EMC²
where information lives®

White Paper

EMC CLARiiON SnapView Clones

A Detailed Review

Abstract

This white paper discusses SnapView™ clones—fully populated point-in-time copies of LUNs (logical units) that allow partial synchronization between the source and destination LUNs. Common uses for SnapView clones, as well as configuration and management information, are provided. General performance details are also provided.

January 2008

Part Number H2100.2

Table of Contents

Executive summary

SnapView™ clones are fully populated point-in-time copies of LUNs (logical units) that allow incremental synchronization between the source and destination LUNs. When combined with SnapView snapshots, which provide users point-in-time *views* of data, clones provide *fully populated,* point-in-time copies, maximizing users' flexibility in using their storage environment. The point-in-time copy allows users to perform additional storage management tasks with minimal impact to the production data. These tasks include:

- Backup/recovery
- Application testing
- Warehousing
- Data movement

All of these tasks can use the point-in-time copy of the data to minimize impact on the production server.

Introduction

SnapView clones provide users the ability to create fully populated binary copies of LUNs within a single storage system. Once populated, clones can be fractured from the source and presented to a secondary server to provide point-in-time replicas of data. Clones share some similarities with the other SnapView feature: snapshots.

Audience

This white paper is intended for anyone who needs to explain or implement SnapView clones. It is intended to provide a general understanding of clones, and also some relevant performance information. Readers who intend to implement and manage clones on a regular basis will need to consult the product documentation for more complete coverage of operations and procedures with clones.

Comparison with SnapView snapshots

Clones and snapshots are each point-in-time views of a source LUN. Like snapshots, clones are both readable and writeable when fractured. Additionally, data from a clone or a snap session can be restored back to the source (for instance, to recover the source data in the event of a data corruption). Additionally, like snapshots, each clone source can have up to eight clones. The essential difference between clones and snapshots is that clones are binary copies of their sources—with fully populated data in the LUNs—rather than the pointer-based model of snapshots, with copy-on-first-write data stored in the reserved LUN. This difference yields two significant benefits to using clones: additional data protection and performance. Users should keep in mind that these differences also relate to differences in capacity requirements and the availability of clones relative to snapshots.

Restores to the source LUN

In addition to providing point-in-time views of data for concurrent access, clone and snapshot replicas provide users a way to recover data in the event of data corruption on the source LUN. The data from the SnapView replica can be used to restore the contents of a source LUN—virtually instantaneously—to the point in time associated with that replica.

For best results in a restore operation, users should proactively verify the state of the data in the replicas as soon as they are created to ensure that the replica used for restoring the source LUN will result in the intended view of the data. Such validation includes running consistency checks and other application-specific measures to ensure data integrity. The SnapView restore operation for clones and snapshots is much faster than standard data recovery processes (disk-based or tape-based) because the restore is

incremental—and it occurs in the background—so the source LUN will appear to be instantly restored. (Details of this operation are discussed in the section "Clone reverse synchronization.")

Protection from some hardware failures

If a source LUN experiences a double point of failure that is hardware-related rather than software-related, clones offer another level of protection. Since the clone is defined on separate drives from the source LUN, the clone continues to be available even if the source LUN is not. In the unlikely event that a source LUN experiences a multiple-drive failure, a bus failure, or some other storage-system failure affecting some but not all hardware components, the clone would remain accessible. In this event, users may opt to put the clone into production if the source volume is deemed unrecoverable (or needed more quickly than it can be recovered). This is essentially a failover to another device within the same storage system. Should this be necessary, users should keep in mind that the clone will have a point-in-time reference that is likely different from the source; so any source data that has not yet been updated to the clone would be lost.

Performance

Because the clone is *a full copy* of the data rather than *pointers* to the data, once it is fractured it can service parallel I/O—for instance, running backups—without incurring any performance impact on the source LUN. This is in contrast to the performance impact when using snapshots—which includes spindle contention due to the pointer model (where reads from the secondary server are directed to the source LUN), as well as copy-on-first-write activity (where original blocks on the source LUN are copied to the reserved LUN).[1] When using clones, users should ensure that the clones are placed on drives that belong to a different RAID group than the source LUN, so that I/O requests targeted at the clone do not access the production LUN disks. Fractured clones can have I/O serviced by the peer SP from the source LUN, which provides additional load-balancing flexibility over what is available with snapshots.

Capacity requirements and availability

Users should keep in mind that because clones are exact replicas of their source LUNs, they will always take up the same space as the source LUN. This means that clones will generally require more space than SnapView reserved LUNs, since the reserved LUNs store only the copy-on-first-write data[2]. Additionally, this means that upon first use, the clone will not be available until the initial synchronization of data from the source to the clone has completed. After that point, the clone can be incrementally resynchronized. Regarding synchronizations, users should keep in mind that the performance benefit with clones is only realized when the clones are fractured; allow time for the initial synchronization of the clones, as well as ongoing incremental resynchronizations of the clone. These operations will have some performance impact on the source LUN, so they should be conducted at non-peak times. The section "Performance considerations with clones" provides a more complete discussion of performance considerations with sync operations.

Table 1 summarizes the similarities and differences between SnapView clones and SnapView snapshots.

Table 1. Snapshot and clone features

Feature	Snapshots	Clones
Is read-/writeable	Yes	Yes
Provides instant restore	Yes	Yes
Is available as soon as it is created	Yes	No, data must be synchronized first

[1] *CLARiiON SnapView Snapshots and Snap Sessions Knowledgebook* and SnapView product documentation provide more information on SnapView snapshots.

[2] The exception to this is when every chunk on the source LUN is written to and must therefore be copied into the snapshot cache. Thus, the entire LUN is copied and that—in addition to the corresponding metadata describing it—results in the contents of the snap reserved LUN being larger than the source LUN itself.

Can be used to restore a corrupted source LUN	Yes	Yes
Can be used if hardware failure occurs on source LUN (for example, multidrive or bus fault)	No	Yes, assuming hardware failure does not affect clone components (SP, bus, drives, and so on)
Occupies less space than the source LUN	Yes, only a fraction (depends on the rate of change of the source LUN)	No
Access does not affect source LUN performance after replica is updated	No, impact includes spindle contention for dual reads, and also copy on first writes	Yes, when fractured

Comparison with MirrorView/Synchronous

Users familiar with MirrorView™ can think of clones as mirrors *within* storage systems, as opposed to *across* storage systems. The essential distinction between clones and mirrors lies in the different purposes for which they were designed. MirrorView was designed primarily for *disaster* recovery and the normal state is for the primary and mirror to be actively mirroring (in sync), whereas clones would more likely be used for *corruption* recovery.

Following on these intended uses, mirrors are typically in sync because their purpose is to protect against data *loss*. In contrast, clones are typically fractured, making them point-in-time copies because their purpose is to protect against data *corruption*. In other words, the fact that the clone has contents from an earlier point in time is the basis of its protection: The user can restore a corrupted source LUN to the earlier point in time of the clone. Accordingly, the recovery procedure varies between clones and mirrors. In a disaster scenario in which the production LUN becomes unavailable, production can be failed over to the mirrored LUN. Generally, clones are protecting against a data corruption, and therefore can be used to incrementally restore the source LUN to an earlier point-in-time copy of the data (presumably before the corruption occurred). As discussed in the previous section, however, there are also possible scenarios where a double hardware failure occurs in the storage system that affects the source LUN; in this scenario, the user would be able to redirect production to the clone—much like the failover to the mirrored LUN when using MirrorView.

Users should keep in mind the essential difference between a synchronous copy and a point-in-time copy: a point-in-time copy will not have the most up-to-date data. Therefore, recovery with a clone may result in some data loss since it will restore the production data to an earlier point in time. In those cases, it is presumed that some loss of transactions might be acceptable in order to recover a larger dataset. This tradeoff should be considered when selecting the synchronization policies—that is, how often the clone is resynchronized—and, according to best practices, validated to ensure the state of the data on the clone.

One other difference between clones and mirrors is that with MirrorView, the mirrored LUN is never accessible for server I/O (reads *or* writes), except via a SnapView snapshot or if the mirrored LUN is promoted to be used for production. Clones, on the other hand, are accessible to servers when fractured, and are therefore readable and writeable. When writes have been made to a clone, the user can either elect to copy over those writes with data from the source LUN, or to copy those writes back to the source LUN, depending on the synchronization option selected.

As of FLARE® release 24, clones can be used in conjunction with mirrors, to provide a full copy of the data on the source LUN or additional readable/writeable copies of the mirror data at the remote site. This new capability allows users to take advantage of both data protection types for a particular source LUN. Now, LUNs can be protected against data corruption with a rapid recovery capability (via clones) *and* be mirrored for disaster recovery purposes.

Like MirrorView, the source and the clone must be of the same physical size, although they can be different RAID types and/or different disk types. The section “Performance considerations with clones” provides a more complete discussion of performance considerations for different RAID types and different disk types.

Table 2 summarizes the similarities and differences between SnapView clones and MirrorView/Synchronous.

Table 2. Comparison between SnapView clones and MirrorView/Synchronous

Feature	Clones	Mirror
Location of copy	Same storage system	Secondary storage system
Concurrent access to copy	Yes, when fractured	No, only when mirrored LUN is promoted for use as production
Requires same amount of space (per copy) as source LUN	Yes	Yes
Available if hardware failure on source LUN	Yes, assuming hardware failure does not affect clone components (SP, bus, drives, and so on)	Yes, even if complete primary storage system becomes unavailable

Business operations supported by SnapView

SnapView enables users to perform various tasks on a given dataset as required in a typical business environment—without having to compromise data access. Although the actual *implementation* of clones varies significantly from snapshots, the *uses* for clones are very similar to those of snapshots: to allow parallel processing without impacting the performance of the production application. This allows users to perform supplementary tasks that can be run concurrent to production operations, without compromising the performance of the production processing. These tasks may include disk-based:

- Backup and recovery operations
- Decision support and data testing
- Data warehousing
- Data reporting
- Data propagation
- User training

Backup and recovery

The point-in-time *copy* offered by clones, in addition to the point-in-time *view* offered by snapshots, allows users to choose the feature that best suits their backup environment. Because a snapshot is created instantly, it can be used in conjunction with a backup procedure as soon as it is created. However, a snapshot-based backup typically has an associated performance impact on the production LUN because the backup procedure usually results in a read on the production LUN. A clone, on the other hand, requires some additional time to initially create the full data copy (subsequent synchronizations will be faster because they will be incremental). Once the copy is made and fractured from the production LUN, the backup procedure will operate completely independent of the production LUN, and thus eliminate the performance impact that accompanies the snapshot-based backup.

For recovering the source LUN, users can perform a restore using either SnapView snapshots or clones. Some situations will require recovering from backups, however. For example, if a single file is lost, restoring an entire file system to a previous point in time to recover a single file may not be the choice that meets the customer recovery objectives. The other option is to present the snapshot and clone replica to a server other than the server that accesses the source LUN(s), and transfer the individual files to be restored over the network. Alternatively, EMC Replication Manager can be used to allow simultaneous access of a source LUN and its associated clone or snapshot from a single server, and copy the individual files to be restored to the source LUN.

Decision support and application testing

Clones, like snapshots, can be used with decision support and database testing, where up to eight duplicates of the source LUN can be created, used, modified, and even destroyed without affecting the data on the source LUN. The performance benefit of clones is particularly relevant for decision support and application testing environments, due to the heavy I/O activity.

One specific type of decision-support testing that especially benefits from clones is the ability to perform a software upgrade via a clone, and subsequently test for compatibility and performance. By installing the upgrade on the clone, users can easily revert to the source if any compatibility or performance issues are discovered with the upgrade. And, similarly, if the upgrade is found to be satisfactory, users can remove the clone from the clone group, after ensuring that the server has flushed all cached data to the clone, and use it as the source or merely as an independent LUN with no associated clone group.

Data warehousing

Clones can also be used with data warehousing applications, as well as other third-party applications. By copying data to a clone and fracturing it from the source, the data warehouse application can be populated by the clone and thus avoid any impact on the performance of the production application. Likewise, for other third-party applications, the ability to import the data from a clone rather than the source provides another example of parallel processing facilitated by clones.

Data reporting

For many applications, running reports requires significant system resources and, as such, must be carefully scheduled to minimize the impact on the production system on which they are reporting. Running the reports from a clone, however, is another example of parallel processing—enabling users to run their reports at more strategic and/or convenient times for their reporting needs, rather than having to choose such times according to low-impact windows of opportunity.

Data propagation

Because clones provide users an easy way to develop multiple copies of data, one of the many things that you can do with clone is data propagation. For instance, users may want multiple copies of boot disks, with the appropriate operating-system customizations for their environment. With clones, they can customize a LUN with all of the required information and settings for a server to operate as needed. The user can then make it the source for a clone group—thus serving as a *golden master* copy. Finally, the user can create clones of this boot disk, for use with other servers. Additionally, with the ability to remove clones from the clone group, and subsequently make more clones, it becomes very easy to quickly propagate many copies from the golden master for use throughout the operating environment.

User training

Clones can also be used to help users familiarize themselves with production applications and data without impacting the performance of the production data, or affecting the integrity of the data. The benefits of concurrent access to data without the performance impact or danger of affecting production data apply to this scenario, as well.

Using clones with other CLARiiON replication software

Clones can be used with SnapView snapshots and with SAN Copy™. As of release 24, clones may also be used with MirrorView/S and MirrorView/A (with some restrictions as outlined in the "Using MirrorView with clones" section).

Using SnapView snapshots with clones

Not only can users create clones and snapshots of the same source LUN, but they can also create snapshots of the clones themselves. This provides users a way of having multilevel copies of their data. This can be especially helpful if users want various iterations of datasets in various states of testing. Instead of the clone, these SnapView sessions could be brought online.

If data corruption (or other adverse effects of testing) should occur, the SnapView session could be discarded, thus leaving the unaltered clone in its original, usable state. This would allow ongoing testing with the same base replica, thus minimizing effort in creating the test resources.

Users may also opt to implement multilevel copies in this way to increase the total number of copies of their production data. Figure 1 shows an implementation where the first level of copies is the weekly clone, and the second level of copies is the daily snap session created for each day of the week for each clone.

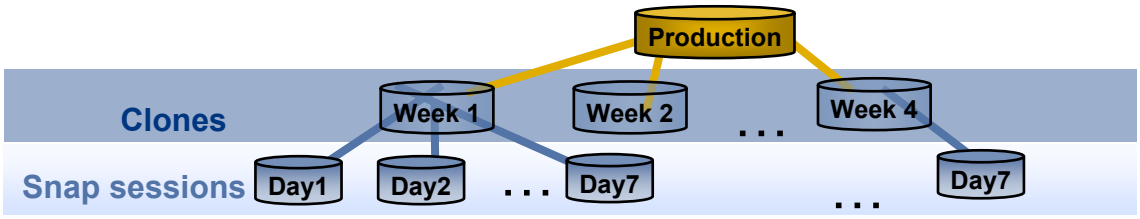

Figure 1. Using snapshots with clones

The snapshot process behaves the same whether it is snapping a clone or a "regular" LUN. In other words, the snapshot uses the same mapping mechanism for clones and LUNs.

Using SAN Copy with clones

In addition to having a copy of data within the same storage system, users often need to have a copy of data on another storage system, as well. Figure 2 illustrates the use of SAN Copy with clones. As shown in the figure, users can opt to use SAN Copy to copy the source LUN or to copy the clone.

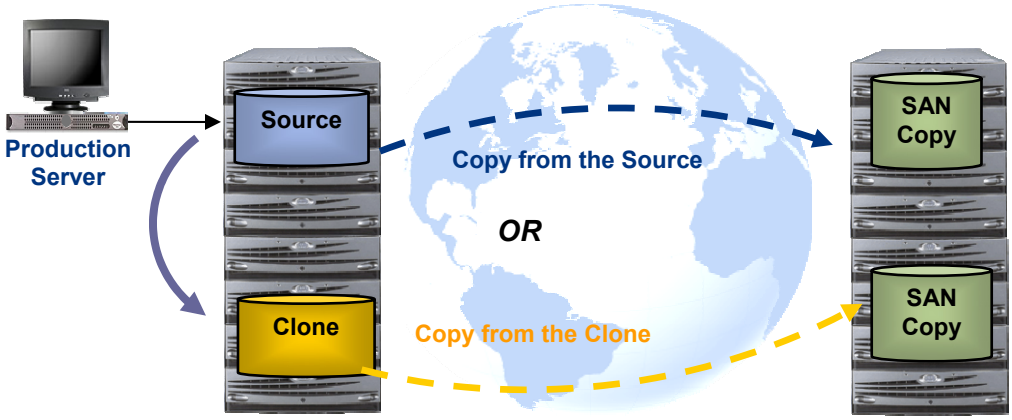

Figure 2. Using SAN Copy with clones

Performance-minded users will often elect to use SAN Copy to copy the clone because the copy can occur without causing any impact to production data.

Using MirrorView with clones

As stated previously, release 24 introduced the ability to clone a mirror. MirrorView/S, or MirrorView/A and SnapView, may share the same source LUN at either the source or remote sites. Users may:

1. Clone a MirrorView primary image (primary image serves as clone source on local storage system).
2. Clone a MirrorView secondary (remote) image (secondary image serves as clone source on remote storage system).
3. Clone a MirrorView primary image *and* clone the MirrorView secondary image.

You cannot mirror a clone, however, as illustrated in Figure 3, SAN Copy must be used to copy clones from one array to another, as shown in Figure 3.

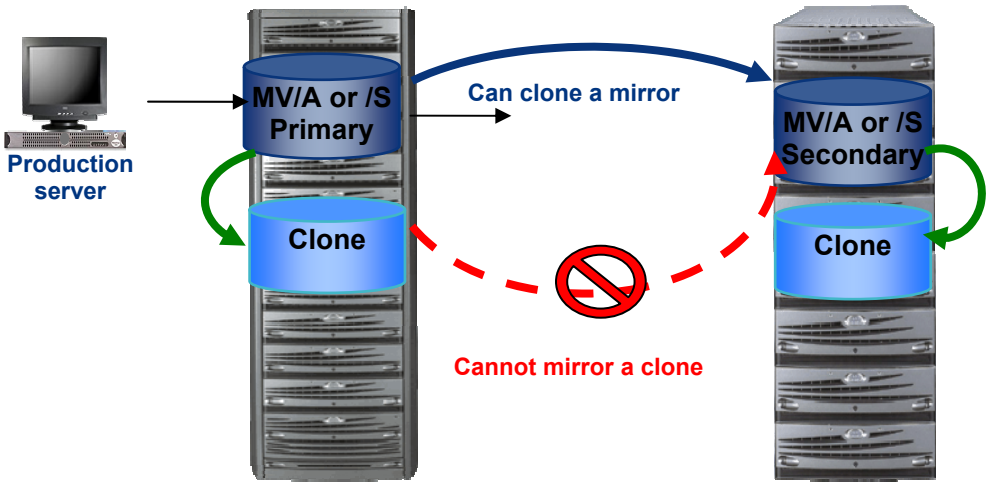

Figure 3. Using MirrorView with clones

At your source site, a clone can enhance an existing disaster recovery implementation by offering a fast local recovery option. A full copy on separate drives, when fractured, protects against both hardware and data corruption errors that affect the source. Implementing clones at the source site removes the need for recovering from the remote location if only a particular source LUN (rather than the entire system) is affected. Recovering locally is a much faster option.

Maintaining a clone of your remote mirror can offer additional protection against corrupted data with MirrorView. When using MirrorView/S all incoming writes on the source side are immediately synchronized with the DR site, meaning that any corruptions to that data are also copied. If a clone of the remote device is available and fractured, a fast recovery to a known good point in time is possible. Having a full copy at the remote site as well can protect further against any hardware failures affecting the remote MirrorView/S and MirrorView/A target.

Another benefit is that your remote site data can now be accessible for read/write activity without incurring the additional performance impact of snapshots. Implementing clones of mirrored LUNs allows you to perform operations at the remote site on copies of production data, while ensuring that a copy of your primary data is still available for disaster recovery. Certain restrictions are in place and enforced by the CLARiiON operating software. A few of these restrictions are noted here:

- Reverse synchronization is not allowed to a secondary mirror image.
- Reverse synchronization is allowed to the primary, but only if the mirror is administratively fractured.
- A secondary image may not be promoted if there is a reverse synchronization occurring on the primary.
- Reverse synchronization is not allowed if the primary is involved in a rollback.

With support for clones of mirrors a new synchronization state is introduced. A state of **Remote Mirror Synchronizing** indicates that the secondary image this clone is associated with is currently undergoing a synchronization or update. If MirrorView/A or MirrorView/S is in a state of **synchronizing**, then any clone associated with the remote mirror will show a state of **remote mirror synchronizing**. If a MirrorView/A pair is updating, then any clone of the secondary will also show the **remote mirror synchronizing** state.

More details for clones of mirrors, including full restrictions, best practices, use cases, and performance, are explained in the white paper *MirrorView Knowledgebook* on EMC Powerlink® (http://Powerlink.EMC.com).

Clone functionality

This section discusses the features of clones, such as fracture and resynchronization, and the process for configuring clone groups. As of release 24, the Configure SnapView Clones Wizard walks the user through the initial configuration process. The underlying principles of setting up a clone group and allocating clone private LUNs are maintained, but many of the steps involved in these tasks are now automated for the user during initial setup. After initial configuration, clones are manipulated through Navisphere® Manager or NaviCLI.

Basic clone features

Data can be copied from the source to the clone—a *synchronization*—or in the other direction, from the clone to the source—a *reverse synchronization*. After the clone has been initially synchronized, it is generally fractured and presented to a secondary server. Upon fracturing the clone, the fracture log keeps track of the regions (*extents*) that have been modified on either the clone or its source after the clone is fractured. The fracture log is a bitmap contained on disk — a region referred to as the clone private LUN — so this record is persistent even in the event of a power outage, ensuring protection and high availability. Subsequent synchronizations or reverse synchronizations are incremental in nature in that they copy only the extents that have changed on the source and/or the clone.

During both synchronization and reverse synchronization, server I/Os (read and write) can continue to the source. The clone, however, is not accessible for secondary server I/Os during either synchronizations or reverse synchronizations; the user must ensure that all server access to the clone is stopped (this includes ensuring that all cached data on the server is flushed to the clone) prior to initiating a synchronization or a reverse synchronization. Failure to remove access will likely cause the server to report write errors or in some cases panic (especially if the clone was used a server boot device).

Users can select the rate at which the data is synchronized to (or reverse-synchronized from) the clone. This way, users can prioritize between storage system CPU cycles directed to synchronization operations versus other I/O processing tasks. Clone synchronization performance information is provided in the "Performance considerations with clones" section.

Clone synchronization

Synchronization is the process of copying data from the source LUN to the clone. Upon creating the association of a clone with a particular source this translates to a full synchronization – all extents on the source LUN are copied to the clone to provide a completely redundant replica. Subsequent synchronizations involve only a copy of any data that has changed on the source since the previous synchronization – overwriting any writes that have occurred directly to the clone from any secondary server that had been accessing it while the clone was fractured. It is essentially an update for the clone. Once synchronized with the incremental updates from the source LUN, the clone is ready to be fractured again to maintain the relevant point-in-time reference.

If a fracture operation is performed while the clone state is synchronizing, a subsequent sync will require a complete copy from the source LUN if the initial sync was not completed. However, if the clone was initially synchronized completely, a subsequent sync will copy only the incremental changes.

A clone LUN can be assigned to the alternate SP from the source LUN; however, the clone LUN will be trespassed during the clone synchronization process, and returned to its SP when it is fractured. For example, if the source LUN is owned by SPA and the clone is owned by SPB, the clone will be trespassed to SPA during the sync, and returned to SPB when fractured.

Clone fracture

A clone must be fractured from the source LUN to achieve the point-in-time representation of the data that the user is trying to obtain, and to make the clone available for secondary server access. Figure 4 shows the menu options to fracture, synchronize, and reverse-synchronize the clone, as well as the options to remove

the clone from the clone group and view clone properties. All functions are accessed within the Navisphere tree by right-clicking the icon of the clone on which the user would like to perform the action.

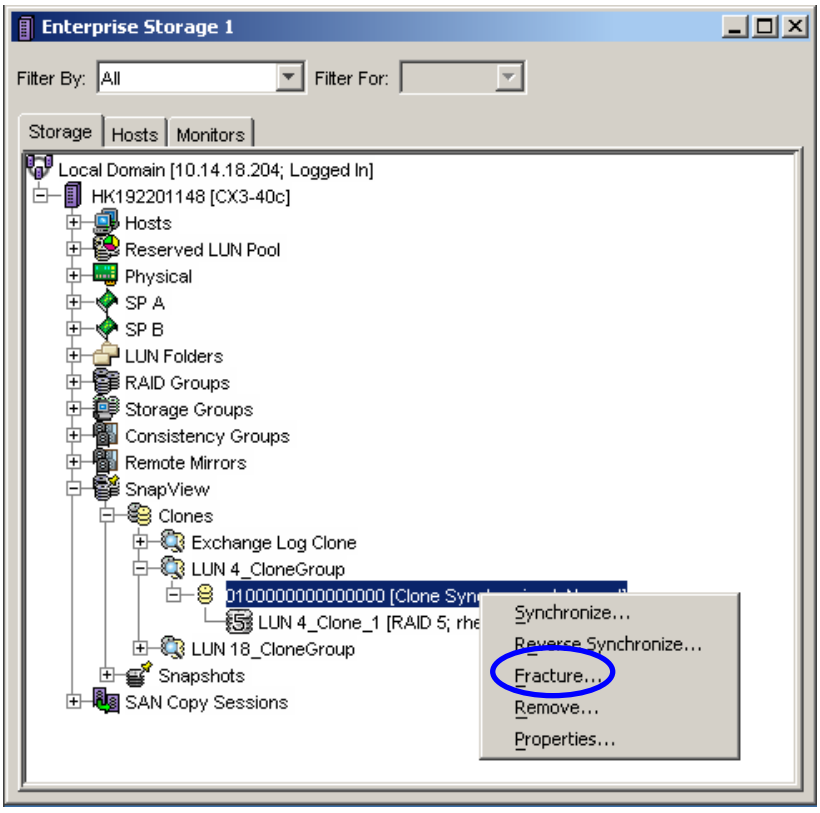

Figure 4. Fracturing a clone

Once fractured, clones can be presented to a secondary server and are available for read and write I/O. Usually, these writes are intended to be discarded—such as in a testing scenario. The next synchronization from the source to the clone discards the writes on the clone, and copies any data written to the source LUN while the clone is fractured. If the user wants to copy the writes made to the clone back to the source LUN, a reverse synchronization would be used.

Clone reverse synchronization

A reverse synchronization is the process of copying data from a clone to the source. A reverse synchronization is typically used to restore a source following a data corruption—assuming that the data corruption occurred after the clone was fractured. As noted previously, users should follow best practices to proactively validate the state of the data on the clone to ensure a successful restore. The user must ensure that all data cached on the server is flushed and that all buffers on the server are discarded prior to initiating a reverse synchronization. The *Navisphere Manager Help* provides details.

Before the reverse-synchronization process begins, the other clones in the clone group are automatically fractured so that users can verify that the restore process was successful, before propagating the restored data changes to the other clones with a subsequent synchronization. In addition, the clone that is used to perform the reverse synchronization operation must be offline. As shown in Figure 5, a restore operation is started using the Tuesday replica, while the other sessions continue to operate without interruption.

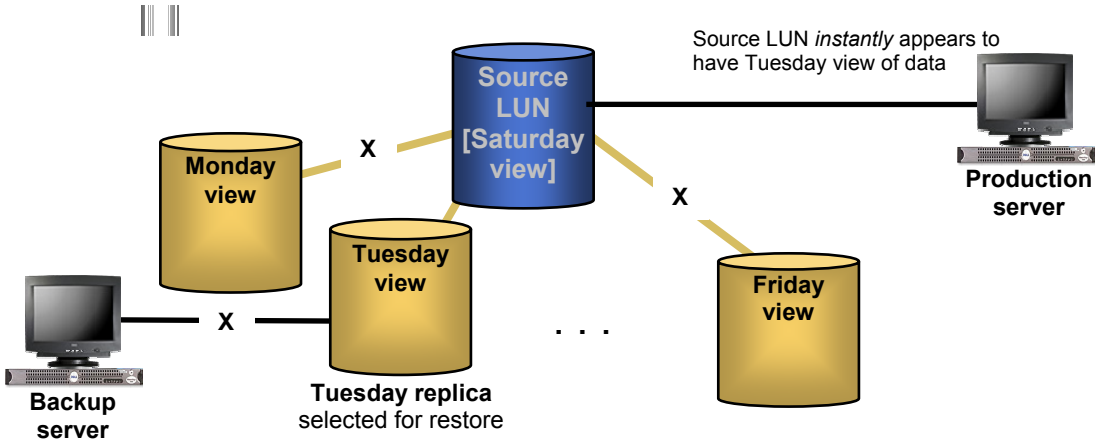

Figure 5. Choosing among replicas for restore operation

During a reverse synchronization, the copy mechanism makes the source seem identical to the clone as soon as the synchronization begins instantaneously. This is accomplished by mirroring the write requests to the source over to the clone and by redirecting the read requests (for extents not yet copied to the source LUN) directly to the clone. Figure 6 illustrates this process, where the changed data on the source LUN is copied over with the original data as preserved on the clone. During the reverse synchronization, incoming writes from the production server are written to the source LUN, and mirrored to the clone, so that upon completion of the reverse synchronization, the clone and the source are byte-for-byte copies.

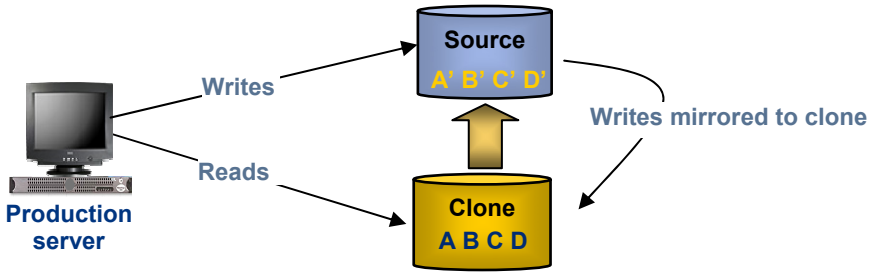

Figure 6. Clones reverse-synchronize

Following the completion of the reverse synchronization, the clone state transitions to "synchronized," and the clone is not fractured until the user explicitly does so (as opposed to the protected restore scenario described next). The other clones, fractured before the reverse synchronization began, remain fractured until the user chooses to resynchronize them.

Clone reverse synchronization, protected restore

An enhanced protection option during a reverse synchronization is to enable the protected restore option for the clone. The protected restore option ensures that when the reverse synchronization begins, the state of the clone is maintained. This means that incoming writes to the source LUN are *not* mirrored to the protected clone. Instead, when the clone is reverse-synchronizing to the source, server writes to and reads from the source are managed using a *copy-on-demand* mechanism (which can be thought of as a reverse copy on first write, for those familiar with SnapView snapshots). Figure 7 illustrates this process, where those areas to which the incoming I/Os are trying to read or write are immediately copied from the clone. Once the area has been copied to the source, the I/O from the production server is accepted.

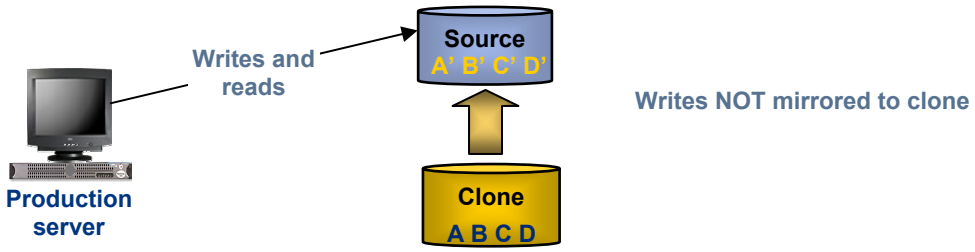

Figure 7. Clones reverse-synchronize with protected restore option enabled

In keeping with the goal of maintaining the point-in-time data on the clone, upon completion of the reverse synchronization with protected restore enabled, the clone will be fractured. Once the source LUN is validated to ensure that the restore occurred as desired and that there is no longer a need to maintain the point-in-time copy of the clone, the clone can be resynchronized, as needed. The clone that was used to reverse-synchronize the source using the protected restore operation can be used to reverse-synchronize back to the source again to allow multiple tries at recovery.

The protected restore feature must initially be enabled at the storage-system level (discussed next); after that, it can be selected for any reverse-sync operation.

Fracture Log Reset

With FLARE release 26, a full-sync capability was added to re-establish data identity between the source and clone if the source or clone LUN becomes corrupted. The **clone –resetfracturelog** command, available within Navisphere CLI, marks all data chunks of the source or clone LUNs as being modified. This allows a user to perform a full synchronization or reverse synchronization.

This switch can only be used if the clone was administratively fractured and the clone is in an out-of-sync or reverse-out-of-sync state. After this switch is executed, the user can perform a full-synchronization or reverse-synchronization operation. For more information about using this command, see the *EMC SnapView Command Line Interface (CLI) Reference* available on Powerlink.

Clone configuration

A source LUN and its associated clones are grouped in a *clone group*. Clone groups are created when the source is first identified, and as clones are added for the source, they become members of the clone group. Upon being added to a clone group, a clone will undergo an initial synchronization so that it contains the contents of the source LUN. The clone will continue to be updated with source LUN contents until it is fractured—and usually presented to a secondary server for the various concurrent activities described previously. The user can then periodically resynchronize the clones to update them with contents from the source LUN, after which the clones are fractured and presented again to the secondary server.

Beginning in release 24, configuring clone groups is accomplished via the Clone Configuration Wizard in the Navisphere Taskbar. This wizard is launched by selecting the Configure SnapView Clones option as shown in Figure 8.

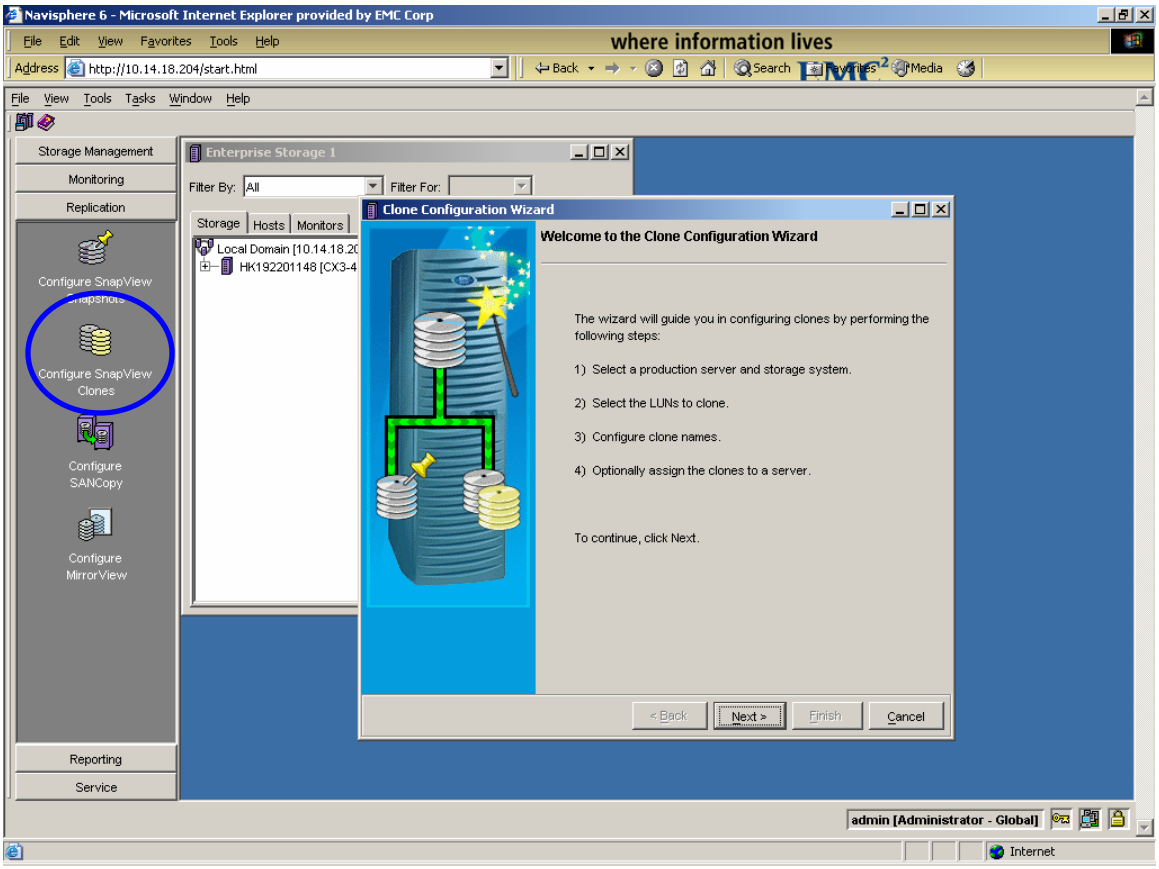

Figure 8. Navisphere Clone Configuration Wizard

The Navisphere Clone Configuration Wizard consolidates all the steps required for configuring clone groups from selection to synchronization. Traditionally, users had to first configure and allocate the clone private LUNs. Then, the user created clone groups. Lastly, clones were added to the individual clone groups. Each of these steps was a discrete task. Now, with the Clone Configuration Wizard, setup of clones consists of one simple-to-follow wizard that encompasses all of these tasks. The Clone Configuration Wizard will:

1. Bind and allocate clone private LUNs (if necessary).
2. Create a clone group with the identified source LUN.
3. Create and add a clone to the clone group
4. Assign the clone to a secondary server storage group, if desired.
5. Initiate synchronization.

The wizard also sets the synchronization rate to **Medium**, the Recovery Policy to **Automatic**, and **Protected Restore** is disabled. These settings are automatically set for the user, reducing the steps that the user has to perform. For a detailed description of these settings, see *Navisphere Manager Help* on Powerlink. For multiple operations, especially when used with other easy-to-use Replication and Storage Management wizards, this saves valuable time when setting up the storage system.

For users accustomed to the earlier methods of creating clone groups and clone private LUNs, or for those who desire greater control over the configuration, the standard Navisphere tree interface and CLI are still accessible to users. Figure 9 shows the traditional dialog box for adding a clone to a clone group. This approach allows the user more granular control of certain properties, such as clone sync rate, that the wizard masks. These controls are discussed in more detail in the following section.

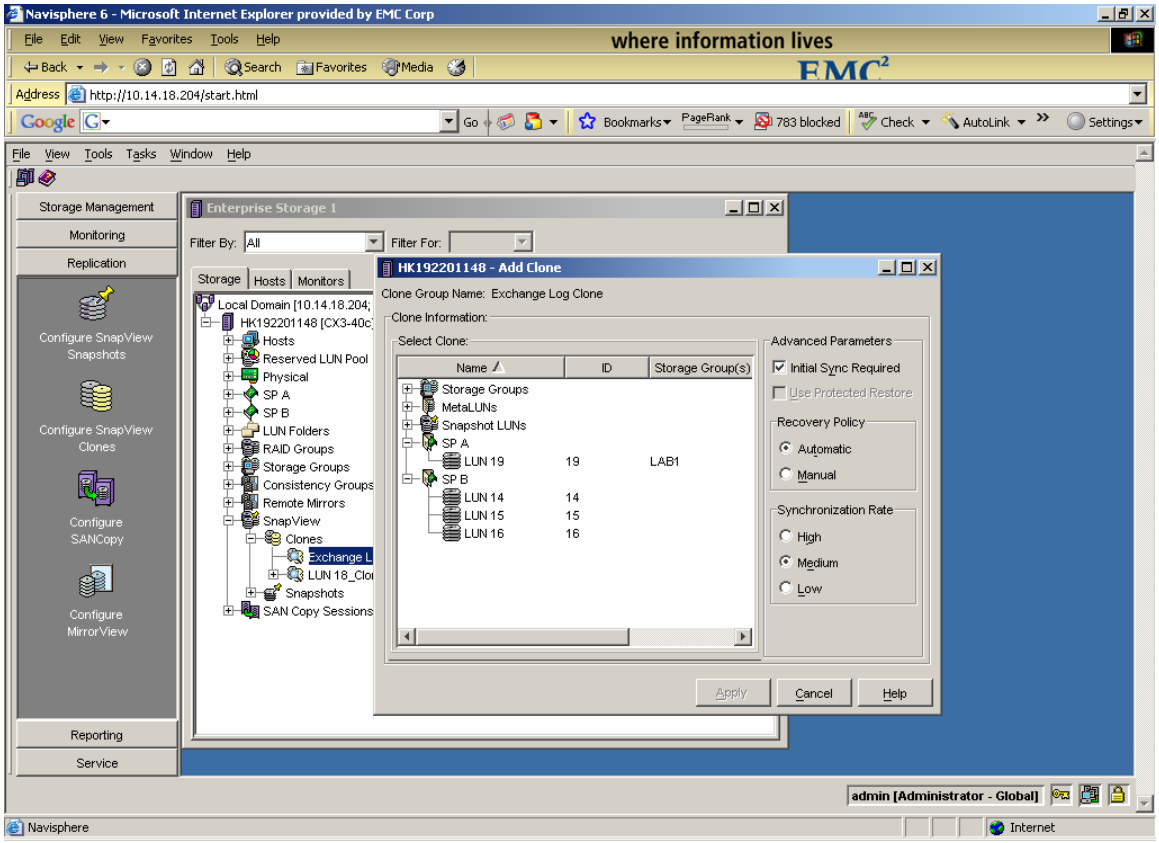

Figure 9. Adding clones to a clone group (Navisphere tree)

When users no longer want a point-in-time clone for a particular source LUN, they may remove the clone from the clone group. When a clone is removed from the clone group, it retains the data it had before being removed, but no longer contains the metadata associated with clones. As such, it functions as an independent LUN at that point.

As with SnapView snapshots, each source LUN may have as many as eight associated clones; thus, eight clones are the maximum allowed in each clone group. The number of clone groups and clones per storage system varies by storage system model and FLARE release. The "Appendix: Clone limits" section provides more information on clone limits.

Clone private LUN

The fracture logs for clones are stored on the clone private LUN. The Clone Configuration Wizard automatically binds and allocates these private LUNs if they have not previously been configured. Figure 10 shows the window for allocating the clone private LUNs for users not utilizing the wizard who choose to configure these LUNs manually.

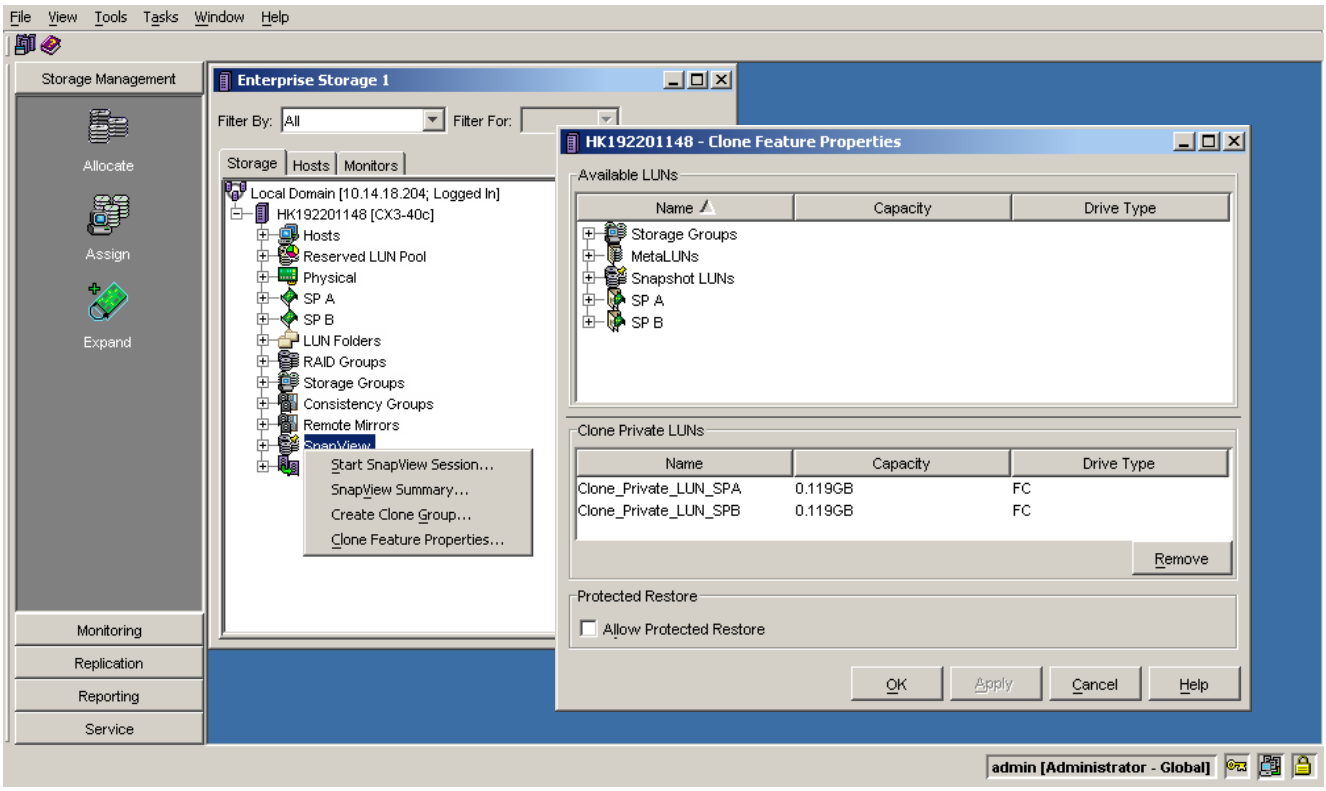

Figure 10. Allocating the clone private LUN

A clone private LUN must be allocated on each SP, and each must be 128 MB (256 MB total). LUNs larger than 128 MB may be used as the clone private LUN, but the space beyond 128 MB will not be used.

Also, users can enable the protected restore feature for SnapView at the storage-system level in this same window. Once enabled at the storage-system level, the feature can be enabled per reverse-sync operation, as needed.

CLARiiON consistency operations

FLARE release 19 added storage-system-based consistency to the SnapView features—including consistent fracture for clones. A consistent fracture enables users to establish a point-in-time set of replicas that maintain dependent write-ordered consistency, which in turn allows users to have a restartable point-in-time replica of multi-LUN datasets.

Dependent write-ordered consistency is typically most relevant in database environments where users have multiple LUNs with related data—for instance, database and log LUNs—that have write-order dependencies among them. To preserve logical data integrity among the various replica members, they must maintain write-order dependency relative to one another. For example, each entry in the database must have the corresponding log entries; otherwise, data corruptions and/or deletions could occur. Likewise, these considerations are applicable when using volume managers to create multi-LUN volumes and file systems.

With the addition of these consistent operations in FLARE release 19, users can now have consistent local replicas using SnapView. With EMC CLARiiON consistency technology, replicas can be owned by either SP. The source LUNs and their corresponding replicas must reside within a single CLARiiON storage system.

Clone consistent fracture operations

SnapView consistent fracture provides users storage-system-based consistent replicas. As with other SnapView operations, the consistent operations can be initiated from either the Navisphere Manager GUI or the Navisphere CLI. This way, users have the flexibility to manage the consistent operations manually (through Navisphere GUI or CLI), or to incorporate the consistent operations into a script using Navisphere CLI.

With SnapView consistent fracture, there is no defined "group" concept or association. The set of objects being operated on together exists at the moment the operation is executed (that is, various clones being fractured), after which no association exists. This allows flexibility from operation to operation, where different objects can be operated on as needed.

Storage-system-based consistency compared to server-based consistency

Users now have the option of employing either storage-system-based or server-based consistency when replicating data. Each method is a different but viable process of maintaining write-order consistency among replicas.

Storage-system-based consistency operates independent of the server application, meaning that it is not necessary to quiesce the application. Instead, the storage system pends any dependent writes during the process of establishing the consistent set of replicas. Because this process is independent of the server and application, the storage-system-based replicas will be in a state comparable to an immediate powerdown (or crash) of the server. As such, the standard database software recovery features—comparable to restarting a server following a poweroff and reboot scenario—will typically need to be run to bring the replicas into a usable state on the secondary server. Additionally, because SnapView operates at the storage-system-level rather than the server-application level, it is especially beneficial in environments that have write-order dependencies across multiple applications. Data warehouses are common examples of such federated environments.

There are scenarios, however, where users may want to momentarily quiesce the application before creating the point-in-time replicas. In these scenarios, users may opt for server-based consistency—for instance, as facilitated through Replication Manager. Replication Manager leverages the native quiesce features within a given application (for example, via VSS for Exchange 2003 or Hot Backup Mode for Oracle) to establish a consistent set of replicas. These replicas, therefore, are valid backup copies of the data, ready for immediate use by the secondary server. Additionally, because Replication Manager operates at the server and application level, it can coordinate replicas of source LUNs belonging to different CLARiiON storage systems.

Thus, replicas created with SnapView consistency operations do not require a server and/or application quiesce, but will likely require additional recovery procedures before being used on the secondary server. (provided that all involved LUNs reside on a single storage system). On the other hand, replicas created using Replication Manager (or other server-based consistency tools) require a short quiesce on the production application, but are more readily available for use by the secondary server.

EMC Replication Manager (RM) supports replicating the file system on which an application resides if RM does not integrate with the application API for the purposes of quiescing. In this case, the CLI capabilities of a given application to quiesce the application prior to the replica can be incorporated into the Replication Manager call-out scripts. Replication Manager also supports the consistent fracture operation for clones and can manage replicas across CLARiiON storage systems; whereas storage-system-based consistency requires that the sources and their replicas reside within one CLARiiON storage system.

Clones consistent fracture

Clones consistent fracture refers to fracturing a set of clones belonging to write-order dependent source LUNs. This means that the associated source LUN for each clone must be unique (the user cannot perform a consistent fracture on multiple clones belonging to the same source LUN). In the Navisphere Manager GUI, the consistent-fracture operation is performed by selecting the desired clones using the standard

multi-set keys (for example, CTRL), and selecting **Fracture** in the menu, as shown in Figure 11. In this scenario of selecting multiple clones, the fracture will be designated as consistent.

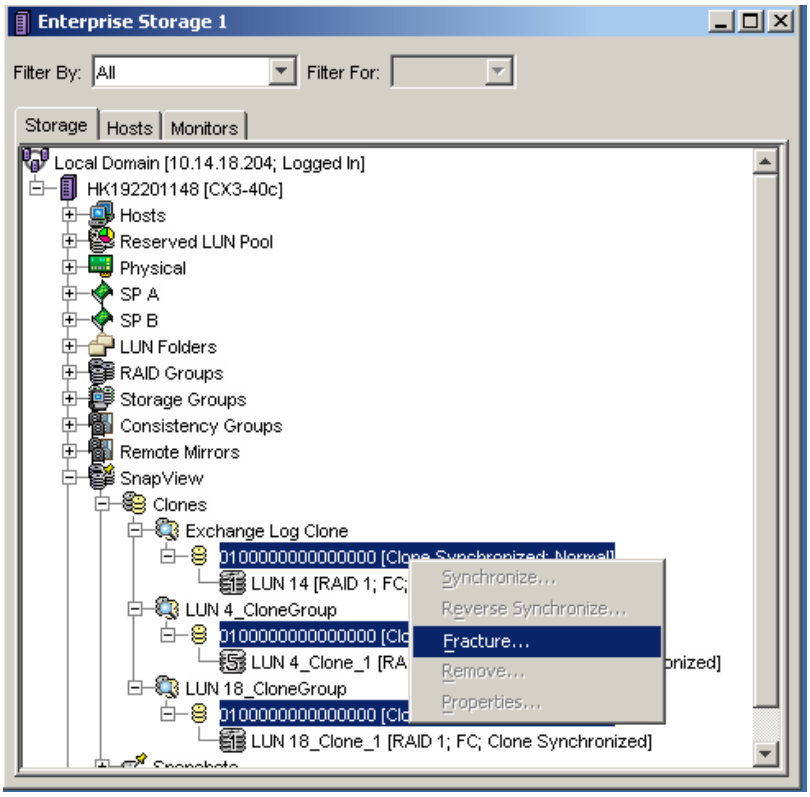

Figure 11. Clone consistent fracture via Navisphere Manager

To perform a consistent fracture in the Navisphere CLI, the user must supply a switch:

```
-consistentfractureclone
```

For the previous example, the full command is:

```
snapview -consistentfractureclones -CloneGroupNameCloneId
"Exchange Log Clone" 0100000000000000 LUN4_CloneGroup
0100000000000000 LUN18_CloneGroup 0100000000000000
```

As with other user-initiated fractures, clones that have been consistently fractured will appear as Administratively Fractured in the clone's properties. If a failure occurs during a consistent fracture, such that one of the clones cannot be fractured, all other clones will be queued for resynchronization. If a clone state is already fractured (admin or system), it is not eligible for inclusion in the set of clones to be consistently fractured.

SnapView consistent fracture limits

Table 3 provides the limits associated with SnapView consistent operations.

Table 3. SnapView consistent fractures

Storage system	CX3-80, CX3-40, CX700, CX500	CX3-20, CX3-10c, CX300
Consistent clone fracture		
Clones being fractured	16	8

Note: A limited number of objects can be included in a single command; once one consistent operation command has been sent to Navisphere, another one can be sent—in a matter of moments, hours, or whatever the replication policies specify.

Reporting

With the advent of the Navisphere Taskbar, users may now create customized reports of their storage-system configuration. A report specific to clones may be initiated that displays all clone groups on the system, their states, and associated LUN numbers.

The Report Wizard is launched by selecting the Generate Report option from the Reporting tab. To generate the report, choose SnapView Clones under Storage Applications from the list of available reports. An example is provided in Figure 12:

Subsystem Name: HK192201148

Report Generated Date and Time: Mon Nov 06 08:33:58 EST 2006

==========

Snapview Clone Report

Clone Private LUNs	Clone_Private_LUN_SPA, Clone_Private_LUN_SPB

Source LUN Information			Clone Information									
Clone Croup Name	Source LUN	Folders	Clone LUN	Clone ID	State	Image Condition	Available for I/O	Dirty	Sync %	Recovery Policy	Sync Rate	Protected Restores	Fractured
Exchange Log Clone	LUN 17	SP A	LUN 14	0100000000000000	Synchronized	Normal	No	No	100	Automatic	Medium	No	No
LUN 18_CloneGroup	LUN 18	SP A	LUN 18_Clone_1	0100000000000000	Synchronized	Normal	No	No	100	Automatic	Medium	No	No
LUN 4_CloneGroup	LUN 4	SP A	LUN 4_Clone_1	0100000000000000	Synchronized	Normal	No	No	100	Automatic	Medium	No	No

Figure 12. SnapView Clone Report

The Report Wizard allows users to get a quick overview of all the clone operations currently taking place on the storage system. This report helps keep track of multiple clones in the storage system and check their status as needed.

Performance considerations with clones

The following information provides a general overview of various performance considerations when using clones. For details around performance expectations for a specific environment, users should engage their account team and designated CLARiiON C-Speed representative, so that all relevant aspects of the environment are factored in as appropriate.

The tests referenced in the following section were conducted on a CX3-80.

Performance of initial clone synchronization

The user-selectable synchronization rates are high, medium, and low; medium is the default. The high synchronization rate devotes a high percentage of the CLARiiON CPU to completing the synchronization as fast as possible, whereas the low and medium synchronization rates allow many other operations to occur during the synchronization. This slows down the synchronization but also minimizes the impact of the synchronization. (As noted, medium is the default, and the rate can be changed while a synchronization is in progress.) Figure 13 shows the bandwidth achievable for synchronizations taking place at high, medium, and low settings. For this test, all drives were 73 GB, 15k rpm 4 Gb/s RAID 5 (4+1) drives, all belonging to a single SP. The SP CPU rates are not directly pictured in Figure 13, so this information will be provided along with an explanation of the bandwidth achieved at the various sync rates. During the synchronization, there was no active I/O to the source LUNs.

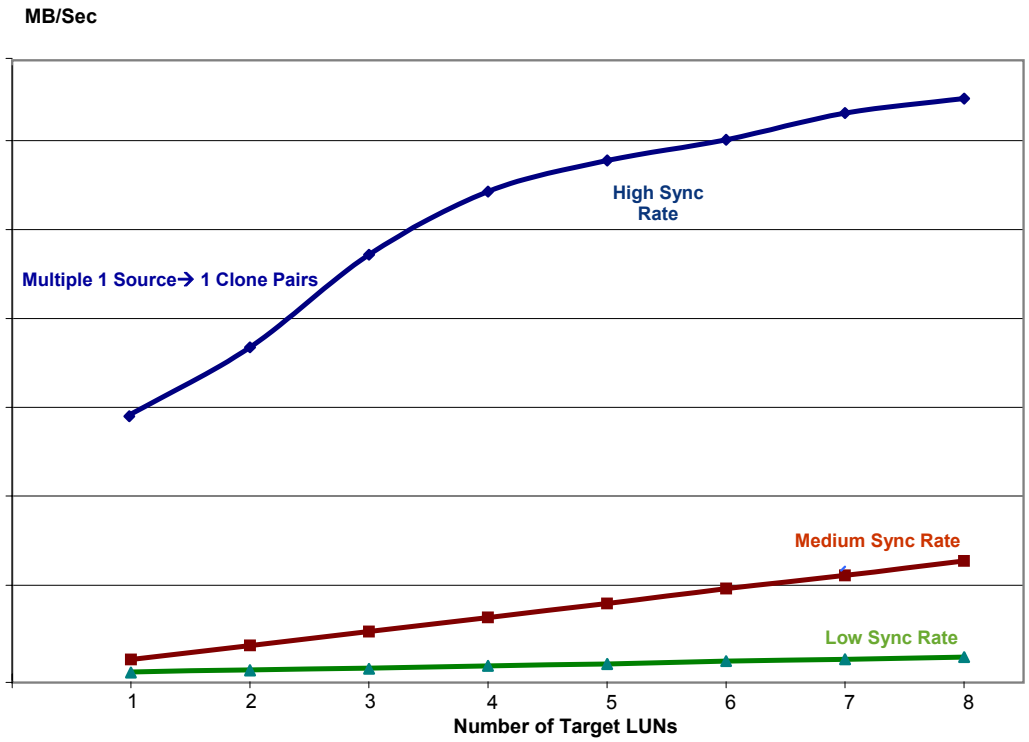

Figure 13. Initial sync rates on a CX3-80—source and clones on Fibre Channel drives

When synchronizing multiple source:clone pairs at a high sync rate (on the CX3-80), the CPU utilization becomes saturated at eight pairs instead of two pairs as with previous FLARE releases. This is due to the changes in the FLARE release 26 code, which uses less CPU resources than FLARE release 24 during synchronization. Users who want to perform clone synchronizations with less impact on the production data should select a lower sync rate. In this test, the impact due to synchronizing eight clones at a medium sync rate was much less; and the impact of synchronizing eight clones at a low sync rate was even lower

when compared to the high-sync rate. Users should keep in mind that actual impact will depend on their particular workload.

As noted, this test was conducted using a single SP. When using both SPs, the bandwidth at low and medium sync rates essentially doubles. However, at high sync rates the bandwidth does not double when a second SP is introduced as there is an increased load to the mirrored write cache. Be sure to consider the time of day and load on the storage system when deciding which rate to select.

Initial synchronization rates compared to incremental resynchronization rates

An initial synchronization is a full sequential read/write operation where all extents are contiguous. The bandwidth achieved during this operation is generally going to be much higher than subsequent resynchronizations because during resynchronizations, the data is contained in noncontiguous extents. The degree of difference between the two will depend on just how noncontiguous the extents to be updated are.

Also, to minimize the overhead of maintaining the fracture log for individual server I/Os, the size of the clone extents that are tracked for change are generally quite a bit larger than the actual server I/Os (the actual size of the extent is proportional to the size of the source LUN). The relevance here is that the resynchronization is at the extent level and not at the individual I/O level.

Finally, users should keep in mind that the performance charts were generated at a period of no activity on the source LUNs. If there is significant I/O activity on the source LUNs during an initial synchronization or an incremental resynchronization, this will also decrease the achievable bandwidth to the clone.

Use of ATA drives with clones

A common question is whether Advanced Technology-Attached (ATA) drives can be used as clones. The performance of ATA drives, in general, will depend on the available write cache. As long as the write cache can absorb the writes to the ATA drives, there will be no noticeable performance difference between writes to Fibre Channel drives and writes to ATA drives. Once the write cache becomes full, however, and needs to flush the data to the ATA drives, the ATA drives may not be able to handle the writes quickly enough—this is when the difference between Fibre Channel and ATA drives will become noticeable.

When using ATA drives for clones, either ATA or SATAII, users should keep in mind that while synchronizing the clone, if the ATA drives cannot handle the flush requests from the write cache quickly enough, the synchronization operation will take longer when performed on the ATA drives—and it will adversely impact other write operations on the array. This is especially important to consider, since ATA drives typically have a higher impact on CPU utilization, and lower bandwidth for random write loads. Additionally, once the clone is synchronized—and until it is fractured—all writes to the source LUN will be simultaneously written to the clone. In this case, if the write cache becomes full, the writes to the clone will be processed at the slower ATA processing rate and thus impact server response time. Finally, given that a key benefit with a clone is that it can be used in the event that the production LUN encounters certain hardware failures, keep in mind that ATA LUNs will rarely be suitable alternate production LUNs.

For these reasons, it is generally not recommended that you implement ATA clones of Fibre Channel source LUNs, but rather only as clones of other ATA drives. With this in mind, it is important to note that all subsequent performance discussions assume clones were created on Fibre Channel drives. Keep in mind that these performance characteristics could vary considerably should the user choose to implement clones on ATA drives.

Server impact during initial synchronization

It is also important to consider the performance impact on the production LUN during the synchronization operation. As noted earlier, clones should be placed on different drives from the source volume to reduce spindle contention. Assuming this practice is followed, reads and writes to the source hit different spindles than reads and writes to the clone. Consequently, the only time that clones will impact source performance is when the clone is being synchronized.

To offset this impact, users are advised to strategically select the times when they resynchronize their clone to minimize the impact to source volume performance. Additionally, as noted previously, users can select the synchronization rate that best suits their environment—essentially choosing between allocating more storage-system resources to complete the copy faster, or allowing the copy process to take place in the background, and thus extending the copy time.

Figure 14 compares the effect on server I/O when selecting among low, medium, and high synchronization rates during the initial clone synchronization on a CX3-80. For this test there were 36 source:clone pairs, spread evenly between SPs. An OLTP simulator was used, with medium- and high-user workloads, and the definition of a transaction consisted of 21 reads and nine writes, plus one write to the transaction log.

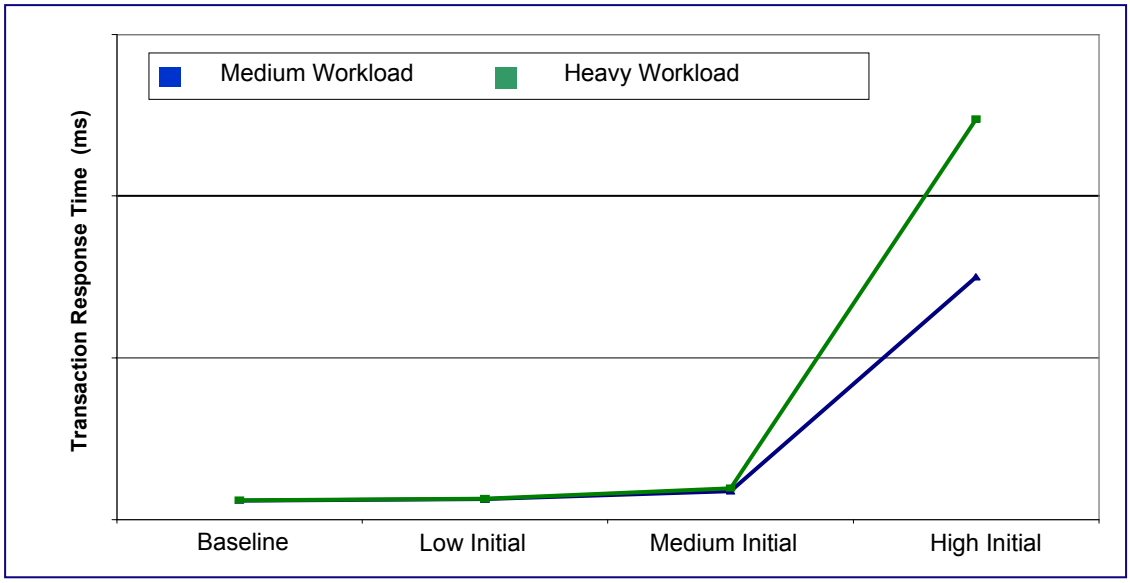

Figure 14. OLTP average response time during initial sync—Fibre Channel

Note that application response times tend to increase significantly when the clone is being synchronized at a high sync rate. When using a low or medium synchronization rate, the impact to the OLTP load is close to negligible. Scaling up to a high sync rate, the response time increases significantly as the mirrored write cache links become saturated and the SP CPU becomes nearly 100 percent utilized. Also, note that at a high initial sync rate, the transaction response time increases significantly when changing the load on the system from medium to high (where load reflects to the number of users).

Figure 15 uses the same test parameters but provides values for transactions per minute (TPM) rather than response time, for users who are more familiar with evaluating performance in terms of TPM.

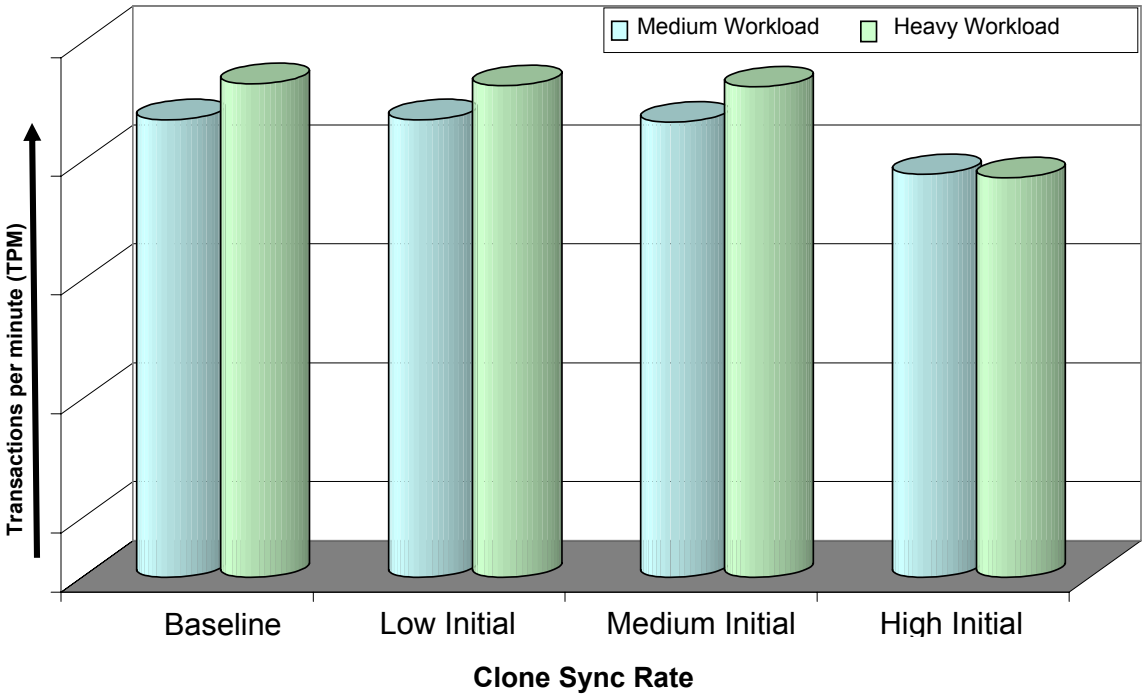

Figure 15. OLTP transactions per minute during initial sync—Fibre Channel

As expected, the trends illustrated in Figure 15 are consistent with those shown in Figure 14. As compared to the baseline, the impact on transactions per minute (TPM) at a low sync rate is almost negligible. Likewise, as the sync rate increases for the Fibre Channel clones, the performance impact on the source also increases. However, due to the changes in FLARE release 26, the TPM numbers depicted for high sync rate increased from FLARE release 24 by 1.5x due to the significant decrease in SP CPU utilization.

With this information in mind, choose a combination of sync rates and time of day for synchronization that works with the application performance that must be sustained. If performance on all applications must be kept to a maximum 24x7, ensure that the lightest production time possible is chosen to perform initial syncs, and that the low sync rate is chosen, rather than the default of medium.

Server impact during clone resynchronization and reverse synchronization, compared to other clone operations

It is important to consider the performance impact on clones during the following clone states:

- Baseline (no clones defined)
- Fractured (no synchronization or mirrored writes occurring)
- Synchronizing (clone is being updated from the source)
- Reverse synchronizing (source is being restored by the clone)
- Synchronized (synchronization has completed; server writes to source are mirrored to clone)

One of the benefits of clones is that once they are fractured, there is no observable impact on source performance. This means that clones can be used for backup operations or any read- and/or write-intensive activity, with no impact to the source LUN. This point is important to consider when trying to decide whether to implement SnapView snaps or clones. Since clones in a fractured state have no observable impact on source performance, the fractured state is more or less equivalent to the baseline, and thus will not be represented on the following two graphs.

Figure 16 and Figure 17 show the results of testing when clones are in each of the other states named previously. The testing parameters were the same as those shown in Figure 14 and Figure 15 and the sync rate for each of these tests was set to high.

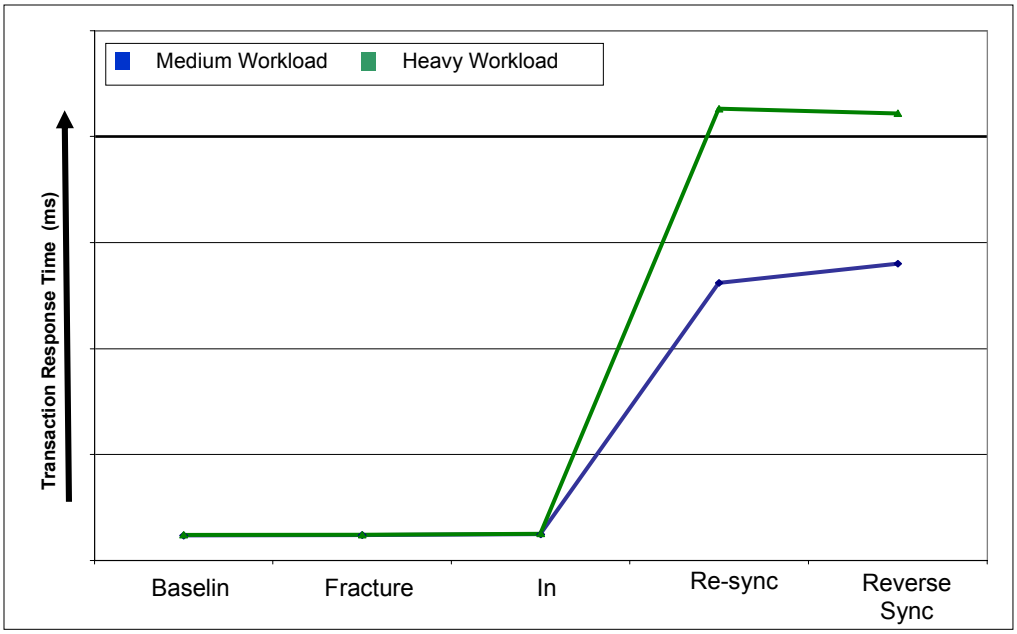

Figure 16. OLTP response time during various states at a high sync rate—Fibre Channel

Figure 16 illustrates the effect that a synchronized, or resyncing clone, at a high sync rate may have to the production server in terms of response time. Figure 17 provides this same information, in terms of TPM.

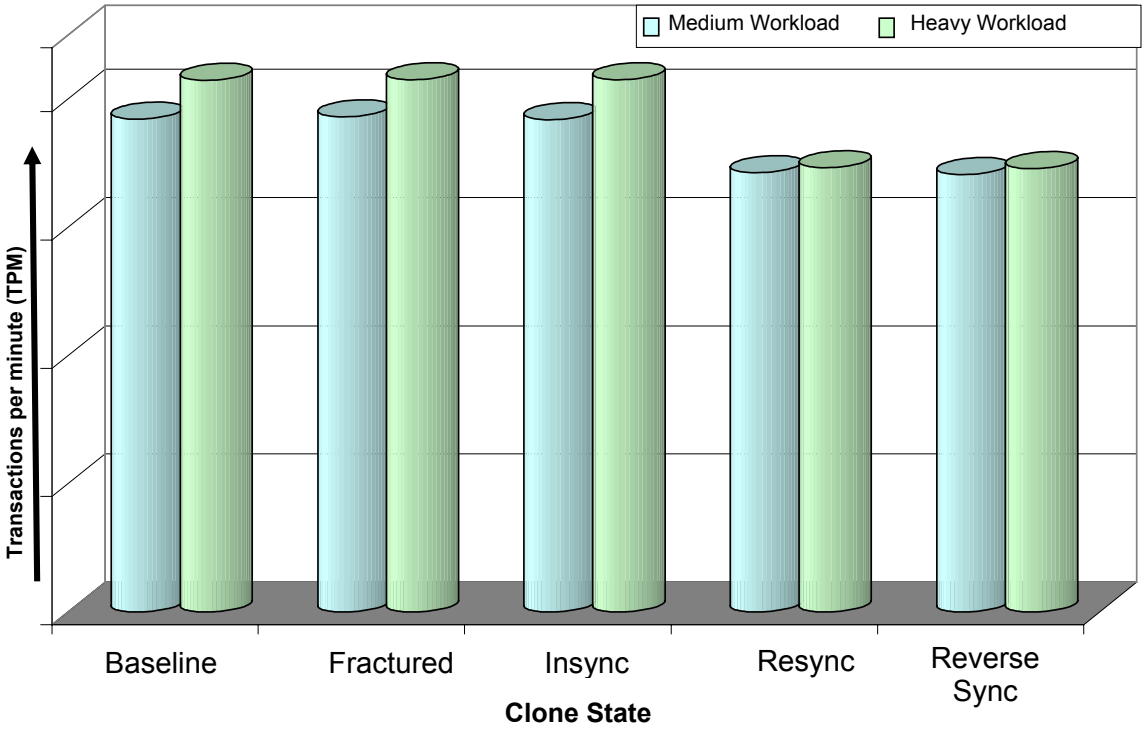

Figure 17. OLTP transactions per minute during various states at a high sync rate—Fibre Channel

As expected, both figures show that the resync operation at a high sync rate creates the most impact on server processing. Once the clone is synchronized (in sync), and before it is fractured, there can be a slight impact on server I/O. This impact is due to the fact that the server writes are mirrored to the clone - essentially performing two writes before acknowledgement for each I/O. It is for this reason that users should fracture the clones as soon as they become synchronized. Not only does this benefit performance, but it also ensures that the point-in-time copy is preserved, protecting your data against any hardware or logical errors on the source.

Looking at performance considerations when using clones then, the key considerations are:

- Clones have almost zero impact on the server I/O when fractured.
- Users can adjust the sync rates to minimize server impact during synchronization operations.
- It is not recommended to use ATA drives as clones for Fibre Channel drives.
- Users should fracture clones once they are synchronized to minimize server impact and to provide for a valid, point-in-time copy of the data.

Given these considerations, in order to minimize impact on source data, users are encouraged to:

- Synchronize clones at off-peak hours, if at all possible.
- Place clones on the same type of drives as source LUNs.
    - If source LUNs are bound on Fibre Channel drives, then clones should be as well.
    - If source LUNs are bound on ATA drives, then clone LUNs may be bound on ATA drives as well.

These are not absolute requirements, but rather are best practices focused on ensuring optimal performance of source LUNs when working with clones.

Conclusion

SnapView clones allow users to create fully populated point-in-time copies of LUNs for the purpose of parallel operations, without impacting production processing. Additionally, with the ability to copy changes from source to clone—or the reverse—SnapView clones provide users flexibility in data protection and data preservation. In combination with instantaneous snapshots of data made available with SnapView, as well as remote data replication made available with MirrorView, SnapView clones round out a robust EMC software suite of business continuity solutions.

References

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

- *Navisphere Manager Help*
- *CLARiiON SnapView Snapshots and Snap Sessions Knowledgebook* white paper
- *EMC CLARiiON Reserved LUN Pool Configuration Considerations* white paper
- *EMC CLARiiON Best Practices for Fibre Channel Storage*: *CLARiiON Release 26 Firmware Update - Best Practices Planning* - Provides general guidelines and best practices for optimizing CLARiiON performance.
- E-Lab Navigator at EMC.com

Appendix: Clone limits

The number of clones and clone groups varies by model and FLARE release. Table 4 and Table 5 summarize these numbers, as well as the total number of clones per storage system. In the tables, an *image* refers to a LUN participating as a SnapView clone. Source LUNs are not included in the supported total image count.

Table 4. Release 24 and 26 clone limits per storage-system model

Platform	CX3-80	CX3-40	CX3-20	CX3-10
Clone sources				
Per storage system [1]	512	256	128	64
Clone images				
Per storage system [2]	1024	512	256	128
Clones per source	Up to 8			

1. Clone sources are no longer counted with clones or mirrors for total image count.

2. As of release 24, SnapView clone limits are *no longer* shared with MirrorView/Synchronous LUN limits.

These limits are enforced by SnapView. This means that as 512 total clone images are allowed on a CX3-40, each of the 256 clone sources on the storage system cannot be populated with the full eight clones per group because that would generate over 2,000 total clone images. Likewise not every clone group on a CX3-20 and CX3-80 can have the maximum number of clones, since again the per-storage-system limit would be reached first. With these limitations in mind, this means that each source LUN may have an average of two clones configured.

Because each clone group has a single source LUN, the number of clone groups is essentially the same as the number of source LUNs that can have clones.

Table 5. Release 19 and 22 clone limits per storage-system model

Platform	CX3-80, CX3-40, CX700	CX3-20, CX500	CX300
Clone sources			
Per storage system [1]	100	50	50
Clone images			
Per storage system [2]	200	100	100
Clones per source	Up to 8		

1. Clone sources are no longer counted with clones or mirrors for total image count.

2. SnapView clone limits are shared with MirrorView/Synchronous LUN limits.

Prior to release 24, the total number of clones per storage system is shared with MirrorView/S, so that there is a total of:

- 200 images (mirrors or clones) allowed on a CX3-80 (release 22) or CX700 (release 19)
- 100 images allowed on a CX3-40 or CX500
- 100 for CX3-20

This total includes both production LUNs, as well as mirrored LUNs for MirrorView. However, as of FLARE release 19, this includes only clone LUNs (and not their sources).

The most up-to-date clone limits are maintained in the *EMC CLARiiON Open Systems Configuration Guide*. Refer to this guide for the latest listing of supported CLARiiON models and associated limits.