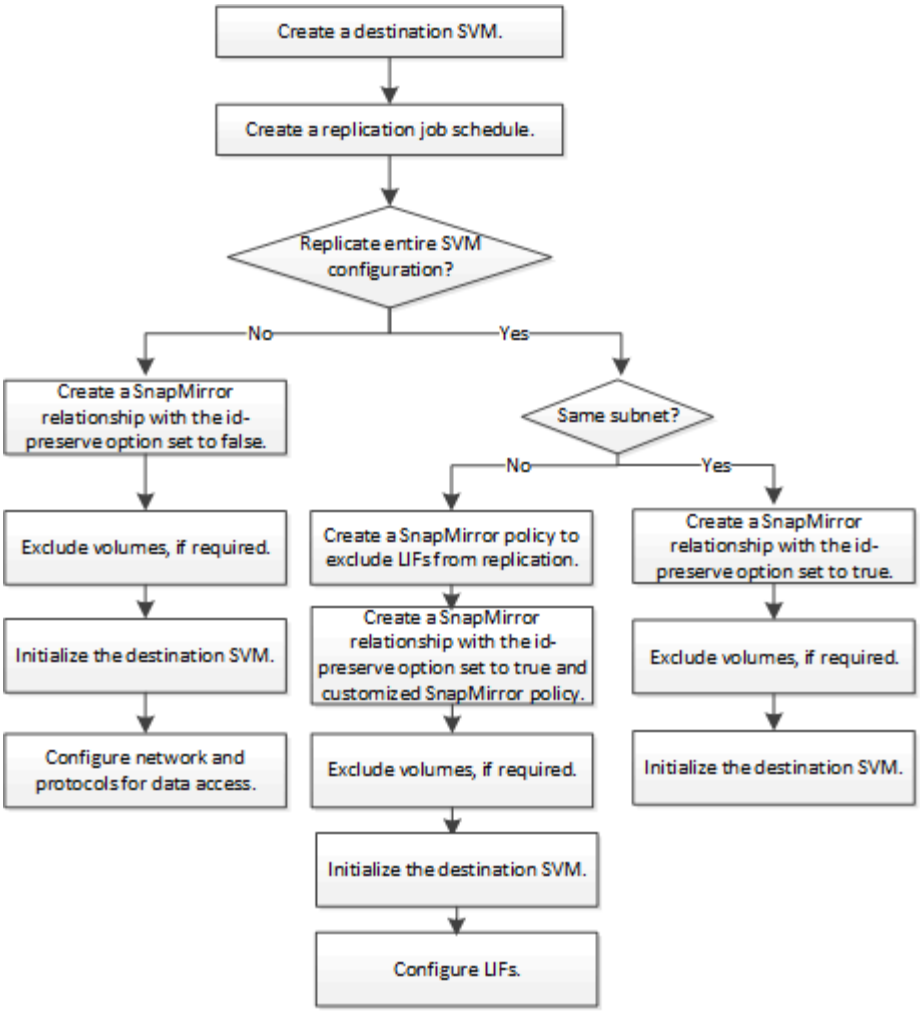
Manage SnapMirror SVM replication

Replicate SVM configurations

SnapMirror SVM replication workflow

SnapMirror SVM replication involves creating the destination SVM, creating a replication job schedule, and creating and initializing a SnapMirror relationship.

This workflow assumes that you are already using a default policy or a custom replication policy.



Criteria for placing volumes on destination SVMs

When replicating volumes from the source SVM to the destination SVM, it's important to know the criteria for selecting aggregates.

Aggregates are selected based on the following criteria:

- Volumes are always placed on non-root aggregates.
- Non-root aggregates are selected based on the available free space and the number of volumes already hosted on the aggregate.

  Aggregates with more free space and fewer volumes are given priority. The aggregate with the highest priority is selected.

- Source volumes on FabricPool aggregates are placed on FabricPool aggregates on the destination with the same tiering-policy.
- If a volume on the source SVM is located on a Flash Pool aggregate, then the volume is placed on a Flash Pool aggregate on the destination SVM, if such an aggregate exists and has enough free space.
- If the `-space-guarantee` option of the volume that is replicated is set to `volume`, only aggregates with free space greater than the volume size are considered.
- The volume size grows automatically on the destination SVM during replication, based on the source volume size.

If you want to pre-reserve the size on the destination SVM, you must resize the volume. The volume size does not shrink automatically on the destination SVM based on the source SVM.

If you want to move a volume from one aggregate to another, you can use the `volume move` command on the destination SVM.

Replicate an entire SVM configuration

You can use the `-identity-preserve true` option of the `snapmirror create` command to replicate an entire SVM configuration.

What you'll need

The source and destination clusters and SVMs must be peered. For more information, see Create a cluster peer relationship and Create an SVM intercluster peer relationship.

For complete command syntax, see the man page.

About this task

This workflow assumes that you are already using a default policy or a custom replication policy.

Beginning with ONTAP 9.9.1, when you use the mirror-vault policy, you can create different Snapshot policies on the source and destination SVM, and the Snapshot copies on the destination are not overwritten by Snapshot copies on the source. For more information, see Understanding SnapMirror SVM replication.

Steps

1. Create a destination SVM:

   `vserver create -vserver` *`SVM_name`* `-subtype dp-destination`

   The SVM name must be unique across the source and destination clusters.

   The following example creates a destination SVM named `svm_backup`:

   ```
   cluster_dst:> vserver create -vserver svm_backup -subtype dp-destination
   ```

2. From the destination cluster, create an SVM peer relationship using the `vserver peer create` command.

   For more information, see Create an SVM intercluster peer relationship.

3. Create a replication job schedule:

   `job schedule cron create -name` *`job_name`* `-month` *`month`* `-dayofweek` *`day_of_week`* `-day` *`day_of_month`* `-hour` *`hour`* `-minute` *`minute`*

   For `-month`, `-dayofweek`, and `-hour`, you can specify `all` to run the job every month, day of the week, and hour, respectively.

   The following example creates a job schedule named `my_weekly` that runs on Saturdays at 3:00 a.m.:

```
cluster_dst::> job schedule cron create -name my_weekly -dayofweek
saturday -hour 3 -minute 0
```

4. From the destination SVM or the destination cluster, create a replication relationship:

   `snapmirror create -source-path SVM_name: -destination-path SVM_name: -type DP|XDP -schedule schedule -policy policy -identity-preserve true`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options.

   The following example creates a SnapMirror DR relationship using the default `MirrorAllSnapshots` policy:

   ```
   cluster_dst::> snapmirror create -source-path svm1: -destination-path
   svm_backup: -type XDP -schedule my_daily -policy MirrorAllSnapshots
   -identity-preserve true
   ```

   The following example creates a unified replication relationship using the default `MirrorAndVault` policy:

   ```
   cluster_dst:> snapmirror create -source-path svm1: -destination-path
   svm_backup: -type XDP -schedule my_daily -policy MirrorAndVault
   -identity-preserve true
   ```

   Assuming you have created a custom policy with the policy type `async-mirror`, the following example creates a SnapMirror DR relationship:

   ```
   cluster_dst::> snapmirror create -source-path svm1: -destination-path
   svm_backup: -type XDP -schedule my_daily -policy my_mirrored -identity
   -preserve true
   ```

   Assuming you have created a custom policy with the policy type `mirror-vault`, the following example creates a unified replication relationship:

   ```
   cluster_dst::> snapmirror create -source-path svm1: -destination-path
   svm_backup: -type XDP -schedule my_daily -policy my_unified -identity
   -preserve true
   ```

5. Stop the destination SVM:

   `vserver stop`

   SVM name

The following example stops a destination SVM named dvs1:

```
cluster_dst::> vserver stop -vserver dvs1
```

6. From the destination SVM or the destination cluster, initialize the SVM replication relationship:

   `snapmirror initialize -source-path` *`SVM_name`*`: -destination-path` *`SVM_name`*`:`

   The following example initializes the relationship between the source SVM, `svm1`, and the destination SVM, `svm_backup`:

```
cluster_dst::> snapmirror initialize -source-path svm1: -destination
-path svm_backup:
```

Exclude LIFs and related network settings from SVM replication

If the source and destination SVMs are in different subnets, you can use the `-discard -configs network` option of the `snapmirror policy create` command to exclude LIFs and related network settings from SVM replication.

What you'll need

The source and destination clusters and SVMs must be peered.

For more information, see Create a cluster peer relationship and Create an SVM intercluster peer relationship.

About this task

The `-identity-preserve` option of the `snapmirror create` command must be set to `true` when you create the SVM replication relationship.

For complete command syntax, see the man page.

Steps

1. Create a destination SVM:

   `vserver create -vserver` *`SVM`* `-subtype dp-destination`

   The SVM name must be unique across the source and destination clusters.

   The following example creates a destination SVM named `svm_backup`:

```
cluster_dst:> vserver create -vserver svm_backup -subtype dp-destination
```

2. From the destination cluster, create an SVM peer relationship using the `vserver peer create` command.

   For more information, see Create an SVM intercluster peer relationship.

3. Create a job schedule:

   `job schedule cron create -name job_name -month month -dayofweek day_of_week -day day_of_month -hour hour -minute minute`

   For `-month`, `-dayofweek`, and `-hour`, you can specify `all` to run the job every month, day of the week, and hour, respectively.

   The following example creates a job schedule named `my_weekly` that runs on Saturdays at 3:00 a.m.:

   ```
   cluster_dst::> job schedule cron create -name my_weekly -dayofweek
   "Saturday" -hour 3 -minute 0
   ```

4. Create a custom replication policy:

   `snapmirror policy create -vserver SVM -policy policy -type async-mirror|vault|mirror-vault -comment comment -tries transfer_tries -transfer -priority low|normal -is-network-compression-enabled true|false -discard -configs network`

   For complete command syntax, see the man page.

   The following example creates a custom replication policy for SnapMirror DR that excludes LIFs:

   ```
   cluster_dst::> snapmirror policy create -vserver svm1 -policy
   DR_exclude_LIFs -type async-mirror -discard-configs network
   ```

   The following example creates a custom replication policy for unified replication that excludes LIFs:

   ```
   cluster_dst::> snapmirror policy create -vserver svm1 -policy
   unified_exclude_LIFs -type mirror-vault -discard-configs network
   ```

5. From the destination SVM or the destination cluster, run the following command to create a replication relationship:

   `snapmirror create -source-path SVM: -destination-path SVM: -type DP|XDP -schedule schedule -policy policy -identity-preserve true|false`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the examples below.

   The following example creates a SnapMirror DR relationship that excludes LIFs:

```
cluster_dst::> snapmirror create -source-path svm1: -destination-path
svm_backup: -type XDP -schedule my_daily -policy DR_exclude_LIFs
-identity-preserve true
```

The following example creates a SnapMirror unified replication relationship that excludes LIFs:

```
cluster_dst::> snapmirror create -source-path svm1: -destination-path
svm_backup: -type XDP -schedule my_daily -policy unified_exclude_LIFs
-identity-preserve true
```

6. Stop the destination SVM:

   `vserver stop`

   `SVM name`

   The following example stops a destination SVM named dvs1:

   ```
   cluster_dst::> vserver stop -vserver dvs1
   ```

7. From the destination SVM or the destination cluster, initialize a replication relationship:

   `snapmirror initialize -source-path` *`SVM`*`: -destination-path` *`SVM`*`:`

   For complete command syntax, see the man page.

   The following example initializes the relationship between the source, `svm1` and the destination, `svm_backup`:

   ```
   cluster_dst::> snapmirror initialize -source-path svm1: -destination
   -path svm_backup:
   ```

After you finish

You must configure the network and protocols on the destination SVM for data access in the event a disaster occurs.

Exclude network, name service, and other settings from SVM replication

You can use the `-identity-preserve false` option of the `snapmirror create` command to replicate only the volumes and security configurations of an SVM. Some protocol and name service settings are also preserved.

What you'll need

The source and destination clusters and SVMs must be peered.

For more information, see Create a cluster peer relationship and Create an SVM intercluster peer relationship.

About this task

For a list of preserved protocol and name service settings, see Configurations replicated in SVM DR relationships.

For complete command syntax, see the man page.

Steps

1. Create a destination SVM:

   `vserver create -vserver SVM -subtype dp-destination`

   The SVM name must be unique across the source and destination clusters.

   The following example creates a destination SVM named `svm_backup`:

   ```
   cluster_dst:> vserver create -vserver svm_backup -subtype dp-destination
   ```

2. From the destination cluster, create an SVM peer relationship using the `vserver peer create` command.

   For more information, see Create an SVM intercluster peer relationship.

3. Create a replication job schedule:

   `job schedule cron create -name job_name -month month -dayofweek day_of_week -day day_of_month -hour hour -minute minute`

   For `-month`, `-dayofweek`, and `-hour`, you can specify `all` to run the job every month, day of the week, and hour, respectively.

   The following example creates a job schedule named `my_weekly` that runs on Saturdays at 3:00 a.m.:

   ```
   cluster_dst::> job schedule cron create -name my_weekly -dayofweek
   "Saturday" -hour 3 -minute 0
   ```

4. Create a replication relationship that excludes network, name service, and other configuration settings:

   `snapmirror create -source-path SVM: -destination-path SVM: -type DP|XDP -schedule schedule -policy policy -identity-preserve false`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the examples below. You must run this command from the destination SVM or the destination cluster.

   The following example creates a SnapMirror DR relationship using the default `MirrorAllSnapshots` policy. The relationship excludes network, name service, and other configuration settings from SVM replication:

```
cluster_dst::> snapmirror create -source-path svm1: -destination-path
svm_backup: -type XDP -schedule my_daily -policy MirrorAllSnapshots
-identity-preserve false
```

The following example creates a unified replication relationship using the default `MirrorAndVault` policy. The relationship excludes network, name service, and other configuration settings:

```
cluster_dst:> snapmirror create svm1: -destination-path svm_backup:
-type XDP -schedule my_daily -policy MirrorAndVault -identity-preserve
false
```

Assuming you have created a custom policy with the policy type `async-mirror`, the following example creates a SnapMirror DR relationship. The relationship excludes network, name service, and other configuration settings from SVM replication:

```
cluster_dst::> snapmirror create -source-path svm1: -destination-path
svm_backup: -type XDP -schedule my_daily -policy my_mirrored -identity
-preserve false
```

Assuming you have created a custom policy with the policy type `mirror-vault`, the following example creates a unified replication relationship. The relationship excludes network, name service, and other configuration settings from SVM replication:

```
cluster_dst::> snapmirror create -source-path svm1: -destination-path
svm_backup: -type XDP -schedule my_daily -policy my_unified -identity
-preserve false
```

5. Stop the destination SVM:

   `vserver stop`

   `SVM name`

   The following example stops a destination SVM named dvs1:

   ```
   destination_cluster::> vserver stop -vserver dvs1
   ```

6. If you are using SMB, you must also configure an SMB server.

   See SMB only: Creating an SMB server.

7. From the destination SVM or the destination cluster, initialize the SVM replication relationship:

   `snapmirror initialize -source-path` *`SVM_name`*`: -destination-path` *`SVM_name`*`:`

After you finish

You must configure the network and protocols on the destination SVM for data access in the event a disaster occurs.

Specify aggregates to use for SVM DR relationships

After a disaster recovery SVM is created, you can use the `aggr-list` option with `vserver modify` command to limit which aggregates are used to host SVM DR destination volumes.

Step

1. Create a destination SVM:

   ```
   vserver create -vserver SVM -subtype dp-destination
   ```

2. Modify the disaster recovery SVM's aggr-list to limit the aggregates that are used to host the disaster recovery SVM's volume:

   ```
   cluster_dest::> vserver modify -vserver SVM -aggr-list <comma-separated-list>
   ```

SMB only: Create a SMB server

If the source SVM has an SMB configuration, and you chose to set `identity-preserve` to `false`, you must create a SMB server for the destination SVM. SMB server is required for some SMB configurations, such as shares during initialization of the SnapMirror relationship.

Steps

1. Start the destination SVM by using the `vserver start` command.

   ```
   destination_cluster::> vserver start -vserver dvs1
   [Job 30] Job succeeded: DONE
   ```

2. Verify that the destination SVM is in the `running` state and subtype is `dp-destination` by using the `vserver show` command.

   ```
   destination_cluster::> vserver show
                                      Admin      Operational Root
   Vserver  Type    Subtype           State      State       Volume
   Aggregate
   -------- ------- ----------       ---------- ----------- ----------
   ----------
   dvs1     data    dp-destination    running    running       -         -
   ```

3. Create a LIF by using the `network interface create` command.

```
destination_cluster::>network interface create -vserver dvs1 -lif NAS1
-role data -data-protocol cifs -home-node destination_cluster-01 -home
-port a0a-101  -address 192.0.2.128 -netmask 255.255.255.128
```

4. Create a route by using the `network route create` command.

```
destination_cluster::>network route create -vserver dvs1 -destination
0.0.0.0/0
-gateway 192.0.2.1
```

Network management

5. Configure DNS by using the `vserver services dns create` command.

```
destination_cluster::>vserver services dns create -domains
mydomain.example.com -vserver
dvs1 -name-servers 192.0.2.128 -state enabled
```

6. Add the preferred domain controller by using the `vserver cifs domain preferred-dc add` command.

```
destination_cluster::>vserver cifs domain preferred-dc add -vserver dvs1
-preferred-dc
192.0.2.128 -domain mydomain.example.com
```

7. Create the SMB server by using the `vserver cifs create` command.

```
destination_cluster::>vserver cifs create -vserver dvs1 -domain
mydomain.example.com
-cifs-server CIFS1
```

8. Stop the destination SVM by using the `vserver stop` command.

```
destination_cluster::> vserver stop -vserver dvs1
[Job 46] Job succeeded: DONE
```

Exclude volumes from SVM replication

By default, all RW data volumes of the source SVM are replicated. If you do not want to protect all the volumes on the source SVM, you can use the `-vserver-dr`

`-protection unprotected` option of the `volume modify` command to exclude volumes from SVM replication.

Steps

1. Exclude a volume from SVM replication:

   `volume modify -vserver` *`SVM`* `-volume` *`volume`* `-vserver-dr-protection unprotected`

   For complete command syntax, see the man page.

   The following example excludes the volume `volA_src` from SVM replication:

   ```
   cluster_src::> volume modify -vserver SVM1 -volume volA_src -vserver-dr
   -protection unprotected
   ```

   If you later want to include a volume in the SVM replication that you originally excluded, run the following command:

   `volume modify -vserver` *`SVM`* `-volume` *`volume`* `-vserver-dr-protection protected`

   The following example includes the volume `volA_src` in the SVM replication:

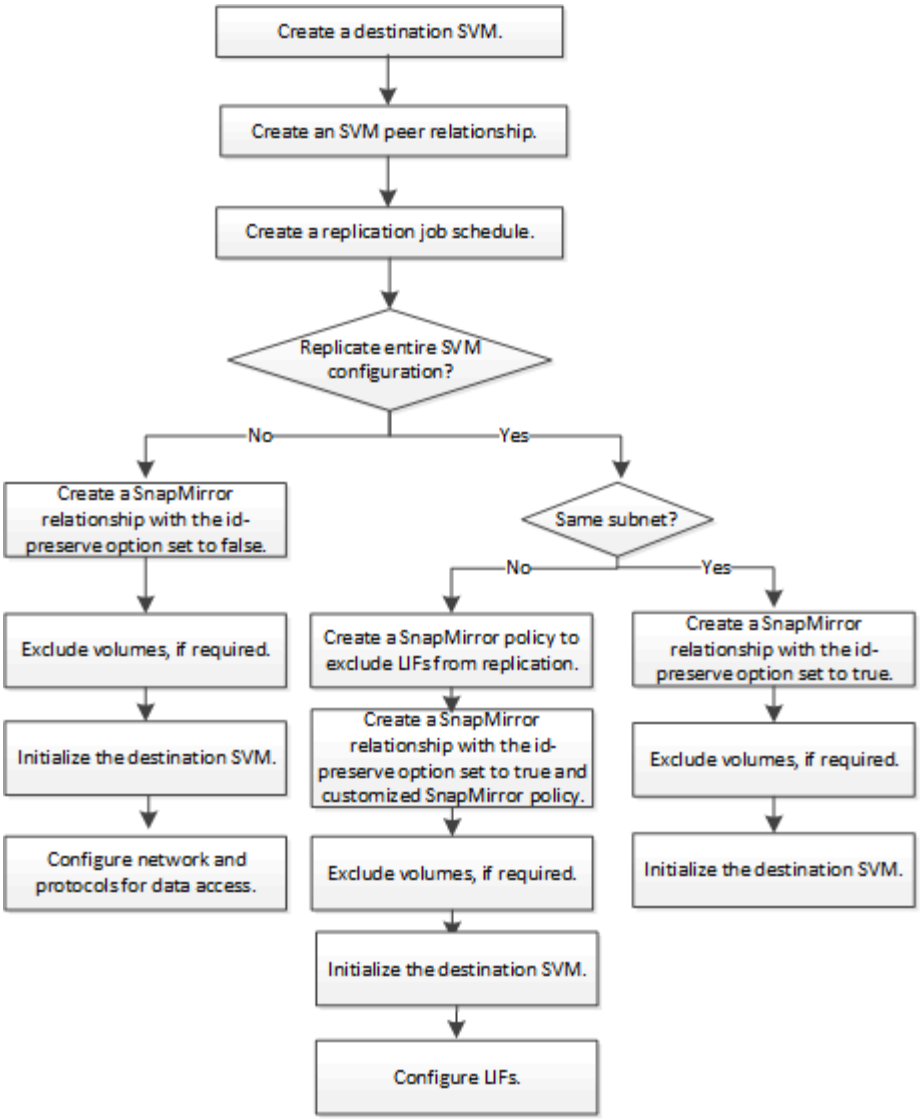
   ```
   cluster_src::> volume modify -vserver SVM1 -volume volA_src -vserver-dr
   -protection protected
   ```

2. Create and initialize the SVM replication relationship as described in Replicating an entire SVM configuration.

Serve data from an SVM DR destination

SVM disaster recovery workflow

To recover from a disaster and serve data from the destination SVM, you must activate the destination SVM. Activating the destination SVM involves stopping scheduled SnapMirror transfers, aborting ongoing SnapMirror transfers, breaking the replication relationship, stopping the source SVM, and starting the destination SVM.



Make SVM destination volumes writeable

You need to make SVM destination volumes writeable before you can serve data to clients. The procedure is largely identical to the procedure for volume replication, with one exception. If you set `-identity-preserve true` when you created the SVM replication relationship, you must stop the source SVM before activating the destination SVM.

About this task

For complete command syntax, see the man page.

> In a disaster recovery scenario, you cannot perform a SnapMirror update from the source SVM to the disaster recovery destination SVM because your source SVM and its data will be inaccessible, and because updates since the last resync might be bad or corrupt.

Steps

1. From the destination SVM or the destination cluster, stop scheduled transfers to the destination:

   `snapmirror quiesce -source-path` *SVM*`:` `-destination-path` *SVM*`:`

You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example stops scheduled transfers between the source SVM `svm1` and the destination SVM `svm_backup`:

```
cluster_dst::> snapmirror quiesce -source-path svm1: -destination-path
svm_backup:
```

2. From the destination SVM or the destination cluster, stop ongoing transfers to the destination:

   `snapmirror abort -source-path` *SVM*`: -destination-path` *SVM*`:`

   You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example stops ongoing transfers between the source SVM `svm1` and the destination SVM `svm_backup`:

   ```
   cluster_dst::> snapmirror abort -source-path svm1: -destination-path
   svm_backup:
   ```

3. From the destination SVM or the destination cluster, break the replication relationship:

   `snapmirror break -source-path` *SVM*`: -destination-path` *SVM*`:`

   You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example breaks the relationship between the source SVM `svm1` and the destination SVM `svm_backup`:

   ```
   cluster_dst::> snapmirror break -source-path svm1: -destination-path
   svm_backup:
   ```

4. If you set `-identity-preserve true` when you created the SVM replication relationship, stop the source SVM:

   `vserver stop -vserver` *SVM*

   The following example stops the source SVM `svm1`:

   ```
   cluster_src::> vserver stop svm1
   ```

5. Start the destination SVM:

   `vserver start -vserver SVM`

   The following example starts the destination SVM `svm_backup`:

   ```
   cluster_dst::> vserver start svm_backup
   ```

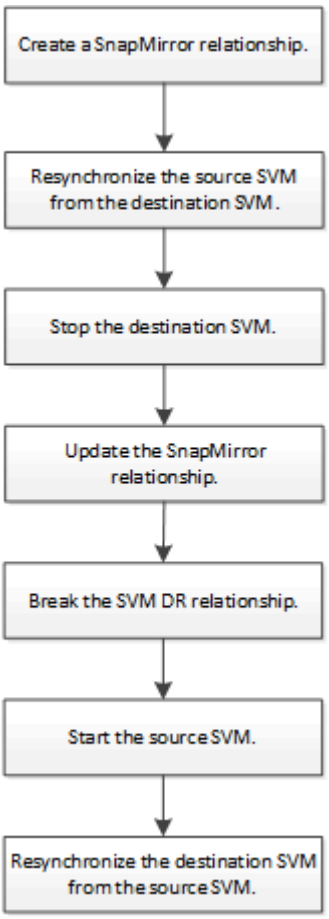
After you finish

Configure SVM destination volumes for data access, as described in Configuring the destination volume for data access.

Reactivate the source SVM

Source SVM reactivation workflow

If the source SVM exists after a disaster, you can reactivate it and protect it by recreating the SVM disaster recovery relationship.



Reactivate the original source SVM

You can reestablish the original data protection relationship between the source and destination SVM when you no longer need to serve data from the destination. The

procedure is largely identical to the procedure for volume replication, with one exception. You must stop the destination SVM before reactivating the source SVM.

What you'll need

If you have increased the size of destination volume while serving data from it, before you reactivate the source volume, you should manually increase max-autosize on the original source volume to ensure it can grow sufficiently.

When a destination volume grows automatically

About this task

This procedure assumes that the baseline in the original source volume is intact. If the baseline is not intact, you must create and initialize the relationship between the volume you are serving data from and the original source volume before performing the procedure.

For complete command syntax on commands, see the man page.

Steps

1. From the original source SVM or the original source cluster, create a reverse SVM DR relationship using the same configuration, policy, and identity-preserve setting as the original SVM DR relationship:

   `snapmirror create -source-path SVM: -destination-path SVM:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example creates a relationship between the SVM from which you are serving data, `svm_backup`, and the original source SVM, `svm1`:

   ```
   cluster_src::> snapmirror create -source-path svm_backup: -destination
   -path svm1:
   ```

2. From the original source SVM or the original source cluster, run the following command to reverse the data protection relationship:

   `snapmirror resync -source-path SVM: -destination-path SVM:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

   > The command fails if a common Snapshot copy does not exist on the source and destination. Use `snapmirror initialize` to reinitialize the relationship.

   The following example reverses the relationship between the original source SVM, `svm1`, and the SVM from which you are serving data, `svm_backup`:

```
cluster_src::> snapmirror resync -source-path svm_backup: -destination
-path svm1:
```

3. When you are ready to reestablish data access to the original source SVM, stop the original destination SVM to disconnect any clients currently connected to the original destination SVM.

   `vserver stop -vserver` *SVM*

   The following example stops the original destination SVM which is currently serving data:

```
cluster_dst::> vserver stop svm_backup
```

4. Verify that the original destination SVM is in the stopped state by using the `vserver show` command.

```
cluster_dst::> vserver show
                                 Admin      Operational Root
Vserver        Type    Subtype   State      State       Volume
Aggregate
--------       ------- ---------- ---------- ----------- ----------
----------
svm_backup     data    default   stopped    stopped     rv
aggr1
```

5. From the original source SVM or the original source cluster, run the following command to perform the final update of the reversed relationship to transfer all changes from the original destination SVM to the original source SVM:

   `snapmirror update -source-path` *SVM*`:` `-destination-path` *SVM*`:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example updates the relationship between the original destination SVM from which you are serving data,`svm_backup`, and the original source SVM, `svm1`:

```
cluster_src::> snapmirror update -source-path svm_backup: -destination
-path svm1:
```

6. From the original source SVM or the original source cluster, run the following command to stop scheduled transfers for the reversed relationship:

   `snapmirror quiesce -source-path` *SVM*`:` `-destination-path` *SVM*`:`

You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example stops scheduled transfers between the SVM you are serving data from, `svm_backup`, and the original SVM, `svm1`:

```
cluster_src::> snapmirror quiesce -source-path svm_backup: -destination
-path svm1:
```

7. When the final update is complete and the relationship indicates "Quiesced" for the relationship status, run the following command from the original source SVM or the original source cluster to break the reversed relationship:

   `snapmirror break -source-path` *SVM*`: -destination-path` *SVM*`:`

   You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example breaks the relationship between the original destination SVM from which you were serving data, `svm_backup`, and the original source SVM, `svm1`:

   ```
   cluster_src::> snapmirror break -source-path svm_backup: -destination
   -path svm1:
   ```

8. If the original source SVM was previously stopped, from the original source cluster, start the original source SVM:

   `vserver start -vserver` *SVM*

   The following example starts the original source SVM:

   ```
   cluster_src::> vserver start svm1
   ```

9. From the original destination SVM or the original destination cluster, reestablish the original data protection relationship:

   `snapmirror resync -source-path` *SVM*`: -destination-path` *SVM*`:`

   You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example reestablishes the relationship between the original source SVM, `svm1`, and the original destination SVM, `svm_backup`:

```
cluster_dst::> snapmirror resync -source-path svm1: -destination-path
svm_backup:
```

10. From the original source SVM or the original source cluster, run the following command to delete the reversed data protection relationship:

    `snapmirror delete -source-path` *SVM*`: -destination-path` *SVM*`:`

    > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination -path` options. See the example below.

    The following example deletes the reversed relationship between the original destination SVM, `svm_backup`, and the original source SVM, `svm1`:

    ```
    cluster_src::> snapmirror delete -source-path svm_backup: -destination
    -path svm1:
    ```

11. From the original destination SVM or the original destination cluster, release the reversed data protection relationship:

    `snapmirror release -source-path SVM: -destination-path SVM:`

    > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination -path` options. See the example below.

    The following example releases the reversed relationship between the original destination SVM, svm_backup, and the original source SVM, `svm1`

    ```
    cluster_dst::> snapmirror release -source-path svm_backup: -destination
    -path svm1:
    ```

After you finish

Use the `snapmirror show` command to verify that the SnapMirror relationship was created. For complete command syntax, see the man page.

Reactivate the original source SVM (FlexGroup volumes only)

You can reestablish the original data protection relationship between the source and destination SVM when you no longer need to serve data from the destination. To reactivate the original source SVM when you are using FlexGroup volumes, you need to perform some additional steps, including deleting the original SVM DR relationship and releasing the original relationship before you reverse the relationship. You also need to release the reversed relationship and recreate the original relationship before stopping scheduled transfers.

Steps

1. From the original destination SVM or the original destination cluster, delete the original SVM DR relationship:

   `snapmirror delete -source-path SVM: -destination-path SVM:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example deletes the original relationship between the original source SVM, svm1, and the original destination SVM, `svm_backup`:

   ```
   cluster_dst::> snapmirror delete -source-path svm1: -destination-path
   svm_backup:
   ```

2. From the original source SVM or the original source cluster, release the original relationship while keeping the Snapshot copies intact:

   `snapmirror release -source-path SVM: -destination-path SVM: -relationship-info -only true`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example releases the original relationship between the original source SVM, svm1, and the original destination SVM, `svm_backup`.

   ```
   cluster_src::> snapmirror release -source-path svm1: -destination-path
   svm_backup: -relationship-info-only true
   ```

3. From the original source SVM or the original source cluster, create a reverse SVM DR relationship using the same configuration, policy, and identity-preserve setting as the original SVM DR relationship:

   `snapmirror create -source-path SVM: -destination-path SVM:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example creates a relationship between the SVM from which you are serving data, `svm_backup`, and the original source SVM, `svm1`:

   ```
   cluster_src::> snapmirror create -source-path svm_backup: -destination
   -path svm1:
   ```

4. From the original source SVM or the original source cluster, run the following command to reverse the data protection relationship:

`snapmirror resync -source-path SVM: -destination-path SVM:`

> You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

> The command fails if a common Snapshot copy does not exist on the source and destination. Use `snapmirror initialize` to reinitialize the relationship.

The following example reverses the relationship between the original source SVM, `svm1`, and the SVM from which you are serving data, `svm_backup`:

```
cluster_src::> snapmirror resync -source-path svm_backup: -destination
-path svm1:
```

5. When you are ready to reestablish data access to the original source SVM, stop the original destination SVM to disconnect any clients currently connected to the original destination SVM.

   `vserver stop -vserver SVM`

   The following example stops the original destination SVM which is currently serving data:

   ```
   cluster_dst::> vserver stop svm_backup
   ```

6. Verify that the original destination SVM is in the stopped state by using the `vserver show` command.

   ```
   cluster_dst::> vserver show
                                    Admin      Operational Root
   Vserver        Type    Subtype    State      State       Volume
   Aggregate
   --------       ------- ---------- ---------- ----------- ----------
   ----------
   svm_backup     data    default    stopped    stopped     rv
   aggr1
   ```

7. From the original source SVM or the original source cluster, run the following command to perform the final update of the reversed relationship to transfer all changes from the original destination SVM to the original source SVM:

   `snapmirror update -source-path SVM: -destination-path SVM:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example updates the relationship between the original destination SVM from which you are serving data,`svm_backup`, and the original source SVM, `svm1`:

```
cluster_src::> snapmirror update -source-path svm_backup: -destination
-path svm1:
```

8. From the original source SVM or the original source cluster, run the following command to stop scheduled transfers for the reversed relationship:

   `snapmirror quiesce -source-path` *`SVM`*`: -destination-path` *`SVM`*`:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination -path` options. See the example below.

   The following example stops scheduled transfers between the SVM you are serving data from, `svm_backup`, and the original SVM, `svm1`:

   ```
   cluster_src::> snapmirror quiesce -source-path svm_backup: -destination
   -path svm1:
   ```

9. When the final update is complete and the relationship indicates "Quiesced" for the relationship status, run the following command from the original source SVM or the original source cluster to break the reversed relationship:

   `snapmirror break -source-path` *`SVM`*`: -destination-path` *`SVM`*`:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination -path` options. See the example below.

   The following example breaks the relationship between the original destination SVM from which you were serving data, `svm_backup`, and the original source SVM, `svm1`:

   ```
   cluster_src::> snapmirror break -source-path svm_backup: -destination
   -path svm1:
   ```

10. If the original source SVM was previously stopped, from the original source cluster, start the original source SVM:

    `vserver start -vserver` *`SVM`*

    The following example starts the original source SVM:

    ```
    cluster_src::> vserver start svm1
    ```

11. From the original source SVM or the original source cluster, delete the reversed SVM DR relationship:

`snapmirror delete -source-path SVM: -destination-path SVM:`

> You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example deletes the reversed relationship between the original destination SVM, svm_backup, and the original source SVM, `svm1`:

```
cluster_src::> snapmirror delete -source-path svm_backup: -destination
-path svm1:
```

12. From the original destination SVM or the original destination cluster, release the reversed relationship while keeping the Snapshot copies intact:

    `snapmirror release -source-path SVM: -destination-path SVM: -relationship-info -only true`

    > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

    The following example releases the reversed relationship between the original destination SVM, svm_backup, and the original source SVM, svm1:

    ```
    cluster_dst::> snapmirror release -source-path svm_backup: -destination
    -path svm1: -relationship-info-only true
    ```

13. From the original destination SVM or the original destination cluster, recreate the original relationship. Use the same configuration, policy, and identity-preserve setting as the original SVM DR relationship:

    `snapmirror create -source-path SVM: -destination-path SVM:`

    > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

    The following example creates a relationship between the original source SVM, `svm1`, and the original destination SVM, `svm_backup`:

    ```
    cluster_dst::> snapmirror create -source-path svm1: -destination-path
    svm_backup:
    ```

14. From the original destination SVM or the original destination cluster, reestablish the original data protection relationship:

    `snapmirror resync -source-path *SVM*: -destination-path *SVM*:`

> You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example reestablishes the relationship between the original source SVM, `svm1`, and the original destination SVM, `svm_backup`:

```
cluster_dst::> snapmirror resync -source-path svm1: -destination-path
svm_backup:
```

Convert volume replication relationships to an SVM replication relationship

You can convert replication relationships between volumes to a replication relationship between the storage virtual machines (SVMs) that own the volumes, provided that each volume on the source (except the root volume) is being replicated, and each volume on the source (including the root volume) has the same name as the volume on the destination.

About this task

Use the `volume rename` command when the SnapMirror relationship is idle to rename destination volumes if necessary.

Steps

1. From the destination SVM or the destination cluster, run the following command to resync the source and destination volumes:

   `snapmirror resync -source-path` *`SVM:volume`* `-destination-path` *`SVM:volume`* `-type DP|XDP -schedule` *`schedule`* `-policy` *`policy`*

   For complete command syntax, see the man page.

   > Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

   The following example resyncs the relationship between the source volume `volA` on `svm1` and the destination volume `volA` on `svm_backup`:

   ```
   cluster_dst::> snapmirror resync -source-path svm1:volA -destination
   -path svm_backup:volA
   ```

2. Create an SVM replication relationship between the source and destination SVMs, as described in Replicating SVM configurations.

   You must use the `-identity-preserve true` option of the `snapmirror create` command when you create your replication relationship.

3. Stop the destination SVM:

`vserver stop -vserver SVM`

For complete command syntax, see the man page.

The following example stops the destination SVM `svm_backup`:

```
cluster_dst::> vserver stop svm_backup
```

4. From the destination SVM or the destination cluster, run the following command to resync the source and destination SVMs:

   `snapmirror resync -source-path SVM: -destination-path SVM: -type DP|XDP -schedule schedule -policy policy`

   For complete command syntax, see the man page.

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.
   >
   > Although resync does not require a baseline transfer, it can be time-consuming. You might want to run the resync in off-peak hours.

   The following example resyncs the relationship between the source SVM `svm1` and the destination SVM `svm_backup`:

   ```
   cluster_dst::> snapmirror resync -source-path svm1: -destination-path
   svm_backup:
   ```

Delete an SVM replication relationship

You can use the `snapmirror delete` and `snapmirror release` commands to delete an SVM replication relationship. You can then delete unneeded destination volumes manually.

About this task

The `snapmirror release` command deletes any SnapMirror-created Snapshot copies from the source. You can use the `-relationship-info-only` option to preserve the Snapshot copies.

For complete command syntax on commands, see the man page.

Steps

1. Run the following command from the destination SVM or the destination cluster to break the replication relationship:

   `snapmirror break -source-path SVM: -destination-path SVM:`

> You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

The following example breaks the relationship between the source SVM `svm1` and the destination SVM `svm_backup`:

```
cluster_dst::> snapmirror break -source-path svm1: -destination-path
svm_backup:
```

2. Run the following command from the destination SVM or the destination cluster to delete the replication relationship:

   `snapmirror delete -source-path` *`SVM`*`: -destination-path` *`SVM`*`:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example deletes the relationship between the source SVM `svm1` and the destination SVM `svm_backup`:

   ```
   cluster_dst::> snapmirror delete -source-path svm1: -destination-path
   svm_backup:
   ```

3. Run the following command from the source cluster or source SVM to release the replication relationship information from the source SVM:

   `snapmirror release -source-path` *`SVM`*`: -destination-path` *`SVM`*`:`

   > You must enter a colon (:) after the SVM name in the `-source-path` and `-destination-path` options. See the example below.

   The following example releases information for the specified replication relationship from the source SVM `svm1`:

   ```
   cluster_src::> snapmirror release -source-path svm1: -destination-path
   svm_backup:
   ```

Manage SnapMirror root volume replication

Manage SnapMirror root volume replication overview

Every SVM in a NAS environment has a unique namespace. The SVM *root volume,* containing operating system and related information, is the entry point to the namespace hierarchy. To ensure that data remains accessible to clients in the event of a node outage

or failover, you should create a load-sharing mirror copy of the SVM root volume.

The main purpose of load-sharing mirrors for SVM root volumes is no longer for load sharing; instead, their purpose is for disaster recovery.

- If the root volume is temporarily unavailable, the load-sharing mirror automatically provides read-only access to root volume data.
- If the root volume is permanently unavailable, you can promote one of the load-sharing volumes to provide write access to root volume data.

Create and initializing load-sharing mirror relationships

You should create a load-sharing mirror (LSM) for each SVM root volume that serves NAS data in the cluster. You can create the LSM on any node other than the one containing the root volume, such as the partner node in an HA pair, or preferably in a different HA pair. For a two-node cluster, you should create the LSM on the partner of the node with the SVM root volume.

About this task

If you create an LSM on the same node, and the node is unavailable, you have a single point of failure, and you do not have a second copy to ensure the data remains accessible to clients. But when you create the LSM on a node other than the one containing the root volume, or on a different HA pair, your data is still accessible in the event of an outage.

For example, in a four-node cluster with a root volume on three nodes:

- For the root volume on HA 1 node 1, create the LSM on HA 2 node 1 or HA 2 node 2.
- For the root volume on HA 1 node 2, create the LSM on HA 2 node 1 or HA 2 node 2.
- For the root volume on HA 2 node 1, create the LSM on HA 1 node 1 or HA 1 node 2.

Steps

1. Create a destination volume for the LSM:

   `volume create -vserver` *`SVM`* `-volume volume -aggregate` *`aggregate`* `-type DP -size size`

   The destination volume should be the same or greater in size than the root volume.

   It is a best practice to name the root and destination volume with suffixes, such as `_root` and `_m1`.

   For complete command syntax, see the man page.

   The following example creates a load-sharing mirror volume for the root volume `svm1_root` in `cluster_src`:

   ```
   cluster_src:> volume create -vserver svm1 -volume svm1_m1 -aggregate
   aggr_1 -size 1gb -state online -type DP
   ```

2. Create a replication job schedule, as described in Creating a replication job schedule.

3. Create a load-sharing mirror relationship between the SVM root volume and the destination volume for the LSM:

   `snapmirror create -source-path` *`SVM:volume|cluster://SVM/volume`* `-destination -path` *`SVM:volume|cluster://SVM/volume`* `-type LS -schedule` *`schedule`*

   For complete command syntax, see the man page.

   The following example creates a load-sharing mirror relationship between the root volume `svm1_root` and the load-sharing mirror volume `svm1_m1`:

   ```
   cluster_src::> snapmirror create -source-path svm1:svm1_root
   -destination-path svm1:svm1_m1 -type LS -schedule hourly
   ```

   The type attribute of the load-sharing mirror changes from `DP` to `LS`.

4. Initialize the load-sharing mirror:

   `snapmirror initialize-ls-set -source-path` *`SVM:volume|cluster://SVM/volume`*

   Initialization can be time-consuming. You might want to run the baseline transfer in off-peak hours.

   For complete command syntax, see the man page.

   The following example initializes the load-sharing mirror for the root volume `svm1_root`:

   ```
   cluster_src::> snapmirror initialize-ls-set -source-path svm1:svm1_root
   ```

Update a load-sharing mirror relationship

Load-sharing mirror (LSM) relationships are updated automatically for SVM root volumes after a volume in the SVM is mounted or unmounted, and during `volume create` operations that include the `junction-path`option. You can manually update a LSM relationship if you want it updated before the next scheduled update.

Load-sharing mirror relationships update automatically in the following circumstances:

- It's time for a scheduled update
- A mount or unmount operation is performed on a volume in the SVM root volume
- A `volume create` command is issued that includes the `juntion-path` option

Step

1. Update a load-sharing mirror relationship manually:

   `snapmirror update-ls-set -source-path` *`SVM:volume|cluster://SVM/volume`*

   The following example updates the load-sharing mirror relationship for the root volume `svm1_root`:

```
cluster_src::> snapmirror update-ls-set -source-path svm1:svm1_root
```

Promote a load-sharing mirror

If a root volume is permanently unavailable, you can promote the load-sharing mirror (LSM) volume to provide write access to root volume data.

What you'll need

You must use advanced privilege level commands for this task.

Steps

1. Change to advanced privilege level:

   ```
   set -privilege advanced
   ```

2. Promote an LSM volume:

   ```
   snapmirror promote -destination-path SVM:volume|cluster://SVM/volume
   ```

   For complete command syntax, see the man page.

   The following example promotes the volume `svm1_m2` as the new SVM root volume:

   ```
   cluster_src::*> snapmirror promote -destination-path svm1:svm1_m2

   Warning: Promote will delete the offline read-write volume
            cluster_src://svm1/svm1_root and replace it with
            cluster_src://svm1/svm1_m2. Because the volume is offline,
            it is not possible to determine whether this promote will
            affect other relationships associated with this source.
   Do you want to continue? {y|n}: y
   ```

   Enter `y`. ONTAP makes the LSM volume a read/write volume, and deletes the original root volume if it is accessible.

   > The promoted root volume might not have all of the data that was in the original root volume if the last update did not occur recently.

3. Return to admin privilege level:

   ```
   set -privilege admin
   ```

4. Rename the promoted volume following the naming convention you used for the root volume:

   ```
   volume rename -vserver SVM -volume volume -newname new_name
   ```

   The following example renames the promoted volume `svm1_m2` with the name `svm1_root`:

```
cluster_src::> volume rename -vserver svm11 -volume svm1_m2 -newname
svm1_root
```

5. Protect the renamed root volume, as described in step 3 through step 4 in Creating and initializing load-sharing mirror relationships.

SnapMirror technical details

Use path name pattern matching

You can use pattern matching to specify the source and destination paths in `snapmirror` commands.

`snapmirror` commands use fully qualified path names in the following format: `vserver:volume`. You can abbreviate the path name by not entering the SVM name. If you do this, the `snapmirror` command assumes the local SVM context of the user.

Assuming that the SVM is called “vserver1” and the volume is called “vol1”, the fully qualified path name is `vserver1:vol1`.

You can use the asterisk (*) in paths as a wildcard to select matching, fully qualified path names. The following table provides examples of using the wildcard to select a range of volumes.

`*`	Matches all paths.
`vs*`	Matches all SVMs and volumes with SVM names beginning with `vs`.
`:*src`	Matches all SVMs with volume names containing the `src` text.
`:vol`	Matches all SVMs with volume names beginning with `vol`.

```
vs1::> snapmirror show -destination-path *:*dest*

Progress
Source              Destination  Mirror        Relationship  Total
Last
Path          Type  Path         State         Status        Progress
Healthy Updated
------------ ---- ------------ ------------- -------------- ----------
------- --------
vs1:sm_src2
              DP   vs2:sm_dest1
                                Snapmirrored  Idle           -
true    -
```

Use extended queries to act on many SnapMirror relationships

You can use *extended queries* to perform SnapMirror operations on many SnapMirror relationships at one time. For example, you might have multiple uninitialized SnapMirror relationships that you want to initialize using one command.

About this task

You can apply extended queries to the following SnapMirror operations:

- Initializing uninitialized relationships
- Resuming quiesced relationships
- Resynchronizing broken relationships
- Updating idle relationships
- Aborting relationship data transfers

Step

1. Perform a SnapMirror operation on many relationships:

   `snapmirror command` `{-state state } *`

   The following command initializes SnapMirror relationships that are in an `Uninitialized` state:

   ```
   vs1::> snapmirror initialize {-state Uninitialized} *
   ```

Ensure a common Snapshot copy in a mirror-vault deployment

You can use the `snapmirror snapshot-owner create` command to preserve a labeled Snapshot copy on the secondary in a mirror-vault deployment. Doing so ensures that a common Snapshot copy exists for the update of the vault relationship.

About this task

If you use a combination mirror-vault fan-out or cascade deployment, you should keep in mind that updates will fail if a common Snapshot copy does not exist on the source and destination volumes.

This is never an issue for the mirror relationship in a mirror-vault fan-out or cascade deployment, since SnapMirror always creates a Snapshot copy of the source volume before it performs the update.

It might be an issue for the vault relationship, however, since SnapMirror does not create a Snapshot copy of the source volume when it updates a vault relationship. You need to use the `snapmirror snapshot-owner create` to ensure that there is at least one common Snapshot copy on both the source and destination of the vault relationship.

Steps

1. On the source volume, assign an owner to the labeled Snapshot copy you want to preserve:

   `snapmirror snapshot-owner create -vserver` *`SVM`* `-volume` *`volume`* `-snapshot` *`snapshot`* `-owner` *`owner`*

   The following example assigns `ApplicationA` as the owner of the `snap1` Snapshot copy:

   ```
   clust1::> snapmirror snapshot-owner create -vserver vs1 -volume vol1
   -snapshot snap1 -owner ApplicationA
   ```

2. Update the mirror relationship, as described in Updating a replication relationship manually.

   Alternatively, you can wait for the scheduled update of the mirror relationship.

3. Transfer the labeled Snapshot copy to the vault destination:

   `snapmirror update -source-path` *`SVM:volume|cluster://SVM/volume`*`, … -destination -path` *`SVM:volume|cluster://SVM/volume`*`, … -source-snapshot snapshot`

   For complete command syntax, see the man page.

   The following example transfers the `snap1` Snapshot copy

   ```
   clust1::> snapmirror update -vserver vs1 -volume vol1
   -source-snapshot snap1
   ```

   The labeled Snapshot copy will be preserved when the vault relationship is updated.

4. On the source volume, remove the owner from the labeled Snapshot copy:

   `snapmirror snapshot-owner delete -vserver` *`SVM`* `-volume` *`volume`* `-snapshot` *`snapshot`* `-owner` *`owner`*

   The following examples removes `ApplicationA` as the owner of the `snap1` Snapshot copy:

```
clust1::> snapmirror snapshot-owner delete -vserver vs1 -volume vol1
-snapshot snap1 -owner ApplicationA
```

Compatible ONTAP versions for SnapMirror relationships

You should verify that the source and destination volumes are running compatible ONTAP versions before creating a SnapMirror data protection relationship.

Version-independence is not supported for SVM replication.

SnapMirror DR relationships

For SnapMirror relationships of type “DP” and policy type “async-mirror”:

In the following table, the column on the left indicates the ONTAP version on the source volume, and the top row indicates the ONTAP versions you can have on your destination volume.

Source	Destination												
	8.2	8.3	9.0	9.1	9.2	9.3	9.4	9.5	9.6	9.7	9.8	9.9.1	9.10.1
8.2	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No
8.3	No	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No
9.0	No	No	Yes	Yes	Yes	No	No	No	No	No	No	No	No
9.1	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No	No
9.2	No	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No
9.3	No	No	No	No	No	Yes	Yes	Yes	No	No	No	No	No
9.4	No	No	No	No	No	No	Yes	Yes	Yes	No	No	No	No
9.5	No	No	No	No	No	No	No	Yes	Yes	Yes	No	No	No
9.6	No	No	No	No	No	No	No	No	Yes	Yes	Yes	No	No
9.7	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	No
9.8	No	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes

9.9.1	No	No	No	No	No	No	No	No	No	No	No	Yes	Yes
9.10.1	No	No	No	No	No	No	No	No	No	No	No	No	Yes

Interoperability is not bidirectional.

Unified replication relationships

For SnapMirror relationships of type “XDP”, using on premises or Cloud Volumes ONTAP releases:

The asterisk (*) after the release version indicates a Cloud Volumes ONTAP-only release.

Table 2: Interoperability for ONTAP version 8.3.x and later

ONTAP version…	Interoperates with previous ONTAP versions…															
	8.3.x	8.3.2 P4	9.0	9.1	9.2	9.3	9.4	9.5	9.6	9.7	9.8	9.9.0 *	9.9.1	9.10.0*	9.10.1	9.11.0*
8.3.x	Yes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
8.3.2 P4	Yes	Yes	-	-	-	-	-	-	-	-	-	-	-	-	-	-
9.0	Yes	Yes	Yes	-	-	-	-	-	-	-	-	-	-	-	-	-
9.1	Yes	Yes	Yes	Yes	-	-	-	-	-	-	-	-	-	-	-	-
9.2	No	Yes	Yes	Yes	Yes	-	-	-	-	-	-	-	-	-	-	-
9.3	No	No	Yes	Yes	Yes	Yes	-	-	-	-	-	-	-	-	-	-
9.4	No	No	Yes	Yes	No	Yes	Yes	-	-	-	-	-	-	-	-	-
9.5	No	No	No	Yes	No	Yes	Yes	Yes	-	-	-	-	-	-	-	-
9.6	No	No	No	Yes	No	Yes	No	Yes	Yes	-	-	-	-	-	-	-
9.7	No	No	No	No	No	Yes	No	Yes	Yes	Yes	-	-	-	-	-	-
9.8	No	No	No	No	No	Yes	No	Yes	Yes	Yes	Yes	-	-	-	-	-

9.9.0*	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	-	-	-	-
9.9.1	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	-	-	-
9.10.0*	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	-	-
9.10.1	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	-
9.11.0*	No	No	No	No	No	No	No	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes

Locate the higher, more recent ONTAP version in the left column, and in the top row locate the lower ONTAP version to determine interoperability. Interoperability is bidirectional.

SnapMirror limitations

You should be aware of basic SnapMirror limitations before creating a data protection relationship.

- A destination volume can have only one source volume.

  A source volume can have multiple destination volumes. The destination volume can be the source volume for any type of SnapMirror replication relationship.

- You can fan out a maximum of eight destination volumes from a single source volume.
- You cannot restore files to the destination of a SnapMirror DR relationship.
- Source or destination SnapVault volumes cannot be 32-bit.
- The source volume for a SnapVault relationship should not be a FlexClone volume.

  The relationship will work, but the efficiency offered by FlexClone volumes will not be preserved.