

SnapView Foundations

Upon completion of this module, you will be able to:

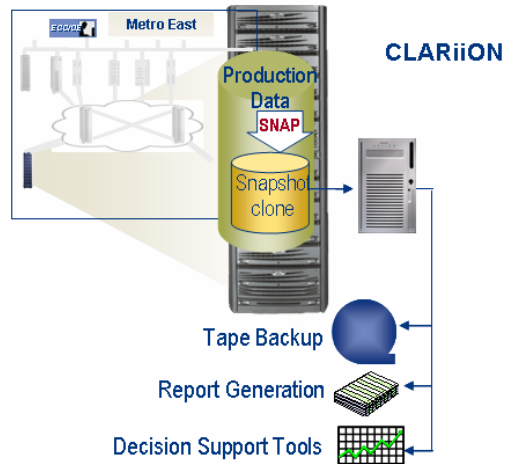
- Describe the Business Continuity needs for application availability and recovery
- Describe the functional concepts of SnapView on the CLARiiON Storage Platform
- Describe the benefits of SnapView on the CLARiiON Storage Platform
- Identify the differences between the Local Replication Solutions available in SnapView

The objectives for this module are shown here. Please take a moment to read them.

EMC SnapView

Creates point-in-time views or point in time copies of logical volumes

- Allows parallel access to production data with SnapView Snapshots and Clones
- Snapshots are pointer based snaps that require only a fraction of the source disk space
- Clones are a full volume copy but require equal disk space
- SnapView snapshots and clones can be created and mounted in seconds and are read and write capable



SnapView is an array software product that runs on the EMC CLARiiON. Having the software resident on the array has several advantages over host-based products. Since SnapView executes on the storage system, no host processing cycles are spent managing information. Storage-based software preserves your host CPU cycles for your business information processing, and offloads information management to the storage system, in this case, the CLARiiON. Additionally, storage-based SnapView provides the advantage of being a singular, complete solution that provides consistent functionality to all CLARiiON connected server platforms.

EMC SnapView allows companies to make more effective use of their most valuable resource, information, by enabling parallel information access. Instead of traditional sequential information access that forces applications to queue for information access, SnapView allows multiple business processes to have concurrent, parallel access to information.

SnapView creates logical point-in-time views of production information through Snapshots, and point-in-time copies through Clones. Snapshots use only a fraction of the original disk space, while Clones require an equal amount of disk space as the source.

EMC Education Services

EMC²
where information lives™

SnapView Snapshots

- Uses Copy on First Write Technology
 - Fast snapshots from production volume
 - Takes a fraction of production space
 - Remains “connected” to the production volume
- Creates instant snapshots which are immediately available
 - Stores changed data from a defined point-in-time
 - Utilizes production for unchanged data
- Offers multiple recovery points
 - Up to eight snapshots can be established against a single source volume
 - Snapshots of Clones are supported (up to eight snapshots per Clone)
- Accelerates application recovery
 - Snapshot “roll back” feature provides instant restore to source volume



© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 3

Mercer Road
EMC Education Services

A SnapView snapshot is not a full copy of your information; it is a logical view of the original information, based on the time the snapshot was created. Snapshots are created in seconds and can be retired when no longer needed. Snapshots can be created quickly and can be deleted at will.

In contrast to a full-data copy, a SnapView snapshot usually occupies only a fraction of the original space. Multiple snapshots can be created to suit the need of multiple business processes. Secondary servers see the snapshot as an additional mountable disk volume. Servers mounting a snapshot have full read/write capabilities on that data.



SnapView Foundations

SNAPVIEW TERMINOLOGY



© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 4 

This section will define some terms used within SnapView.

SnapView – Terminology

- **Production host**
 - Server where customer applications execute
 - Source LUNs are accessed from production host
 - Utility to start/stop Snapshot Sessions from host provided - admsnap
 - Snapshot access from production host is not allowed
- **Backup (or secondary) host**
 - Host where backup processing occurs
 - Offloads backup processing from production host
 - Snapshots are accessed from backup host
 - Backup media attached to backup host
 - Backup host must be same OS type as production host for filesystem access (not a requirement for image/raw backups)

Some SnapView terms are defined here. The Production host is where customer production applications are executed. The Secondary host is where the snapshot will be accessed from.

Any host may have only one view of a LUN active at any time. It may be the Source LUN itself, or one of the 8 permissible snapshots. No host may ever have a Source LUN and a Snapshot accessible to it at the same time.

If the snapshot is to be used for testing, or for backup using filesystem access, then the production host and secondary host must be running the same operating system. If raw backups are being performed, then the filesystem structure is irrelevant, and the backup host need not be running the same operating system as the production host.

SnapView – Terminology (continued)

- **Source LUN**
 - Production LUN – this is the LUN from which Snapshots will be made
- **Snapshot**
 - Snapshot is a “frozen in time” copy of a Source LUN
 - Up to 8 R/W Snapshots per Source LUN
- **Reserved LUN Pool**
 - Private area used to contain copy on first write data
 - One LUN Pool per SP – may be grown if needed
 - All Snapshot Sessions owned by an SP use one LUN Pool
 - Each Source LUN with an active session is allocated one or more Reserved LUNs

The Source LUN is the production LUN which will be snapped. This is the LUN which is in use by the application, and will not be visible to secondary hosts.

The snapshot is a point-in-time view of the LUN, and can be made accessible to a secondary host, but not to the primary host, once a SnapView session has been started on that LUN.

The Reserved LUN Pool – strictly 2 areas, one pool for SPA and one for SPB – holds all the original data from the Source LUN when the host writes to a chunk for the first time. The area may be grown if extra space is needed, or, if it has been configured as too large an area, it may be reduced in size. Because each area in the LUN Pool is owned by one of the SPs, all the sessions that are owned by that SP use the same LUN Pool. We'll see shortly how the component LUNs of the LUN Pool are allocated to Source LUNs.

SnapView – Terminology (continued)

- **SnapView Session**

- SnapView Snapshot mechanism is activated when a Session is started
- SnapView Snapshot mechanism is deactivated when a Session is stopped
- Snapshot appears “off-line” until there is an active Session
- Snapshot is an exact copy of Source LUN when Session starts
- Source LUN can be involved in up to 8 SnapView Sessions at any time
- Multiple Snapshots can be included in a Session

- **SnapView Session name**

- Sessions should have human readable names
- Compatibility with admsnap – use alphanumerics, underscores

Having a LUN marked as a Source LUN – which is what happens when a Snapshot is created on a LUN – is a necessary part of the SnapView procedure, but it isn't all that is required. To start the tracking mechanism and create a virtual copy which has the potential to be seen by a host, we need to start a session. A session will be associated with one or more Snapshots, each of which is associated with a unique Source LUN. Once a Session has been started, data will be moved to the SnapView cache as required by the COFW (Copy On First Write) mechanism. To make the Snapshot appear on-line to the host, it is necessary to activate the Snapshot. These administrative procedures will be covered shortly.

Sessions are identified by a Session name, which should identify the session in a meaningful way. An example of this might be 'Drive_G_8am'. These names may be up to 64 characters long, and may consist of any mix of characters. Remember, the utilities, such as admsnap, make use of those names, often as part of a host script, and that the host operating system may not allow certain characters to be used. Quotes, triangular brackets, and other special characters may cause problems; it is best to use alphanumerics and the underscore.



SnapView Foundations

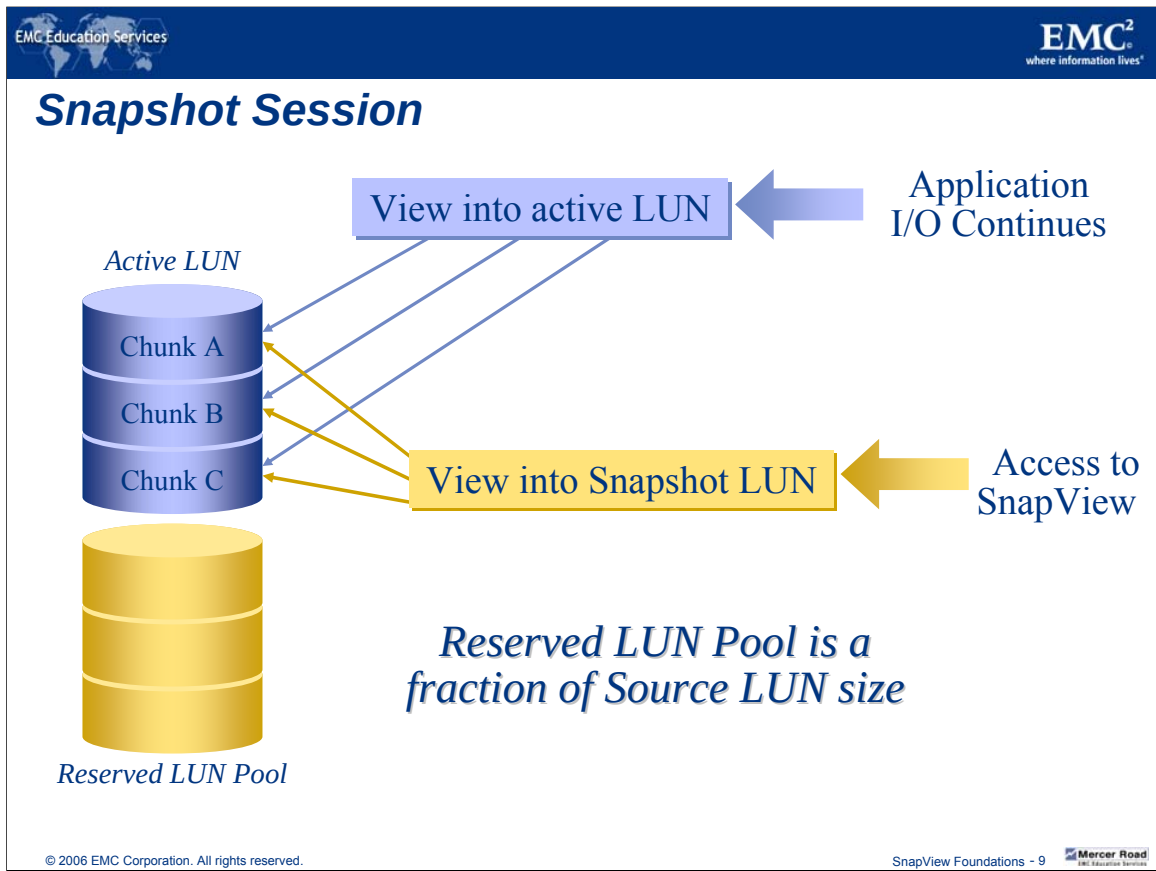
THEORY OF OPERATION

© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 8



This section will look at the theory of operation of SnapView.



When you create a snapshot, a portion of the previously created Reserved LUN Pool is zeroed, and a mount point for the snapshot LUN is created. The newly created mount point is where the secondary host(s) will attach to access the snapshot.

SnapView – SnapView Sessions

- **Start/stop Snapshot Sessions**
 - Can be started/stopped from Manager/CLI or from production host via admsnap
 - Requires a Session name
- **Snapshot Session administration**
 - List of active Sessions available
 - From management workstation only
 - Session statistics
 - From management workstation only
 - Snapshot Cache usage
 - Performance counters
 - Analyzer tracks some statistics

Once the Reserved LUN Pool is configured and snapshots created on the selected Source LUNs, we now start the Snapshot Sessions. That procedure may be performed from the GUI, the CLI, or admsnap on the Production host. The user needs to supply a Session Name – this name will be used later to activate a snapshot.

When Sessions are running, they may be viewed from the GUI, or information may be gathered by using the CLI. All sessions are displayed under the Sessions container in the GUI.

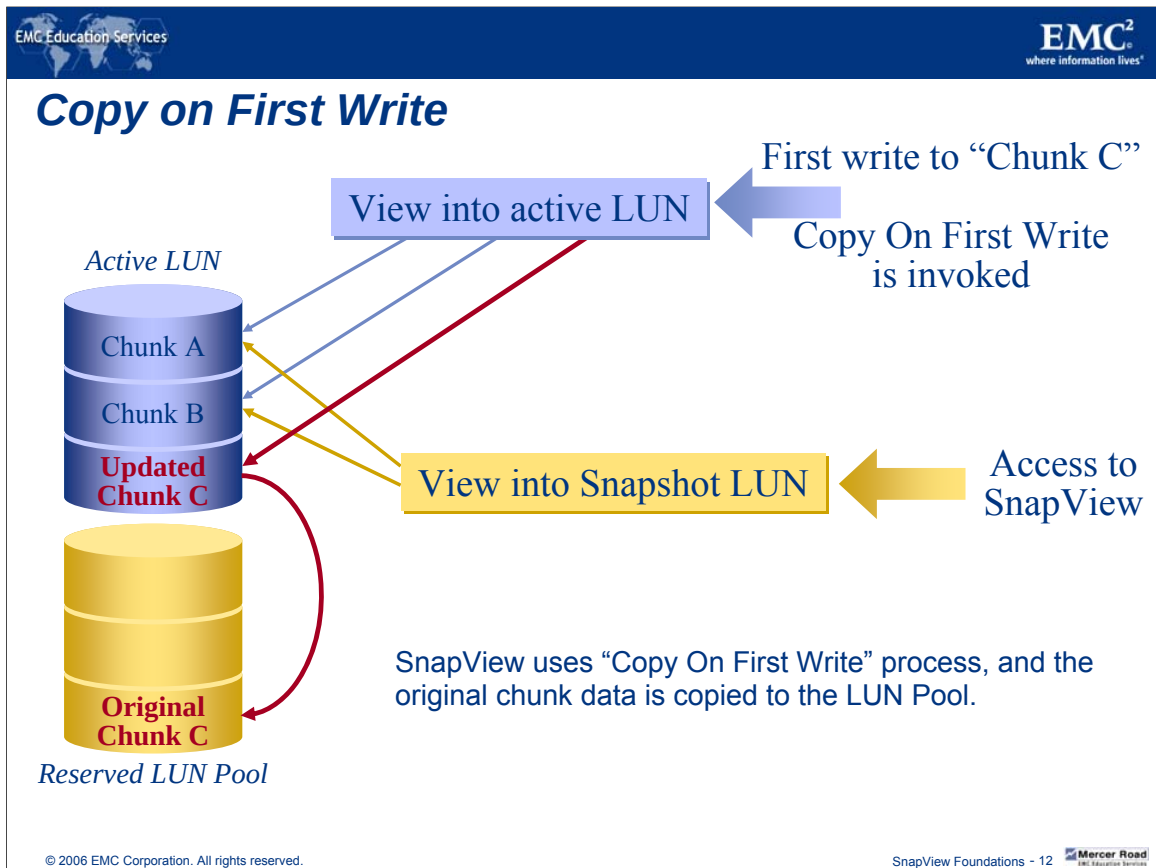
SnapView – Copy on First Write

- **Allows efficient utilization of copy space**
 - Uses a dedicated Reserved LUN Pool
 - LUN Pool typically a fraction of Source LUN size for a single Snapshot
- **Saves original data chunks – once only**
 - Chunks are a fixed size - 64 KB
 - Chunks are saved when they're modified for the first time
- **Allows consistent “point in time” views of LUN(s)**

The Copy On First Write mechanism involves saving an original data block into snapshot cache, when that data block in the active filesystem is about to be changed. The use of the term “block” here may be confusing, because this block is not necessarily the same size as that used by the filesystem or the underlying physical disk. Other terms may be used in place of “block” when referring to SnapView – the official term is ‘chunk’.

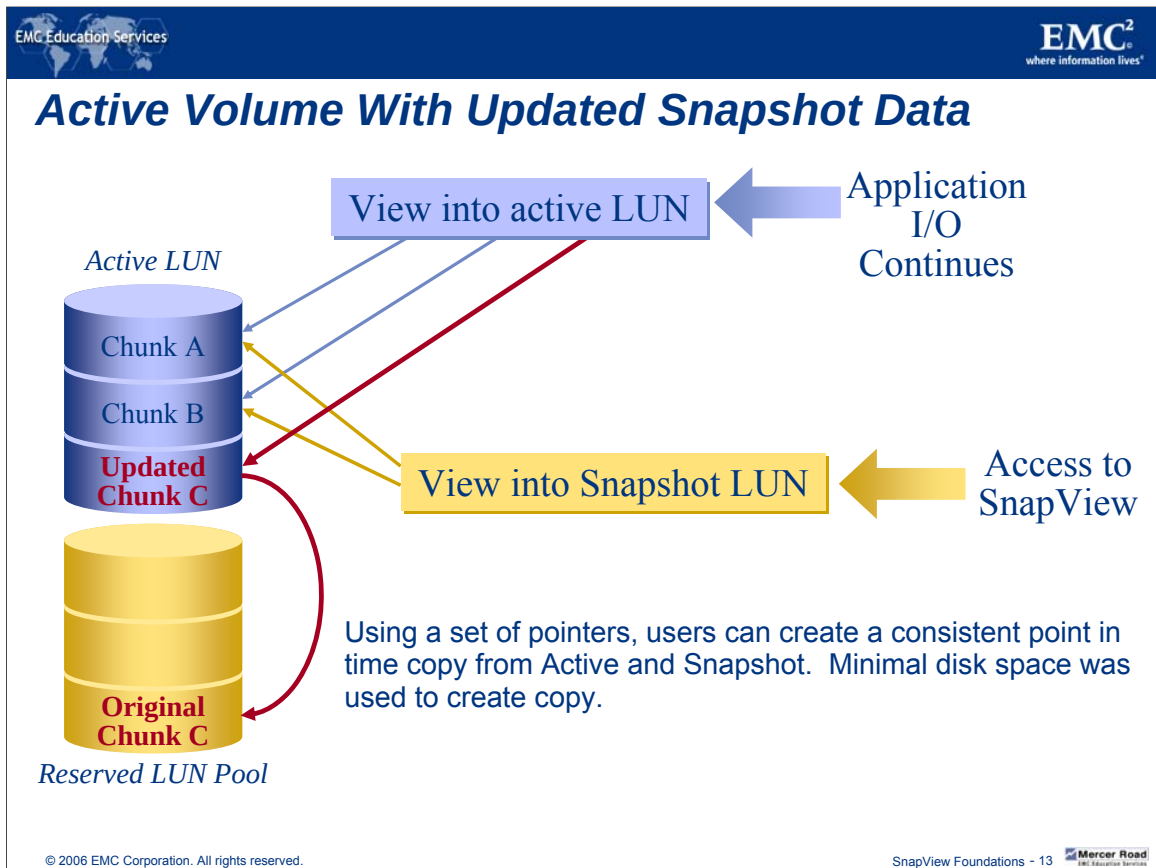
The chunk is saved only once per snapshot – SnapView allows multiple snapshots of the same LUN. This ensures that the view of the LUN is consistent, and, unless writes are made to the snapshot, will always be a true indication of what the LUN looked like at the time it was snapped.

Saving only chunks that have been changed allows efficient use of the disk space available; whereas a full copy of the LUN would use additional space equal in size to the active LUN, a snap may use as little as 10% of the space, on average. This depends greatly, of course, on how long the snap needs to be available and how frequently data changes on the LUN.



SnapView uses a process called "Copy On First Write" (COFW) when handling writes to the production data during a running session.

For example, let's say a snapshot is active on the production LUN. When a host attempts to write to the data on the Production LUN, the original Chunk C is first copied to the Reserved LUN Pool, then the write is processed against the Production LUN. This maintains the consistent, point-in-time copy of the data for the ongoing snapshot.



Once the Copy On First Write has been performed, the pointer is redirected to the block of data in the Reserved LUN Pool. This maintains the consistent point in time of the snapshot data, while minimizing the additional disk space required to create the snapshot that is now available to another host for parallel processing.

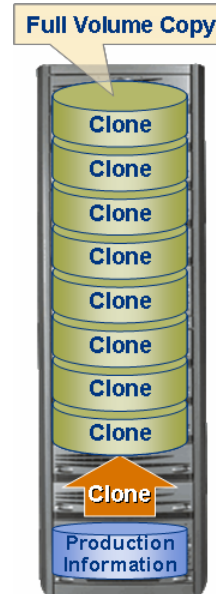
SnapView – Activating/Deactivating Snapshots

- **Activating a Snapshot**
 - Makes it visible to secondary host
- **Deactivating a Snapshot**
 - Makes it inaccessible (off-line) to secondary host
 - Does not flush host buffers (unless performed with admsnap)
 - Keeps COFW process active

To make the snapshot visible to the host as a LUN, the Snapshot needs to be activated. Activation may be performed from the GUI, from the CLI, or via admsnap on the Backup host. Deactivation of a snapshot makes it inaccessible to the Backup host. Normal data tracking continues, so if the snapshot is reactivated at a later stage, it will still be valid for the time that the session was started.

SnapView Clones – (Business Continence Volumes)

- Overall highest service level for backup and recovery
 - Fast sync on first copy, faster syncs on next copy
 - Fastest restore from Clone
- Removes performance impact on production volume
 - De-coupled from production volume
 - 100% copy of all production data on separate volume
 - Backup operations scheduled anytime
- Offers multiple recovery points
 - Up to eight Clones can be established against a single source volume
 - Selectable recovery points in time
- Accelerates application recovery
 - Instantly restore from Clone, no more waiting for lengthy tape restore



Clones offer us several advantages in certain situations. Because copies are physically separate, residing on different disks and RAID groups from the Source LUN, there is no impact from competing I/Os. Different I/O characteristics, such as a database applications with highly random I/O patterns or backup applications with highly sequential I/O patterns running at the same time, will not compete for spindles. Physical or logical (human or application error) loss of one will not affect the data contained in the other.





SnapView Clones and SnapView Snapshots

- Each SnapView Clone is a full copy of the source
 - Creating initial Clone requires full sync
 - Incremental syncs thereafter
- Clones may have performance improvements over snapshots in certain situations
 - No Copy On First Write mechanism
 - Less potential disk contention depending on write activity
- Each Clone requires 1x additional disk space

	Snapshots	Clones
Elements per Source	8	8
Sources per storage system	100 Sources *	50 Clone Groups *
Elements per storage system	800 sessions * 300 snapshots *	100 total images *

* Indicates different limits for different CLARiiON models

© 2006 EMC Corporation. All rights reserved.
SnapView Foundations - 16



To begin, let's look at how SnapView Clones compare to SnapView snapshots.

Where both Clones and Snapshots are each point-in-time views of a Source LUN, the essential difference between them is that Clones are exact copies of their Sources – with fully populated data in the LUNs – rather than being based on pointers, with Copy on First Write Data stored in a separate area. It should be noted that creating Clones will take more time than creating Snapshots, since the former requires actually copying data.

Another benefit to the Clones having actual data, rather than pointers to the data, is the performance penalty associated with the Copy on First Write mechanism. Thus, Clones generate a much smaller performance load on the Source (than Snapshots).

Because Clones are exact replicas of their Source LUNs, they will generally take more space than Reserved LUNs, since the Reserved LUNs are only storing the Copy on First Write data. The exception would be where every chunk on the Source LUN is written to, and must therefore be copied into the Reserved LUN Pool. Thus, the entire LUN is copied and that, in addition to the corresponding metadata describing it, would result in the contents of the Reserved LUN being larger than the Source LUN itself.

The Clone can be moved to the peer SP for load balancing, but it will automatically get trespassed back for syncing.

SnapView is supported on the FC4700, and on all CX-series CLARiiONs except the CX200.

EMC Education Services

EMC²
where information lives[®]

SnapView Feature Limit Increases for Flare Release 19

Array	CX700	CX500	CX300
SnapView BCVs			
BCV Images <i>(sources no longer counted with BCVs for total image count)</i>			
per Storage System ^[1]	200	100	100
BCVs per Source	Up to 8		
BCV Sources			
per Storage System	100	50	50
			N/A

© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 17

Mercer Road
EMC Education Services

CX700 limits are 100 Clone Groups/array, and 200 images per array, where an image is a Clone, MV/s primary, or MV/s secondary (no longer includes Clone Sources).

[¹] SnapView BCV limits are shared with MirrorView/Synchronous LUN limits

Source and Clone Relationships

- **Adding Clones**
 - Must be exactly equal size to Source LUN
- **Remove Clones**
 - Cannot be in active sync or reverse-sync process
- **Termination of Clone Relationship**
 - Renders Source and Clone as independent LUNs
 - Does not affect data

Because Clones on a CLARiiON use MirrorView technology, the rules for image sizing are the same – source LUNs and their Clones must be exactly the same size.

Synchronization Rules

- Synchronizations from Source to Clone or reverse
- Fracture Log used for incremental syncs
 - Saved persistently on disk
- Host Access
 - Source can accept I/O at all times
 - Even when doing reverse sync
 - Clone cannot accept I/O during sync

Clones must be manually fractured following synchronization. This allows the administrator to pick the time that the clone should be fractured, depending on the data state. Once fractured, the Clone is available to the secondary host.

EMC Education Services

EMC²
where information lives™

Clone Synchronization

- “Refresh” Clones with contents of Source
 - Overwrites Clone with Source data
 - Using Fracture Log to determine modified regions
 - Host access allowed to Source, not to Clone

Clone 1 refreshed to Source LUN state

Source LUN

Production Server

Clone 1

Clone 2

...

Clone 8

Backup Server

X

© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 20

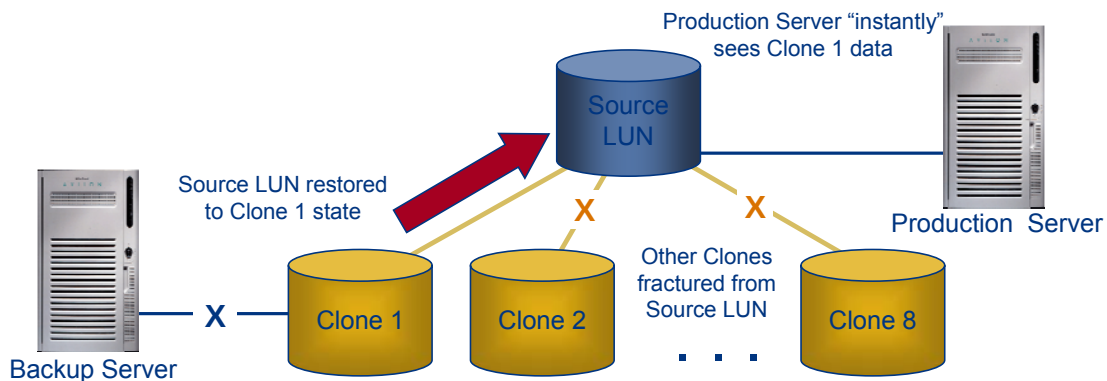
Mercer Road
EMC Education Services

Clone Synchronization copies source data to the clone. Any data on the clone will be overwritten with Source data.

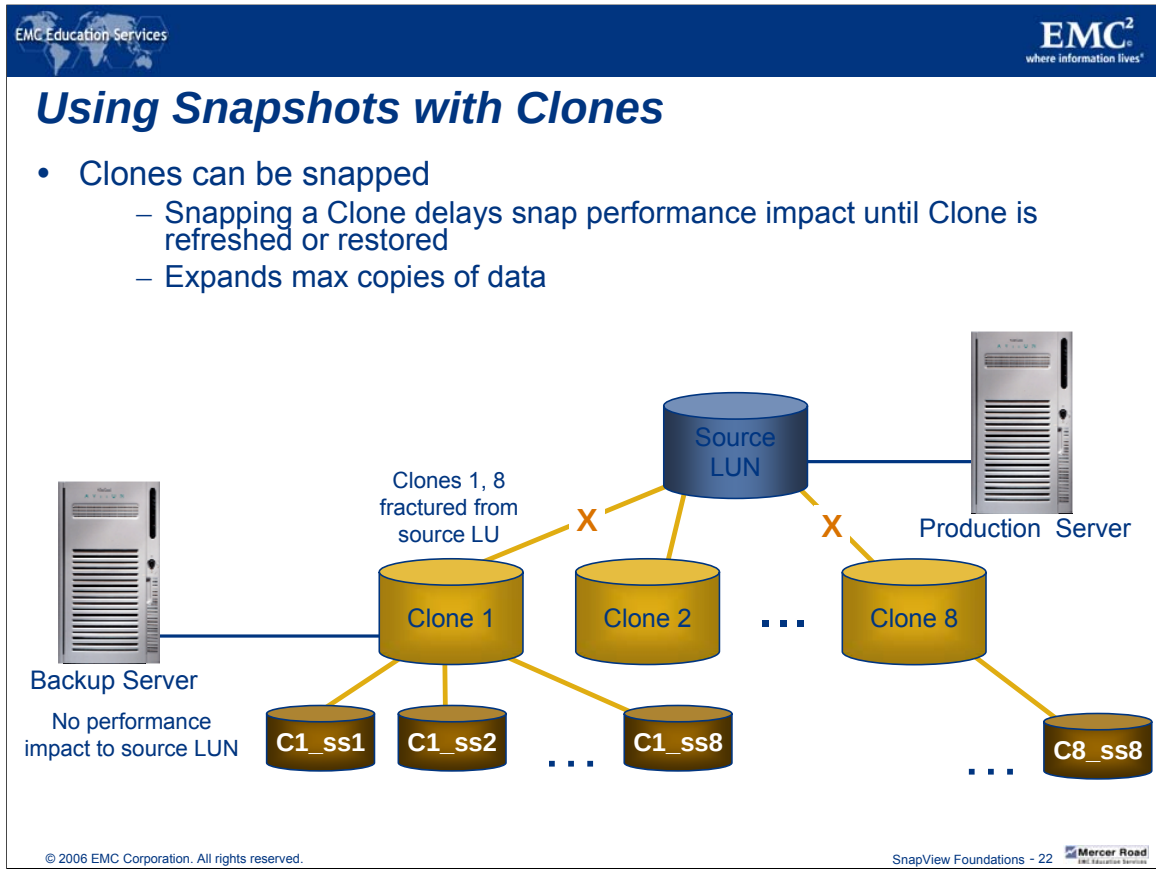
Source LUN access is allowed during sync with use of mirroring. The Clone, however, is inaccessible during sync. Any attempted host I/Os will be rejected.

Reverse Synchronization

- Restore Source LUN with contents of Clone
 - Overwrites Source with Clone data
 - Using Fracture Log to determine modified regions
 - Host access allowed to Source, not to Clone
 - Source “instantly” appears to contain Clone data



The Reverse Synchronization copies Clone Data to the Source LUN. Data on the Source is overwritten with Clone Data. As soon as the reverse-sync begins, the source LUN will seem to be identical to the clone. This feature is known as an “instant restore”.



Snapshots can be used with clones. So, taken to an extreme, this would offer 8 snapshots per clone, times 8 clones, plus the 8 clones, plus the additional 8 snapshots directly off the source – for a total of 80 copies of data!

SnapView Clone Functionality

- Clone Private LUN
 - Persistent Fracture Log
- Reverse Synchronization
 - Instant Restore
 - Protected Restore

Next, we'll look at clone functionality – with particular emphasis on those features that differentiate our product from our competition.

SnapView Clone Private LUN (CPL)

- **Contains persistent fracture log**
 - Tracks modified regions (“extents”) between each Clone and its source
 - Allows incremental resyncs – in either direction
- **128 MB private LUN on each SP**
 - Must be 128 MB/SP (total of 256 MB)
 - Pooled for all Clones on each SP
 - No other Clone operations allowed until private LUNs created

The Clone Private LUN contains the fracture log, which allows for incremental resyncs of data. This reduces the time taken to resync, and allows customers to better utilize the clone functionality.

Because it's stored on disk, it is persistent, and thus can withstand SP reboots/failures, as well as array failures. This allows customers to benefit from the incremental resync, even in the case of a system going down.

A Clone Private LUN is a 128 MB LUN that is allocated to each SP, and it must be created before any other Clone operations can commence.

Reverse-Sync – Protected Restore

- **Non-Protected Restore**

- Host→Source writes mirrored to Clone
 - Reads are re-directed to Clone
- When Reverse-sync completes:
 - Reverse-sync'ed Clone remains unfractured
 - ❖ Other Clones remain fractured

- **Protected Restore**

- Host→Source writes not mirrored to Clone
- When Reverse-sync completes:
 - All Clones are fractured
 - ❖ Protects against Source corruptions
- **Configure via individual Clone property**
 - **Must be globally enabled first**

Another major differentiating feature is our ability to offer a “protected restore” clone – this is essentially your “golden copy” clone.

To begin with, we’ll discuss what happens when protected restore is not explicitly selected. In that case, the goal is to send over the contents of the clone and bring the clone and the source to a perfectly in-sync state. To do that, writes coming into the source are mirrored over to the clone that is performing the reverse-sync. Also, once the reverse sync completes, the clone remains attached to the source.

On the other hand, when restoring a source from a “golden copy” clone, the golden copy needs to remain as-is. This means that the user wants to be sure that nothing from the source can affect the contents of the clone. So, for a protected restore, the writes coming into the source are NOT mirrored to the protected clone. And, once the reverse sync completes, the clone is fractured from the source.

Reverse-Sync – “Instant Restore”

- “Copy on Demand”
 - Host requests I/O to Source
 - Extent immediately copied from Clone
 - Host I/O is allowed to Source
 - Copying of extents from Clone continues
- For uninvolved extents, host I/O to source allowed, bypassing “Copy on Demand”

Reverse synchronizations will have the effect of making the source appear as if it is identical to the clone at the commencement of the synchronization. Since this “copy on demand” mechanism is designed to coordinate the host I/Os to the source (rather than the clone), host I/Os cannot be received by the clone during synchronization.

SnapView Consistent Operations: Fracture and Start

- **What is it?**
 - User-controlled (or scripted) consistent operations within Clones and SnapView layered drivers new in R19
 - “Consistent Fracture” – Fracturing a set of Clones consistently
 - “Consistent Start” – Starting a SnapView session consistently
- **How is it used?**
 - User defines set of Clone LUNs at beginning of Fracture
 - User defines set of source LUNs at beginning of Start
 - Performed with Navisphere or admsnap (SnapView sessions only)

New with the Release of Flare Code 19, a consistent fracture is when you fracture more than one clone at the same time in order to preserve the point-in-time restartable copy across the set of clones. The SnapView driver will delay any I/O requests to the source LUNs of the selected clones until the fracture has completed on all the clones (thus preserving the point-in-time restartable copy on the entire set of clones). A restartable copy is a data state having dependent write consistency and where all internal database/application control information is consistent with a Database Management System/application image.

The clones you want to fracture must be within different Clone Groups. You cannot perform a consistent fracture between different Clone Groups. You cannot perform a consistent fracture between different storage systems.

If there is a failure on any of the clones, the consistent fracture will fail on all of the clones. If any clones within the group were fractured prior to the failure, the software will re-synchronize those clones.

Consistent fracture is supported on CX-Series storage systems only. If you have a CX600 or CX700 storage system, you can fracture up to 16 clones at the same time. If you have another supported CX-Series storage system, you can only fracture up to 8 clones at the same time. A maximum of 32 consistent fracture operations can be in progress simultaneously per storage system.

If you consistent fracture while synchronizing, you will be Out-Of-Sync, which is allowed but may not be a desirable data state.

SnapView – Consistent Operations Overview

- **Consistent Operations**

- Maintains ordered writes across the set of member LUNs
 - Critical for dependent write consistency
- Set can span SPs within one array, but not across arrays
- All or nothing; operation performed on all set members or none

- **No “group” concept or association**

- Allows server-centric control, rather than array-centric control
 - Admsnap can split file systems and volumes by name
 - Set of LUNs that comprise file systems and volumes can change
 - Scripts that use admsnap are not modified when sets change
- No bond on the source LUNs after the operation
 - Source LUNs can still participate in other SnapView operations
- Managed via Navi GUI, CLI, or admsnap (Snap sessions only)
 - Simple extensions (switches)

Problems can occur if dependent writes occur out of sequence. This results in data lacking logical consistency relative to each other. Snap sessions reflect different time references and commands are performed on a group, or not at all.

Consistent Operations – Limits

- SnapView Consistent Sessions
 - CX600/700 – 16 Source LUNs
 - CX300/400/500 – 8 Source LUNs
 - Counts as one of the 8 Sessions per Source LUN allowed
- SnapView Clones Consistent Fracture
 - CX600/700 – 16 Clone LUNs
 - CX300/400/500 – 8 Clone LUNs
 - Set cannot include more than 1 Clone for any given Source
- All limits are enforced by the array
- Not supported on AX100 or FC4700

This slide shows the current limits for SnapView Consistent Sessions and Consistent Fractures.

SnapView Clones – Consistent Fracture

- **Fracturing Clones consistently**
 - Associated source LUN must be unique for each clone specified
 - User cannot pick multiple clones for same source LUN
 - Fractured Clones will appear as “Administratively Fractured” in the Clone’s properties
 - User cannot consistently fracture a set of Clone LUNs if one of them is already fractured (Admin or System)
 - If the clone is synchronizing, it will be Out-Of-Sync, which is allowed but may not be a desirable data state
 - If the clone is reverse-synchronizing, it will be Reverse-Out-Of-Sync, which is allowed but may not be a desirable data state
- **No group association maintained for the set of Clone LUNs after fracture completes**
- **If a failure occurs during consistent fracture:**
 - Info provided to determine which clone failed and why
 - Clones fractured to this point will be queued to resync
 - If the clone was in the midst of reverse-sync’ing, it will be queued to resume the reverse sync

A consistent fracture is when you fracture more than one clone at the same time in order to preserve the point-in-time restartable copy across the set of clones. The SnapView driver will delay any I/O requests to the source LUNs of the selected clones until the fracture has completed on all the clones (thus preserving the point-in-time restartable copy on the entire set of clones). A restartable copy is a data state having dependent write consistency and where all internal database/application control information is consistent with a Database Management System/application image. The clones you want to fracture must be within different Clone Groups. You cannot perform a consistent fracture between different Clone Groups.

If there is a failure on any of the clones, the consistent fracture will fail on all of the clones. If any clones within the group were fractured prior to the failure, the software will re-synchronize those clones.

Consistent fracture is supported on CX-Series storage systems only. If you have a CX600 or CX700 storage system, you can fracture up to 16 clones at the same time. If you have another supported CX-Series storage system, you can only fracture up to 8 clones at the same time. A maximum of 32 consistent fracture operations can be in progress simultaneously per storage system.

If you consistent fracture while synchronizing, you will be Out-Of-Sync, which is allowed but may not be a desirable data state.

SnapView Sessions – Consistent Start

- **Starting Consistent Sessions**
 - “Consistent” is just an attribute of Snap session
 - No conversion from consistent to non-consistent or visa-versa
 - Session name uniquely identifies consistent session on array
 - Cannot be started if session name already exists on the array
 - Cannot add Source LUNs to consistent session after it has started
 - Non-consistent session can add more LUNs after session has started
 - Can issue “consistent start” on session with one Source LUN
 - May be protection from having other LUNs added to the session
- **All other session functionality same as SnapView sessions pre-Saturn**
 - Counts as one of the 8 Sessions per Source LUN allowed
- **If a failure occurs during consistent start:**
 - Info provided to determine which source failed and why
 - Session will be stopped

A consistent session name cannot already exist on the array (for either consistent or non-consistent sessions). Likewise, a non-consistent session cannot use the same name as a currently running consistent session. If a session is already running, the user will receive an error when trying to start consistent session and an already-started session will not be stopped.

SnapView Consistent Start Limitations and Restrictions

- Cannot perform other operations on session while the Consistent Start is in progress, including:
 - Administrative Stop of the session
 - Rollback of the session
 - Activation of any snapshots against the session
- Cannot perform a Consistent Start of a session on a Source LUN currently involved in another consistent operation
 - MirrorView/A – performs an internal consistent mark operation which could interfere with the consistent start.
 - Once the Consistent Mark is complete the Consistent Start is allowed.
 - Another Consistent Start on the same LUN – once the Consistent Start is completed the next Consistent Start is allowed.
 - *Does NOT interfere with Clones Consistent Fracture code*

You cannot perform a Administrative Stop of the session while the Consistent Start is in progress:

- Non-Administrative Stops (cache full, cache errors, etc) are queued up and the session will stop after the Consistent Start finishes.
- Under certain conditions, the Consistent Start will fail instead and perform a stop; thus causing the Administrative stop to fail.



SnapView Foundations

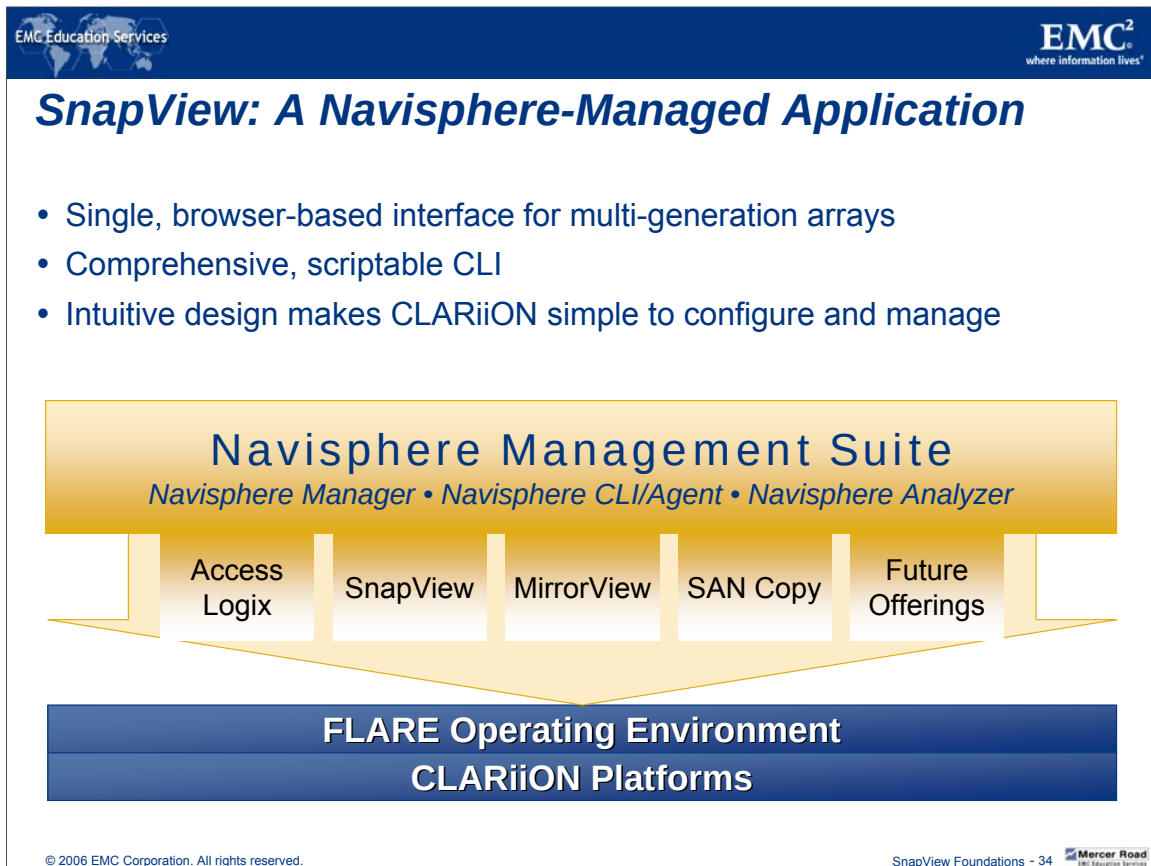
MANAGEMENT OPTIONS



© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 33 

Let's now turn to management options with SnapView.



This slide graphically represents the CLARiiON software family.

The most important thing to notice is that all functionality is managed via the Navisphere Management Suite, and all advanced operations are carried down to the hardware family via the FLARE Operating Environment.

Navisphere Manager is the single management interface to all CLARiiON storage system functionality.

FLARE performs advanced RAID algorithms, disk-scrubbing technologies, and LUN expansion (metaLUNs) to name a few of the many things FLARE is capable of doing.



SnapView Foundations

ENVIRONMENT INTEGRATION



© 2006 EMC Corporation. All rights reserved.

SnapView Foundations - 35 

This section discusses integration of SnapView in an environment.

SnapView Application Integration

- SnapView offers Application Integration Modules for:
 - MS Exchange (RMSE)
 - RMSE supports Exchange 2000, 2003 and 5.5 on W2K
 - RMSE supports Exchange 2003 on the W2K3 platform
 - Requires one CLARiiON array and two servers
 - Uses Clones (and Snapshots) only - there is no MirrorView support
 - SQL Server (RMSE)
 - GUI and CLI allows validation and scheduling
 - SQL Server 2000 on Windows 2000, 2003
 - Uses MS VDI (Virtual Device Interface) to perform online cloning and snapshots

RMSE (Replication Manager Standard Edition) is EMC's second generation (SnapView Integration Module for Exchange was the first). RMSE builds on our experience with a more comprehensive product offering. RMSE allows the creation of hot splits of Exchange and SQL Server databases and volumes. It provides Rapid Recovery when the database experiences corruption. It also allows for larger mailboxes with no disruption to the database. Additionally, RMSE can use both Full SAN Copy and Incremental SAN Copy technology for data migration. Replication types are listed below.

- Snapshots only
- Clones only
- Clones with Snapshots

SnapView Application Example: Exchange Backup and Recovery

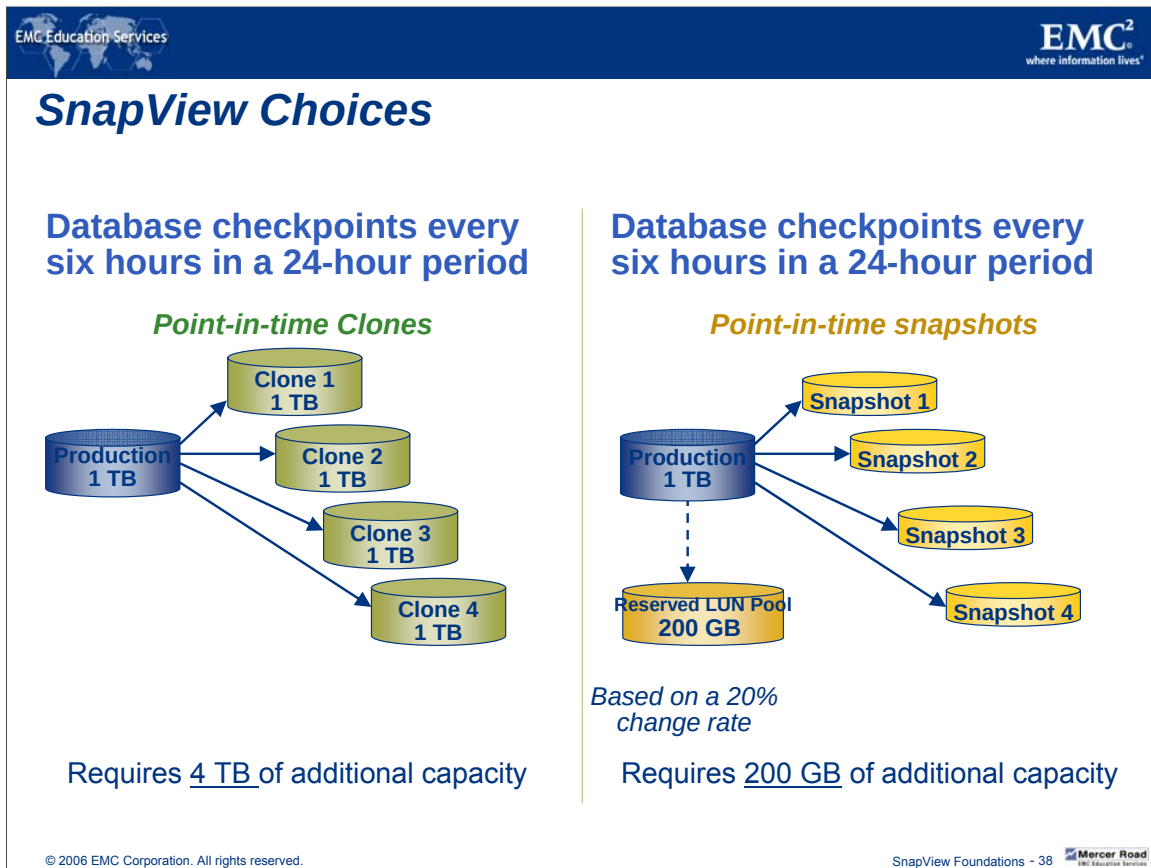
- **Simplified, easy-to-use backup and recovery**
 - Designed for Exchange Administrator's use
 - Easy-to-use scheduler for automated backups
- **Faster, reliable recovery**
 - Leverages SnapView instant restore from RAID-protected Clones
- **Faster, reliable backup**
 - Backup any time needed from snapshot
 - Clone "hot split" technology coupled with automated Microsoft corruption check
- **Enables Exchange consolidation**
 - Backup and recovery times no longer bottleneck to database growth

Most servers today have the power to handle many more users. So, if you can manage to recover a larger database within your allotted recovery window, then you can save costs by consolidating Exchange users onto fewer machines. RMSE for Exchange product is one way to use SnapView to help lower costs for your business.

RMSE integration makes it easy to create disk-based replicas (Clones) of Exchange databases during normal business hours and run backup at your leisure. Server cycles are restored to Exchange servers, allowing faster responses for Exchange users.

Restoring Exchange mailboxes from a disk-based replica using SnapView is much faster than utilizing tape to restore.

EMC's RMSE solution provides a simplified way to actually scan the Exchange server's system log to check for Exchange database corruption, and it also runs an Exchange-supplied corruption utility to ensure there are no "torn pages" on the Clone that would make the database unrecoverable or corrupt. This ensures that the database is valid prior to backup or restore. Other vendors consider this as an option, but this is mandatory for EMC's method.



In order to improve data integrity and reduce recovery time for critical applications, many users create multiple database checkpoints during a given period of time. To maintain application availability and meet service level requirements, a point-in-time copy (such as a SnapView Clone) can be non-disruptively created from the source volumes, and used to recover the database in the event of a database failure or database corruption.

Creating a checkpoint of the database every six hours would require making four copies every 24 hours; therefore, creating four point-in-time copies per day of a 1 TB database would require an additional 4 TB of capacity.

To reduce the amount of capacity required to create the database checkpoints, a logical point-in-time view can be created instead of a full volume copy. When creating a point-in-time view of a source volume, only a fraction of the source volume is required. The capacity required to create a logical point-in-time view depends on how often the data is changed on the source volume after the view has been created (or “snapped”). So in this example, if 20% of the data changes every 24 hours, only 200 GB (1 TB x 20% change) is required to create the same number of database checkpoints.

This capability lowers the TCO required to create the multiple database checkpoint by requiring less capacity. It also can increase the number of checkpoints created during a 24-hour period by requiring only a fraction of the capacity compared to a full volume copy, thus increasing data integrity and improving recoverability.

Module Summary

Key points covered in this module:

- Functional concepts of SnapView on the CLARiiON Storage Platform
- Benefits of SnapView on the CLARiiON Storage Platform
- Differences between the Local Replication Solutions available in SnapView

These are the key points covered in this training. Please take a moment to review them.