White Paper

EMC CLARiiON SnapView Snapshots and Snap Sessions Knowledgebook

A Detailed Review

Abstract

This white paper describes the EMC® SnapView™ product and its optional features. Additionally, it defines associated terminology, as well as operational characteristics combined with performance considerations.

April 2008

Table of Contents

Executive summary

This white paper describes SnapView™ functionality and associated terminology. The focus is on operational characteristics. The following pages discuss what SnapView snapshots are, how they work, and the benefits they bring to storage area network (SAN) environments.

Introduction

SnapView is an optional software package for the EMC® CLARiiON® storage product. Using SnapView, users can create a point-in-time view—or multiple views—of a logical unit (LUN), which can subsequently be made accessible to another host, or simply held as a point-in-time copy for possible restoration. For instance, a system administrator can make the snapshot session accessible to a backup host so that the production host can continue processing on the source LUN without the downtime traditionally associated with backup processes. SnapView can also be used in conjunction with other secondary applications (such as decision-support systems) by providing a view of the production data that can be used without impacting the integrity of the source data on the active LUN.

SnapView snapshots use a pointer-and-copy-based design; a memory map keeps track of chunks (groups of blocks) of data. As write requests are made to the active LUN (the *source LUN*), the chunks are copied to a reserved area consisting of nonvolatile space on private LUNs (the *reserved LUN pool*[1]*)*, and the memory map is updated with the new location of each of these chunks. This process, referred to as *copy on first write*, occurs only when the first write request is made to any chunk on the source LUN, since once the original data is safely copied to the reserved LUN, any number of writes can be made to the same chunk without losing the original view of the data.

The source LUN, the reserved LUN, and the memory map work together to create the snapshot. The snapshot is made visible to a secondary host when the snapshot is activated to a SnapView session (given that a snapshot has been defined and has been added to a storage group connected to a host).

The SnapView software provides users another feature called clones. The clone feature of SnapView works similarly to the snapshot feature, and is discussed in a separate white paper, titled *EMC CLARiiON SnapView Clones – A Detailed Review.*

Audience

The intended audience for this paper is customers, partners, and EMC field personnel requiring information about the features available with SnapView snapshots. This paper also discusses implementation details for configuring and using SnapView snapshots.

Terminology

The following SnapView terms are provided to help clarify the components and features of SnapView.

Reserved LUN pool — Collection of LUNs used to support the pointer-based design of SnapView. As the first SnapView session is started on a given source LUN, a reserved LUN is assigned to the source LUN. If a SnapView session runs long enough for the assigned reserved LUN to be filled, the next available LUN in the reserved LUN pool will be assigned to the source LUN. Reserved LUNs are thus assigned on a per-source-LUN basis, such that source LUNs have a one-to-many relationship to their reserved LUNs.[2]

[1] In versions preceding SnapView 2.3, the reserved LUN was called the *SnapView cache*.
[2] In early revisions of SnapView, the reserved LUNs (called cache LUNs in those versions of software) were shared resources, assigned on a per-SP basis and shared among all source LUNs owned by that SP. Starting with SnapView version 1.3, the reserved LUNs were assigned on a per-SP basis primarily to support the multi-session enhancement and, subsequently, session persistence. In this way, the concept of a

Rollback — Enables recovery of a source LUN by copying the data in the reserved LUN back to the source LUN. If the rollback is performed while a snapshot is still active to this session, the snapshot writes will be copied to the source LUN. If the snapshot is deactivated, the original session data will be copied to the source LUN.

SnapView session — Process of defining the point-in-time designation by invoking copy-on-first-write activity for updates to the source LUN. Starting a session assigns a reserved LUN to the source LUN if no other sessions are running on this same source LUN. Note that as far as this session is concerned, until a snapshot is activated, the point-in-time copy is not visible to any servers. However, we are tracking the source LUN so we can, at any time in the future, activate a snapshot to this session in order to present the point-in-time image (when the SnapView session was started) to a host. As noted earlier, each source LUN can have up to eight sessions.

SnapView snapshot — The defined virtual device that is presented to a host and enables visibility into running sessions. The snapshot will be defined under a source LUN in such a way that activation of that snapshot will only be allowed on any running sessions belonging to that same source LUN. A snapshot can only be assigned to a single session; thus, to have two active snapshots for the same source LUN, you must have two separate sessions running in which to activate two separate snapshots. Active snapshots are fully read and write-capable. Once the snapshot is deactivated, however, all writes to the snapshot will be deleted.

Source LUN — The LUN containing production data on which you want to start a SnapView session, and optionally activate a snapshot to that session. [3]

SnapView snapshot components

As illustrated in Figure 1, the logical view of the snapshot device is a composite of the unchanged data chunks[4] on the source volume and the data chunks that have been saved on the reserved LUN, which consists of both copy-on-first-write data and any writes to the snapshot device. Because SnapView snapshots are both readable and writeable, this means that a user can write to the snapshot device. Snapshot writes are saved on the same reserved LUN as the copy-on-first-write data.

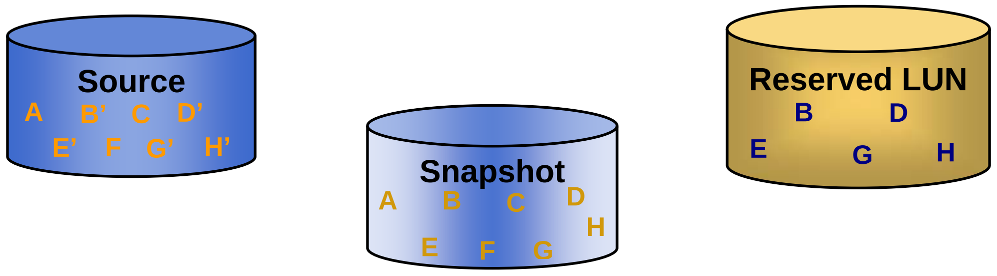


Figure 1. Writes to the source LUN cause writes to the reserved LUN pool

In Figure 1, some of the data on the source volume has changed since the session started. Those chunks are appended with a prime (') to indicate that the data changed. A copy-on-first-write operation is generated to first save those chunks to the reserved LUN before changing them on the source. Note that all chunks on the source volume appended with a prime exist in their original state in the reserved LUN (blue letters).

pool of LUNs composing the reserved LUN pool still exists; the concept now applies at the source LUN level, however, rather than at the SP level.

[3] Early revisions of SnapView were limited to one session per source LUN; starting with SnapView version 1.3, up to eight sessions can be started on a single source LUN.

[4] The SnapView chunk size is 64 KB.

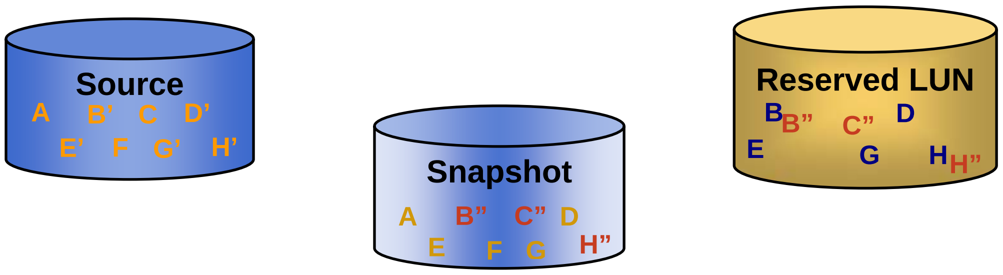


Figure 2. Writes to SnapView snapshots with data saved on the reserved LUN

In Figure 2, for any areas that were changed via the snapshot, those chunks are also written to the reserved LUN (red letters). In some cases, these writes were made to the same areas that were already in the reserved LUN (for example, chunks B and H). Also in some cases, these writes were made to areas that had not yet changed on the source; that is, there was not yet an associated copy-on-first-write operation for these chunks (for example, chunk C). Starting with release 24 of FLARE, a snapshot write will not incur a copy-on-first write operation from the source if that chunk had not yet changed on the source.

Using SnapView snapshots

SnapView tasks can be managed by one of the following tools: Navisphere® Manager user interface (UI), Navisphere command line interface (CLI), or admsnap. With Navisphere GUI and Navisphere CLI, users have an option of managing SnapView tasks through a UI or a CLI interface. The admsnap utility provides host-specific capabilities, such as bringing snapshots online. This paper will focus on using the Navisphere UI or CLI. Information on admsnap is provided in the "Appendix" section.

Installation

SnapView software resides on the CLARiiON storage system. Starting with FLARE® release 14, the SnapView package is installed at the factory. To enable the SnapView feature in the storage system, the SnapView enabler package is installed with a non-disruptive upgrade (NDU) process that supports online installation of software—that is, without requiring that the storage system be brought down during the upgrade.

SnapView appears in the Navisphere UI management window and SnapView commands are supported within the Navisphere CLI command list. Users must also have Access Logix™ enabled, as snapshots are presented to hosts that do not contain the source LUNs that have been snapped. You can do this in the Navisphere GUI by right-clicking the storage system, selecting the **properties** tab, and selecting **Data Access** under the **Storage Access** tab.

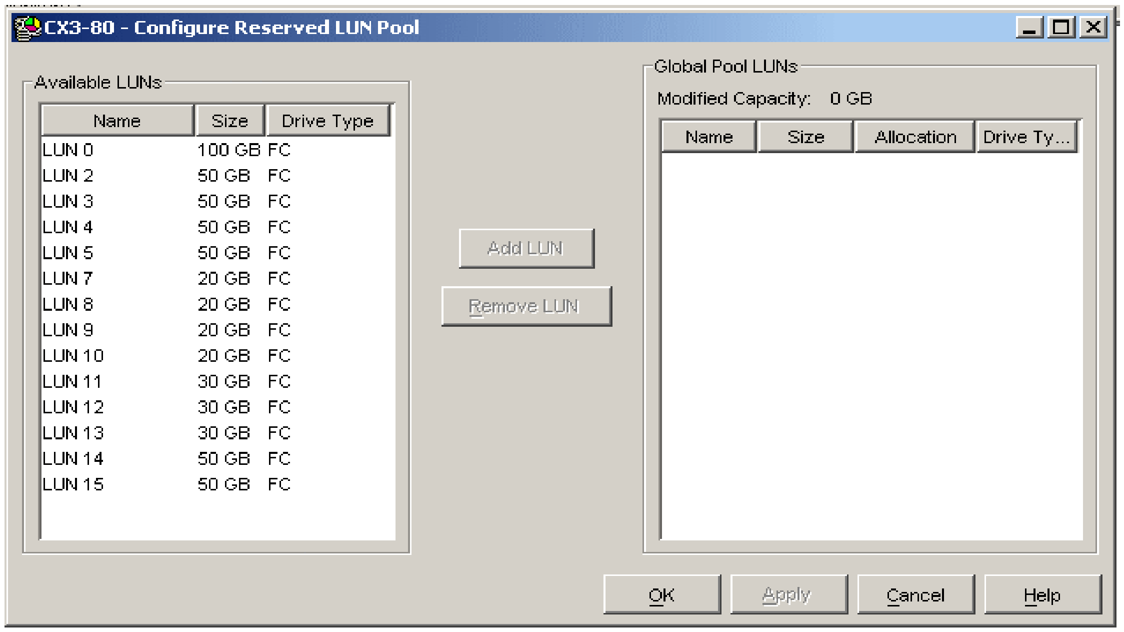
Configuring the reserved LUN pool

After installing SnapView on the CLARiiON storage system, you must determine the number and size of LUNs you will need to add to the reserved LUN pool. The requirements vary for each installation, depending on the environment. Consider the following when determining how to allocate reserved LUNs:

- Rate of change of source LUN data
- Expected concurrent session count
- Expected duration of SnapView sessions
- Expected snapshot write rate

As a guide, consider relatively small LUNs, but remember that at least one LUN must be allocated for any source LUN that has a running SnapView session.

With FLARE release 24, LUNs are assigned to a global reserved LUN pool area as shown in Figure 3. Therefore, each SP can access any LUN in the reserved LUN pool. Once a reserved LUN gets allocated to a source LUN, the reserved LUN is owned by the SP that owns the source LUN. When an SP failure occurs, if the source LUN or its associated snapshot LUN trespass, the reserved LUNs also trespass to the surviving SP.



Name	Size	Drive Type
LUN 0	100 GB	FC
LUN 2	50 GB	FC
LUN 3	50 GB	FC
LUN 4	50 GB	FC
LUN 5	50 GB	FC
LUN 7	20 GB	FC
LUN 8	20 GB	FC
LUN 9	20 GB	FC
LUN 10	20 GB	FC
LUN 11	30 GB	FC
LUN 12	30 GB	FC
LUN 13	30 GB	FC
LUN 14	50 GB	FC
LUN 15	50 GB	FC

Figure 3. Configuring the reserved LUN pool

When you start the first session on a source LUN, one reserved LUN will be assigned to that source LUN. If the reserved LUN becomes full during the time this session is running, the next available reserved LUN will be assigned automatically to the source LUN. If there are no more reserved LUNs available in the reserved LUN pool, then the session that caused the reserved LUN to become full will be terminated automatically. Any reserved LUNs being used by that session will return to the reserved LUN pool, available for use by other sessions.

To help users monitor the reserved LUN usage, SnapView will send an alert via Event Monitor when the combined reserved LUN usage per source LUN reaches 50 percent, then at 75 percent, and then every 5 percent thereafter. This way, users are alerted as the reserved LUN begins filling, and additional LUNs can be added to the reserved LUN pool if needed.

If a user consumes the designated maximum number of reserved LUNs allowed per storage system, no additional reserved LUNs can be added when needed. As a result, sessions that need additional reserved LUN space will not be able to obtain that additional space, and therefore those sessions will be terminated at the point that they exceed the allocated reserved LUN space.[5]

As noted earlier, reserved LUNs are assigned to the source LUN when the session is started. Additional reserved LUNs are assigned to the source LUN as needed; for example, when more space is needed for copy-on-first-write data that is being saved in the reserved LUN. Thus, there is a one-to-one (or, in some cases, one-to-many) correspondence between source LUNs and their associated reserved LUN(s).

In this design of an aggregate pool of reserved LUNs, each reserved LUN is assigned to its source LUN according to availability; the first free LUN is assigned first, and so on. Since users cannot designate which reserved LUNs are assigned to which source LUNs, EMC recommends that you make these LUNs a

[5] SnapView limits (as of the date of this publication) are provided in the "Appendix" section. The most up-to-date limits can be found in the *CLARiiON Open Systems Configuration Guide*.

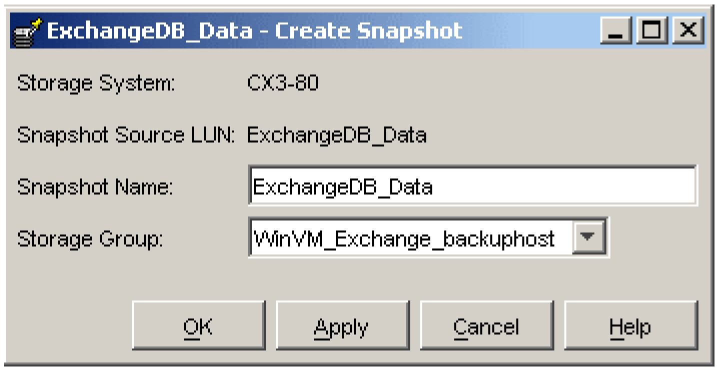
uniform size. Any source LUN can use any LUN in the reserved LUN pool that is not allocated. The maximum number of reserved LUNs assigned to this global pool will be confined to the storage-system-defined maximum count for reserved LUNs.[6] Please note that for AX4 systems running Navisphere Manager each SP has its own separate reserved LUN pool.

For more information on reserved LUN pool considerations, refer to the *CLARiiON Reserved LUN Pool Configuration Considerations* white paper on EMC Powerlink®.

Creating a SnapView snapshot

With the necessary software components installed and LUNs allocated to the reserved LUN pool, we can start using the SnapView feature. At this point, you can choose whether you want to start the session first, or create the snapshot first. For this discussion, we'll begin by creating a SnapView snapshot.

Creating the snapshot enables the allocation of an offline device to a storage group, which is ready to be activated to a running session whenever the user desires. Following the procedure for starting a session via the Navisphere UI (refer to the "Starting a SnapView session" section on page 12), right-click a source LUN and select **Create Snapshot**. This brings up the dialog box shown in Figure 4.


ExchangeDB_Data - Create Snapshot

Storage System: CX3-80

Snapshot Source LUN: ExchangeDB_Data

Snapshot Name: ExchangeDB_Data

Storage Group: WinVM_Exchange_backuphost

Figure 4. Creating a snapshot

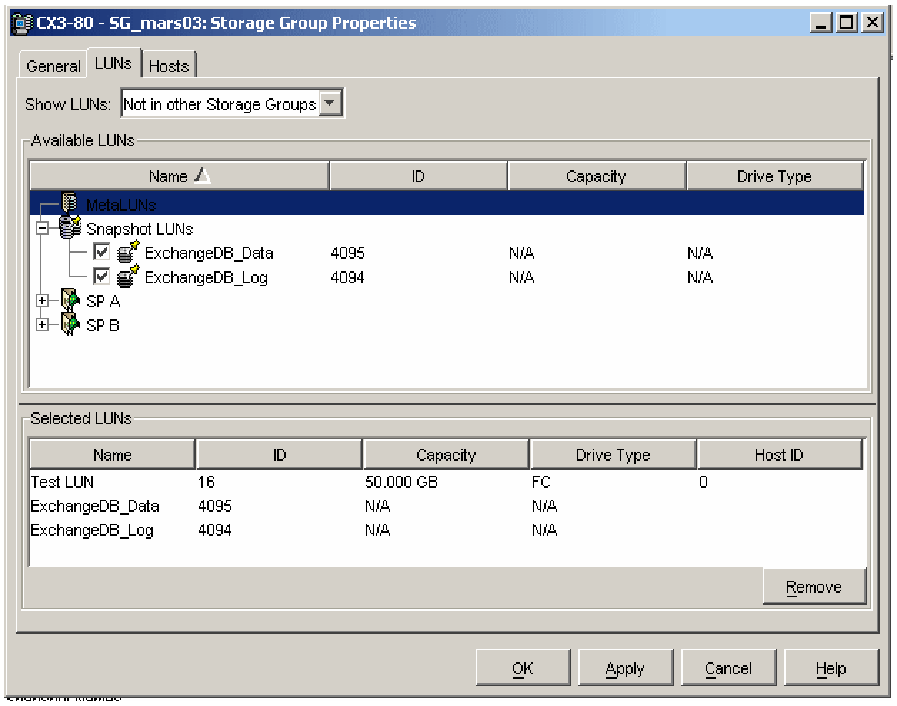
From this dialog box, you create or define a snapshot for source **ExchangeDB_Data** (LUN200). This snapshot will be offline until activated to a running SnapView session. You can also select which host will see this snapshot once it is activated to a session by assigning it to a storage group. Alternatively, you can do this in a separate step, as shown in Figure 5.

The appropriate Naviseccli command to perform the same function is:

```
naviseccli -h <SP A IP address> snapview -createsnapshot 200 -snapshotname
ExchangeDB_Data
```

If you choose not to assign the snapshot to a storage group when creating the snapshot (or if it becomes necessary to move a snapshot out of one storage group and into another), follow the standard procedure for storage allocation. That is, using the Navisphere UI you can access the storage group properties for a given host, and select which storage resources you wish to assign to that host—inclusive of LUNs and/or snapshots.

[6] The "Appendix" in this paper specifies the SnapView limits at the time this paper was published. Please refer to the most recent version of the *Open Systems Configuration Guide* for the most up-to-date SnapView limits.


CX3-80 - SG_mars03: Storage Group Properties
General
LUNs
Hosts
Show LUNs: Not in other Storage Groups
Available LUNs
Remove
OK
Apply
Cancel
Help

Name	ID	Capacity	Drive Type
MetaLUNs			
Snapshot LUNs			
ExchangeDB_Data	4095	N/A	N/A
ExchangeDB_Log	4094	N/A	N/A
SP A			
SP B			

Selected LUNs

Name	ID	Capacity	Drive Type	Host ID
Test LUN	16	50.000 GB	FC	0
ExchangeDB_Data	4095	N/A	N/A	
ExchangeDB_Log	4094	N/A	N/A	

Figure 5. Assigning a snapshot to a storage group

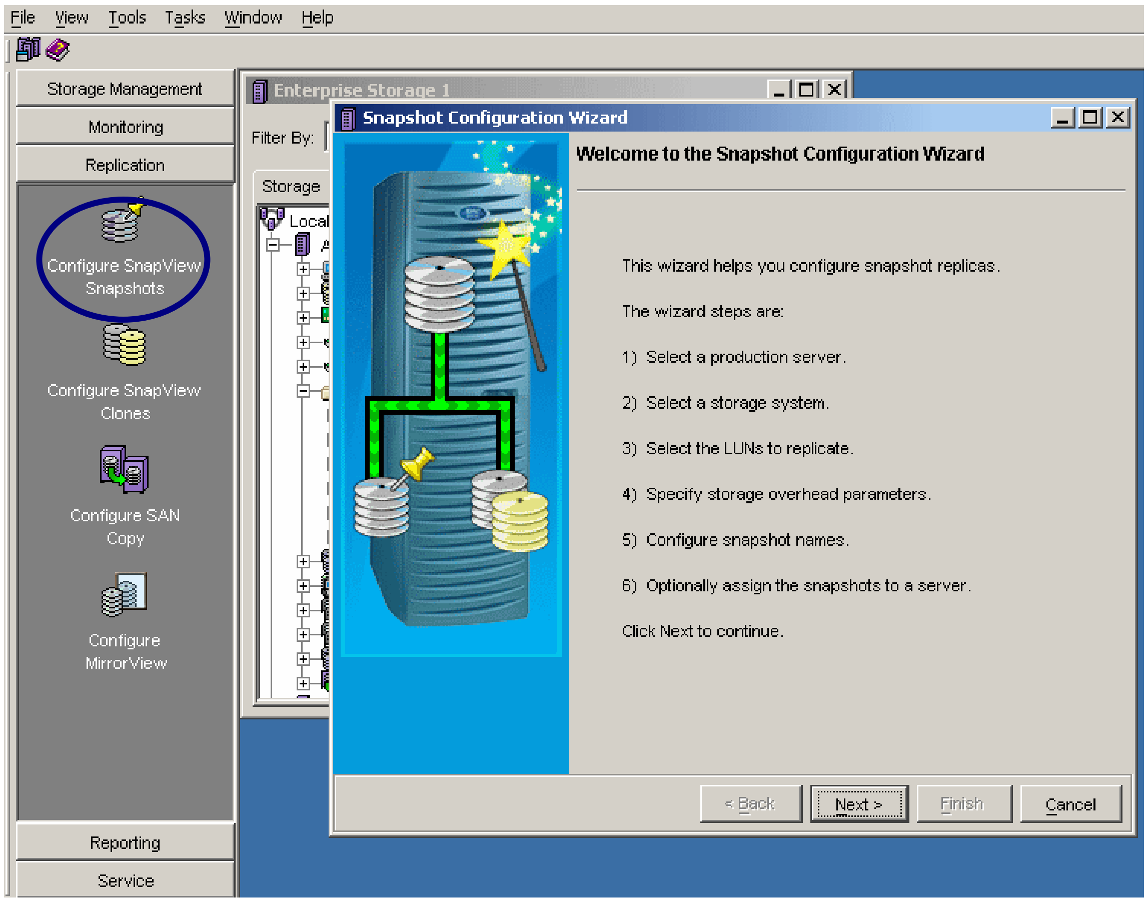
The appropriate Naviseccli command to perform the same function is:

```
naviseccli -h <SP address> snapview -storagegroup -addsnapshot -gname
WinVM_Exchange_backuphost -snapshotname ExchangeDB_data
```

Once snapshots have been defined they will remain inactive, seen as offline devices by the connected operating system, and come online only when activated to a running SnapView session.

Using the SnapView Snapshot Configuration Wizard

Beginning in release 24, configuration of reserved LUNs and snapshots for one or more source LUNs can be automated through the SnapView Snapshot Configuration Wizard. This wizard is launched by selecting **Configure SnapView Snapshots** as illustrated in Figure 6.



Welcome to the Snapshot Configuration Wizard

This wizard helps you configure snapshot replicas.

The wizard steps are:

1) Select a production server.

2) Select a storage system.

3) Select the LUNs to replicate.

4) Specify storage overhead parameters.

5) Configure snapshot names.

6) Optionally assign the snapshots to a server.

Click Next to continue.

Figure 6. SnapView Snapshot Configuration Wizard

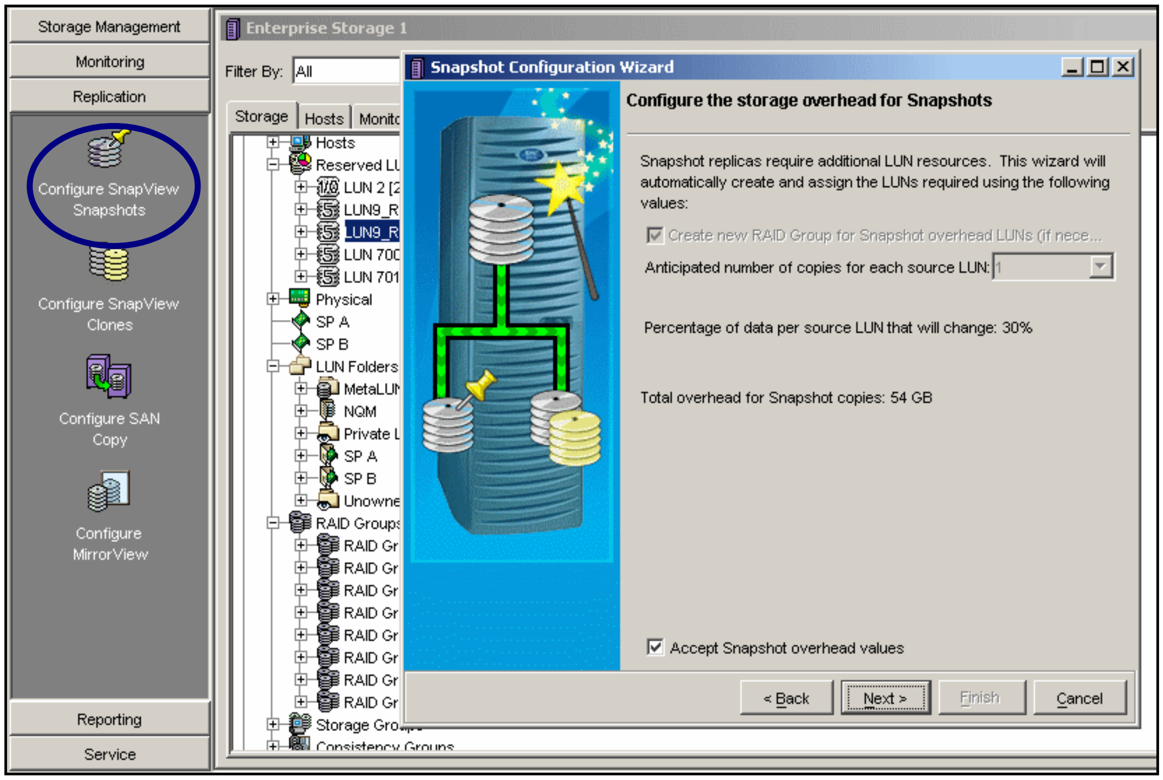
The Navisphere SnapView Snapshot Configuration Wizard consolidates all the steps required for configuring reserved LUNs and snapshots for one or more source LUNs. As shown in Figure 6, the Snapshot Configuration Wizard will:

1. Bind and allocate new reserved LUNs for the selected source LUN(s).
2. Create the snapshot(s) for the identified source LUN(s), if desired.
3. Assign the snapshot(s) to a secondary server storage group, if desired.

Two reserved LUNs, which are usually allocated from different RAID groups than the source LUN, are created for each source LUN. Starting with release 26, RAID 6 is supported with SnapView. The wizard does not use LUNs that are already allocated from the reserved LUN pool. If you run the wizard a second time with the same selected source LUN, it attempts to remove the old reserved LUNs after creating the new ones, provided that the old LUNs are unallocated. The wizard will choose a RAID group for reserved LUNs if it meets the following criteria:

- Does not contain the source LUN
- Contains fewest server-visible LUNs.
- Contains fewest clones and mirror images
- Contains the most free space

The wizard also defaults the reserved LUN size to 30 percent of the size of source LUN. This size can be changed by clearing the **Accept Snapshot overhead values** checkbox and selecting the appropriate size (with a minimum of 20 percent) as shown in Figure 7.



Configure the storage overhead for Snapshots

Snapshot replicas require additional LUN resources. This wizard will automatically create and assign the LUNs required using the following values:

☑ Create new RAID Group for Snapshot overhead LUNs (if nece...

Anticipated number of copies for each source LUN: 1

Percentage of data per source LUN that will change: 30%

Total overhead for Snapshot copies: 54 GB

☑ Accept Snapshot overhead values

< Back | Next > | Finish | Cancel

Figure 7. Choosing the sizes of reserved LUNs

The standard Navisphere tree interface (object-based menu) or Navisphere CLI is still accessible for users who are accustomed to the traditional methods of creating and configuring reserved LUNs and snapshot LUNs (as described previously), or who desire greater control over the configuration.

Starting a SnapView session

As noted earlier, the user can either create the snapshot first *or* start the snap session first. Since in this discussion we began by creating a snapshot, we will now discuss the process of starting a snap session.

When a session is started, a reserved LUN is assigned to each source LUN if no other session is already running on that source LUN. The storage system will start tracking the source LUN, invoking copy-on-first-write operations for data areas that are changed on the source LUN.

When starting a session, you have two options:

- **Single-LUN session —** You may wish to have a session running on a single source LUN. If using the Navisphere UI, right-click the desired LUN and select **Start Session**. Enter a session name; then click **OK**, and the session will start.
- **Multi-LUN session —** For consistency, you may want to start a session containing multiple source LUNs when you have as an interdependent data set (for example a database consisting of several LUNs). With the addition of these consistent operations in FLARE release 19, users can now have consistent local replicas using SnapView sessions owned by either SP.

As with other SnapView operations, **start** can be initiated in the Navisphere Manger GUI or Navisphere CLI. Admsnap can also be used to issue the **start** command. You also have the option of managing consistent operations manually (through Navisphere Manager or CLI), or to incorporate the consistent operations into a script (using Navisphere CLI or admsnap, as applicable).

If using the Navisphere UI, right-click the storage system, and select **Start Session**. This opens a dialog box in which you select the source LUNs for the session. Give the session a name and click **OK** to start, as shown in Figure 8.

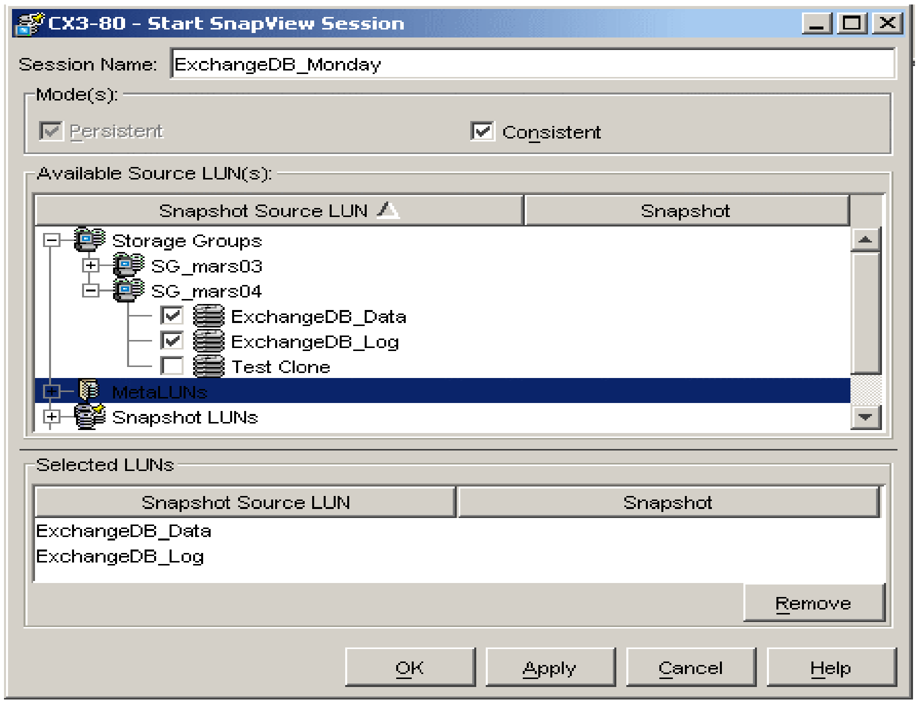


Figure 8. Selecting LUNs in a multi-LUN session

In Figure 8, multiple source LUNs are selected, even though some of the LUNs do not have any snapshots created. Remember, starting the session starts the process of tracking changes and preserving data. You can do that at any time, and then create a snapshot and activate it to enable visibility into that point-in-time view. Starting with release 24, all SnapView sessions are configured as persistent sessions as shown in Figure 8. For additional details on persistent sessions, please see the “Appendix.” To make a session consistent, select **Consistent** in the **Optional Modes** section of the **Start SnapView Session** dialog box.

The appropriate Naviseccli command to perform this function is:

```
naviseccli -h <SP A IP address> snapview -startsession Exchange_DB_Monday
-lun Exchange_DB_data Exchange_DB_log -consistent
```

As shown here, if the session spans both SPs, you only need to specify a single SP and the software will start the session on both SPs.

You can also start a session in admsnap (which is located on the production server) with this command:

```
admsnap start -s Exchange_DB_Monday -o Exchange_DB_data Exchange_DB_log -c
```

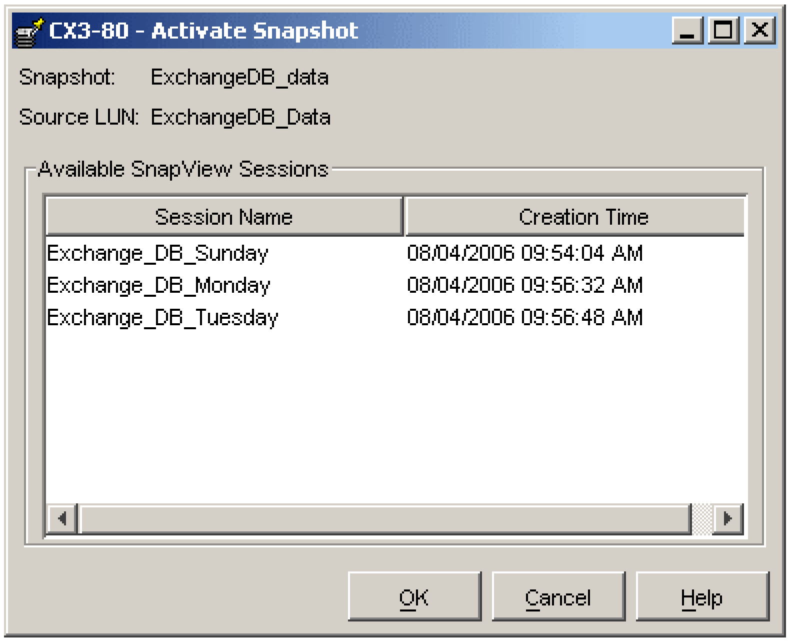
The **-consistent** or **-c** switch mentioned in the above Naviseccli and admsnap commands designates the session as consistent. With a consistent start operation, sessions containing LUNs owned by both SPs can be started with a single command. This also means that, once a consistent session has been started using a given name, it is not possible to issue a subsequent **startsession** command using the same session name. The session must either be stopped, or the user must select a different session name for the subsequent session. In other words, consistent session names are now unique and cannot be reused until the consistent session is stopped; and when starting a consistent session, the user must specify all LUNs up front, since

LUNs cannot be added after the start command is issued. Note that a consistent session is counted as one of the eight sessions per source, and as one of the total sessions allowed per array.

Activating a snapshot to a session

Once you have at least one SnapView session running and a snapshot created for a given source LUN, you can activate a snapshot to a session. This action essentially associates a snapshot to the point-in-time view provided by a particular session. Using the Navisphere UI, you can activate a snapshot by right-clicking the snapshot and selecting **Activate**.

Figure 9 shows three running sessions on the same source LUN. The start date and time are displayed for each so the user knows which point-in-time view of data will be presented for the activated snapshot.



Snapshot: ExchangeDB_data

Source LUN: ExchangeDB_Data

Session Name	Creation Time
Exchange_DB_Sunday	08/04/2006 09:54:04 AM
Exchange_DB_Monday	08/04/2006 09:56:32 AM
Exchange_DB_Tuesday	08/04/2006 09:56:48 AM

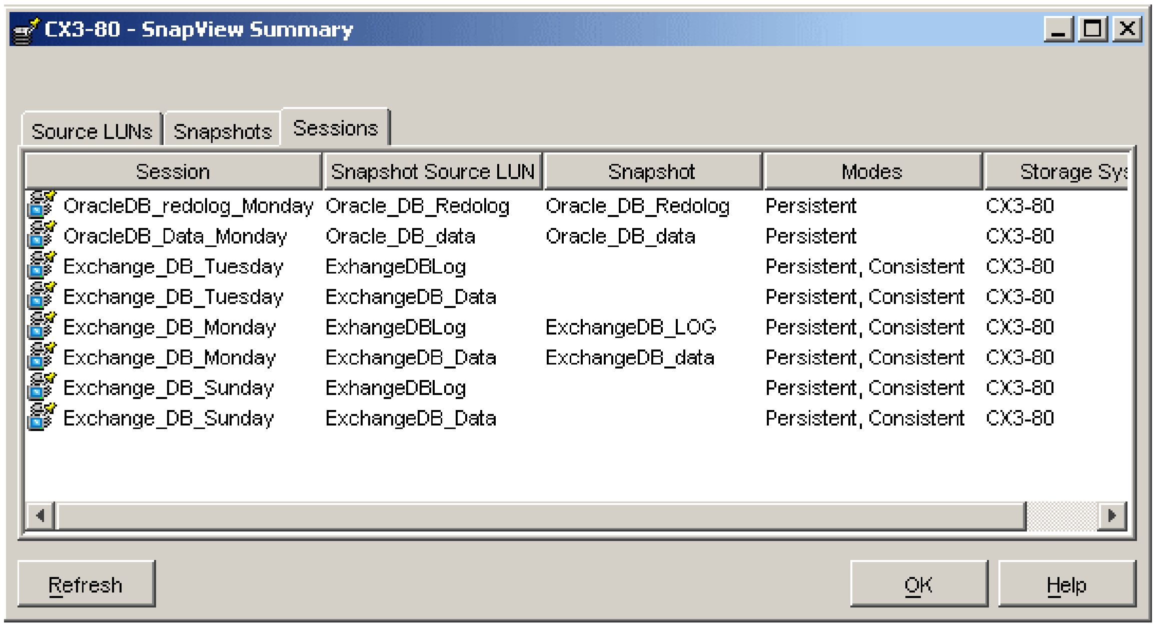
Figure 9. Activating a snapshot to a session

The appropriate Naviseccli command to perform the same function is:

```
naviseccli -h <SP address> snapview -activatesnapshot Exchange_DB_Monday
```

If the snapshot is already in a storage group and allocated to a host, following activation the connected host should be able to see this point-in-time copy of source LUN data after a bus rescan at the host level.

SnapView provides a summary window displaying all running sessions and which source LUN(s) those sessions are associated with. Any snapshots that have been activated to sessions are displayed as well.



Session	Snapshot Source LUN	Snapshot	Modes	Storage Sys
OracleDB_redolog_Monday	Oracle_DB_Redolog	Oracle_DB_Redolog	Persistent	CX3-80
OracleDB_Data_Monday	Oracle_DB_data	Oracle_DB_data	Persistent	CX3-80
Exchange_DB_Tuesday	ExhangeDBLog		Persistent, Consistent	CX3-80
Exchange_DB_Tuesday	ExchangeDB_Data		Persistent, Consistent	CX3-80
Exchange_DB_Monday	ExhangeDBLog	ExchangeDB_LOG	Persistent, Consistent	CX3-80
Exchange_DB_Monday	ExchangeDB_Data	ExchangeDB_data	Persistent, Consistent	CX3-80
Exchange_DB_Sunday	ExhangeDBLog		Persistent, Consistent	CX3-80
Exchange_DB_Sunday	ExchangeDB_Data		Persistent, Consistent	CX3-80

Figure 10. SnapView summary

As shown in Figure 10, several snapshots are listed as being associated with the sessions that have been started, indicating that these snapshots have been activated to that particular session. Additionally, you will see that there are some single-LUN sessions (`OracleDB_Data_Monday` and `OracleDB_Redolog_Monday`), as well as multi-LUN sessions (`Exchange_DB_Monday`). As noted, multi-LUN sessions provide users with a simpler method of handling interdependent datasets. For multi-LUN sessions, each source LUN represented in the session has a separate snapshot (if the user wants to view the contents from a host).

Host access to snapshots

In the previous example, multiple sessions are running on the source LUN for which we have created a snapshot—each with different start times. You can choose to activate a snapshot to any one of those sessions, to provide host visibility to the session point-in-time view (as long as the source LUN has a defined snapshot available and not activated for another session). As an example, you could start a session on a source LUN for each day of the week. If at some point you want to recover a file that was created on Tuesday but either was deleted or became corrupt after Thursday, you could activate a snapshot for the session started on Wednesday and copy the file from the available snapshot—thus implementing online file recovery.

Each host will have different requirements to see the new online device, from a simple rescan or import, to a reboot (often required when you add any LUN to a storage group) The admsnap host-based utility will also assist with this process. Please refer to the SnapView Command Line interface documentation for more details on bringing snapshot (and other) devices online.[7]

EMC supports adding a snapshot to the same storage group containing that snapshot's source LUN only if Replication Manager is used to modify the disk signatures. Additionally, EMC supports this in VMware ESX environments, provided that the snapshot is presented to a different virtual machine than the one

[7] In some cases, when storage groups are first created and storage is initially presented to a host, it may be necessary to reboot the host to allocate HBA resources before you can access the snapshot. It may be most efficient to reboot the host immediately after creating the snapshot(s) and allocating it to a storage group and host. This way, when you reboot the host, it will pick up the offline device(s) and, once you activate the snapshot, you will be able to bring that device online to the operating system without having to reboot. Users should refer to OS-specific documentation for details regarding such procedures.

owning the source LUN. This is due to potential file-system labeling conflicts: remember, the snapshot will be an identical copy of the source LUN when it's activated and some operating systems may not like detecting two or more devices or volumes with identical labels. This is a recommendation, and there is no lock in place to prevent you from doing this. It is expected that snapshots will be presented to hosts other than the host connected to the source LUN, such as a separate backup server. With release 26, SnapView supports hosts that are configured using failovermode 4 (Asymmetric Active/Active feature) of the CLARiiON. For more details on how SnapView snapshots work with failovermode 4, please see the *EMC CLARiiON Asymmetric Active/Active Feature (ALUA)* white paper.

Deactivating snapshots

You can deactivate the snapshot from the session once you have finished accessing it. This does not stop the session—it merely removes the association between the snapshot and the session. Additionally, it deletes all writes to the snapshot saved in the reserved LUN as shown in Figure 11.

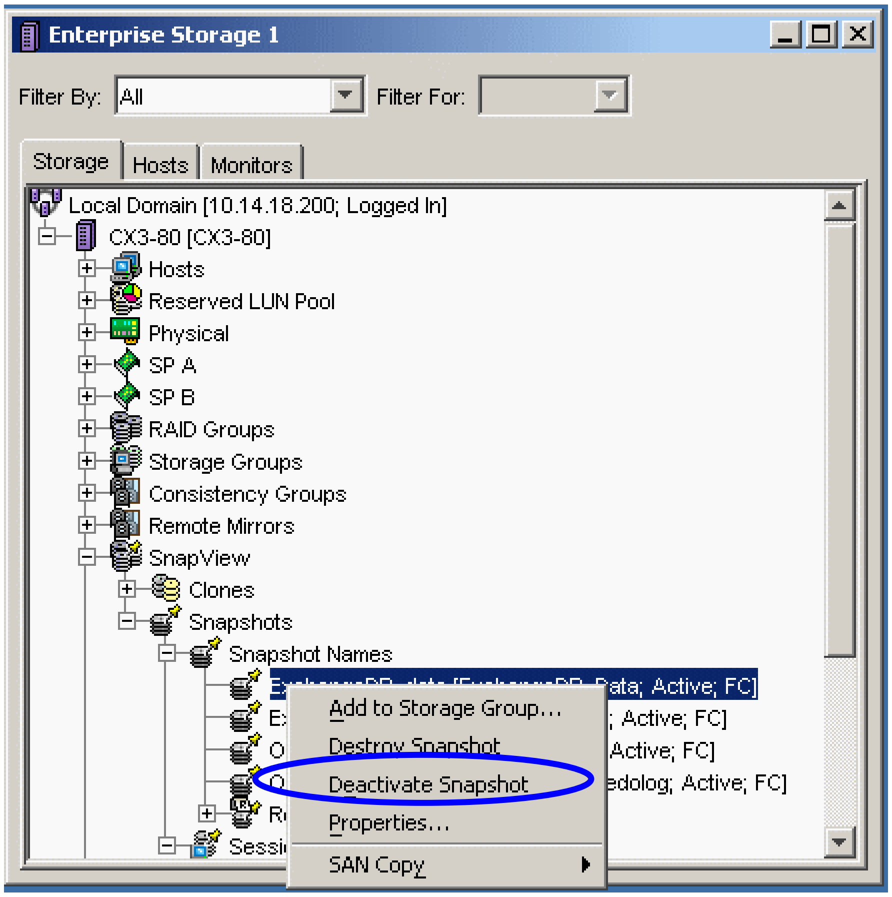


Figure 11. Deactivating snapshots

You typically deactivate a snapshot for a given session when you want to:

- Associate the snapshot to another session (that is, provide a Tuesday view of the data, rather than, say, the current Monday view)
- Associate another snapshot to the session (that is, provide the Monday view of the data to another host, via a second snapshot)
- Refresh the view of the snapshot, with the original point-in-time view of the session as it was first created (that is, if a user wrote to the snapshot and wanted to remove the writes, refreshing the snapshot with the original Monday view of the data)

When preparing to deactivate a snapshot, you should first clear the memory buffers on the secondary host. This step is important, since the operating system accessing the snapshot may have buffered information it expects to write to the snapshot. Not flushing the host buffers is dangerous because if you deactivate a snapshot on the Monday session and then activate the snapshot on the Tuesday session, the data in host buffers would be written to the Monday session and thus potentially corrupt the Tuesday view of data (that is, for the snapshot; not for the source). After flushing the host buffers, you can deactivate the snapshot, which will render it offline from the host's perspective. You may leave the snapshot in the storage group and allocated to the host, awaiting activation to the next session (which will bring it back online again).

Reactivating snapshots

If you have a running session that's being used for testing software, you may decide to revert the snapshot back to the state that it was in when you first activated it. To do this, deactivate the snapshot and then activate it again to the original session name. Any changes that were made to the snapshot by the test environment will be lost, and you're back to the start point of the data as it was when the session was created.

Alternatively, you could activate the snapshot to another session for the same source. Following the earlier example, this would allow you to change the view of data presented to the host—for instance, shifting from the Monday view to the Tuesday view.

Destroying snapshots

When you no longer need the snapshot for a given host, you can remove it from the storage group (you should deactivate it first, as described in the "Deactivating" section). Like adding snapshots to storage groups, users will follow the standard procedure for deallocating storage resources—by accessing the storage group properties for a given host, and removing the snapshots from the **Selected LUNs** pane. This returns the snapshots into the **Available LUNs** pool.

If you no longer need the snapshot for any host in your environment, you can then destroy it. To destroy a snapshot, right-click the snapshot and select **Destroy Snapshot** as shown in Figure 12.

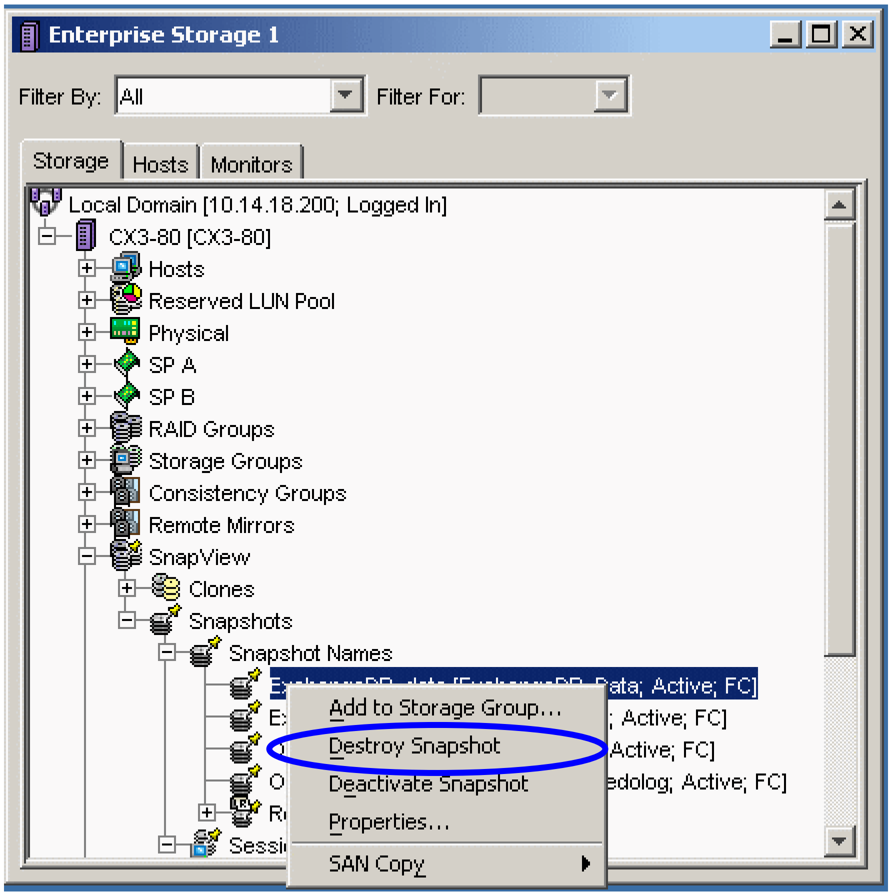


Figure 12. Destroying snapshots

The appropriate Naviseccli command to perform the same function is:

```
    naviseccli -h <SP address> snapview -rmsnapshot Exchange_DB_Monday
```

Stopping SnapView sessions

When you have no further use of the SnapView session, you can stop it. This clears all copy-on-first-write data stored in the reserved LUN. If this reserved LUN is used only for the session you are stopping (that is, if this is the only session for a given LUN), that reserved LUN returns to the reserved LUN pool.
If there are other sessions using a given reserved LUN, that reserved LUN will not be returned to the reserved LUN pool until all the sessions using that LUN have been stopped[8].

You would typically stop a given session when:

- You reach the eight-session limit per source LUN and want to retire a session so that you could start another.
- You no longer need the point-in-time data preserved by the session.

In the Navisphere UI, right-click the session to select the **Stop Session** option shown in Figure 13.

[8] These also include sessions started for incremental SAN Copy and MirrorView.

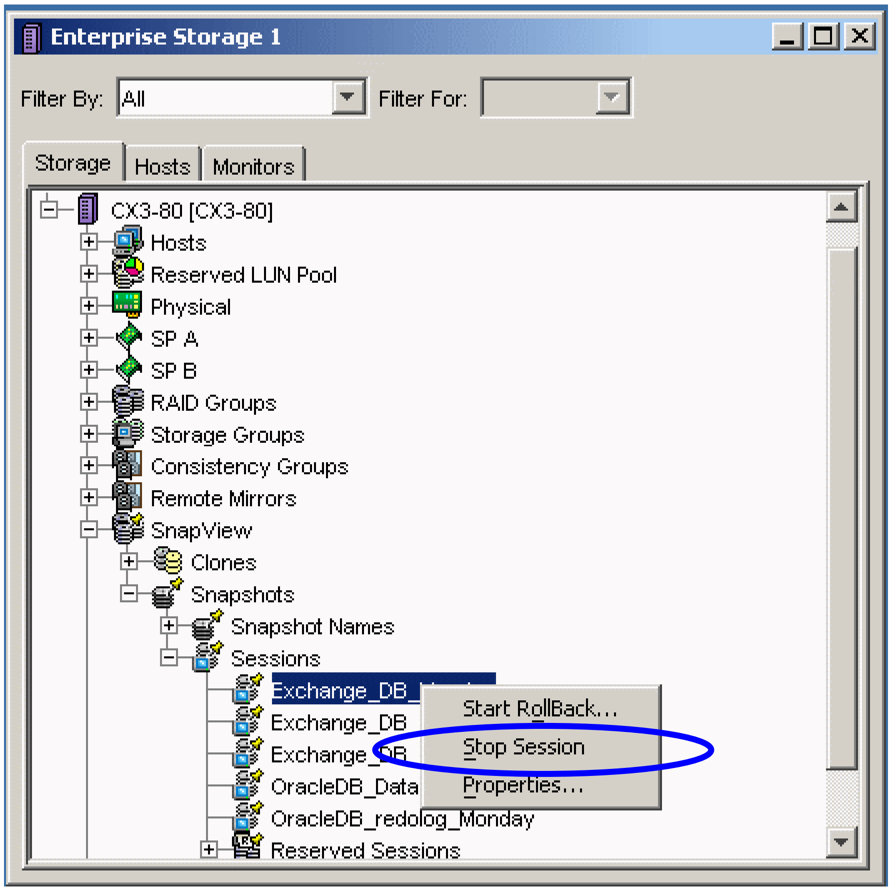


Figure 13. Stopping sessions

The appropriate Naviseccli command to perform the same function is shown here:

```
naviseccli -h <SP address> snapview -storagegroup -stopsession
Exchange_DB_Monday
```

SnapView rollback

In addition to providing point-in-time views of data for concurrent access, these point-in-time sessions also provide a way to recover data in the event of a data corruption on the source LUN. SnapView rollback allows you to restore the contents of a source LUN almost instantaneously to the point in time that any of the SnapView sessions for that source LUN were started. As shown in Figure 14, the rollback operation reverses the pointer-and-copy process by copying the original data on the reserved LUN back to the source LUN.

In this illustration, it is Tuesday at 1 P.M., and the user has encountered a problem on the production LUN. The user corrects the problem by going back to the last known good copy of the source data, which in this case is the data that was on the source at 9 A.M. when it was captured by the snap session that was started at that time.

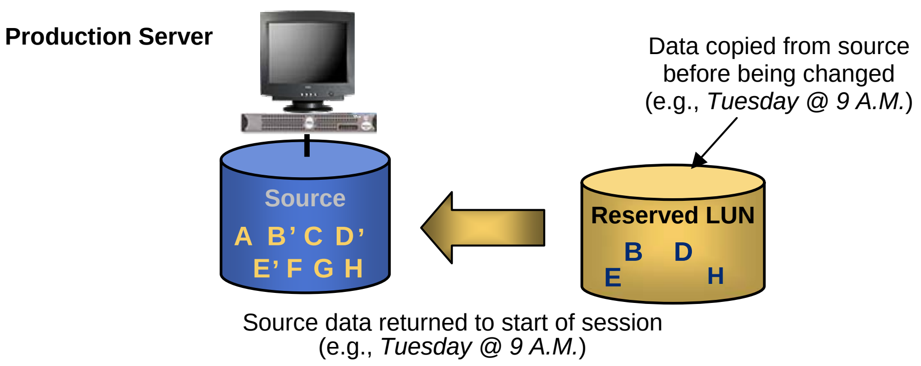

Production Server
Data copied from source before being changed (e.g., *Tuesday @ 9 A.M.*)
Source
A B' C D'
E' F G H
Reserved LUN
B D
E H
Source data returned to start of session (e.g., *Tuesday @ 9 A.M.*)

Figure 14. SnapView rollback operation

Rollback operates independently of any of the other sessions and/or snapshots that may be defined on a source LUN. Therefore, it is completely nondestructive to the source LUN's other sessions or associated snapshots. In other words, if a user has started multiple sessions and has snapshots activated on some or all of these sessions, the data relative to each session is not affected by the rollback operation.

As shown in Figure 15, a rollback operation is started using the Tuesday session, while the other sessions continue to operate without interruption.

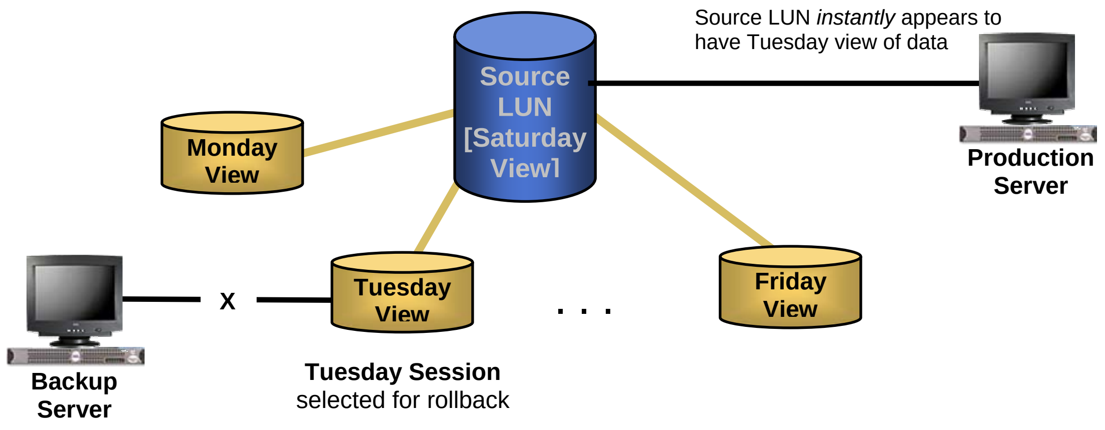

Source LUN *instantly* appears to have Tuesday view of data
Source LUN [Saturday View]
Monday View
Production Server
Tuesday View
. . .
Friday View
X
Backup Server
Tuesday Session selected for rollback

Figure 15. Choosing among sessions for SnapView rollback operation

Any session can be used to perform a rollback operation. However, the rollback operation for a given LUN must complete before another rollback operation for that same LUN is started. In addition, in order to preserve consistency on the source LUN, a source LUN can be involved in only one rollback operation at a time.

Host access during rollback

During a rollback operation, the source LUN is available for host reads and writes while the copy operation takes place in the background. At the start of the rollback operation, the user needs to flush the host buffers on the source and on the activated snapshot used for rollback. It is also necessary to take the source device offline and then bring it back online after the rollback session has started. Additionally, if a snapshot device is to remain activated to the session that is used for rollback, that snapshot must be taken offline until the rollback completes to prevent data inconsistency due to writes to the secondary server. Taking the snapshot **offline** makes it inaccessible for host I/O, which ensures consistent, unchanging data for the duration of the background copy. Once the rollback completes, the user can bring the snapshot back online, thereby making it accessible to a secondary host.

During the rollback process, the source LUN appears to have been instantly restored. This is accomplished by directing reads and writes to the source volume in such a way that I/O requests are satisfied effectively.

Reads to areas that have been identified for copying back to the source are redirected to the reserved LUN, and read directly from that LUN. Writes to areas that have been identified for copying back to the source generate a process by which the corresponding chunk(s) are immediately copied from the reserved LUN to the source, after which the write request is satisfied directly from the source LUN. In this way, new data written from the host supersedes the data being rolled back.

During the rollback operation, all sessions maintain their data and continue to perform copy-on-first-write operations as usual. Also, any activated snapshots remain accessible for host I/O (except, as noted previously, the snapshot activated on the session used for rollback).

Rollback operations

As with other SnapView functions, rollback operations are managed via Navisphere Manager or Navisphere CLI.

Starting rollback

To start a rollback operation using the UI, right-click the session and select **Start Rollback**, as shown in Figure 16.

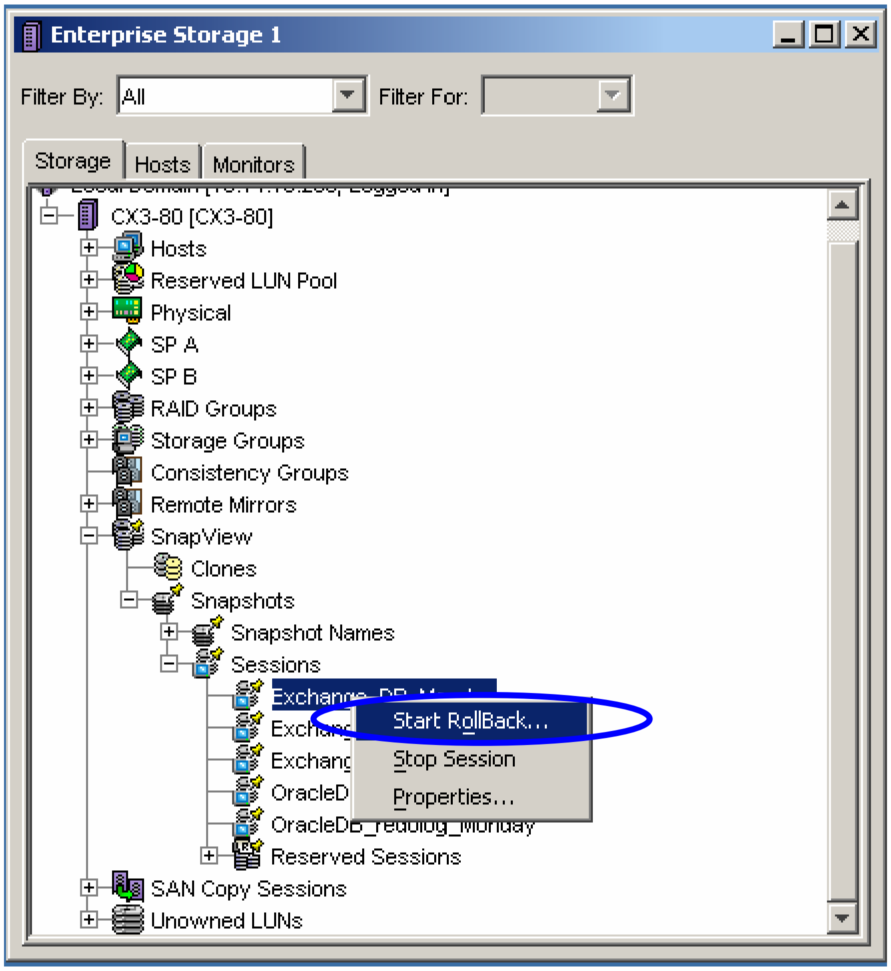


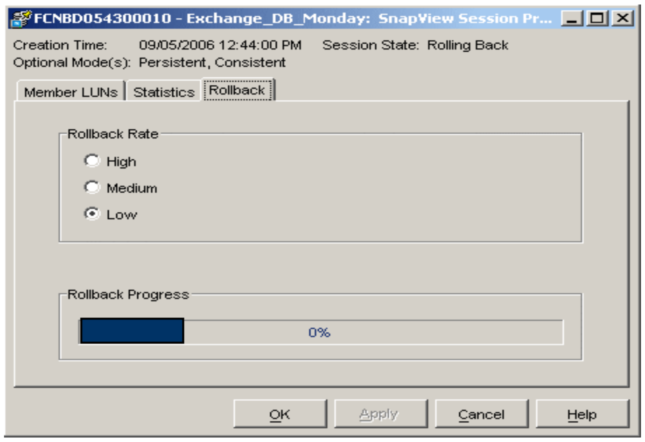
Figure 16. Starting a SnapView rollback operation from the Navisphere UI

Alternatively, the Navisphere CLI rollback command can be used:

```
naviseccli -h <SPName> SnapView -startrollback SessionName
```

When starting a rollback operation, users select a desired rollback rate: **High**, **Medium**, or **Low**[9]. The default rollback rate is **Medium**. **High** completes operations faster but may affect storage system performance for host I/0 requests, **Low** completes updates slower but minimizes the impact on other storage operations.

Once the rollback operation is started, users can view the rollback copy progress and adjust the copy speed. For instance, if the rollback is proceeding too slowly, the rollback rate can be changed from **Medium** to **High**, as shown in Figure 17.



Creation Time: 09/05/2006 12:44:00 PM Session State: Rolling Back
Optional Mode(s): Persistent, Consistent

Rollback Rate: High [✓ Low] Medium

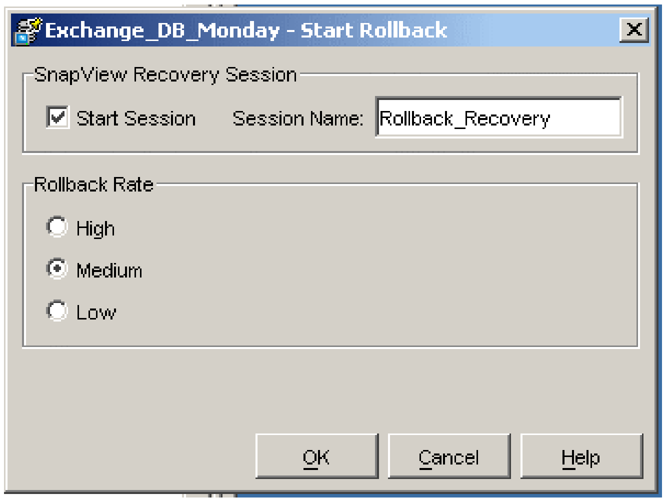
Rollback Progress: 0%

Figure 17. Managing SnapView rollback from the Navisphere UI

Rollback recovery session

To provide an easy way to *undo* a rollback operation, users are prompted to create a *recovery session* when the rollback operation begins. This session is simply one of the eight allowable sessions per source LUN. Users who opt to create this session essentially create a new session at that given point in time.

For example, on Saturday at 9 A.M. a user plans to initiate a rollback operation from a session started earlier that week. Before starting, the user creates a recovery session, as shown in Figure 18.



SnapView Recovery Session

[✓ Start Session] Session Name: Rollback_Recovery

Rollback Rate: High [✓ Medium] Low

Figure 18. SnapView rollback recovery session

[9] For additional details on rollback background copy rates, refer to “Rollback and performance.”

This gives the user the option to discard the changes made by the rollback session. The user proceeds with the rollback and, upon completion, realizes this is not the desired dataset. The user can then return to the Saturday A.M. view of the data by initiating a subsequent rollback operation using the recovery session just created, or by choosing another session for the rollback operation.

Rollback with modified snapshots

As discussed earlier, SnapView snapshots are both readable and writeable. Snapshot writes are saved on the same reserved LUN(s) as the copy-on-first-write data. With SnapView rollback, you can retrieve those writes outside of the snapshot. As long as the snapshot existed, the writes would be available. But once the snapshot was expired, the writes were discarded from the reserved LUN. With SnapView rollback, however, these writes can now be rolled back to the source volume.

If users perform a rollback operation from the session with an activated snapshot, those writes made to the snapshot are copied back to the source volume. In this way, unless a snapshot is deactivated before starting the rollback operation, snapshot writes are copied back to the source. As shown in Figure 19, this would result in the writes to the snapshot being copied to the source LUN. In Figure 19, chunks shown in red with a quotation mark (**”)** represent the writes made to the snapshot.

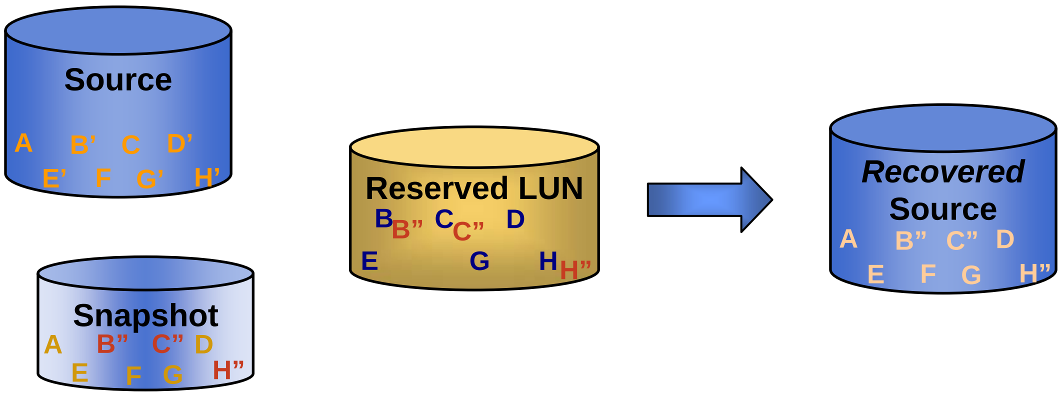


Figure 19. Rollback with modified snapshot

If, on the other hand, users prefer to have a view of the source as it appeared when the session was first started, they can simply deactivate the snapshot *before* performing the rollback operation. The writes to the snapshot will be discarded (those in red, with the **”** designation), leaving just the original copy-on-first-write data (shown in Figure 20).

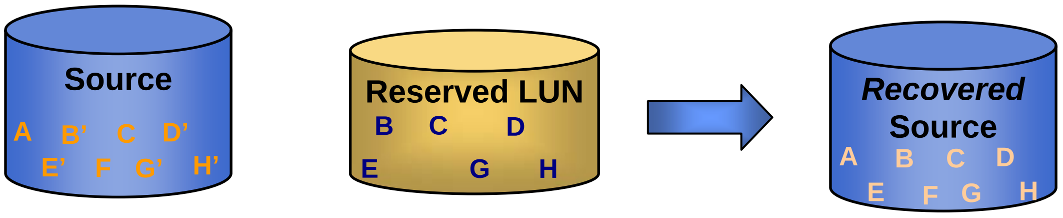


Figure 20. Rollback with deactivated snapshot

Consequently, the rollback operation copies only the original data back to the source. As a result, the source is returned to the same point-in-time view as when the session was first started.

Rollback recommendations and requirements

The following recommendations are provided to help ensure successful rollback operations.

Proactive data verification

In anticipation of being able to use a given session for rollback recovery, it is advisable that users proactively verify the SnapView session data by activating a snapshot on the session and mounting the snapshot to a secondary host to run through consistency checks. This way, users can be sure that they have usable, recoverable instances of the data, in case they have to perform a rollback operation from one of these sessions.

Rollback failure and recovery

Once begun, a rollback operation runs to completion; rollback operations cannot be aborted. If the session being rolled back runs out of reserved LUN space (for instance, due to new copy-on-first-write operations), the rollback operation will complete before the session is stopped. Then, once the rollback operation is finished, the session that was rolling back is stopped. SnapView rollback has a checkpointing capability. That is, in the event of a trespass—manual or automatic—the background copy restarts at the last checkpoint. Likewise, in the event of an SP reboot, rollback keeps track of the background copy process so that it can be restarted once the SP comes back online.

Rollback impact on the reserved LUN pool

As mentioned, the rollback operation could increase copy-on-first-write activity for other sessions since the purpose of rollback is to change the source LUN. This is especially true if the **Recovery Session** is selected before performing the rollback. As the copy-on-first-write activity increases, the reserved LUN(s) may fill up more quickly.[10] In light of this increased activity, users may want to proactively allocate additional reserved LUNs when performing rollback operations.

Rollback operations using consistent (multi-LUN) sessions

SnapView consistency is significant during a data restore when you roll back a session. SnapView consistent operations allow a consistent restore where all LUNs in the session are included in the rollback operation. In other words, if a session contains two LUNs, then both LUNs participate in the rollback operation, as shown in Figure 21.

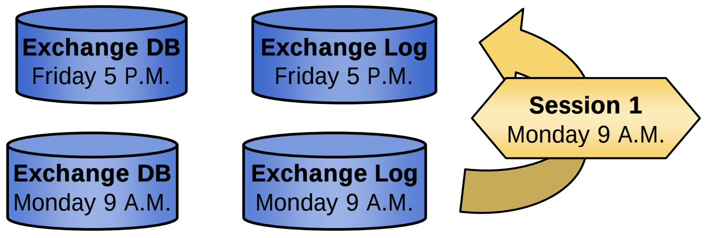


Figure 21. Rollback operation with multi-LUN sessions

Typically, this would be the desired effect, since it's presumed that a multi-LUN session was created for the purpose of having a set point-in-time view that applies to all LUNs in the session. However, if this is not the intended effect, users should create single-LUN sessions that can be used to roll back the contents of each LUN individually.

The following best practices must be followed to ensure that you maintain a consistent point-in-time copy of your data:

When starting a consistent session, it is important to remember that if you activate snapshots to the session and make any data changes, the data changes will change the original point-in-time data of the snap

[10] As noted earlier, users should keep in mind that as the reserved LUN(s) allocated to the source LUN begin to fill up, Event Monitor alerts the user at 50 percent, 75 percent, and at every 5 percent thereafter.

session. Should a rollback be necessary, you may need to deactivate the snapshot, thus bringing the session data back to the point in time when the session was started. Note that if you do this, you may also remove the verification and recovery writes, which means that they will have to be done again on the source LUN. Upon determining whether the snapshot writes should be copied to the source LUN, you can issue the **rollback** command to return each of the associated source LUNs to the point in time at which the session was started.[11] *Furthermore, to maintain a consistent point-in-time view of the session data among all member LUNs, do not stop a consistent session on an individual member.* While this is allowed, in most cases it produces undesired effects since removing a member renders the set incomplete and no longer consistent.

Rollback operations with LUN partitions

Users who have employed volume management tools to partition LUNs must likewise remember that since rollback operations are applied to the entire LUN, such an operation changes the contents of all LUN partitions on a given LUN. In other words, if there are two partitions on a single LUN—for instance, partitions F and G—there is not a way for users to roll back only one part of the LUN (that is, just partition F).

Similarly, if a logical volume spans multiple LUNs, it is critical to also roll back the LUN to the same point in time to maintain consistency for all related partitions. The most reliable way to ensure this is to make certain that the LUN containing the other partition(s) is included in the same session. For instance, if drive R has two mount points, contained on LUNs 4 and 5—you must start a multi-LUN session encompassing both LUNs 4 and 5, so that a rollback operation will roll back both partitions to the same point in time.

SnapView interoperability with other CLARiiON replication software

SnapView snapshots can be used with SnapView clones, MirrorView™, and SAN Copy™. The operations and recommendations when using SnapView snapshots with this replication software are outlined below.

Local recovery: Using SnapView snapshots with clones

Since SnapView snapshots can be used in conjunction with SnapView clones; users can replicate a source production LUN using SnapView clones to create a full recoverable copy of their production data. The clone copy can then be used as a source for starting a SnapView session. Multiple copies of a clone volume can be taken using SnapView snapshots to avoid the copy-on-first-write activity on the source production volume. A SnapView session must be started on the clone when it is in a synchronized or consistent state.

To preserve other point-in-time clone copies during SnapView rollback operations on a clone source, users must fracture all clone(s) before starting the rollback operation on the source. This requirement protects the clone because the rollback operation changes are not propagated to the clone or secondary mirror image until the user has had a chance to verify that the changes are correct. Once the user has confirmed that the result of the rollback operation was as expected, the clone can be reestablished. The data copied to the source during the rollback is then copied to the clone using the synchronization operation.

Remote recovery: Using SnapView snapshots with MirrorView

Since SnapView snapshots can be used with both primary and secondary MirrorView images, SnapView rollback provides an extra level of data protection for MirrorView. Should the primary image experience data corruption from the host, the data corruption is mirrored to the secondary image(s) as well[12], thus corrupting all mirror copies of the data. If SnapView sessions are running on the primary and/or secondary

[11] If the user chooses to copy the chanted data instead of restoring the source to the original point-in-time that the replica was created; these warnings do not apply.
[12] An exception is if the secondary image is fractured; in this case, mirroring is stopped so the corruption is not mirrored to the secondary image.

images, however, you can restore the image by performing a rollback operation. Given the importance of maintaining a robust disaster recovery environment, EMC strongly recommends that users perform consistency checks as a matter of course to ensure that the snap session data is reliable.

When performing a rollback operation on a MirrorView primary image that has one or more secondary images, users must first fracture the image(s) before starting the rollback operation.

SnapView snapshots with SAN Copy

A snapshot activated by a SnapView session of the source LUN can be used as a source for a full SAN Copy session. That way, the source LUN does not need to be offline during the copy operation. For incremental SAN Copy sessions, a snapshot of the destination SAN Copy LUN can be taken and mounted to a host for testing and verification of the data.

SnapView snapshots uses

SnapView has many uses; the most frequently used ones include application testing, backup, and recovery.

Application testing

SnapView makes it very easy to create multiple data replicas instantly, allowing you to perform application testing by running various tests on a standard set of data. These data replicas can then be mounted on separate hosts and made accessible to users as needed. If the results of the testing need to be propagated to the source, SnapView rollback provides a very easy mechanism for doing so. As noted, users must correctly flush host buffers when performing rollback operations to modify source data.

Application backup and recovery

The instant point-in-time *views* offered by SnapView snapshots allow users to run backup procedures on these snapshots while the production is online. SnapView snapshots, when used in conjunction with SnapView clones, reduce the performance impact that accompanies the snapshot-based backup on the production LUN. The clones are full point-in-time copies of the production LUN (which provides hardware and software protection), and backups can be generated from the snapshot copies taken of the clone copies.

The SnapView snapshot rollback feature can also be used to provide a fast and easy way to switch among various versions of applications and/or application data. For instance, if you wish to run a large query that might potentially corrupt the databases, you may want to start a SnapView session before performing the upgrade, and then run the query. That way, if the source LUN is corrupted, SnapView rollback provides a very easy way to return to the original point-in-time version of the application. By applying the rollback using the session that was started just before running the query, you are able to return to the previous good point-in-time copy— almost immediately.

Performance considerations with SnapView snapshots

Various factors should be considered when appraising performance with CLARiiON replication software on the storage system. Some of these factors include the following: I/O load, I/O profile, LUN placement, additional load on the SP, additional LUNs, and additional I/O imposed by SnapView.

Impact due to concurrent access and copy on first write

Because of the pointer-and-copy-based design of SnapView, users need to keep in mind that they may incur spindle contention when reading from snapshots. For instance, if users are presenting snapshots to backup hosts for the purpose of running backup operations, it is quite possible that the reads from these backup operations could impact the production read performance. Also, keep in mind that the reads from the reserved LUN will be random even though the backup stream is sequential.

Another potential impact on performance occurs when the production host is generating high write activity to the source LUN during a running SnapView session. In this case, more load is placed on the SP and the RAID group of the reserved LUN(s) because of additional reads from the source LUN and writes to the reserved LUN. The additional impact may increase the write response times for the production host.

Since SnapView divides the source LUN into 64 KB chunks this means that for a given server application write of 4 KB to the source, the corresponding 64 KB chunk will be copied into the reserved LUN—assuming this is the first update to this particular chunk of data after the session started. Hence, subsequent writes to the source LUN that fall within chunks that have already been written to the reserved LUN will not incur any additional copy on first write. Over time, therefore, the copy-on-first-write activity declines. In the same way, over time, as more data is accumulated in the reserved LUN, access to the snapshot affects the production LUN less as reads and writes to the snapshot will go direct to the saved data in the reserved LUN(s). If the copy-on-first-write activity results in copying the data that will be read for backup, it would be beneficial to hold off the backup operations until the bulk of the copy-on-first-write activity has been completed so that the reads for the backup operation will be directed to different spindles than the production volume.

Figure 22 illustrates the effect of an initial increase of transaction response times and the gradual decline of copy-on-first-write activity over time.

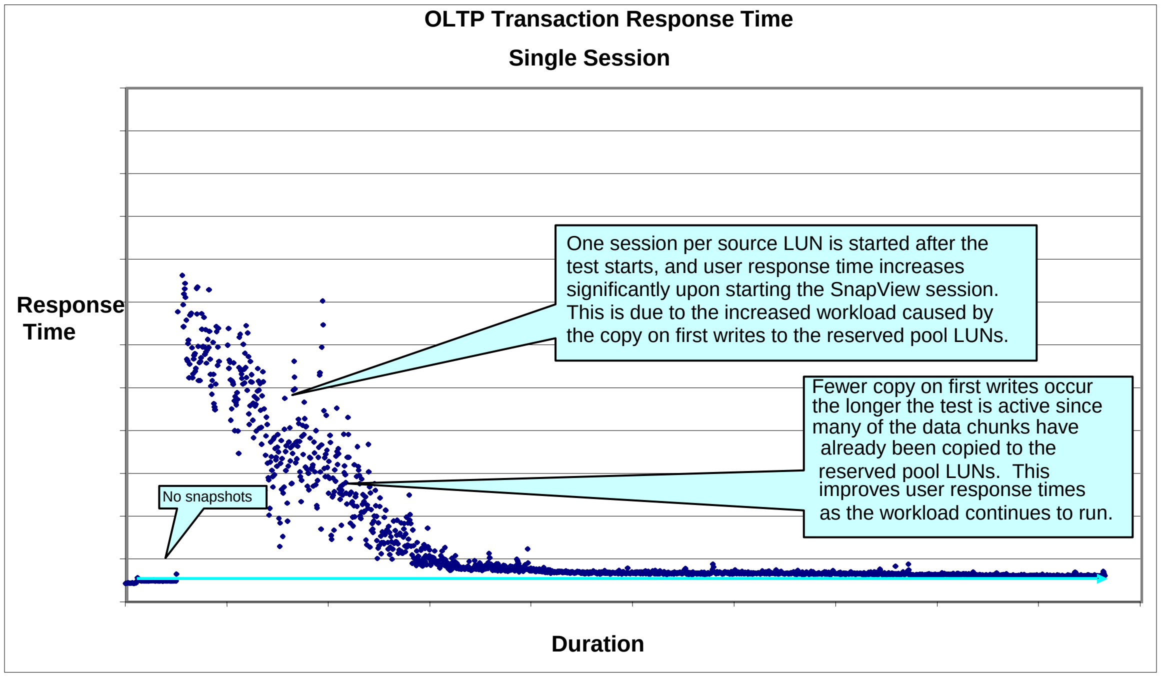


Figure 22. Response time during single SnapView session

As shown in this graph, the response time for the production application increases significantly after the start of a SnapView session due to the increased workload caused by the copy-on-first-write activity. However, after a period of time, the response time of the application returns to a baseline due to the fact that the majority of copy-on-first-write operations have completed; that is, additional writes to the production volume are hitting those areas for which the copy of original data to the reserved LUN has already occurred. This means that additional writes will not experience the response time latency associated due to copy on first write.

When multiple sessions are running, the effect is similar, as shown in Figure 23.

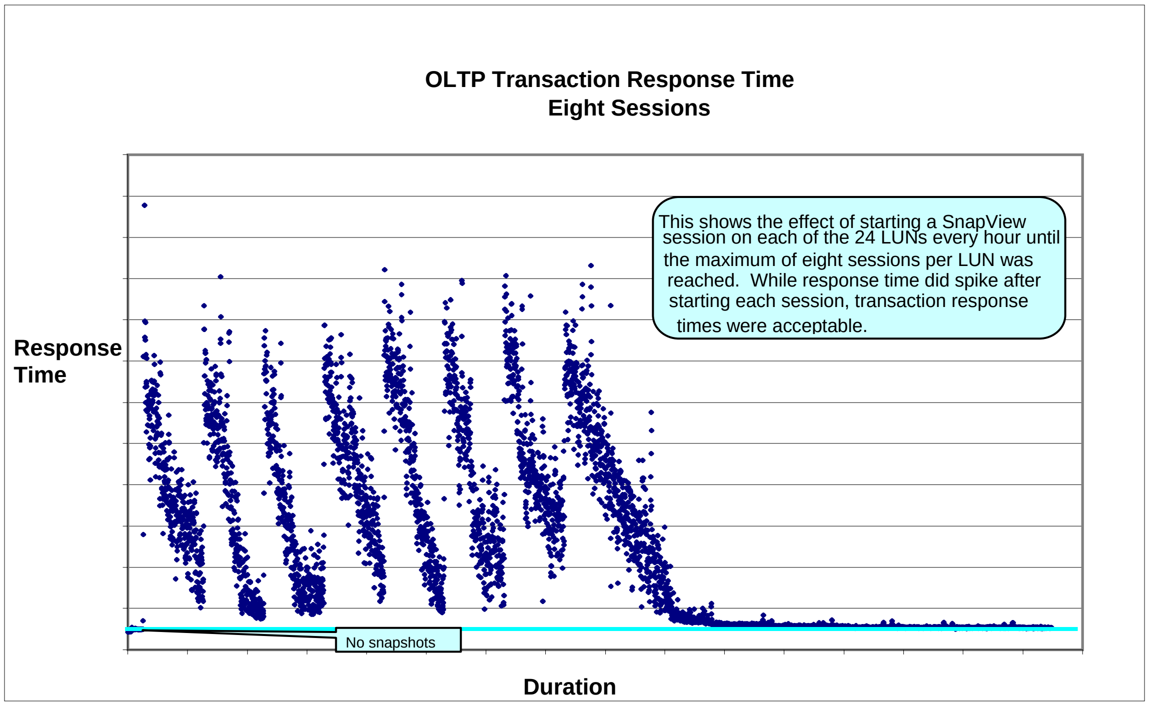


Figure 23. Response time during multiple SnapView sessions

As seen in Figure 23, the response time during the copy-on-first-write operations again increases due to copy-on-first-write activity. The figure also shows that the response time will ultimately return to a baseline—although with multiple sessions running, this return period is somewhat extended. Because of this affect, users typically plan to delay backup operations until the bulk of copy-on-first-write activity completes, if such a delay does not adversely impact backup schedules.

Characterizing I/O from the host enables you to determine characterization of I/O with SnapView sessions. Random writes incur more copy-on-first-write operations with SnapView. Rewrites have reduced impact, as explained later. It's worth noting that over time, the longer a SnapView session is running, the fewer copy-on-first-write operations that take place. Sequential I/O writes will likely incur less copy-on-first-write activity since the writes may fall within the 64K chunk size used by SnapView.

Considerations with MirrorView/Synchronous

As discussed previously, SnapView can be used with MirrorView to provide an extra level of recovery on the remote image. The snap session can be used for file-level recovery by being activated to a snapshot mounted on a secondary host; additionally, the snap session can be used to provide LUN-level restore via rollback. However, users should consider the effect of the response time due to the copy-on-first-write operation, since writes to mirrored LUNs are not acknowledged to the production host until the write to the secondary image is complete. As a result, the secondary write may have to wait for the copy-on-first-write activity to complete before an acknowledgement is sent to the primary storage system; this will increase the response time to the production host.

Rollback and performance

Except for the period of time during which a rollback operation is actively occurring, the SnapView rollback driver is in passive mode, allowing all I/O to simply pass through. When a rollback operation starts, the rollback driver assumes the dual role of managing the background copy processes, as well as managing the I/O requests being made to the source LUN. (The "SnapView rollback" section provides more information.)

Rollback background copy rate

Because the rollback operation impacts I/O to the source LUN, the background copy process provides the user the option to select the rate of speed at which the data is copied. As noted earlier, **High** minimizes the number of other operations allowed between copy operations, and **Low** maximizes that number. Figure 24 illustrates the various throughput rates during each rollback rate.

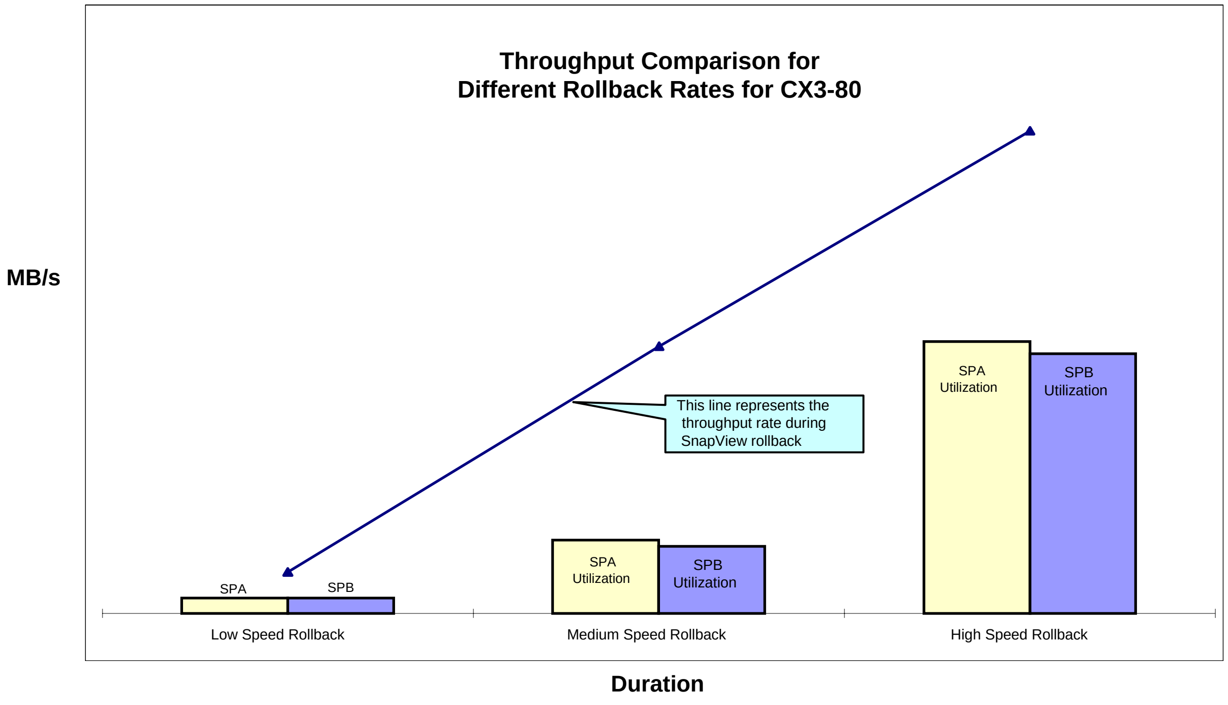


Figure 24. Rollback throughput rates

As noted earlier, the user can modify the rollback rate at any point during the rollback operation. Copy completion progress is also provided to help to determine how far along a copy is and whether the rate needs to be increased or decreased accordingly.

Effect of rollback on copy-on-first-write activity

Since the rollback operation is changing the contents of the source LUN, it will likely generate copy-on-first-write operations for other sessions. This is especially true if a recovery session has been created before starting the rollback operation. Typically, sessions experience the greatest copy-on-first-write activity in the first several minutes of creation. Over time, as the bulk of the data that is going to change during an active session has done so, the copy-on-first-write activity decreases—and even stops. If a recovery session is started before performing a rollback operation, users should anticipate that same high rate of copy-on-first-write activity at the beginning of the recovery session—which, as with other sessions, occurs in the first several minutes of the rollback operation.

Once the rollback operation has finished and the copy-on-first-write activity associated with the recovery session (and perhaps other sessions) subsides; the performance impact associated with SnapView rollback likewise subsides.

Conclusion

SnapView snapshots provide flexible options to enable instantaneous parallel data access facilities. This instant capability reduces the time production environments would normally have to be offline for purposes of backups or decision support type operations.

In addition to providing for concurrent access to data, SnapView also provides the ability to instantly restore source data in the event of a data corruption. Any snap session can be used to perform the rollback operation.

You can manage SnapView manually using the Navisphere UI method, or scripted using Naviseccli. The admsnap utility provides additional command line facilities—specifically for host interaction with snapshots. The command line interface provided by Naviseccli and admsnap facilitates automated operations such as recurring backup jobs. SnapView leverages the power of the CLARiiON CX3 series, adding functionality without compromise.

References

The following titles can be found on Powerlink except where noted:

- *Navisphere Manager Help*
- *EMC CLARiiON SnapView Clones – A Detailed Review* white paper
- *EMC CLARiiON Reserved LUN Pool Configuration Considerations* white paper
- E-Lab Navigator (also at EMC.com)

Appendix

The following sections provide readers with the feature set differences between SnapView Express and Manager, a short reference to the number of SnapView resources allowed per storage system model, and admsnap commands. Each section instructs readers where to obtain more in-depth information as needed.

Differences between SnapView Express and SnapView Manager

SnapView Express comes packaged with all AX4-5 systems, while SnapView Manager is an optional software package available only on AX4-5 running Navisphere Manager. The AX4-5 is the only system that supports SnapView Express and SnapView Manager. Table 1 outlines the differences between SnapView Express and SnapView Manager.

Table 1. Comparison between SnapView Express and SnapView Manager

Feature Set	AX4 (SnapView Express)	AX4 (SnapView Manager)
Supports SnapView clones	No	Yes
Assigns snapshot and source LUN to the same server	No	Yes
Has rollback support	No	Yes
Has the option to select placement of reserved LUNs	No	Yes

Limits

For most up-to-date SnapView limits information on different CLARiiON storage systems, refer to the most recent version of the *CLARiiON Open Systems Configuration Guide*. For ease of reference, Table 2, Table 3, and Table 4 provide SnapView limits as defined at the time of this printing.

Table 2. SnapView limits for the CX3 series platforms

Platform	CX3-80	CX3-40	CX3-20
SnapView snapshots			
Source LUNs and/or maximum number of reserved LUNs			
Per storage system	512	256	128
Sessions			
Per source LUN	8		
Source LUNs in consistent session	16	16	8
Snapshots			
Per storage system	2048	1024	512
Per source LUN	8		

Table 3. SnapView limits for the CX series platforms

Platform	CX700	CX500	CX300
SnapView snapshots			
Source LUNs and/or maximum number of Reserved LUNs			
Per storage system	100	50	25
Sessions			
Per source LUN		8	
Source LUNs in consistent session	16	16	8
Snapshots			
Per storage system	300	150	100
Per source LUN		8	

Table 4. SnapView limits for the AX4 platform

Platform	AX4 (Express)	AX4 (Navisphere Manager)
SnapView snapshots		
Source LUNs		
Per storage system	16	64
Sessions		
Per array	16	256
Source LUNs in consistent session	N/A	8
Max Reserved LUNs		
Per array	64	64
Source LUNs in consistent session	8	
Snapshots		
Per source LUN	1	8
Source LUNs in consistent session	N/A	8

Admsnap

The command line utility admsnap allows hosts connected to a CLARiiON SAN to interact with the SnapView software components on the CLARiiON storage system. It can be used to manage the SnapView sessions from both the production and secondary hosts outside of the Navisphere Manager interface. The admsnap utility acts as a client of the SnapView product and functions by issuing SCSI commands to the storage system.

The admsnap utility provides the necessary tools for systems connected to the SAN to use the SnapView functionality. The utility is beneficial because it provides customers with the ability to activate and deactivate sessions from the secondary host. In many cases, the only requirement for the production host is to start and stop sessions.

For admsnap to interact properly with the storage system, two software components must be installed on the host:

- Navisphere Agent — The Agent pushes host information to the storage system. This information is necessary so that admsnap can translate host device names to CLARiiON LUN numbers.
- admsnap — The admsnap software must be installed on the production and backup servers in order to establish the communications between the two systems. This is a host requirement. Once admsnap has been installed, the host can issue all of the available commands to the storage system.

The five main functions that the admsnap program can perform for snapshots on the storage system are:

- Start
- Stop
- Activate
- Deactivate
- Flush

Start

The `start` command is run from the production host. It starts a SnapView session on the device that is specified on the command line or from within the script. Before issuing a `start` command, the information contained within the source LUN should be quiesced. Do this by issuing the **`admsnap flush`** command, or using the **`sync`** command on UNIX hosts.

The syntax of the `start` command is:

```
admsnap start -s <session-name> -o <device-name> -c
```

In lieu of `<device-name>`, users may substitute drive letters or file-system names. The `-c` is optional. If added, the session will run in consistent mode across a set of source LUNs.

Stop

The admsnap `stop` command stops an active session, making any active snapshot unavailable to secondary hosts. Since it does not unmount or unmap the snapshot from the host, users or applications attempting to read or write to this area will receive an error message. As such, stopping a session with an active snapshot should be avoided if possible. The preferred method is to first unmount the snapshot from the secondary host, deactivate the snapshot from the session, and then issue the `stop` command. Stopping the session frees all disk and memory resources consumed by that session. If this is the last session stopped for the source LUN, the reserved LUN is returned to the owning SP's reserved LUN pool and can then be used by the storage system for other sessions. It is still necessary to unmount the snapshot from UNIX-based hosts and NT servers.

The syntax of the `stop` command is:

```
admsnap stop -s <session-name> -o <device_name>
```

In lieu of `<device-name>`, users may substitute drive letters or file-system names.

Activate

The `activate` command is issued from the secondary host to access the snapshot. For the snapshot to be activated properly, the following requirements must be satisfied:

- A session has been started for the LUN.
- Snapshot has been defined for the LUN and is assigned to the storage group for this host.

The `activate` command issues the appropriate SCSI commands to scan the bus for all active CLARiiON storage system SnapView sessions accessible by defined snapshots in the storage group connected to this host. The scan returns a list of all sessions that have available snapshots assigned to the storage group. When a session name is provided during the `activate` command, admsnap attempts to match it to the list of names returned and, if successful, activates the session and brings the snapshot online.

The syntax of the command is:

```
admsnap activate -s <session-name> [-o device_name]
```

The `-o` is optional; if you omit it, the software searches all devices for the appropriate device that matches the session name which takes additional time.

Deactivate

The `deactivate` command is used to unmount a snapshot. For Windows platforms, the `deactivate` command flushes all data from host buffers and unmounts the drive. For UNIX systems, however, the `umount` command would be used instead to flush all data and unmounts the drive.

Deactivating a snapshot has no effect on the session: the host has flushed its memory buffers to disk, and the partition has been unmounted. All original data within that session continues to be written to the reserved LUN until the session is explicitly stopped. As such, a snapshot may be activated on the session again after the deactivation completes. This can be useful in development environments when, after some period of testing, a user would like to refresh the data to the point in time when the session started.

The syntax of the command is:

```
admsnap deactivate -s <session-name> [-o device_name]
```

Flush

The `flush` command is used exclusively by Windows systems by forcing the data contained within the file-system buffers out to disk. To ensure that all cached data has been written to disk, use the flush command on the production host before starting a snapshot session. In addition, to ensure that all data has been written from the host buffers out to the snapshot, use the `flush` command on the second host before issuing a `deactivate` command. For Solaris, the `flush` command flushes all data and clears all buffers. For all other UNIX systems, the command is not supported; use the UNIX `sync` command instead.

The syntax of the `flush` command is:

```
admsnap flush -o <device-name>
```

A session name is unnecessary with `flush`**.**

For additional information on admsnap, refer to the *EMC SnapView admsnap and Command Line Interface (CLI) Administrator's Guide*.

SnapView persistent sessions

Starting with release 24 of FLARE, all SnapView sessions are persistent by default. For non-persistent sessions, the SnapView memory map that maintains the point-in-time session data by tracking the copy-on-first-write operations resides primarily in SP memory. When sessions are defined to be persistent, the memory map entries are copied to the nonvolatile reserved LUN, along with the copy-on-first-write data. This way, if the SP became unavailable, the session data—and the pointers defining it—remains available, even if the owning SP is not.

SnapView chunk map and persistence

The saved chunk map plays a very important role in SnapView persistence. It is the structure by which all of the vital information about copy-on-first-write data to the reserved LUNs is stored.

With each new copy-on-first-write operation performed for this snapshot session, the chunk map entry is updated and copied to the reserved LUN[13]. After both the copy-on-first-write data and the chunk map entry are safely copied on the reserved LUN, the source LUN is updated and the successful write is subsequently acknowledged to the host. In this way, if something fails at any point in the copy-on-first-write operation, the host will simply retransmit.

[13] SnapView chunks are defined at 64 KB; the SnapView chunk maps are sized at less than .1 percent of the source LUN.

In the event of an SP reboot, or any failure that requires moving a session to the peer SP, the stored chunk map can be retrieved, so that the mapping of which chunks have been written to the reserved LUN can be re-created.

Persistence through reboot and/or trespass

Enabling persistence for a session allows the session to remain active during SP reboots. For instance, if a non-disruptive upgrade (NDU) occurs, each SP is rebooted in a serial fashion. Also, some SP failures require that the system administrator reboot the SP. The snapshot session remains active through either of these scenarios since the source LUN trespass is followed by the associated reserved LUN trespass to the peer SP. Once the rebooted SP comes back online and the peer SP reboots, the source and associated reserved LUN trespass occurs again. The rebooted SP reads the persistent map on the reserved LUN and rebuilds the memory map. With this information, the SP then remaps the pointers to the original data and to the copy-on-first-write changed data housed in the reserved LUN, thus making active snapshot sessions available.

Likewise, SnapView sessions can also maintain persistence in the event that a peer SP must trespass the source LUNs (for example, SP failure, HBA failure, switch failure, path failure, etc.). When a source LUN is trespassed by the peer SP, the associated reserved LUN(s)—which must always be owned by the same SP as the source LUN—will, along with the source LUN, change ownership to the peer SP. This is referred to as a *gang trespass*, in which case the peer SP takes temporary ownership of the source LUNs and the reserved LUN(s), which contain the copy-on-first-write data and the memory map. As with the reboot scenario described earlier, the peer SP reads the data in the reserved LUN(s) and rebuilds the corresponding map entries. Users should keep in mind that, as with all trespass scenarios, the peer trespass may incur performance penalties on the CLARiiON storage system if one SP becomes overloaded.