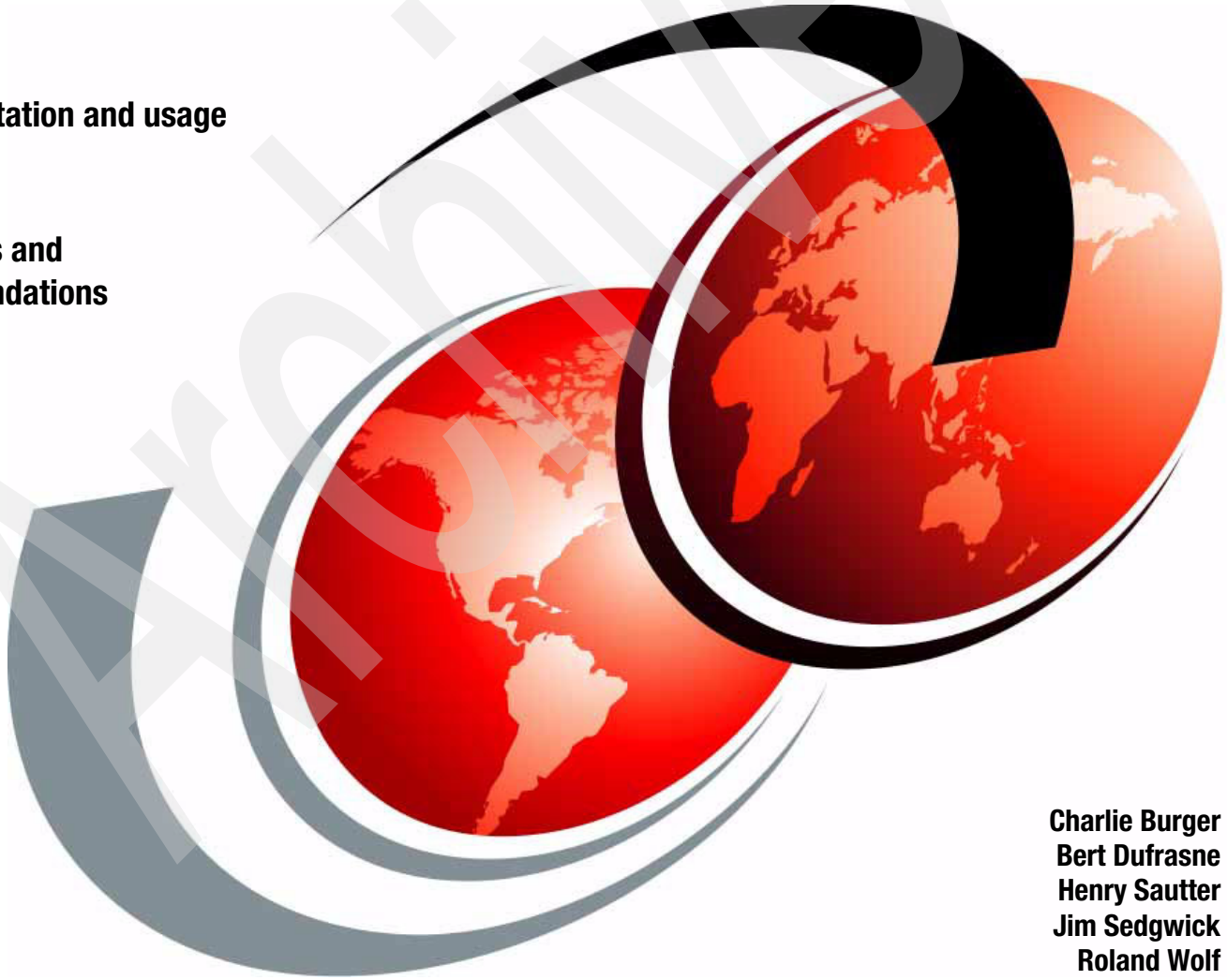
IBM System Storage DS8000 Series: IBM FlashCopy SE

- Concept and design
- Implementation and usage examples
- Guidelines and recommendations



Charlie Burger
Bert Dufrasne
Henry Sautter
Jim Sedgwick
Roland Wolf

Redpaper

ibm.com/redbooks

Archived

Archived

International Technical Support Organization

IBM System Storage DS8000 Series: IBM FlashCopy SE

February 2008

Archived

REDP-4368-00

Note: Before using this information and the product it supports, read the information in “Notices” on page v.

Archived

First Edition (February 2008)
This edition applies to the DS8000 R3 as announced October 23,2007.

© Copyright International Business Machines Corporation 2008. All rights reserved.
Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 2008. All rights reserved.

Archived

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

© Copyright IBM Corp. 2008. All rights reserved.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

DB2®
DFSMS™
DFSMSdss™
DS6000™
DS8000™
FlashCopy®
IBM®
MVS™
Redbooks®
Redbooks (logo) ®
RACF®
RMF™
S/390®
System z™
System Storage™
TotalStorage®
z/OS®
z/VSE™
zSeries®

Other company, product, or service names may be trademarks or service marks of others.

Archived

Preface

In October 2007, IBM® announced a space efficient FlashCopy® capability for the DS8000™ platform. We call this new function IBM FlashCopy SE. It is a licensed function of the DS8000 family. It does not replace, but rather complements the standard FlashCopy function.

In this Redpaper, we start with a general overview of FlashCopy SE, where we discuss its concepts and intended usage. We summarize the concepts of the standard FlashCopy function and review its different options in the context of FlashCopy SE. We also have a detailed discussion of the FlashCopy SE design and implementation. As an added benefit, we provide guidance for the definition and usage of FlashCopy SE from both an open system and z/OS® environments and provide illustrations using the DS GUI or DSCLI interfaces.

The team that wrote this paper

A team of specialists from around the world working at the International Technical Support Organization, San Jose Center produced this paper.

Charlie Burger is a Certified Consulting IT Specialist with Advanced Technical Support in the United States (US). Charlie started his IBM career in Pittsburgh, Pa in 1978 and worked as a large systems Customer Engineer (CE) and then as Systems Support Representative for the Midwest Area Systems Center (ASC) until 1991. His next assignment was to write MVS™ education courses for EMEA until 1994. He then moved to San Jose to support DFSMS™, specifically VSAM and catalogs. Recently, Charlie's primary focus is on Advanced Copy Services supporting the ESS, DS6000™, and DS8000.

Bert Dufrasne is an IBM Certified IT Specialist and Project Leader for System Storage™ disk products at the International Technical Support Organization, San Jose Center. He has worked at IBM in various IT areas. He has written many IBM Redbooks® and has also developed and taught technical workshops. Before joining the ITSO, he worked for IBM Global Services as an Application Architect. He holds a degree in Electrical Engineering.

Henry Sautter is a Consulting IT Specialist with Advanced Technical Support in the US. He has 15 years of experience with S/390® and IBM disk storage hardware and Advanced Copy Services functions while working in Tucson Arizona. His previous 13 years of experience include IBM Processor microcode development and S/390 system testing while working in Poughkeepsie, NY. He has worked at IBM for 28 years. Hank's areas of expertise include enterprise storage performance and disaster recovery implementation for large systems and open systems. He writes and presents on these topics. He holds a BS degree in Physics.

Jim Sedgwick is an IT Specialist in the US. He has 20 years of experience in the storage industry. He spent five years with IBM as a Printer Design Engineer after receiving his Mechanical Engineering degree from NCSU. Jim's current area of expertise includes enterprise storage performance and copy services. He writes and presents on both subjects.

Roland Wolf is a Certified Consulting IT Specialist in Germany. He has 21 years of experience with S/390 and disk storage hardware and also in SAN and storage for open systems. He is working in Field Technical Sales Support for storage systems. His areas of expertise include performance analysis and disaster recovery solutions in enterprises utilizing the unique capabilities and features of the IBM disk storage servers, ESS, and DS6000/DS8000. He contributes to various IBM Redbooks publications including, ESS,

© Copyright IBM Corp. 2008. All rights reserved.

DS6000, and DS80000 Concepts and Architecture, and DS6000 / DS8000 Copy Services. He holds a PhD in Theoretical Physics.

Thanks to the following people for their contributions to this project:

Deanna Polm
International Technical Support Organization, San Jose Center

John Bynum

Gail Spear, Lee LaFrese, Peter Kimmel, Jana Jamsek, Sony Williams, Rosemary McCutchen, Craig Gordon, Ingrid Stey, Werner Bauer. Lisa Gundy, Sony Williams, Kevin Gibble, Stephen West, Alison Pate, Brian Sherman, Chip Jarvis.

Become a published author

Join us for a two-to-six week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Archived

1

IBM FlashCopy SE

This chapter is a high level introduction to the concept and design of IBM FlashCopy SE and a discussion of its intended usage.

Archived

© Copyright IBM Corp. 2008. All rights reserved.

1.1 IBM FlashCopy SE overview

IBM FlashCopy SE is functionally not very different from the standard FlashCopy. The concept of space efficient with IBM FlashCopy SE relates to the attributes or properties of a DS8000 volume. As such, a space efficient volume could be used as any other DS8000 volume. However, the intended and only recommended use is as target volume in a FlashCopy relationship.

When a normal volume is created, it occupies the defined capacity on the physical drives. A space efficient volume does not occupy physical capacity when it is initially created. Space gets allocated when data is actually written to the volume, which allows the FlashCopy target volume capacity to be thinly provisioned (in other words, smaller than the full capacity of the source volume). Therefore, when you plan for FlashCopy you can provision less disk capacity when using FlashCopy SE than when using standard FlashCopy, which can help lower the amount of physical storage that is needed by many installations.

Consider the common practice of creating a FlashCopy relationship (without background copy) to spin data off to tape, for example. In this instance, although the backup application is reading data from the FlashCopy target volume and sending it to tape, no data is actually copied to the FlashCopy target volume. Data is being read from the FlashCopy source volume and is being sent directly to tape. Because the standard FlashCopy target volume is fully provisioned (size is equal to or greater than that of the FlashCopy source volume), we could argue that the FlashCopy target volume is a waste of valuable storage resources. FlashCopy SE was created to address exactly this kind of situation. FlashCopy SE gives you the capability to create a space efficient volume as the FlashCopy target.

As far as the FlashCopy source-target relationship is concerned, the FlashCopy SE target volume acts like a standard (fully provisioned) FlashCopy target volume, but implies that a full background copy is not in effect (the nocopy option is enforced), which makes sense because, if you have requirements for a full background copy, you have no intended need for space efficiency.

FlashCopy SE accomplishes space efficiency by pooling the physical storage requirements of many FlashCopy SE volumes into a common *repository*. A mapping structure is created to keep track of where the FlashCopy SE volume's data is physically located within the repository.

Refer to Figure 1-1 on page 3, as we take a closer look at the architecture of FlashCopy SE. The FlashCopy SE target volume itself is a *virtual* volume and has no actual storage allocation, which is why it appears as transparent in the picture.

However, FlashCopy SE is not “free” storage. The repository is an object within an extent pool. In some sense, the repository is similar to a fully provisioned volume within the extent pool. The repository has a physical size and a logical size. The physical size of the repository is the amount of space that is allocated in the extent pool. The intended overall benefit of FlashCopy SE is that the repository is smaller than the full capacity of the FlashCopy source volumes.

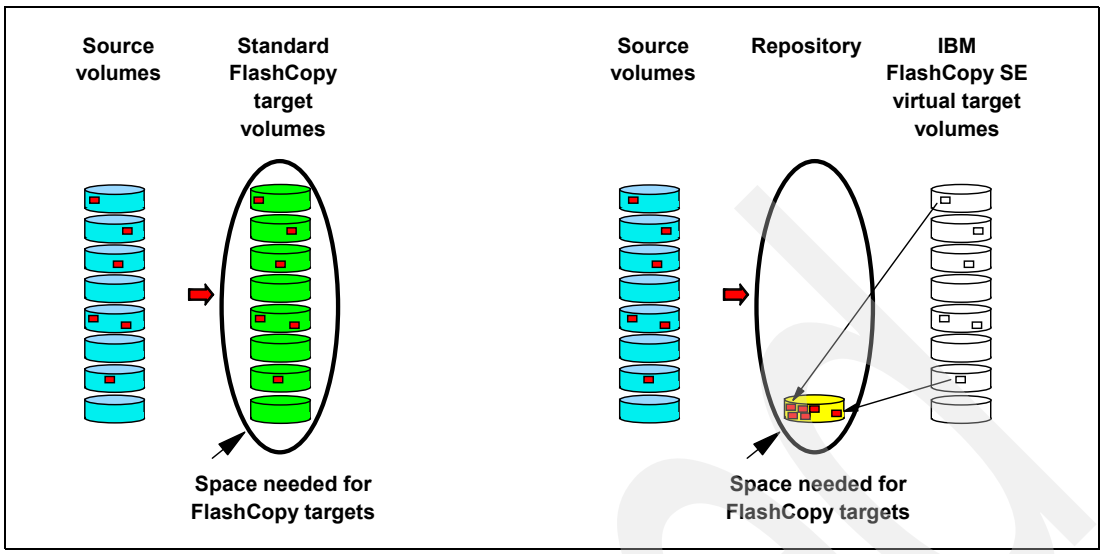


Figure 1-1 IBM FlashCopy SE concept

The repository provides the physical space requirements for space efficient volumes. As tracks are destaged for space efficient volumes, storage for the tracks is obtained from the segments that are assigned in the repository. The data for a space efficient volume is stored in the repository, but it is only accessible from the space efficient volume. The host operating systems do not have access to the repository.

Important: The host operating systems do not have access to the repository.

It is important to understand the trade-offs and performance implications to decide where implementing FlashCopy SE is advantageous and where you should avoid it. Standard FlashCopy generally has superior performance to FlashCopy SE. We discuss performance in more details in 3.3, “Performance planning for FlashCopy SE” on page 20.

See also 3.5, “Guidelines and recommendations” on page 24.

Note: You can order the FlashCopy SE feature for any IBM System Storage DS8000 series and require DS8000 Licensed Machine Code (LMC) level 5.3.0xx.xx (bundle version 63.0.xx.xx), or later.

Keep an eye on the following Techdocs flash for up-to-date FlashCopy SE implementation considerations and recommendations:

http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10617

Archived

2

FlashCopy concepts and terminology

In this chapter, we review the DS8000 FlashCopy function and a describe the different features (copy and nocopy, incremental, multiple targets). In parallel, we also point out what is changing or different in the context of FlashCopy SE.

In this chapter, we discuss:

- What is FlashCopy
- Background COPY and NOCOPY
- Reads and writes during FlashCopy relationship
- FlashCopy options

Archived

© Copyright IBM Corp. 2008. All rights reserved.

2.1 What is FlashCopy

FlashCopy creates an instant copy of a volume at a specific point-in-time, which is why it is often referred to as Point-in-Time Copy, instantaneous copy or time zero (t0) copy. FlashCopy is a hardware (storage system based) function that the software invokes.

By doing a FlashCopy, a relationship is established between a *source* and a *target*. Both are considered to form a FlashCopy *pair*.

After you issue a FlashCopy command, a FlashCopy relationship is created between the source and target volumes and they become available for processing with full read/write access. Typically, large applications, such as databases, have their data spread across several volumes, and all of their volumes with dependent data should be FlashCopied at exactly the same point-in-time or FlashCopied while maintaining the order of their dependent writes. FlashCopy offers *Consistency Groups*, which allows you to FlashCopy multiple volumes while maintaining the order of the volumes' dependent writes. It takes only a few seconds to establish relationships for tens to hundred or more volume pairs.

Two variations of FlashCopy are available:

- Standard FlashCopy

  Standard FlashCopy uses a normal volume as target volume. This target volume has to have the same size (or larger) as the source volume, and that space is allocated in the storage subsystem.

- IBM FlashCopy SE

  IBM FlashCopy SE uses space efficient volumes as FlashCopy target volumes. A space efficient volume has a virtual size that is equal to the source volume size. However, space is not allocated for this volume at the beginning when the volume is created and the FlashCopy is initiated; instead, space is allocated in the repository when a first update is made to original tracks on the source volumes and those tracks are copied to the FlashCopy SE target volume. Writes to the SE target also consume repository space.

FlashCopy and FlashCopy SE are optional and distinct licensed features of the IBM DS8000. Both features can coexist on a DS8000.

FlashCopy can be invoked by:

- DS CLI
- DS Storage Manager GUI
- TPC for Replication
- DFSMSdss™
- TSO
- ICKDSF
- z/OS API

Some of the reasons for using FlashCopy to make copies of data are:

- Backup processing
- Data mining
- Creating an environment for testing
- Creating an environment for development
- Creating data for reporting
- Archiving

Archived

2.1.1 Basic concepts

The following are basic characteristics of the FlashCopy operation:

- Establish FlashCopy relationship

  When you start FlashCopy, the relationship between source and target is established by creating a pointer table, including a bitmap for the target. This process is completed very quickly and makes the copy appear to be instantaneous.

  While the FlashCopy relationship is being created, the DS8000 holds off I/O activity to the volume by putting the source volume in a SCSI *queue full* state (open systems) during which, any new I/Os that are issued receive a queue full error and are automatically reissued by the host bus adapter. For z/OS, the system is placed in an *extended long busy* condition, where no user disruption or intervention is required. I/O activity resumes when the FlashCopy is established.

  If all bits for the bitmap of the target are set to their initial values, this means that no data block was copied so far. The data in the target is not modified during bitmap set up. At this first step, the bitmap and the data look as illustrated in Figure 2-1.

  The target volume in Figure 2-1 and Figure 2-2 on page 8 can be a normal volume or a space efficient volume. In both cases the logic is the same.

  The difference between standard FlashCopy and FlashCopy SE is where the physical storage resides. For standard FlashCopy it is a normal volume, for IBM FlashCopy it is a repository (see Figure 2-5 on page 10).

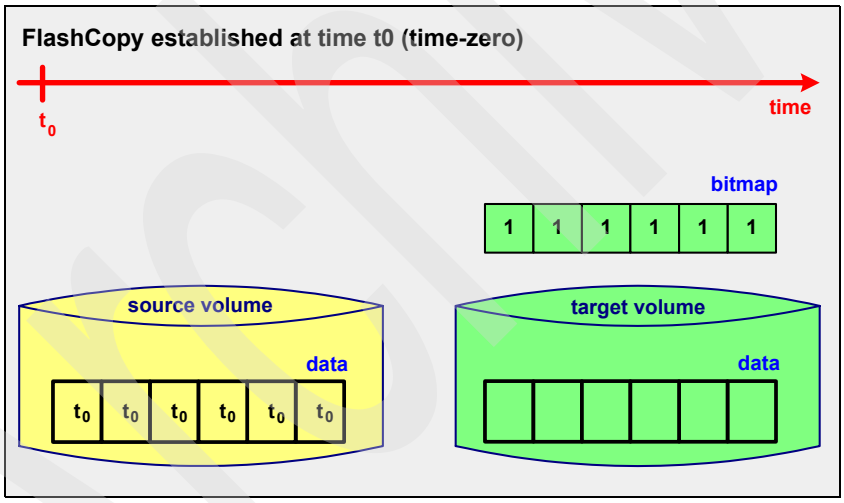


Figure 2-1 FlashCopy at time t0

Background COPY and NOCOPY

As result of the FlashCopy, either all physical blocks from the source volume are copied (full copy), or — when using the *nocopy* option — only those parts that are changing in the source data since the FlashCopy was established are copied.

The specification of background copy or nocopy has no effect upon when the target volume is used for reads or writes.

With the standard FlashCopy, background copy is the default, while nocopy is now enforced with FlashCopy SE.

Background copy

When you invoke the *copy* option, and the establish process completes, a background process is started that copies all data from the source to the target. After this process is finished, and if there were no updates on the target, the picture we get is similar to the Figure 2-2.

If not explicitly defined as *persistent*, the FlashCopy relationship ends as soon as all data is copied. If you specified the *persistent* FlashCopy option, the FlashCopy relationship must be withdrawn explicitly.

Only the standard FlashCopy allows a full background copy. FlashCopy SE has no such function, but remember that both features can coexist.

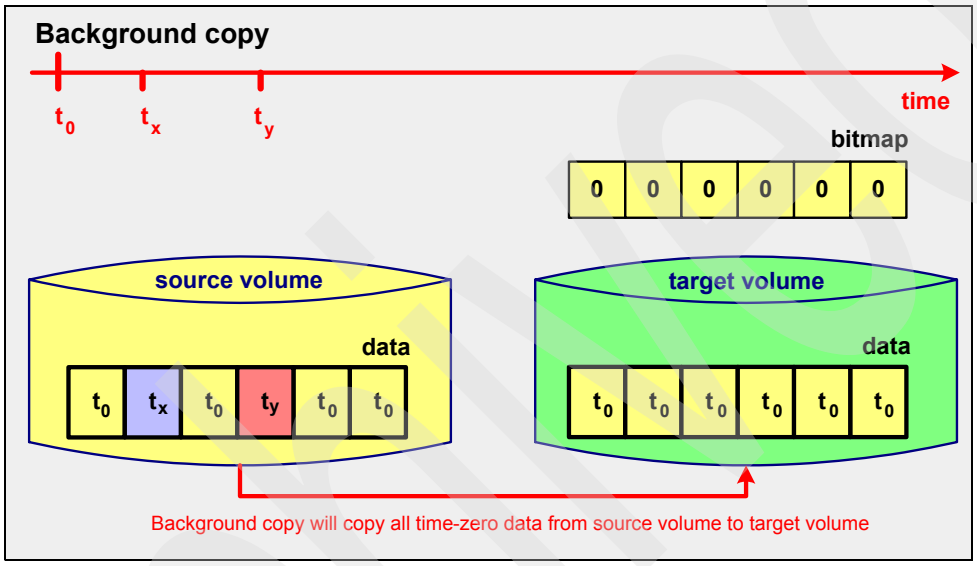


Figure 2-2 Target volume after full volume FlashCopy relationship finished

If there are writes to the target, then the picture we get is similar to Figure 2-3.

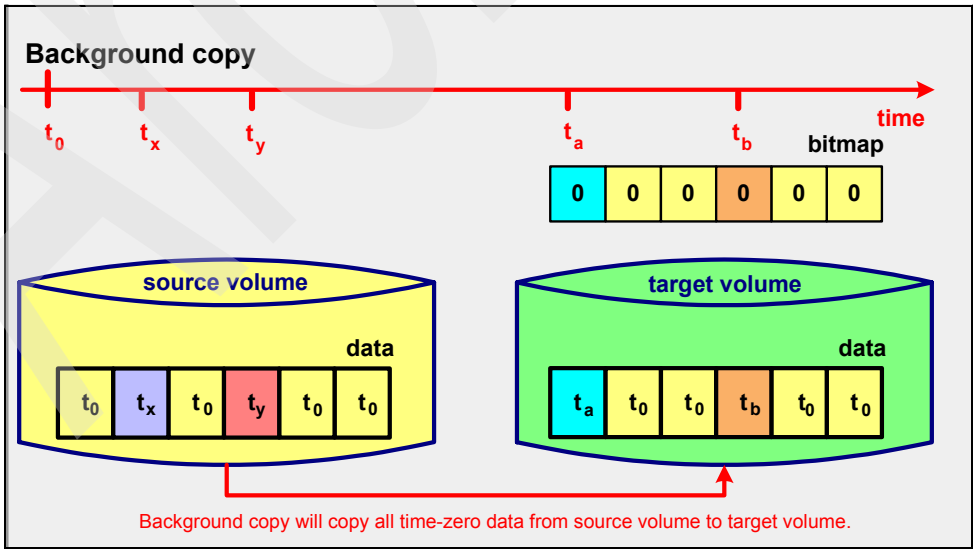


Figure 2-3 FlashCopy after updates to the target volume

Nocopy option

If FlashCopy is established using the *nocopy* option, the result is as shown in Figure 2-4 and Figure 2-6 on page 11. The relationship lasts until it is explicitly withdrawn or until all data in the source volume is modified. Blocks for which no write occurred on the source or on the target stays as they were at the time when the FlashCopy was established.

The *nocopy* option is default for FlashCopy SE when using the DSCLI (for DFSMSdss you have to specify FCNOCOPY or MODE(NOCOPY) for TSO). The FlashCopy is a set of tracks that can consist of an entire volume or just a selected set of tracks. Attempts to read/write data already copied proceed as normal while attempts to process data not yet copied is intercepted and “Copy on Demand” or read from source is performed as needed. All of this processing creates the effect of an instant copy.

Note FlashCopy SE must always be a **`nocopy`** relationship.

Reads and writes during the FlashCopy relationship

After the relationship is established, it is possible to perform read and write I/Os on both the source and the target. Assuming that the target is used for reads only while production is ongoing, things will look as illustrated in Figure 2-4.

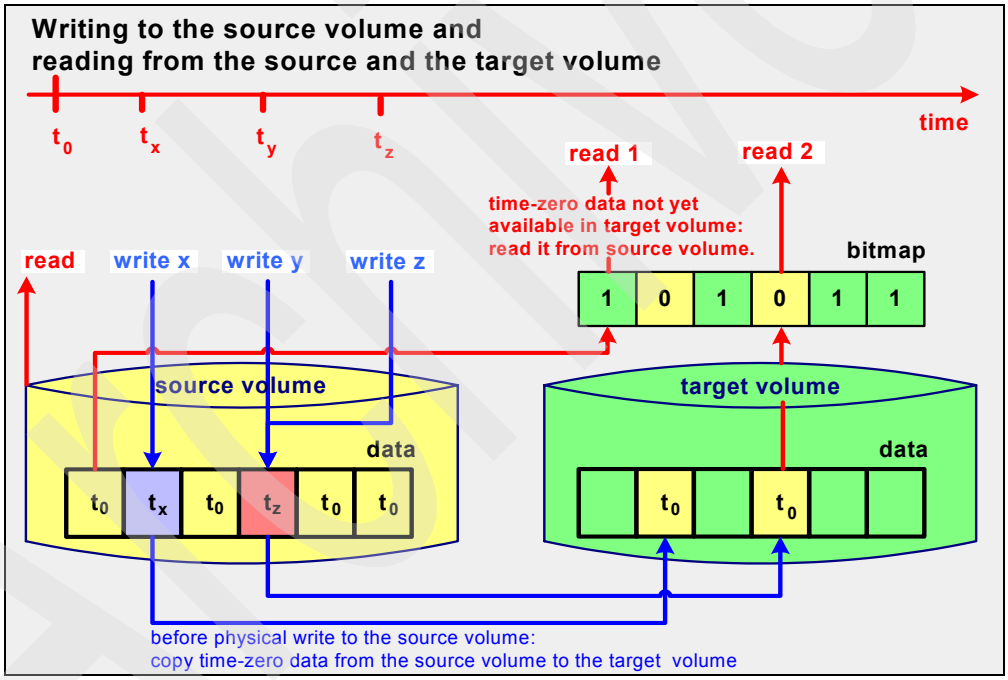


Figure 2-4 Reads from source and target volumes and writes to source volume

- Reading from the source

  The data is read immediately. See Figure 2-4 or Figure 2-5 on page 10.

- Writing to the source

  Whenever data is written to the source volume while the FlashCopy relationship exists, the storage subsystem makes sure that the time-zero-data is copied to the target volume prior to overwriting it in the source volume. When the target volume is a space efficient volume, the data is actually written to a repository (Figure 2-5 on page 10).

  To determine if the data of the physical track on the source volume needs to be copied to the target volume, the bitmap is analyzed. If the bitmap identifies that the time-zero data is

not available on the target volume, the data is copied from source to target. If it states that the time-zero data was already copied to the target volume, no further action is taken. See Figure 2-4 on page 9 or Figure 2-5.

It is possible to use the target volume immediately, for reading data and also for writing data. Figure 2-5 shows reads and writes for FlashCopy SE.

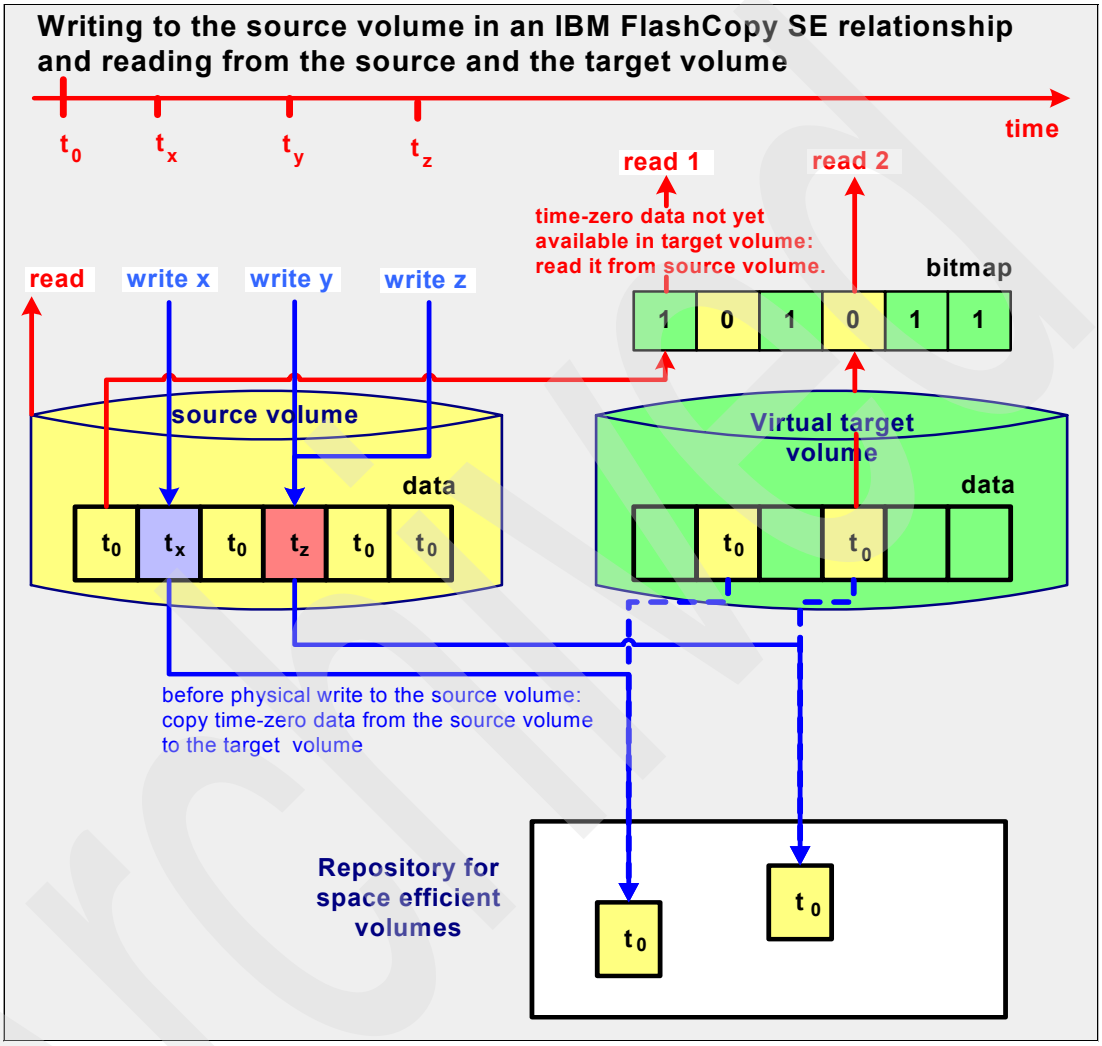


Figure 2-5 Reads from source and target volumes and writes to source volume for FlashCopy SE relations

The following list explains the various reads and writes to and from the source and the target:

- Reading from the target

  Whenever a read-request goes to the target while the FlashCopy relationship exists, the bitmap identifies if the data must be retrieved from the source or from the target. If the bitmap states that the time-zero data has not yet been copied to the target, then the physical read is directed to the source. If the time-zero data was already copied to the target, then the read is performed immediately against the target. See Figure 2-4 on page 9 or Figure 2-5.

- Writing to the target

  When data is written to the target volume while the FlashCopy relationship exists, the storage subsystem makes sure that the bitmap is updated, which ensures that the time-zero data from the source volume never overwrites updates that were done directly to the target volume. Figure 2-6 illustrates writing to the target volume.

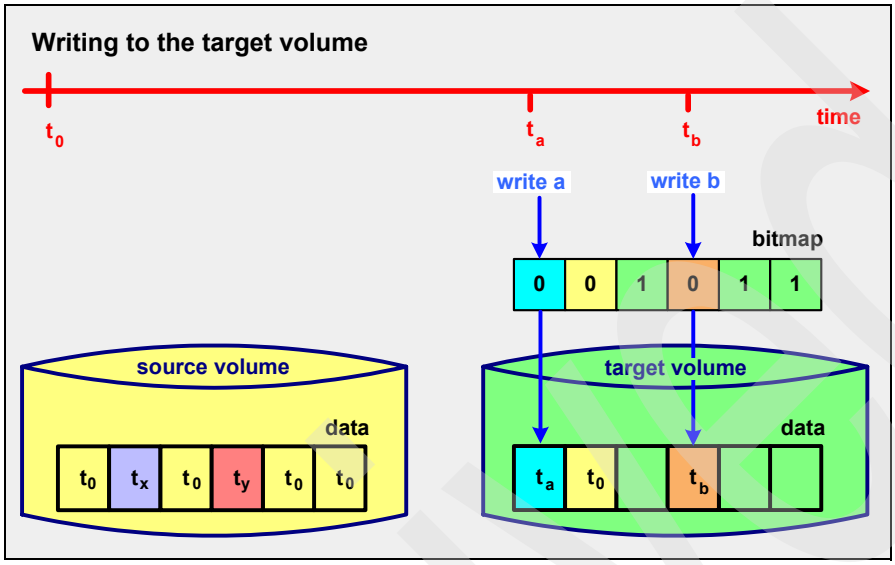


Figure 2-6 Writes to target volume

- Terminating the FlashCopy relationship

  The FlashCopy relationship is *automatically ended* when all tracks are copied from the source volume to the target volume. The relationship can also be *explicitly withdrawn* by issuing the corresponding commands. If the *persistent* FlashCopy option or nocopy was specified, then the FlashCopy relationship must be withdrawn explicitly.

  A FlashCopy SE relationship ends when it is withdrawn. When the relationship is withdrawn, there is an option to release the allocated space of the space efficient volume.

2.2 FlashCopy options

You can perform a FlashCopy at the volume level, on a data set (z/OS and z/VSE™ only) or on a selected set of tracks.

Restriction: FlashCopy SE does not support data set level FlashCopy.

We also explained that you can specify FlashCopy with either background copy or nocopy. There are additional options available when invoking FlashCopy:

- Switching from nocopy to background copy
- Persistent relationship
- Multiple relationships
- Incremental FlashCopy
- Consistency group FlashCopy
- Inband FlashCopy
- Fast Reverse Restore enabled FlashCopy
- Performing Fast Reverse Restore

2.2.1 Nocopy to background copy

If you have a FlashCopy relationship that was invoked with nocopy, or if a background copy has not yet completed, the target volume becomes unusable if the relationship is withdrawn before all of the source tracks are copied to the target. If you want to have the target volume usable after the relationship is withdrawn, you can convert the nocopy to background copy. After the background copy completes, the relationship is withdrawn and the target volume is usable. You may want to do this, for example, if you wanted to FlashCopy the target volume (a FlashCopy target cannot be a FlashCopy source at the same time).

FlashCopy SE: It is not possible to convert a nocopy to a background copy when the target is a space efficient volume. Any attempts to do so, fail.

2.2.2 Persistent FlashCopy

If you have a background copy and the background copy is completed, the relationship is withdrawn. There is no way to query the source volume to see what the last FlashCopy target was if the relationship is withdrawn.

If you specify that the FlashCopy is persistent, the relationship is maintained even after the background copy completes. The relationship is then manually withdrawn.

You may want to consider using persistent if you have procedures where you FlashCopy a source volume to a target one night and then FlashCopy it to a different target the following night. With persistent FlashCopy, you can query the source to see where it was last flashed, withdraw the relationship, and then flash to a different target, which lessens the chance of accidentally flashing to the wrong target.

FlashCopy SE: FlashCopy SE supports the persistent option.

2.2.3 Multiple relationships

A single source volume can have up to 12 targets of which only one can be incremental.

FlashCopy SE: FlashCopy SE supports multiple relationships. However, having too many targets can affect performance. For up-to-date information and recommendations, refer to the following Web page:
http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10617

2.2.4 Incremental FlashCopy

Incremental FlashCopy provides the capability to only background copy those tracks that changed since the last increment was taken, which reduces the amount of data to process in background copy and thus the amount of time it takes for the background copy to complete. When the FlashCopy relationship is established, the change data recording (which also turns on the persistent flag), and background copy parameters are used. An incremental flash can be performed in either direction.

A bitmap is used to track changes to both the source and target volumes. After the initial incremental FlashCopy is established, which does a background copy on the entire source on the first FlashCopy, the next increment will only background copy those tracks that were

updated since the last increment. It also background copies those tracks that were updated on the target and need to be overlaid. To maintain the incremental relationship, specify change recording with each increment. If you do not specify change recording, the relationship is withdrawn after the incremental background copy completes.

If you did an incremental FlashCopy A > B, you do not have to wait for the background copy to complete before you issue another increment A > B. However, if you did an increment A > B and now want to do an increment B > A, you must wait until the A > B background copy completes.

2.2.5 Consistency Group FlashCopy

FlashCopy is often used to make copies of data that crosses the volume boundary. In cases where the data is dependent upon each other, the data that crosses boundaries needs to be consistent. That is, the order of the dependent writes must be maintained. *FlashCopy Consistency Group* provides a mechanism for achieving a consistent data copy across multiple volumes without requiring that the application I/O be quiesced.

In the case of production data, application impact must be minimized. Prior to Consistency Group FlashCopy, you have to first quiesce the application, or in the case of DB2®, use the **SET LOG** commands, establish their FlashCopy relationships, and then restart the application. This process is disruptive and causes application outages or data unavailability for an unacceptable period of time.

You can specify Consistency Group for FlashCopy by using the *freeze* option, which holds off initiation and completion of write I/O to the source volumes until a **thaw** command is issued or, by default, when two minutes have passed.

The FlashCopy command and FlashCopy FREEZE is on a volume basis, but the THAW command is at the LSS level, which means that if there were more than one set of volumes using consistency, the **thaw** command would affect all of the Consistency Groups.

The target of each source volume is within one physical disk subsystem, but source volumes within a Consistency Group can span physical disk subsystems.

FlashCopy SE: You can invoke Consistency Group FlashCopy using nocopy, so you can use it with FlashCopy SE.

2.2.6 Inband FlashCopy

Inband FlashCopy allows FlashCopy requests to be issued remotely through an existing PPRC link. Inband FlashCopy is useful if the host at the recovery site is not online. The Inband option eliminates the need for a host connection from the local site to the remote subsystem for FlashCopy backup.

The FlashCopy request must be issued at a host processor that is connected to the PPRC primary volume, with the PPRC secondary volume specified as the FlashCopy source.

You can issue all supported full-volume FlashCopy commands with the Inband option, except the THAW portion of Consistency Group processing.

Fast Reverse Restore is also not supported with Inband because you need to issue the command to the remote FlashCopy target volume (not the remote FlashCopy source).

FlashCopy SE: An Inband FlashCopy can be a FlashCopy SE.

2.2.7 Fast Reverse Restore Enabled FlashCopy

A *Fast Reverse Restore Enabled FlashCopy* is a FlashCopy relationship that is established with attributes that allow a Fast Reverse Restore to be performed upon the FlashCopy pair. The attributes are:

- Change Recording
- Persistent (Change Recording invokes persistent automatically)
- Background nocopy

A Fast Reverse Restore Enabled FlashCopy relationship is most often found in Global Mirror environments, although Global Mirror is not required for Fast Reverse Restore.

FlashCopySE: A Fast Reverse Restore Enabled FlashCopy can be space efficient because this function does not specify background copy.

2.2.8 Fast Reverse Restore

A *Fast Reverse Restore* is performed upon a FlashCopy pair that is established as Fast Reverse Restore Enabled. The Fast Reverse Restore copies the target back to the source and physically copies from the target to the source those tracks that were previously physically copied to the target while the Fast Reverse Restore relationship was enabled.

A Fast Reverse Restore FlashCopy is most often done when you create consistent data after a planned or unplanned outage in a Global Mirror environment.

FlashCopy SE: You can use Fast Reverse Restore with FlashCopy SE.

The Fast Reverse Restore capability was enhanced with R3 microcode. In R3, you no longer need to have Change Recording specified on the relationship in order to do Fast Reverse Restore. It is possible to have multiple targets and use Fast Reverse Restore to restore any ONE of them. The one consideration for using Fast Reverse Restore on one of the relationships is that prior to the Fast Reverse Restore, all other targets must be removed. Therefore, you need to be careful about picking the correct relationship for the Fast Reverse Restore.

Archived

3

FlashCopy SE implementation and usage

FlashCopy SE is essentially based on the space efficient characteristics of the target volume. In this chapter, we explain how space efficient volumes were designed and implemented, including some considerations for recovery procedures in the event that the Repository becomes full. We also discuss capacity planning and performance considerations. We have a section where we illustrate the process of defining a set of space efficient volumes and describe changes to the DSCLI commands and DS GUI for space efficient volume support.

Archived

© Copyright IBM Corp. 2008. All rights reserved.

3.1 Space efficient volumes

IBM FlashCopy SE uses a new type of volume as FlashCopy target volumes: space efficient volumes.

When a normal volume is created it occupies the defined capacity on the physical drives. A space efficient volume does not occupy physical capacity when it is created. Space gets allocated when data is actually written to the volume. The amount of space that gets physically allocated is a function of the amount of data changes that are performed on a volume. The sum of all defined space efficient volumes can be larger than the physical capacity available. This function is also called *thin provisioning*.

Space efficient volumes can only be created when the FlashCopy SE licensed feature is installed on the DS8000. Space efficient volumes are seen by a server just like normal volumes. A server cannot see any difference.

Note: In the current implementation, space efficient volumes are supported as FlashCopy target volumes only.

The idea with space efficient volumes is to save storage when it is only temporarily needed, which is the case with FlashCopy when you use the *nocopy* option. This type of FlashCopy is typically used with the goal of taking a backup from the FlashCopy target volumes. Without the use of space efficient volumes, target volumes consume the same physical capacity as the source volumes. However, these target volumes were often nearly empty because space is only occupied on-demand, when a write to the source volume occurs. Only changed data is copied to the target volume. A space efficient volume only uses the space needed for updates to the source volume.

3.2 Repository for space efficient volumes

The definition of space efficient (SE) volumes begins at the extent pool level. SE volumes are defined from *virtual space* in that the size of the SE volume does not initially use physical storage. However any data written to an SE volume must have enough physical storage to contain this write activity. The repository provides this physical storage.

The repository is an object within an extent pool. It is similar to a volume within the *extent pool*. The repository has a physical size and a logical size. The physical size of the repository is the amount of space that is allocated in the extent pool. It is the physical space that is available for all space efficient volumes, in total, in this extent pool. The repository is striped across all ranks within the extent pool. There can only be one repository per extent pool.

Important: The size of the repository and virtual space is part of the extent pool definition. Each extent pool can have an SE volume repository, but this physical space cannot be shared between extent pools.

The logical size of the repository is the sum of virtual storage that is available for space efficient volumes. As an example, there could be a repository of 100 GB reserved physical storage, and you defined a logical capacity of 200 GB. In this case, you could define 10 LUNs with 20 GB each. So the logical capacity can be larger than the physical capacity. Of course, you cannot fill all the volumes with data because the total physical capacity is limited to 100 GB in this example.

Note: In the current implementation of space efficient volumes, it is not possible to expand the size of the repository, neither the physical capacity of the repository or the virtual capacity. Therefore, you must plan carefully for the size of the repository before you use it. If a repository needs to be expanded, you must delete all space efficient volumes within this extent pool. Then, delete and recreate the repository with a different size.

Space for a space efficient volume is allocated when a write occurs, more precisely, when a destage from the cache occurs. The allocation unit is a track, which is 64 KB for open systems LUNs or 57 KB for CKD volumes, which you must consider when you plan for the size of the repository. The amount of space that gets physically allocated might be larger than the amount of data that was written. If there are 100 random writes of, for example, 8 KB (800 KB in total), we probably allocate 6.4 MB (100 x 64 KB). If there are other writes changing data within these 6.4 MB, there will be no new allocations at all.

Because space is allocated in tracks, and the system needs to maintain tables where it places the physical track and how to map it to the logical volume, there is some overhead involved with space efficient volumes. The smaller the allocation unit the larger the tables and the overhead. The DS8000 has a fixed allocation unit of a track, which is a good compromise between processing overhead and allocation overhead.

Summary: Virtual space is created as part of the extent pool definition. This virtual space is mapped onto the repository (physical space) as needed. Virtual space equals the total space of the intended FlashCopy source volumes or space to contain the size of SE volumes that are intended to be used for other purposes. No actual storage is allocated until write activity occurs to the SE volumes.

Figure 3-1 illustrates the concept of space efficient volumes.

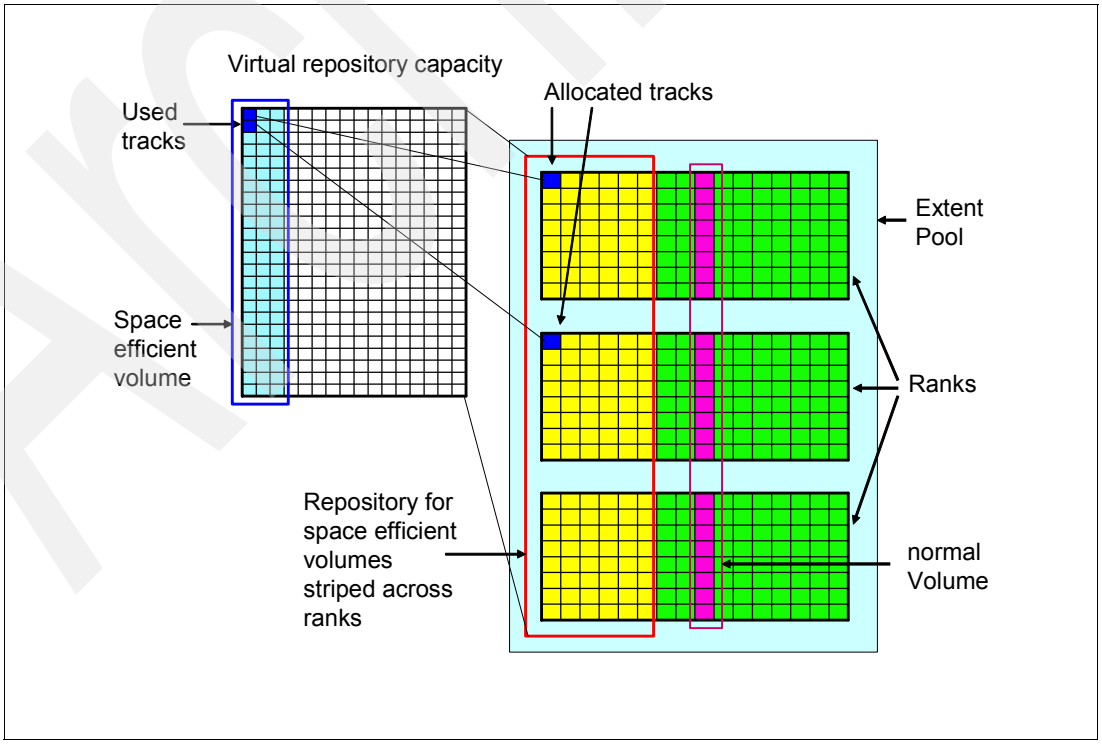


Figure 3-1 Concept of space efficient volumes

3.2.1 Capacity planning for FlashCopy SE

Proper sizing of the amount of physical space required in the repository is essential to the operation of space efficient volumes. Currently, the physical size of the repository cannot be increased after it is defined. Using all of the available physical space makes the SE volumes unavailable to the host. For Global Mirror applications, the last Consistency Group must be maintained, so no updates are allowed to the FlashCopy source, which results in a write inhibit condition on the source, and the Global Mirror pair suspends. It is essential that you not underestimate the initial allocation of physical space that is made available to SE volumes.

Important: It is essential to properly size the repository because you cannot expand its capacity after it is created.

We expect that in many cases 20% of the source volume size is a good value (Beyond 20% of utilization, performance might degrade significantly).

The amount of space that is needed for a space efficient volume depends on two factors:

- The data change rate on the source volumes
- The lifetime of the FlashCopy SE relationship

You can get information about the write activity with the help of *IBM TotalStorage Productivity Center (TPC) for Disk* or, for z/OS, by using host applications, such as RMF™, to collect I/O statistics.

From the write data rate MB/s, we can estimate the amount of changed data by multiplying this number with the planned lifetime of the FlashCopy SE relationship. Let us assume a set of volumes for a 1 TB database. We can assume an average of 3 MB/s write activity. Within 10 hours (36 000 seconds) we update about 100 GB, which is about 10% of the capacity. In many cases, the change rate is much lower. However, we cannot be sure that this amount of change is identical with the capacity needed for the repository. There are two factors that are important here:

- The needed capacity in the repository can be higher because there is always a full track (64 KB) that is copied to the repository for any first change to the source track, even if it is only 4 KB, for example.
- The needed capacity in the repository can be lower because several changes to the same source data track does not change anything in the repository.

Because in most cases we do not know the workload, we can assume both effects even out each other.

Calculating the repository size with the help of standard FlashCopy

If you already have the standard FlashCopy feature, and you plan to use FlashCopy SE but you do not yet have the feature, and you want to plan for the needed capacity (or the capacity you can save), there is an easy way to calculate the needed capacity, which applies whether FlashCopy SE is going to be used in addition to or instead of standard FlashCopy:

- For a source volume that you want to copy later with FlashCopy SE, establish a standard FlashCopy with the *nocopy* option onto a temporary regular target volume.
- Immediately after the FlashCopy check the OutOfSyncTracks with the `lsflash -l` command. In z/OS, you can also use the TSO FCQUERY or ICKDSF FLASHCPY QUERY command.
- Wait as long as you plan to keep your FlashCopy target volumes, and then repeat the `lsflash -l` command, and check the OutOfSyncTracks.

- Calculate the difference of the OutOfSyncTracks from the first and last query and multiply it by 64 KB. This gives you the allocated space in a repository for the source volume in question, if you would have used FlashCopy SE.
- You can repeat the procedure for all volumes you want to copy with FlashCopy SE, and you will get a size for the repository.

If you currently use *incremental* FlashCopy and you plan to use FlashCopy SE, you can check the OutOfSyncTracks with the **`lsflash -l`** command right after you issue the **`resyncflash`** command to get the changed tracks since the last FlashCopy.

When you determine the average change rate on the source volumes, you should double that capacity to be on the safe side, and take this as the size of your repository.

Repository overhead

There is some capacity needed in the repository for internal tables. The size depends on the physical and logical size of the repository. The space for these internal tables is allocated in addition to the specified repository size when the repository is created. Usually, this additional storage is in the range of about 2% of the repository capacity. However, if you define your virtual capacity much larger than the physical capacity, you will get another ratio.

From the following equation, you can obtain an estimate for the additional capacity (**`repoverh`**) that gets allocated with a repository with a certain repository capacity (**`repcap`**) and a certain virtual capacity (**`vircap`**):

```
repoverh (GB) = 0.01 * repcap (GB) + 0.005 * vircap (GB)
```

If a repository of 5,000 GB is created and a virtual capacity of 50,000 GB is specified, for example, about 300 GB gets allocated in addition to the specified 5,000 GB.

3.2.2 Releasing repository space

Repository space that contains data for space efficient volumes is released:

- When you use the release space option with a FlashCopy DSCLI **`rmflash`** command. For z/OS, options are provided on the withdraw ANTRQST API, DFSMSdss DUMP FCWITHDRAW, and ICKDSF FlashCopy withdraw command to release space during withdraw processing.
- When a FlashCopy relationship is created.
- As part of the creation of a Global Mirror Consistency Group.
- As part of volume initialization (using **`ICKDSF INIT`** or DSCLI commands **`initckdvol`** and **`initfbvol`**).

When the repository becomes full, the host can no longer access the SE volumes. Releasing space is needed to recover from conditions that cause the repository to reach 100% utilization. There are thresholds that you can set to trigger messages to you when the repository reaches 15% and 0% of available space remaining. You may also set a threshold to provide an indication of reaching another threshold value. SNMP alerts are generated when a threshold is reached. For z/OS, a new MVS message is generated to indicate the state of the repository (**`IEA499E`**).

Using space efficient volumes as part of a Global Mirror configuration has a slightly different result when the repository becomes full. The last Consistency Group must be maintained, so no updates are allowed to the FlashCopy source, which results in a write inhibit condition on the source, and the Global Mirror pair will suspend. Releasing space in this situation can be

more complicated due to the need to maintain consistent data until GM can be successfully restarted.

3.3 Performance planning for FlashCopy SE

FlashCopy SE has additional overhead compared to standard FlashCopy.

Data from source volumes are copied to *virtual* target volumes. Actually the data is written to a repository, and there is a mapping mechanism to map the physical tracks to the logical tracks (see Figure 3-2). Each time a track in the repository is accessed, it has to go through this mapping process.

Consequently the attributes of the volume that is hosting the repository are important considerations when you plan a FlashCopy SE environment.

Figure 3-2 illustrates updates to source volumes in a FlashCopy SE environment.

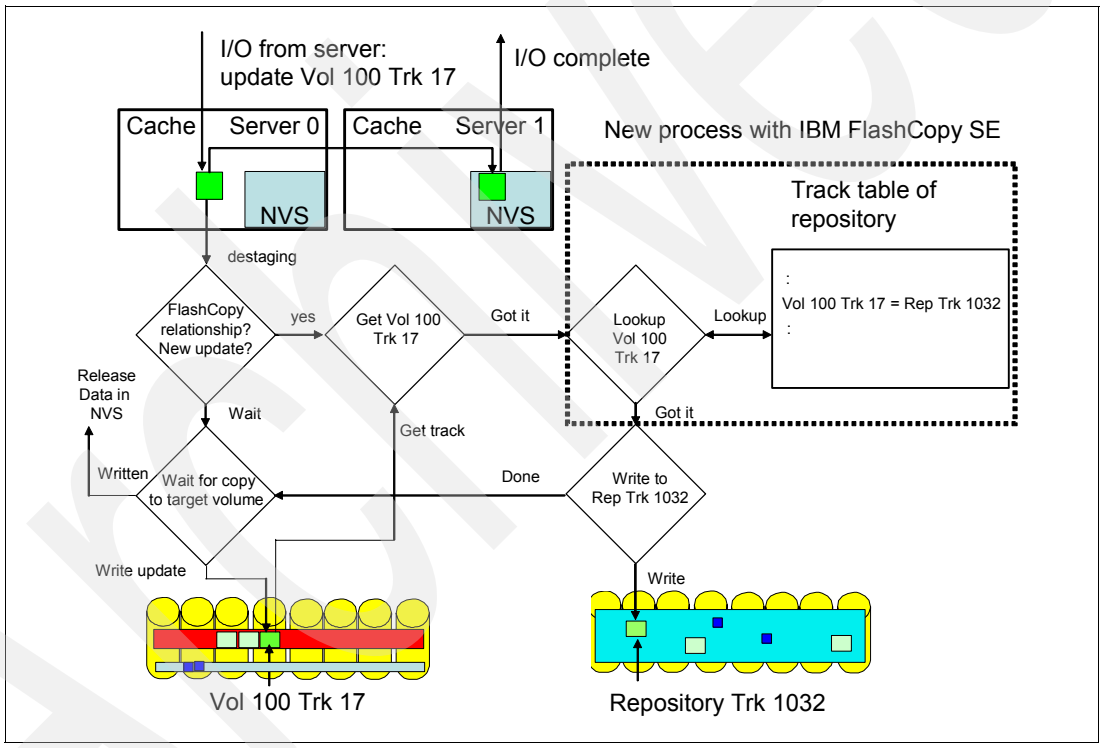


Figure 3-2 Updates to source volumes in a FlashCopy SE relationship

Because of space efficiency, data is not physically ordered in the same sequence on the repository disks as it is ordered on the source. Processes that might access the source data in a sequential manner might not benefit from sequential processing when accessing the target.

Another important consideration for FlashCopy SE is that we always have *nocopy* relationships. A full copy is not possible. If there are many source volumes that have targets in the same extent pool, all updates to these source volumes cause write activity to this one extent pool's repository. We can consider a repository as something similar to a volume. So we have writes to many source volumes being copied to just one *volume,* the repository.

There is less space in the repository than the total capacity (sum) of the source volumes, so you may be tempted to use less disk spindles (DDMs). By definition, fewer spindles mean less performance. You can see how careful planning is needed to achieve the required throughput and response times from the space efficient volumes. A good strategy is to keep the number of spindles roughly equivalent, but just use smaller/faster drives (but do not use FATA drives), for example, if your source volumes are 300 GB 15K RPM disks that are using 73 GB 15 KRPM disks, on the repository may provide both space efficiency and excellent repository performance.

Another possibility is to consider RAID 10 for the repository, although that goes somewhat against space efficiency (you might be better off using standard FlashCopy with RAID 5 than SE with RAID 10). However, there might be cases where trading off some of the space efficiency gains for a performance boost justifies RAID 10. Certainly if RAID 10 is used at the source, you should consider it for the repository (note that the repository always uses striping when in a multi-rank extent pool).

Storage pool striping has good synergy with the repository (volume) function. With storage pool striping, the repository space is striped across multiple RAID arrays in an extent pool, which helps to balance the volume skew that may appear on the sources. It is generally best to use four RAID arrays in the multi-rank extent pool that is intended to hold the repository, and no more than eight.

Finally, try to use at least the same number of disk spindles on the repository as the source volumes. Avoid severe “fan in” configurations, such as 32 ranks of source disk being mapped to an eight rank repository because this type of configuration will likely have performance problems unless the update rate to the source is very modest.

Also, although it is possible to share the repository with production volumes on the same extent pool, use caution when doing this as contention between the two could impact performance.

In summary, you can expect a very high random write workload for the repository. To prevent the repository from becoming overloaded, you can do the following:

- Have the repository in an extent pool with several ranks (a repository is always striped). Use at least four ranks, but no more than eight.
- Use fast 15K RPM and small capacity disk drives for the repository ranks.
- Avoid placing repository and standard volumes in the same extent pool.

Of course the above recommendations are not required, but you should consider them in your planning for FlashCopy SE.

Because FlashCopy SE does not need a lot of capacity (if your update rate is not too high), you might want to make several FlashCopies from the same source volume, for example, you might want to make a FlashCopy several times a day to set checkpoints, to protect your data against viruses, or for other reasons.

Of course creating more than one FlashCopy SE relationship for a source volume increases the overhead because each first change to a source volume track has to be copied several times for each FlashCopy SE relationship; therefore, you should keep the number of concurrent FlashCopy SE relationships to a minimum or test how many relationships you can do without affecting your application performance too much.

The current recommendation is to have no more than four targets for one CKD source and to only have one-to-one relationships for open LUNs.

3.4 Suggested use cases

FlashCopy SE is not a global replacement for standard FlashCopy; however, there are certain use cases that lend themselves to SE. When considering use cases, remember this general principle: the copy should be temporary and the update percentage to the source (or target) should be modest. FlashCopy SE is optimized for use cases where about 5% of the source volume is updated during the life of the relationship. If more than 20% of the source is expected to change, then standard FlashCopy is likely a better choice. Durations for typical use cases are expected to generally be less than eight hours unless the source data has little write activity. You can also use FlashCopy SE for copies that you plan to keep long term, if it is known that the FlashCopy relationship will experience few updates to the source and target volumes.

If there is no way to estimate how long the relationship may persist, then you probably should first consider standard FlashCopy.

Backup to tape

The primary reason that most of you will want to use FlashCopy SE is to make a temporary copy of a volume that is then backed up to tape, which makes sense provided that creating the copy is not immediately followed by a massive string of updates to the source volumes, for example, a database reorg. FlashCopy SE followed by a full volume restore of the source data is not appropriate. By using a FlashCopy, your application can keep writing to the source while backing up a point-in-time to tape.

Figure 3-3 illustrates backup to tape.

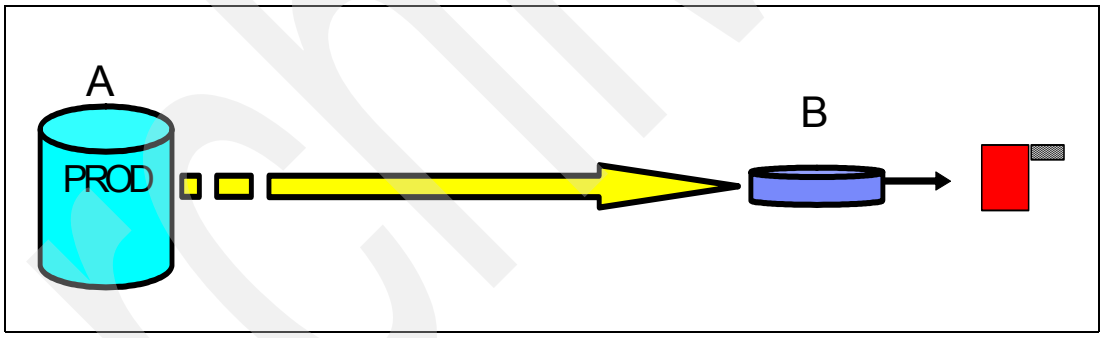


Figure 3-3 Backup to tape

For dumping, to tape the repository must be greater than the amount of production updates during the tape operation.

Online backup and restore

Another anticipated use case for FlashCopy SE is to create temporary copies for application development or disaster recovery (DR) testing. Creating periodic checkpoints or online backups with FlashCopy SE may also help to protect against database corruption or other error conditions. All of these applications are generally not very performance intensive. Other applications with moderate update activity may also lend themselves to FlashCopy SE use.

Figure 3-4 illustrates online backup and restore.

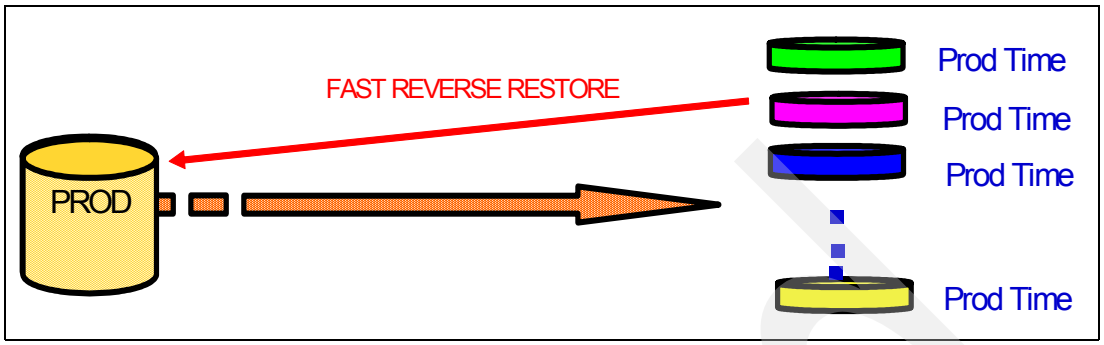


Figure 3-4 Online backup and restore

For online backups, the repository must be greater than all of the updates to all copies of the data since the copies were established. Frequent FlashCopy SE copies to a small number of copy sets limits the physical storage requirements. Reflashing older copies also releases the repository space used.

Remember also that the new microcode now allows you to do a Fast Reverse Restore without having the change recording indicator set.

z/OS Global Mirror configurations

In a z/OS Global Mirror configuration, when mirrored pairs are suspended, additional updates are "recorded" at the local site. These updates need to be sent to the remote site during the re-sync process. A consistent copy of the data is not available until the re-sync process completes, which is why we recommend that a FlashCopy of the consistent data be taken before the re-sync begins. Figure 3-5 illustrates z/OS Global Mirror and FlashCopySE.

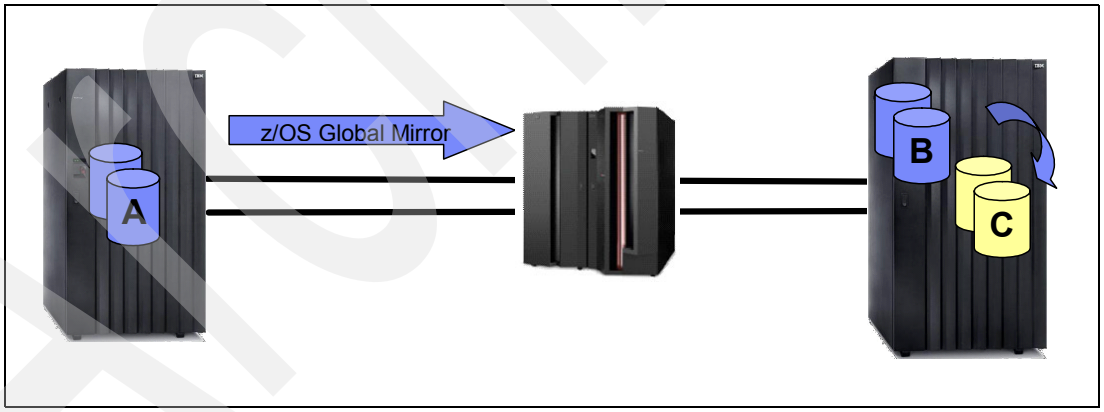


Figure 3-5 z/OS Global Mirror and FlashCopy SE

When space efficient volumes are used for this purpose, the repository must be large enough to hold the original data that is updated from the local site. If, for example, 20,000 tracks are out-of-sync, then the repository must be larger than 20,000 tracks. Each individual storage pool repository must be able to hold the updates from the corresponding pools on the primary. Physical repository space is not shared among storage pools. If one storage pool runs out of space, the Consistent Data Copy cannot be maintained.

Archive

Global Mirror and Metro Global Mirror configurations

Using FlashCopy SE with Global Mirror or Metro/Global Mirror is also a possible use case. When the Consistency Group Interval (CGI) is low, the “C” volumes certainly are temporary. Each time a new CG is formed, a new Flash is created. However, if the CGI grows long due to constrained bandwidth or write bursts, it is possible to fill the repository, which may cause a suspension of GM. Figure 3-6 illustrates Global Mirror and FlashCopy SE.

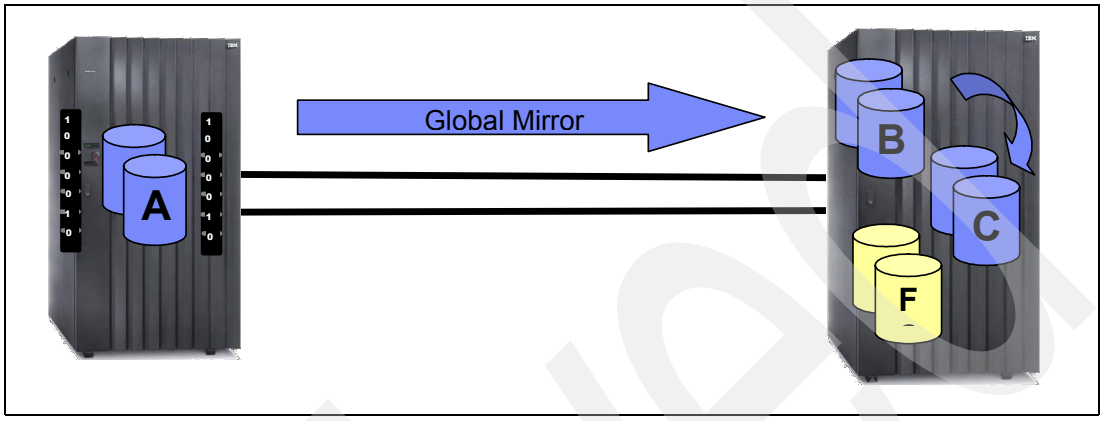


Figure 3-6 Global Mirror and FlashCopy SE

Low Recovery Point Objective (RPO) means that the "B" to "C" FlashCopies are frequently re-flashed. The physical storage requirements must be greater than the amount of updates during each Consistency Group interval. Longer RPO times results in more physical space being required in the repository.

When GM is paused CG formation stops, but data is still being transferred to the remote site, which requires increasing amounts of repository space. The Global Copy pairs should be suspended to limit the repository space used.

3.5 Guidelines and recommendations

Guidelines and recommendations are available for FlashCopy SE. Many of these will change as the licensed internal code continues to be optimized. Many of these will be loosened or removed.

Keep an eye on the following Techdocs flash for up-to-date FlashCopy SE implementation considerations and recommendations:

http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10617

Archive

3.6 DS8000 GUI and DSCLI commands for SE volumes

As with the configuration of any logical volume, a space efficient volume is configured from an extent pool. There are unique characteristics for an extent pool to support space efficient volumes, but the extent pool can support both normally provisioned volumes and space efficient volumes.

When you define an extent pool that will back space efficient volumes, you have to specify virtual capacity and repository capacity. The following sections contains steps on how you can create an extent pool that can back normally provisioned volumes and space efficient volumes.

3.6.1 Creating an extent pool for the repository

Issue the **mkextpool** command to create the fixed block or CKD extent pool. We recommend that you have the repository in at least a four-rank extent pool, which balances the workload over more ranks. Performance may degrade in extent pools with more than eight ranks.

1. Enter the **mkextpool** command at the dscli command prompt with the following parameters and variables:

   ```
   dscli> mkextpool -rankgrp 0 -stgtype fb my_extpool
   ```

   Specify **-stgtype ckd** for CKD extents.

2. Press Enter. A successful execution displays the following message:

   ```
   Date/Time: August 16, 2007 7:07:54 PM PDT IBM DSCLI Version: 0.0.0.0 DS:
   IBM.2107-1300101 Extent Pool P1 successfully created.
   dscli>
   ```

   Note: The unique name that you assigned to the extent pool does not display in the process message. However, when you issue the **lsextpool** command, the extent pool name is displayed.

3. Verify the extent pool assignments by issuing the **lsextpool** command when you are done creating the extent pools. Use the -l parameter to display a full report for the extent pools that are assigned to the storage unit. Enter the **lsextpool** command at the dscli command prompt with the following parameters and variables:

   ```
   dscli>lsextpool -dev IBM.2107-1300101 -l
   ```

After you create the extent pool, you can define space efficient storage in the extent pool and then define space efficient volumes.

3.6.2 Defining the repository

Before you can create a space efficient volume, you have to create a repository for space efficient volumes. There are new DS CLI commands and DS GUI options to deal with repositories. The new DS CLI commands are:

mksestg	Creates a repository.
rmsestg	Deletes a repository.
chsestg	Changes the properties of a repository (currently you cannot change the size of a repository).
lssestg	Lists all repositories.

showsestg Shows details for a repository in a specified extent pool.

There can be one repository per extent pool. A repository has a physical capacity that is available for storage allocations by space efficient volumes and a virtual capacity that is the sum of all LUN/volume sizes of the space efficient volumes. The physical repository capacity is allocated when the repository is created.

Using the DSCLI

Example 3-1 shows the creation of a repository using the DS CLI. If there are several ranks in the extent pool, the repository's extents are striped across the ranks.

The minimum repository size is 16 GB. The maximum repository size is determined by the capacity of the extent pool. However, take into account the capacity overhead for the repository (see "Repository overhead" on page 19). The virtual capacity can be larger than the extent pool capacity.

Tip: If you want to define all storage in an extent pool for a repository, use the DS GUI because it tells you what capacity is allocated for the repository, including the overhead (see Figure 3-8 on page 28).

Example 3-1 Creating a repository for space efficient volumes

```
dscli> mksestg -repcap 100 -vircap 200 -extpool p53
Date/Time: October 17, 2007 11:59:12 IBM DSCLI Version: 5.3.0.977 DS: IBM.2107-7520781
CMUC00342I mksestg:: The space efficient storage for the extent pool P53 has been created
successfully.
dscli>
```

vircap Specifies virtual capacity. The minimum size is 16 GB for virtual or repository capacities.

-repcap or **-reppercent** Specifies repository size. The **-repcap** specifies actual repository size and the **-reppercent** specifies a percentage of virtual capacity.

-captype Optionally specifies all capacity unit types with gb (default), cyl, or blocks.

-recapthreshold Optionally sets the user warning threshold. This is the repository threshold, not the virtual threshold. Defaults to 0% available (100% used).

chsestg Changes SE storage, but currently, it can only change user warning threshold. It cannot change either virtual or repository capacity sizes.

Use this information for all of the extent pools in which you want to define space efficient storage.

Using the DS GUI

Figure 3-7 shows the creation of a repository by the DS GUI.

Archived

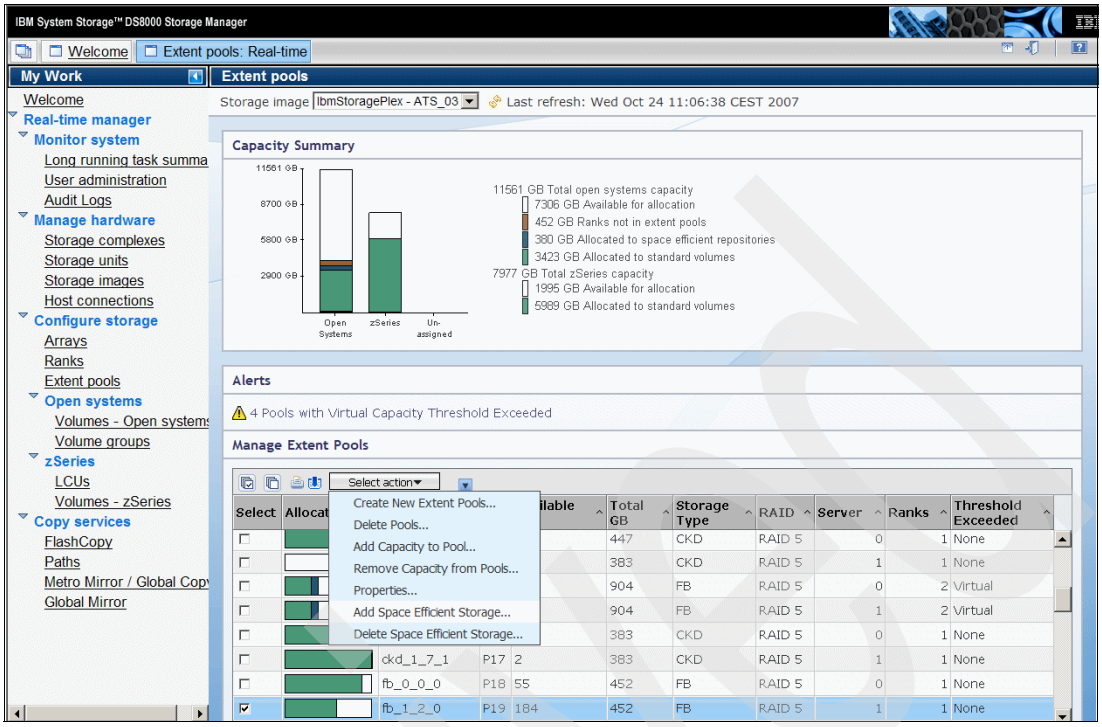


Figure 3-7 Creation of a repository by the DS GUI

To create a repository using the DS GUI:

1. Select **Real-time manager**.
2. Select **Configure storage**.
3. Select **Extent pools**.
4. Select the check box next to an extent pool where you want to create the repository.
5. From Select action, select **Add Space Efficient Storage**.

In Figure 3-8 on page 28, you can specify the physical size of the repository that is allocated on the ranks and the virtual capacity for all space efficient volumes within this extent pool. There are also options to set thresholds for warnings when the repository fills up. Of course there are similar options for the DS CLI. For the complete syntax, see *IBM System Storage DS8000: Command-Line Interface User's Guide,* SC26-7916.

When you create a repository with a certain repository capacity, the actual capacity that is allocated in the extent pool is somewhat larger than the specified capacity to hold some internal tables.

You also have the option to reserve some percentage of virtual capacity, which is helpful if many users are allowed to create volumes, and you want to prevent suddenly running out of space. If some capacity is reserved, an administrator can release the rest of the capacity and he is aware now, that free capacity is short. You can delete a repository using the **`rmsestg`** command. The **`lssestg`** command provides information about all repositories in the DS8000.

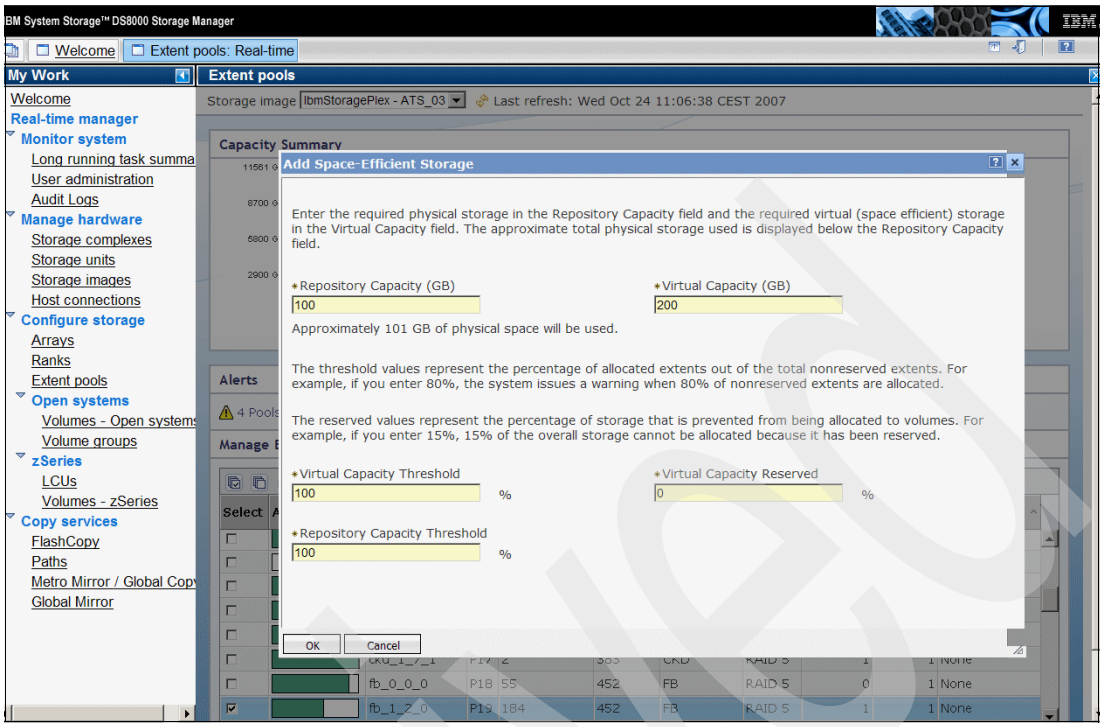


Figure 3-8 Specifying the size of a repository

You can also use the **showsestg** command, which returns the same information as **lssestg -l**, but adds:

- repcapalloc and vircapalloc as percentages
- repcap and vircap capacities in cylinders
- repcap and vircap capacities in blocks
- %repcapthreshold

Example 3-2 shows the output of the **showsestg** command. Check the repcapalloc value if you are interested in how much capacity within the repository is used.

Example 3-2 Getting information about a space efficient repository

```
dscli> showsestg p53
Date/Time: October 17, 2007 1:30:53 IBM DSCLI Version: 5.3.0.977 DS: IBM.2107-7520781
extentpoolID     P53
stgtype          fb
datastate        Normal
configstate      Normal
repcapstatus     below
%repcapthreshold 0
repcap (2^30B)   100.0
repcapblocks     209715200
repcapcyls       -
repcapalloc      0.0
%repcapalloc     0
vircap           200.0
vircapblocks     419430400
vircapcyls       -
vircapalloc      0.0
%vircapalloc     0
overhead         3.0dscli>
```

Figure 3-7 on page 27 shows that the DS GUI also has an action **Delete Space Efficient Storage** to delete a repository, and Figure 3-9 shows the DS GUI when you select the action **Properties**.

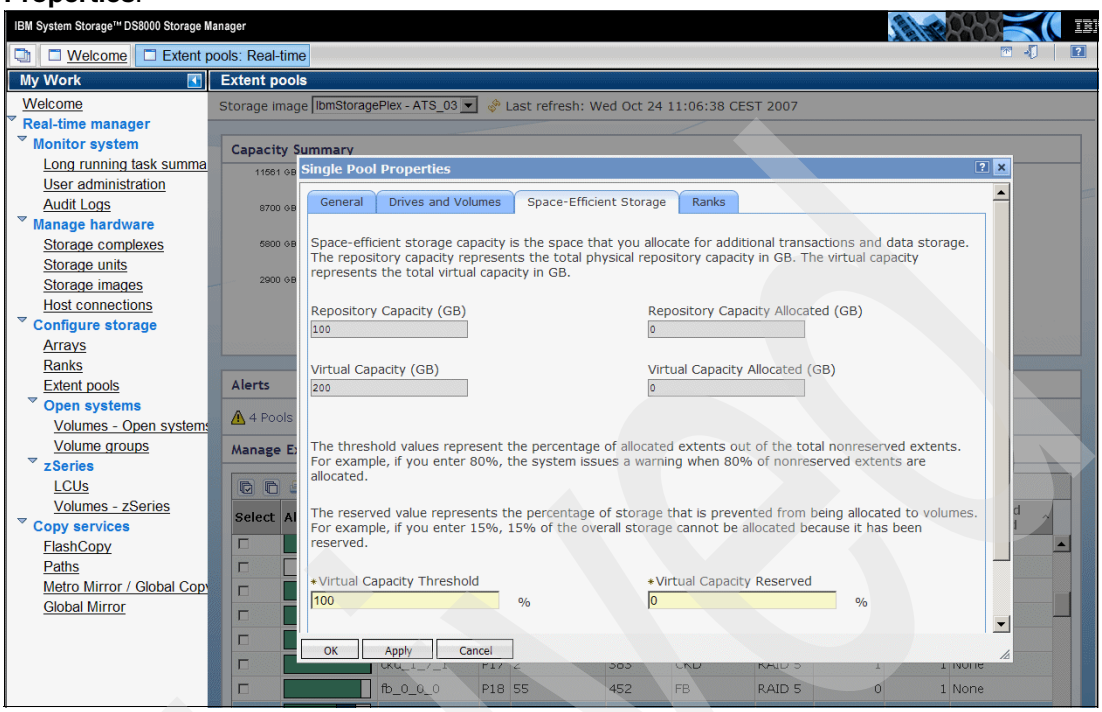


Figure 3-9 Properties of a repository

3.6.3 Creating space efficient volumes

Now that we have a repository, we can create space efficient volumes within this repository.

Working with the DS CLI

You can create a space efficient volume by specifying the **-sam tse** (storage allocation method track space efficient) parameter on the **mkfbvol** command, as shown in Example 3-3.

Example 3-3 Creating a space efficient volume

```
dscli> mkfbvol -extpool p53 -cap 40 -name ITSO-1721-SE -sam tse 1721
Date/Time: October 17, 2007 3:10:13 CEST IBM DSCLI Version: 5.3.0.977 DS: IBM.2107-7520781
CMUC00025I mkfbvol: FB volume 1721 successfully created.
dscli>
```

When we list space efficient repositories with the **lssestg** command, as shown in Example 3-4, we can see that in extent pool P53 we have a virtual allocation of 40 extents (GB), but that the allocated (used) capacity repcapalloc is still zero.

Example 3-4 Getting information about space efficient repositories

```
dscli> lssestg -l
Date/Time: October 17, 2007 3:12:11 PM CEST IBM DSCLI Version: 5.3.0.977 DS: IBM.2107-7520781
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P4           ckd      Normal    Normal      below                     0          64.0    1.0          0.0          0.0
P47          fb       Normal    Normal      below                     0          70.0  282.0          0.0        264.0
P53          fb       Normal    Normal      below                     0         100.0  200.0          0.0         40.0
dscli>
```

This allocation comes from the volume we just created. Use **lsfbvol** (**lsckdvol** for CKD volumes) to display SE volumes. An SE volume shows "TSE" in the sam column. The command **showfbvol** (**showckdvol** for CKD volumes) will also show "TSE" for the sam attribute, but in addition, the **repcapalloc** attribute shows repository capacity used by the specified volume in GBs. Example 3-5 illustrates the **showfbvol** command.

Example 3-5 Checking the repository usage for a volume

```
dscli> showfbvol 1721
Date/Time: October 17, 2007 3:29:30 PM CEST IBM DSCLI Version: 5.3.0.977 DS:
IBM.2107-7520781
Name             ITSO-1721-SE
ID               1721
accstate         Online
datastate        Normal
configstate      Normal
deviceMTM        2107-900
datatype         FB 512
addrgrp          1
extpool          P53
exts             40
captype          DS
cap (2^30B)      40.0
cap (10^9B)      -
cap (blocks)     83886080
volgrp           -
ranks            0
dbexts           0
sam              TSE
repcapalloc      0
eam              -
reqcap (blocks) 83886080
dscli>
```

Working with the DS GUI

Figure 3-10 shows the creation of a space efficient volume using the DS GUI. You have to select an extent pool with a repository. You can identify such an extent pool having a non-zero value in the **Available Virtual GB** column.

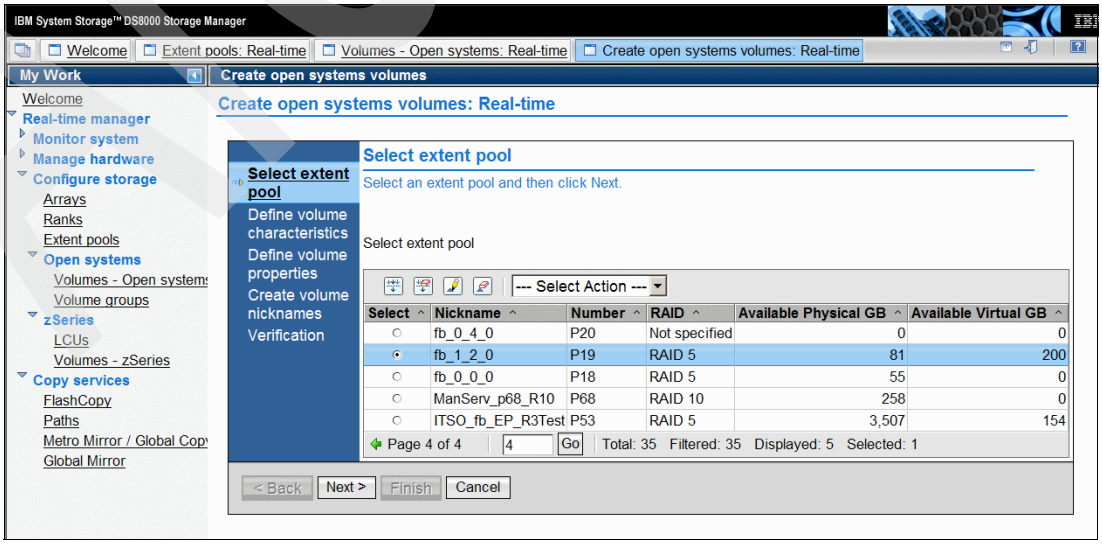


Figure 3-10 Selecting an extent pool with a repository for space efficient volumes

The Define volume characteristics window, Figure 3-11, is where we define that we want to create a space efficient volume.

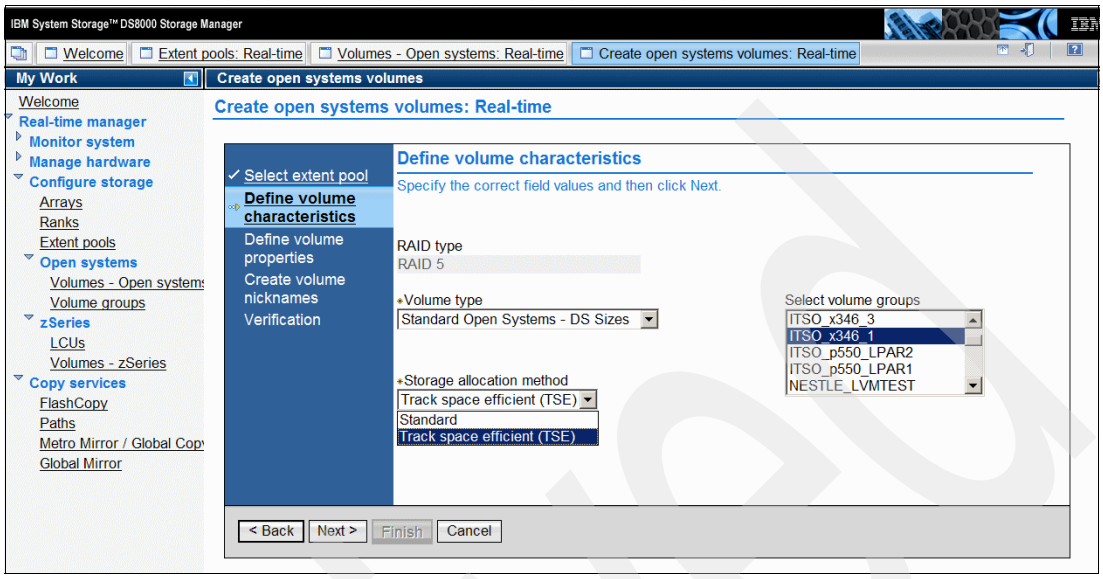


Figure 3-11 Creation of a space efficient volume

To create a space efficient volume, you must select the Storage allocation method **Track space efficient (TSE)**. The remaining steps are the same as for a standard volume.

Archive

Archived

4

FlashCopy SE in open environments

In this chapter, we discuss and illustrate using FlashCopy SE in open environments. We provide details about changes that were made to the various interfaces to support FlashCopy SE.

If you are working in a z/OS mainframe environment, refer to Chapter 5, “Using FlashCopy SE in a z/OS environment” on page 45.

Archived

© Copyright IBM Corp. 2008. All rights reserved.

4.1 Performing FlashCopy SE operations

The operations that you can perform with FlashCopy SE are nearly the same as with standard FlashCopy; however, there are two general exceptions:

- Whenever you want to make a FlashCopy onto a space efficient volume (for example with **mkflash**, **resyncflash,** or **reverseflash**) you have to specify the option **-tgtse** to allow the target volume to be track space efficient.
- On any FlashCopy command the **-cp** (full copy) option is not allowed if the target is a space efficient volume.

You can still perform a standard FlashCopy (onto a fully provisioned volume) even if you specified that a FlashCopy onto a space efficient volume is allowed.

If you work with the DS GUI, there are equivalent options.

Restriction: It is not possible to create a full background copy of the source volume on the target volume when the target volume is a space efficient volume.

4.1.1 Creating and resynchronizing the FlashCopy SE relationship

Now let us do a space efficient FlashCopy. Assume you have two volumes, one standard volume and a space efficient volume. Both volumes can be:

- In different extent pools
- In different LSSs
- Anywhere within the same DS8000 (however in the same LPAR in an LPAR machine).

Working with the DS CLI

In Example 4-1, we have a standard volume (1720) and a space efficient volume (1740). You can identify the space efficient volume by the sam attribute **TSE** (you have to specify the **-l** option to see this attribute).

Example 4-1 List of standard and space efficient volumes

```
dscli> lsfbvol -l 1720-1740
Date/Time: October 24, 2007 3:28:12 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
Name          ID   accstate datastate configstate deviceMTM datatype extpool sam      captype cap (2^30B) ..... volgrp reqcap (blocks) eam
==========================================================================================================================..... ==========================================
ITSO-STD-1720 1720 Online   Normal    Normal      2107-900  FB 512   P3       Standard      DS        25.0 ..... V13           52428800 rotatevols
ITSO-SE-1740  1740 Online   Normal    Normal      2107-900  FB 512   P3            TSE      DS        25.0 ..... V13           52428800 -
```

When we want to establish a FlashCopy SE relationship between the two volumes, we can use any option that is available for standard FlashCopy, such as **-record** or **-persist**, for example, but we *must* specify **-tgtse** and we *cannot* specify **-cp**, which means we establish a *nocopy* relationship. In Example 4-2, we establish a FlashCopy SE relationship.

Example 4-2 Establishing a FlashCopy SE relationship

```
dscli> mkflash -tgtse -record -persist 1720:1740
Date/Time: October 24, 2007 1:53:08 PM CEST IBM DSCLI Version: 5.3.0.991 DS:
IBM.2107-7503461
CMUC00137I mkflash: FlashCopy pair 1720:1740 successfully created.
```

If we had tried to do a full copy by specifying the **-cp** option, we would get a message, as shown in Example 4-3 on page 35.

Example 4-3 Trying to do a full copy

```
dscli> mkflash -tgtse -record -persist -cp 1720:1740
Date/Time: October 24, 2007 3:49:52 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUN02649E mkflash: 1720:1740: The task cannot be initiated. Either you did not specify the
Permit space efficient Target or Secondary option, or at least one of the options that you
have specified is not supported for a space efficient target or secondary volume.
```

Example 4-4 shows the result of an **lsflash -l** command. We can see that BackgroundCopy is disabled when we do the space efficient FlashCopy. The AllowTgtSE Enabled attribute indicates that it actually is a FlashCopy SE relationship.

Example 4-4 Listing a space efficient relationship

```
dscli> lsflash -l 1720
Date/Time: October 24, 2007 3:57:12 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled TargetWriteEnabled
=============================================================================================================================
1720:1740 17     0           60      Disabled   Enabled   Enabled    Disabled   Enabled           Enabled


BackgroundCopy OutOfSyncTracks DateCreated                    DateSynced                     State AllowTgtSE
=============================================================================================================
Disabled       409600          Wed Oct 24 15:55:55 CEST 2007 Wed Oct 24 15:55:55 CEST 2007 Valid Enabled
```

The **lsflash** command has an option to show only FlashCopy SE relationships: **-tgtse.** When you use this command, specify a range of volume addresses where you want to look for FlashCopy SE relationships. Example 4-5 uses the **lsflash -tgtse** command to list Flashcopy SE relationships.

Example 4-5 Listing FlashCopy SE relationships

```
dscli> lsflash -tgtse 1700-1750
Date/Time: October 25, 2007 2:13:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled TargetWriteEnabled BackgroundCopy
=============================================================================================================================
1720:1740 17     0           60      Disabled   Disabled  Disabled   Disabled   Enabled           Enabled           Disabled
```

In Example 4-6, we show the **resyncflash** command with some additional options just to show that we can use them, as in standard FlashCopy operations. Only the **-tgtse** parameter is important for FlashCopy SE.

Example 4-6 Resynchronizing a FlashCopy SE pair

```
dscli> resyncflash -record -persist -tgtpprc -tgtinhibit -tgtse 1720:1740
Date/Time: October 24, 2007 4:16:41 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00168I resyncflash: FlashCopy volume pair 1720:1740 successfully resynchronized.
```

A normal **reverseflash** operation is not possible for a FlashCopy pair that is in a nocopy relationship (a FlashCopy SE relation always is a *nocopy* relationship), but you can do a Fast Reverse Restore operation by using the command **reverseflash -fast**.

Working with the DS GUI

When you set up a FlashCopy SE relationship using the DS GUI, you have to select the option **Allow target volumes to be space efficient volumes** on the Define relationship type window, shown in Figure 4-1.

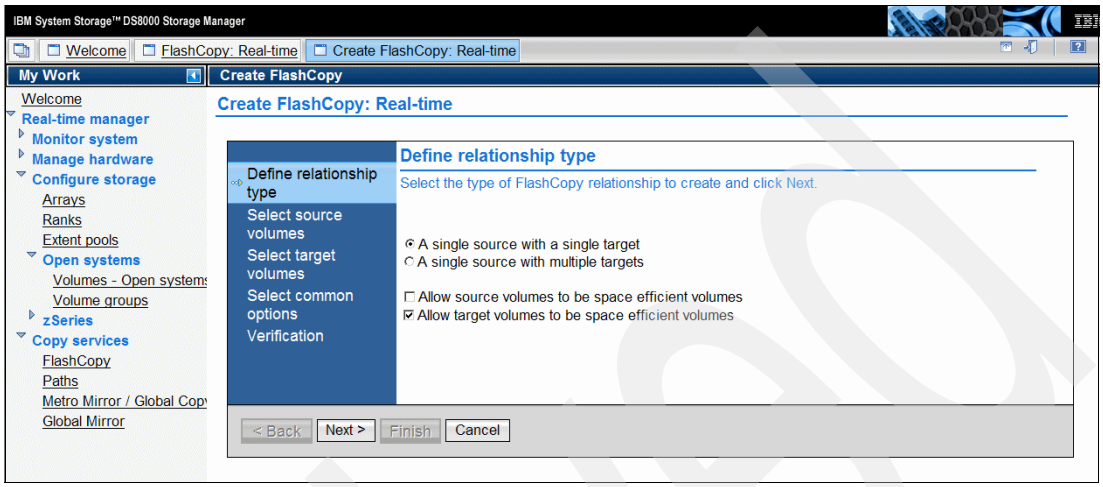


Figure 4-1 Creating a FlashCopy SE with the DS GUI

On the next window, Figure 4-2, you define your source volume, as usual. When you come to the Select target volume panel, you want to select a space efficient volume. You can recognize such volumes by examining the Storage Allocation attribute, which is **TSE** for space efficient volumes.

The other parameters that you can specify are the same as the parameters in standard FlashCopy. Do not select the option **Initiate background copy** because FlashCopy SE does not allow a full copy. If you select the option, you will get an error message, and the FlashCopy will fail. For the same reason, do not select the action *I***nitiate background copy** in the window in Figure 4-3 on page 37.

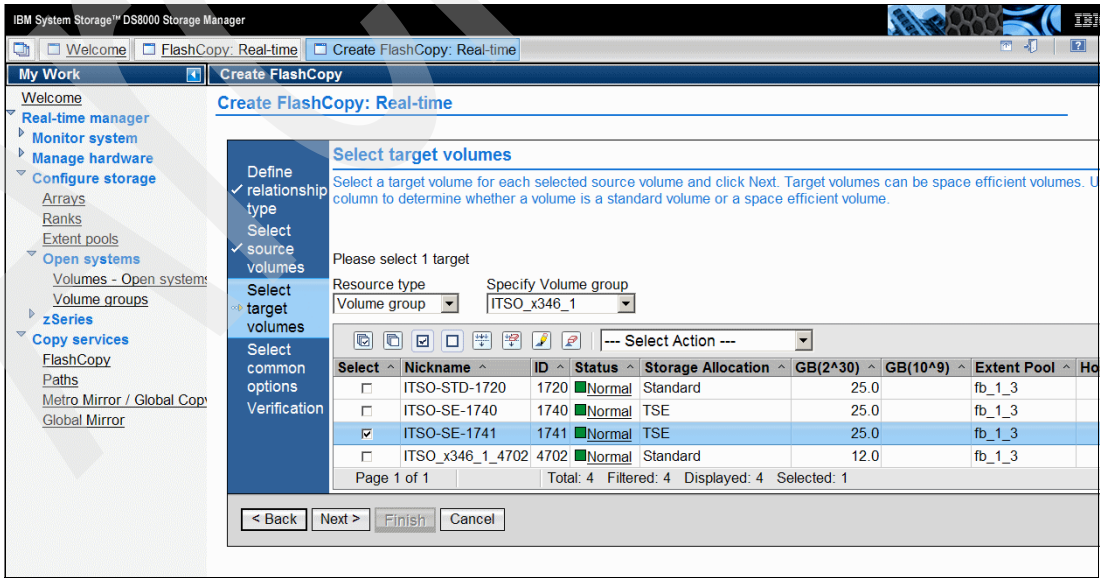


Figure 4-2 Selecting a space efficient target

Select the **Properties** action in the panel shown in Figure 4-3.

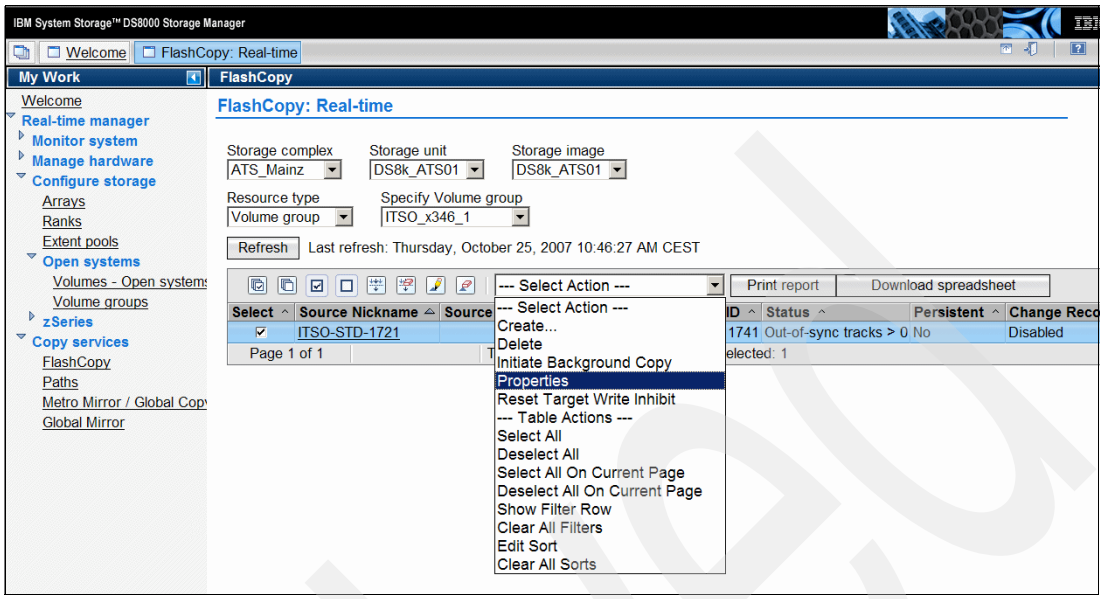


Figure 4-3 Getting information about a FlashCopy SE relationship

Figure 4-4 shows the options that are in effect for this FlashCopy SE relationship. In the current implementation of FlashCopySE, you cannot change the last two options, **Relationship failed if space efficient target volume full** and **Source write inhibited if space efficient target volume full**.

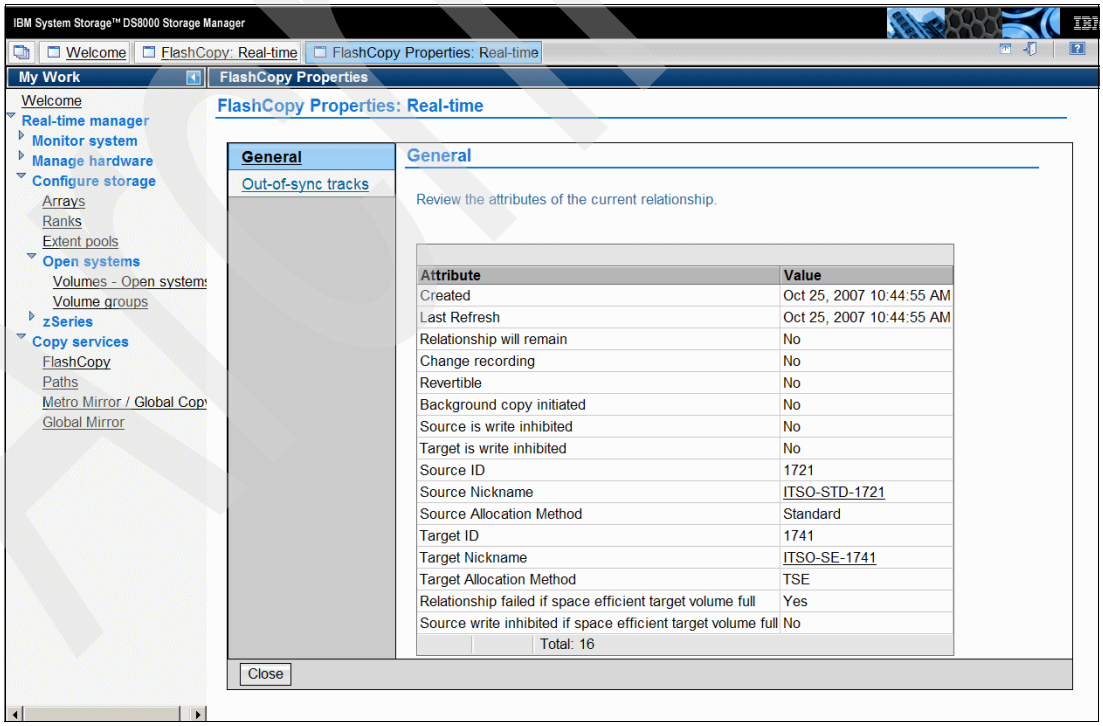


Figure 4-4 Properties of a FlashCopy SE relationship

If you want to check how much space is actually allocated for a space efficient volume, go to **Real-time Manager** → **Configure Storage** → **Volumes - Open systems**, and select a space efficient volume, then select the **Properties** action. In Figure 4-5, we can see that the volume we selected currently occupies 0.3 GB of physical storage.

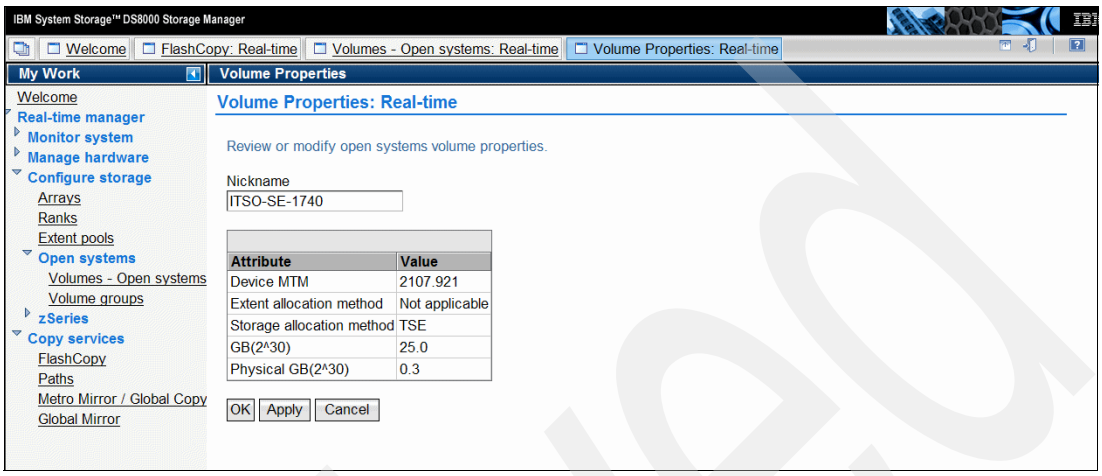


Figure 4-5 Checking the allocated space for a space efficient volume

In a similar way, select **Real-time Manager** → **Configure Storage** → **Extent pools** to select an extent pool with a virtual capacity (which means the extent pool has a repository). Select the **Properties** action for that extent pool. Figure 4-6 shows an example for an extent pool with an allocated repository capacity of t 4.7 GB.

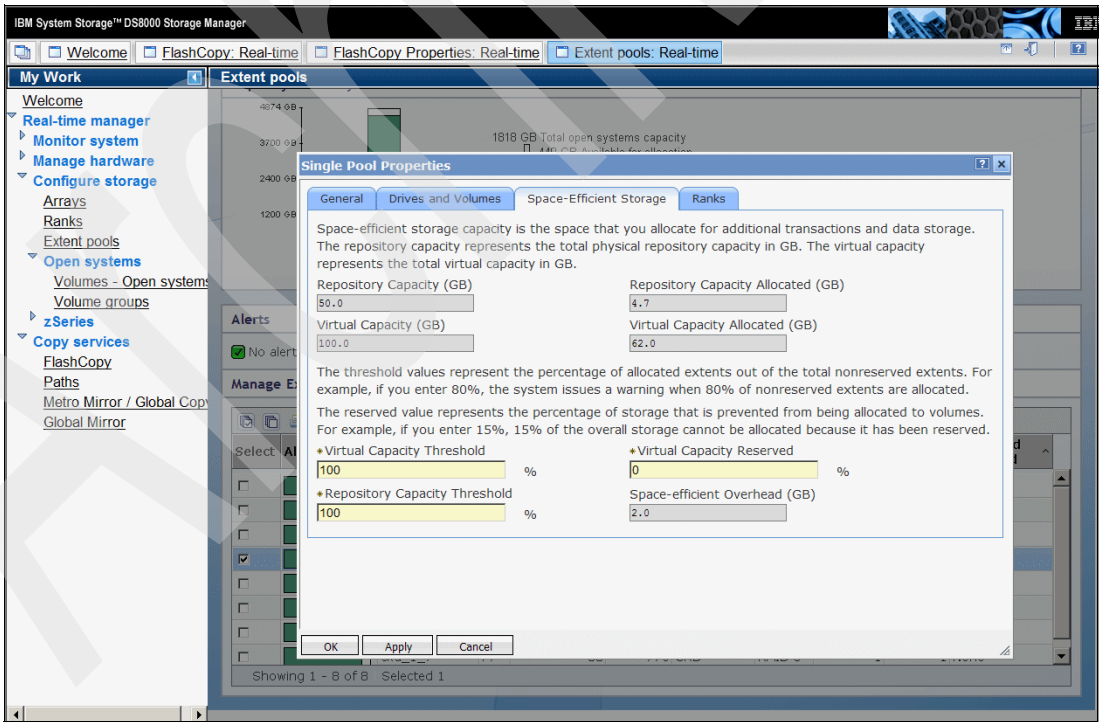


Figure 4-6 Properties of an extent pool with a repository for space efficient storage

4.1.2 Removing FlashCopy relationships and releasing space

When changes are made to the FlashCopy source volume, before a destage to the source volume happens, the original data track of the source volume that is to be modified is copied to the repository that is associated with the space efficient target volume. Consequently, the repository gets filled with data, as shown in Example 4-7.

Example 4-7 A repository is filled with data

```
dscli> lssestg -l
Date/Time: October 24, 2007 4:50:02 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           fb      Normal    Normal      below                       0           50.0  100.0         0.0         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 4:56:14 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           fb      Normal    Normal      below                       0           50.0  100.0         2.6         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:05:02 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           fb      Normal    Normal      below                       0           50.0  100.0         8.9         37.0
```

If you use a normal **rmflash** command to withdraw a FlashCopy relationship, the target volume is in an undefined state, as is always the case with *nocopy* relationships when they are withdrawn; however, the allocated space for that space efficient volume is still allocated and uses up space in the repository.

There are four ways to get rid of that space:

- By specifying the **-tgtreleasespace** on the **rmflash** command
- By using the **initfbvol -action releasespace** command
- By doing a new FlashCopy SE onto the space efficient target volume
- By deleting a space efficient volume with the **rmfbvol** command

Similar functions are available when you use the DS GUI.

Using the DS CLI

You can release space when you withdraw a FlashCopy SE relationship. The **rmflash** command has an option to release space: **-tgtreleasespace**, as Example 4-8 illustrates. When you withdraw the FlashCopy SE relationship with the **-tgtreleasespace** option, the space for the space efficient target volume is released, not immediately, but after a while when all data of your space efficient volume is gone.

Example 4-8 Releasing space when withdrawing a relationship

```
dscli> rmflash -tgtreleasespace -quiet 1720:1740
Date/Time: October 24, 2007 5:21:18 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00140I rmflash: FlashCopy pair 1720:1740 successfully removed.
dscli> lssestg -l
Date/Time: October 24, 2007 5:21:25 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           fb      Normal    Normal      below                       0           50.0  100.0         8.0         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:21:35 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
```

Archived

```
==========================================================================================================================
P3          fb      Normal    Normal      below                    0          50.0  100.0        2.6         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:21:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3          fb      Normal    Normal      below                    0          50.0  100.0        0.0         37.0
```

However, the space efficient volume still exists with the same virtual size, and you can reuse it for another FlashCopy SE relationship.

If the virtual size of the space efficient volume does not match the size of your new source volume, you can dynamically expand the virtual size with the **chfbvol -cap newsize volume** command. However, you cannot make the volume smaller. If you want to make it smaller, you must delete the volume and re-create it with a different size.

If you did not specify the **-tgtreleasespace** parameter on the **rmflash** command, you can use the **initfbvol -releasespace volume** command to release space for the specified volume. Example 4-9 uses the **initfbvol -releasespace** command to release space.

Example 4-9 Releasing space with the initfbvol command

```
dscli> initfbvol -action releasespace 1740
Date/Time: October 25, 2007 9:35:32 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00337W initfbvol: Are you sure that you want to submit the command releasespace for the
FB volume 1740?[Y/N]:y
CMUC00340I initfbvol:: 1740: The command releasespace has completed successfully.
```

Important: Your DS8000 userID needs Administrator rights to use the **initfbvol** command.

After you issue the **initfbvol -releasespace** command for a volume, that volume is empty. All space is released; however, the virtual volume still exists.

Working with the DS GUI

When you intend to withdraw a FlashCopy SE relationship by selecting the **Delete** action in the panel shown in Figure 4-3 on page 37, a subsequent panel is displayed where you have to confirm your action. This panel has an option that is check-marked by default: **Eliminate data and release allocated target space on space efficient target volumes**.

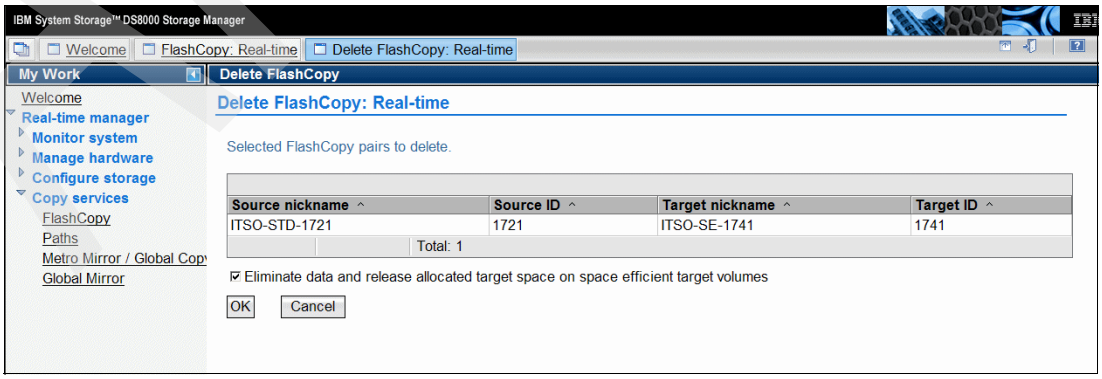


Figure 4-7 Withdrawing a FlashCopy SE relationship with the DS GUI

If you do not select the option, and you do not intend to use the space efficient volume for another FlashCopy right now, and you do not want to delete it either, you can initialize the space efficient volume to release space by selecting the action **Initialize TSE Volume** in the **Real-time Manager** → **Configure Storage** → **Volumes - Open systems** panel after you select the volume you want to initialize, as shown in Figure 4-8.

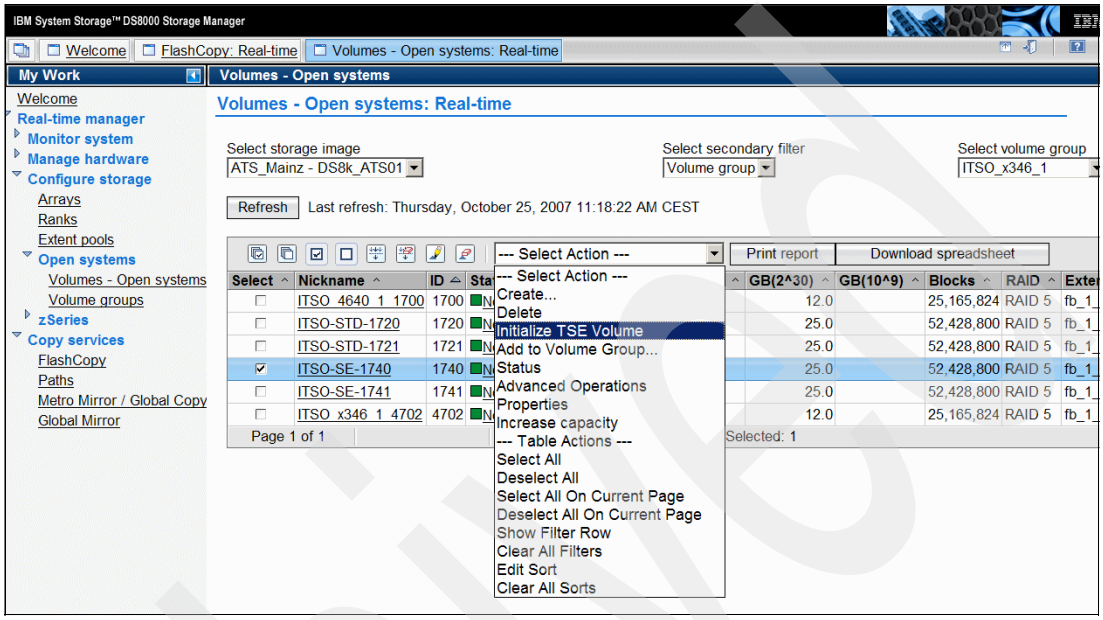


Figure 4-8 Initializing a space efficient volume

4.1.3 Using other FlashCopy SE operations

There are other FlashCopy operations in a standard FlashCopy that we have not yet discussed so far:

- **commitflash**
- **revertflash**
- **unfreezeflash**
- **setflashrevertible**

You can use these commands with FlashCopy SE in the same way that you use the commands with standard FlashCopy, without any change. The **setflashrevertible** command has a **-tgtse** option that you must specify when dealing with space efficient volumes.

The same actions are available in the DS GUI.

You can use the following set of remote FlashCopy commands with FlashCopy SE in the same way that you use them with standard FlashCopy:

- **commitremoteflash**
- **resyncremoteflash**
- **lsremoteflash**
- **mkremoteflash**
- **revertremoteflash**
- **rmremoteflash**
- **setremoteflashrevertible**

For some commands, you must specify the **-tgtse** option if you work with space efficient volumes. We discussed this parameter for the local FlashCopy commands.

For more information about the commands, see *IBM System Storage DS8000: Command-Line Interface User´s Guide*, SC26-7916.

4.1.4 Working with space efficient volumes

You can work with a space efficient volume in the same way that you work with a normal volume. You can mount space efficient volumes to a server or host, read from it, write to it, use it in remote copy relations, and so on. Space efficient volumes, however, are not supported for production. If you try to use a space efficient volume, other than as a FlashCopy target, it does *not* cause a failure, but it is not supported.

Important: Space efficient volumes are supported as FlashCopy target volumes exclusively.

4.1.5 Monitoring repository space and out-of-space conditions

Because you can over-provision space efficient volumes, which means that the sum of all virtual volume sizes can be larger than the physical repository size, you have to keep an eye on the free space that is available in the repository.

Setting a threshold for a repository

By default there is a warning threshold at 15% and 0% free space left in the repository.You can set your own warning threshold to get informed when the repository reaches a certain filling level. In Example 4-10, we set the threshold to 50% (a more practical threshold is at 20% space left).

Example 4-10 Setting a notification threshold for the repository

```
dscli> chsestg -repcapthreshold 50 p3
Date/Time: October 25, 2007 3:09:32 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00343I chsestg:: The space efficient storage for the extent pool P3 has been modified successfully.

dscli> lssestg -l
Date/Time: October 25, 2007 3:09:38 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           fb      Normal    Normal      below                       50           16.0   60.0         0.0        50.0
```

When the repository fills up and the threshold is reached, a warning is sent out, depending on the options you set in the DS8000. If SNMP notification is configured, you receive a trap, as shown in Example 4-11.

Example 4-11 SNMP alert when repository threshold reached

```
2007/10/25 15:22:26 CEST
Space Efficient Repository or Over-provisioned Volume has reached a warning watermark
UNIT: Mnf Type-Mod SerialNm
      IBM 2107-922 75-03461
Volume Type: 0
Reason code: 0
Extent Pool ID: 3
Percentage Full: 50
```

You can also specify a threshold for the repository using the DS GUI, in the panel shown in Figure 4-6 on page 38.

Repository full condition

When the repository becomes full, the FlashCopy SE relationships with space efficient volumes in the full repository will fail at the next write operation (see Example 4-12 for a DS CLI example or Figure 4-9 for an example with the DS GUI).

Example 4-12 State of a FlashCopy SE relationship when the repository is full

```
dscli> lsflash -l 1720
Date/Time: October 25, 2007 3:56:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled
=============================================================================================================
1720:1740 -      -           -       -          -         -          -          -

TargetWriteEnabled BackgroundCopy OutOfSyncTracks DateCreated DateSynced State      AllowTgtSE
=============================================================================================
-                  -              -               -           -          Tgt Failed -
```

When space is exhausted while a FlashCopy SE relationship exists, the relationship is placed in a failed state, which means that the target copy becomes invalid. Writes will continue to the source volume.

If the space becomes depleted, the relationship continues to exist in a failed state, reads and writes to the source continue to be allowed, but updated tracks are not copied to the target. Also, all reads and writes to the target will fail (Figure 4-9), which causes any jobs that are running against the target volume to fail, rather than to succeed without data integrity. To clear this condition for the target volume, withdraw the relationship and release the space on the space efficient volume. You can also establish a new FlashCopy SE relationship, which releases all space in the repository that is associated with the target volume. Another possibility is to use the **`initfbvol`** command to release space.

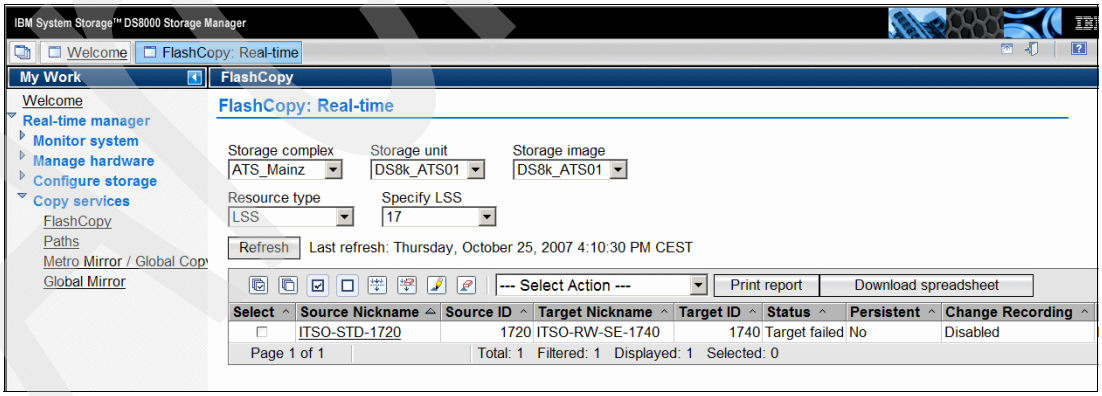


Figure 4-9 Status of a failed FlashCopy SE relation

As space begins to approach depletion on a given repository, the control unit begins to delay writes to the space efficient volumes that are backed by that repository volume, which allows the data in cache to be destaged prior to the space being completely exhausted. This process minimizes the amount of data that gets trapped in NVS when the space is finally exhausted. The delay is based on how many updates are occurring and how much space is left.

In a Global Mirror environment, where the FlashCopy target volumes are space efficient volumes, the behavior of a FlashCopy SE relation is different when the repository becomes full. In this case, you want to keep the FlashCopy relation because it represents a consistent state of the Global Mirror target volumes. The FlashCopy *source* volumes are put in a *write-source inhibit* state. Because these FlashCopy source volumes are *target* volumes of Global Mirror, the Global Mirror pairs are suspended at the next mirror write to the remote volume.

Archived

5

Using FlashCopy SE in a z/OS environment

In this chapter, we discuss and illustrate how to use FlashCopy SE in a z/OS environment.

Archived

© Copyright IBM Corp. 2008. All rights reserved.

5.1 Performing FlashCopy SE operations in a z/OS environment

The operations that you can perform with FlashCopy SE are nearly the same as the operations you can perform with standard FlashCopy; however, there are two general exceptions:

- When you want to make a FlashCopy onto a space efficient volume (for example with the **FCESTABL** TSO command or with DFSMSdss's **COPY FULL** command), you have to specify an option to allow the target volume to be track space efficient. The same is true for **ICKDSF FLASHCPY** establish.
- On any FlashCopy command, the background copy option is not allowed if the target is a space efficient volume.

If you work with the DS CLI or the DS GUI there are equivalent options.

Restriction: You cannot create a full background copy of the source volume on the target volume when the target volume is a space efficient volume.

5.1.1 Creating and resynchronizing FlashCopy SE relationships

Assume that you have two volumes, one standard volume and a space efficient volume. Both volumes can be:

- In different extent pools
- In different LCUs
- Anywhere within the same DS8000 (however in the same LPAR in an LPAR machine).

For performance reasons, both volumes should be managed by the same DS8000 server and, preferably, managed by different device adapters.

Working with Time Sharing Options

When you establish a FlashCopy SE relationship with Time Sharing Option's (TSO) FCESTABL command:

- You *must* specify the new keyword SETGTOK(YES).
- You *must* specify the MODE(NOCOPY) keyword, as we show in Example 5-1.
- You cannot specify the MODE(COPY) or MODE(NOCOPY2COPY) keywords, as shown in Example 5-2 on page 47.
- You cannot specify the INCREMENTAL(YES) keyword as it requires a full copy which is not possible with FlashCopy SE.

Example 5-1 TSO FCESTABL command for a FlashCopy SE pair

```
FCESTABL SDEVN(9631) TDEVN(9633) SETGTOK(YES) MODE(NOCOPY) ONLINTGT(YES)
ANTF0001I FCESTABL COMMAND COMPLETED FOR DEVICE 9631. COMPLETION CODE: 00
```

The request will fail with message ANTF0491E (Example 5-2 on page 47), if the target is a space efficient volume, and you:

- Do not specify the keyword to allow a space efficient target
- Specify NO
- Do not specify MODE(NOCOPY)

Example 5-2 a FlashCopy SE establish fails if MODE(NOCOPY) is not specified

```
FCESTABL SDEVN(9631) TDEVN(9633) SETGTOK(YES) ONLINTGT(YES)
ANTF0491E FLASHCOPY ESTABLISH COMMAND FAILED FOR DEVICE 9631. SPECIFIED TARGET IS SPACE EFF
ANTF0001I FCESTABL COMMAND UNSUCCESSFUL FOR DEVICE 9631. COMPLETION CODE: 08
```

You can use all of the other parameters, such as ONLINTGT, TGTPPRIM, and ACTION(FREEZE), just like you use them in standard FlashCopy relations.

The invocation of an FCQUERY TSO command has information added to indicate if a FlashCopy relationship involves a space efficient volume. The SE column indicates if the source or target volume is space efficient—**Y** means space efficient. If the relation is in a failed state, you will see an **F**. In Example 5-3, we use the FCQUERY command with a space efficient indicator.

Example 5-3 FCQUERY with space efficient indicator

```
FCQUERY DEVN(9631)
ANTF0420I FCQUERY Formatted -3
DEVN SSID LSS CCA  CU     SERIAL       ACT    MAX XC PC CC RV SE  SEQNUM
9631 9006  96  31 2107 000000020781      1  65534  N  N  N  N NY 00000000
```

The new column 'SE' is added, which is a two-character field. The first character indicates the space efficient characteristic of the queried device. The second character indicates the space efficient properties of the FlashCopy target(s) associated with the queried device.

The possible values for the first character in this field are:

Y The queried volume is a space efficient volume.

N The queried volume is a not space efficient volume.

- For a fixed block (OPENDVCS(YES)) or inband (REMOTE(YES)) request, the query is issued using an access volume, so the to space efficient status of that volume does not apply, and a dash (-) is displayed.

The possible values for the second character in this field are:

N The volume is not in a relationship with a space efficient target, and this volume is not a space efficient volume.

Y The volume is in a relationship with one or more space efficient targets.

F The volume is in a relationship with one or more space efficient targets, and at least one of those relationships is in a failed state.

I The volume is in a relationship with one or more space efficient targets, and at least one of those relationships caused this volume to become write inhibited.

B The volume is in a relationship with one or more space efficient targets, and at least one of those relationships caused this volume to become write inhibited. At least one of those relationships is in a failed state.

Working with DFSMSdss

When you work with DFSMSdss:

- You *must* specify the FCSETGTOK(FAILRELATION) keyword to allow a FlashCopy SE relationship onto a space efficient volume. FAILRELATION indicates that FlashCopy to a space efficient volume is permitted and that the relationship will fail if the repository runs out of space.
- You *must* specify FCNOCOPY.

Example 5-4 uses DFSMSdss to copy full onto a space efficient volume.

Example 5-4 DFSMSdss COPY FULL onto a space efficient volume

```
//STEP1  EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//I01      DD UNIT=3390,VOL=SER=RW9630,DISP=SHR
//O01      DD UNIT=3390,VOL=SER=RW9632,DISP=SHR
//SYSIN    DD *
   COPY  FULL INDDNAME(I01)  OUTDDNAME(O01)   PURGE    ALLDATA(*) -
         ADMIN FCNOCOPY FCSETGTOK(FAILRELATION)

ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ADR109I (R/I)-RI01 (01), 2007.310 09:22:50 INITIAL SCAN OF USER CONTROL STATEMENTS COMPL
ADR014I (SCH)-DSSU (02), 2007.310 09:22:50 ALL PREVIOUSLY SCHEDULED TASKS COMPLETED.
ADR016I (002)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (002)-STEND(01), 2007.310 09:22:50 EXECUTION BEGINS
ADR241I (002)-DDTFP(01), TARGET VTOC BEGINNING AT 000001:0000 AND ENDING AT 000010:0014
ADR806I (002)-TOMI (02), VOLUME RW9630 WAS COPIED USING A FAST REPLICATION FUNCTION
```

If the target volume is a space efficient volume, and you do not specify FCSETGTOK, the FlashCopy will fail, as shown in Example 5-5. DFSMSdss will not do a normal copy.

Example 5-5 Result when FCSETGTOK is not specified

```
0ADR849E (002)-DDTFP(01), COPY FROM VOLUME RW9630 TO VOLUME RW9632 FAILED BECAUSE THE
                          OUTPUT VOLUME IS A SPACE EFFICIENT VOLUME
0ADR006I (002)-STEND(02), 2007.310 10:36:54 EXECUTION ENDS
0ADR013I (002)-CLTSK(01), 2007.310 10:36:54 TASK COMPLETED WITH RETURN CODE 0008
```

Support was added for a new RACF® FACILITY class profile, STGADMIN.ADR.COPY.FCSETGT, to allow the installation to protect the new FCSETGTOK keyword.

You also can set the FCSETGTOK(FAILRELATION) parameter in an DFSMS installation exit (ADRUIXIT).

Working with ICKDSF

ICKDSF provides a keyword on its FLASHCPY ESTABLISH request to tell the control unit that the target volume can be a space efficient volume. The default is to not allow the target to be a space efficient volume. The new keyword for the FLASHCPY ESTABLISH is: SETGTOK(NO|YES).

The request will fail with message ICK34029I (Example 5-2 on page 47), if the target is a space efficient volume, and you:

- Do not specify the keyword to allow a space efficient secondary
- Specify NO

Working with the DS CLI

When you want to establish a FlashCopy SE relationship between the two volumes, you can use any option that is available for standard FlashCopy, such as **-record** or **-persist**, for example. However, you *must* specify **-tgtse** and you *cannot* specify **-cp**, which means you can only establish a *nocopy* relationship, as shown in Example 5-6 on page 49.

Example 5-6 Establishing a FlashCopy SE relationship

```
dscli> mkflash -tgtse -record -persist 1720:1740
Date/Time: October 24, 2007 1:53:08 PM CEST IBM DSCLI Version: 5.3.0.991 DS:
IBM.2107-7503461
CMUC00137I mkflash: FlashCopy pair 1720:1740 successfully created.
```

If you attempt to do a full copy by specifying the **-cp** option, the message in Example 5-7 is displayed.

Example 5-7 Trying to do a full copy

```
dscli> mkflash -tgtse -record -persist -cp 1720:1740
Date/Time: October 24, 2007 3:49:52 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUN02649E mkflash: 1720:1740: The task cannot be initiated. Either you did not specify the
Permit space efficient Target or Secondary option, or at least one of the options that you
have specified is not supported for a space efficient target or secondary volume.
```

Example 5-8 shows the result of an **lsflash -l** command. We can see that **BackgroundCopy** is disabled when we do the space efficient FlashCopy. The **AllowTgtSE Enabled** attribute indicates that it actually is a FlashCopy SE relationship.

Example 5-8 Listing a space efficient relationship

```
dscli> lsflash -l 1720
Date/Time: October 24, 2007 3:57:12 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled TargetWriteEnabled
=============================================================================================================================
1720:1740 17     0           60      Disabled   Enabled   Enabled    Disabled   Enabled            Enabled


BackgroundCopy OutOfSyncTracks DateCreated                    DateSynced                    State AllowTgtSE
=============================================================================================================
Disabled       409600          Wed Oct 24 15:55:55 CEST 2007 Wed Oct 24 15:55:55 CEST 2007 Valid Enabled
```

The **lsflash** command has an option to show only FlashCopy SE relationships: **-tgtse**. When you use this command, specify a range of volume addresses where you want to look for FlashCopy SE relationships, as shown in Example 5-9.

Example 5-9 Listing FlashCopy SE relationships

```
dscli> lsflash -tgtse 1700-1750
Date/Time: October 25, 2007 2:13:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled TargetWriteEnabled BackgroundCopy
=============================================================================================================================
1720:1740 17     0           60      Disabled   Disabled  Disabled   Disabled   Enabled            Enabled            Disabled
```

Example 5-10 shows the **resyncflash** command with some additional options to illustrate that you can use the options like you do in standard FlashCopy operations. Only the **-tgtse** parameter is important for FlashCopy SE.

Example 5-10 Resynchronizing a FlashCopy SE pair

```
dscli> resyncflash -record -persist -tgtpprc -tgtinhibit -tgtse 1720:1740
Date/Time: October 24, 2007 4:16:41 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00168I resyncflash: FlashCopy volume pair 1720:1740 successfully resynchronized.
```

A normal **reverseflash** operation is not possible for a FlashCopy pair that is in a nocopy relationship (a FlashCopy SE relation always is a *nocopy* relationship), but you can do a Fast Reverse Restore operation by using the command **reverseflash -fast**.

Working with the DS GUI

When you set up a FlashCopy SE relationship with the DS GUI, you must select the option **Allow target volumes to be space efficient volumes** on the Define relationship type window, which we show in Figure 5-1.

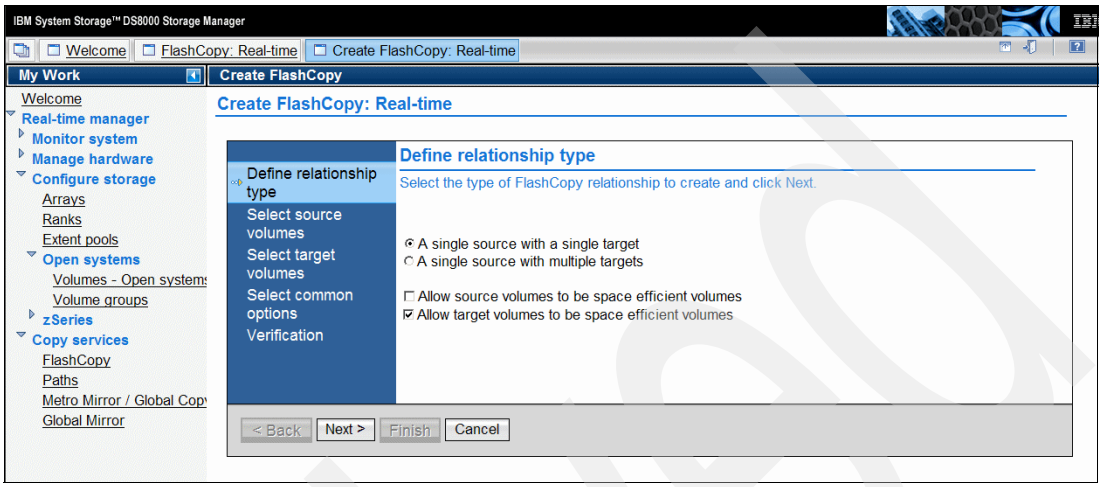


Figure 5-1 Creating a FlashCopy SE with the DS GUI

On the next panel, Figure 5-2, you define your source volume as usual. When you reach the Select target volume panel, select a space efficient volume. You can recognize such volumes by examining the **Storage Allocation** attribute, which is TSE for space efficient volumes.

The other parameters that you can specify are the same ones that you specify in standard FlashCopy. Do not select the option **Initiate background copy** because FlashCopy SE does not allow a full background copy. If you select the option, you will get an error message and the FlashCopy will fail. For the same reason, do not select the action **Initiate background copy** in the panel shown in Figure 5-3 on page 51. However, let us select the **Properties** action in that panel.

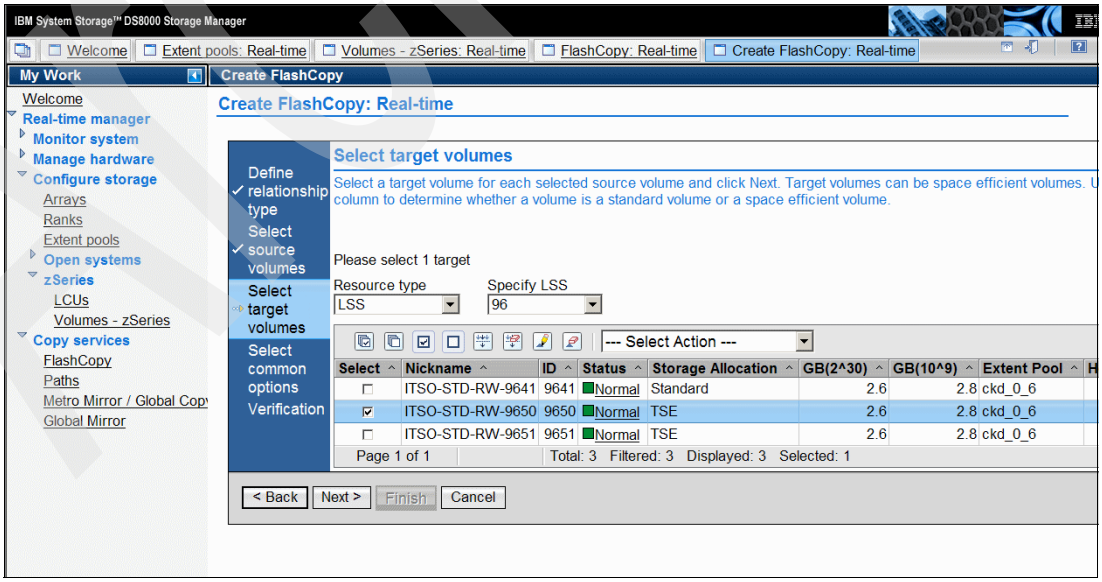


Figure 5-2 Selecting a space efficient target

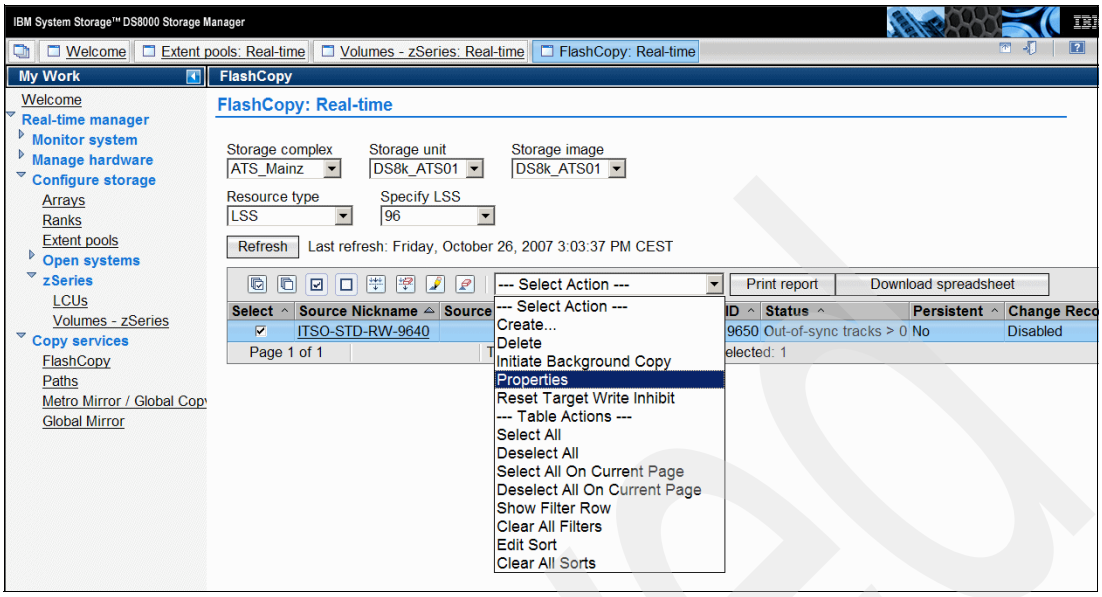


Figure 5-3 Getting information about a FlashCopy SE relationship

Figure 5-4 shows the options that are in effect for this FlashCopy SE relationship. In the current implementation of FlashCopy SE, you cannot change the last two options: **Relationship failed if space efficient target volume full** and **Source write inhibited if space efficient target volume full**.

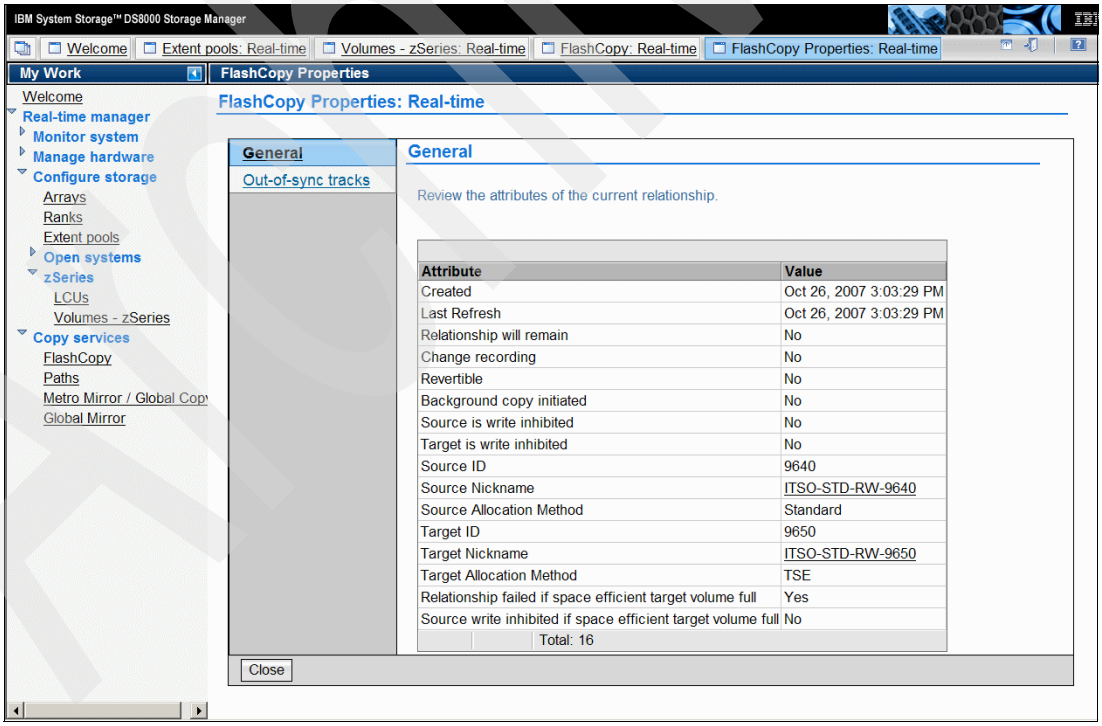


Figure 5-4 Properties of a FlashCopy SE relationship

If you want to check how much space is actually allocated for a space efficient volume, go to **Real-time Manager** → **Configure Storage** → **zSeries®** → **Volumes - zSeries**, select a space efficient volume, and select the **Properties** action.

In Figure 5-5, we see that the volume we selected currently occupies 0.0 GB physical storage.

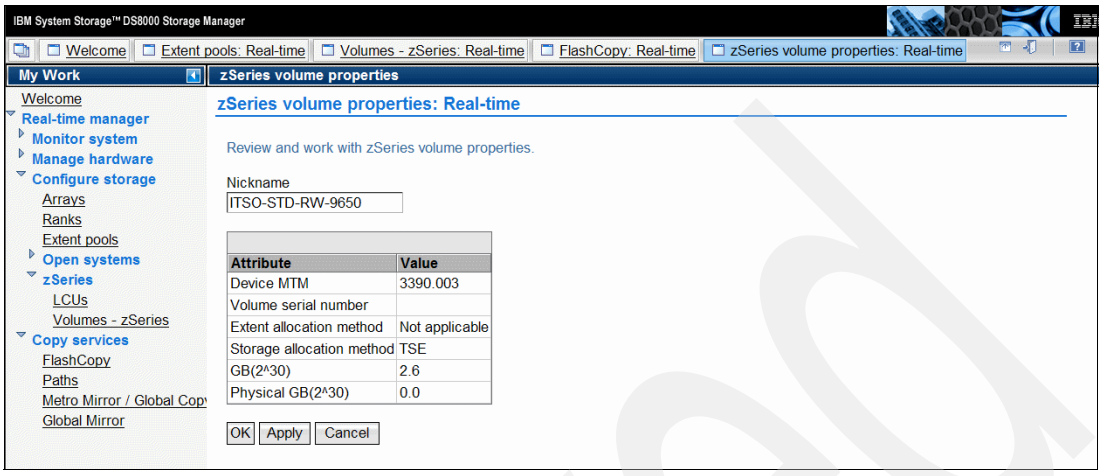


Figure 5-5 Checking the allocated space for a space efficient volume

To select an extent pool with a virtual capacity (which means the extent pool has a repository), select **Real-time Manager** → **Configure Storage** → **Extent pools**, and select the **Properties** action for that extent pool. Figure 5-6 shows an example for an extent pool that has 0.0 GB capacity currently allocated.

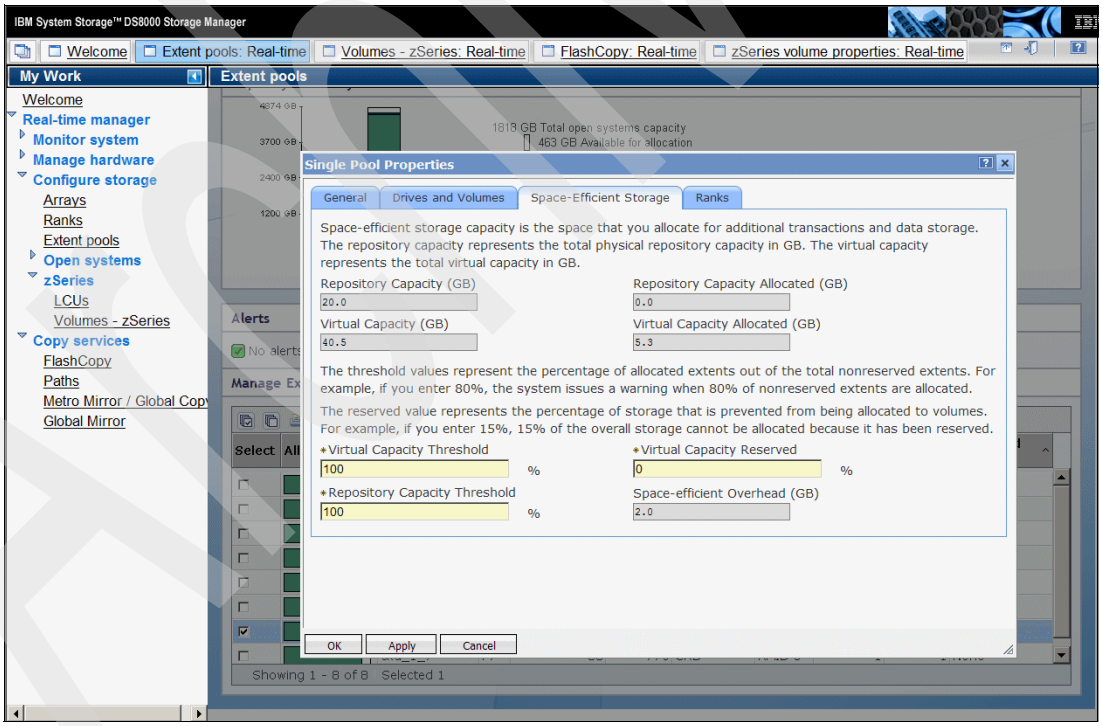


Figure 5-6 Properties of an extent pool with a repository for space efficient storage

5.1.2 Removing FlashCopy relationships and releasing space

When you make changes to the FlashCopy source volume, before a destage to the source volume happens, the original data track on the source volume that is about to be modified is copied to the repository that is associated with the space efficient target volume.

Consequently, the repository gets filled with data, as shown in Example 5-11; however, you can avoid this by checking the repository space with the DS CLI **lssestg** command.

Example 5-11 A repository is filled with data

```
dscli> lssestg -l
Date/Time: October 24, 2007 4:50:02 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                       0           50.0  100.0         0.0         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 4:56:14 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                       0           50.0  100.0         2.6         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:05:02 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                       0           50.0  100.0         8.9         37.0
```

In z/OS, you can get information about the repository or space efficient volumes using the **IDCAMS LISTDATA** command. You can use the SPACEEFFICIENTVOL and the EXTENTPOOLCONFIG keywords, as shown in Example 5-12 and Example 5-13.

Example 5-12 Information about space efficient volumes with the LISTDATA command

```
LISTDATA SPACEEFFICIENTVOL VOLUME(RW9632) UNIT(3390)
IDCAMS  SYSTEM SERVICES                                          TIME: 09:28:43
                      2107 STORAGE CONTROL
                  SPACE EFFICIENT VOLUME REPORT
        STORAGE FACILITY IMAGE ID 002107.922.IBM.75.000000020781
                      SUBSYSTEM ID X'9006'
                    ..........STATUS...........
                      REPOSITORY                              REPOSITORY
DEVICE  VOLSER  SPACE CONSUMED        SIZE  EXT POOL ID         SIZE
9632    RW9632               0        3339         0020        20213
9633    RW9633              11       10017         0020        20213
IDCAMS  SYSTEM SERVICES                                             TIME: 09:28:43
TOTAL NUMBER OF SPACE EFFICIENT VOLUME(S): 2
```

In Example 5-12 and Example 5-13, the Extent Pool is shown as a hexadecimal number. In our example, we queried Extent Pool P32 (as it is called within the DS8000), and it is reported as Extent Pool 20, which is the hexadecimal value of 32.

Example 5-13 Getting information about an extent pool with IDCAMS LISTDATA

```
LISTDATA EXTENTPOOLCONFIG EXTENTPOOLID(32) VOLUME(RW9632) UNIT(3390)
IDCAMS  SYSTEM SERVICES                                          TIME: 09:59:16
                           2107 STORAGE CONTROL
                      EXTENT POOL CONFIGURATION REPORT
        STORAGE FACILITY IMAGE ID 002107.922.IBM.75.000000020781
                      ......EXTENT POOL ID 0003 SUMMARY.....
REPOSITORY FULL WARNING PERCENTAGE:              0
EXT POOL FULL WARNING PERCENTAGE:                0
EXTENT POOL STATUS
       FIXED BLOCK EXT POOL:                          NO
       REPOSITORY CONFIGURED:                         YES
       EXTENT POOL AT WARNING PERCENTAGE:             NO
       EXTENT POOL FULL:                              NO
```

```
                  ...EXTENT POOL 0020 DETAILED REPORT....
EXTENT POOL REPOSITORY STATUS
       REPOSITORY AT WARNING PERCENTAGE:            YES
       REPOSITORY FULL:                              NO
                        SIZE                      ALLOCATED
EXTENT POOL           483042                         444087
REPOSITORY             20213                             11
```

If you were to withdraw a FlashCopy relationship, the target volume would be in an undefined state, as is always the case with nocopy relationships when they are withdrawn, but the allocated space for that space efficient volume is still allocated and uses up space in the repository.

There are several ways to release that space:

- Options are provided on the withdraw ANTRQST API and ICKDSF FlashCopy withdraw command to release space during withdraw processing. For DFSMSdss DUMP FCWITHDRAW, no option is provided, and the space is automatically released for a FlashCopy SE relationship.
- Space is released when a volume is initialized with the ICKDSF INIT command or by using the DS CLI **`initckdvol -action releasespace`** command.
- When you establish a new FlashCopy SE onto the space efficient target volume.
- When you delete a space efficient volume with the **`rmckdvol`** command.

Similar functions are available when using the DS CLI or the DS GUI.

Using the FCWITHDR command in TSO

When you withdraw a FlashCopy SE relationship using the **`FCWITHDR`** TSO command, space for the space efficient volume is not released. You can use ICKDSF INIT to initialize the volume and release its space in the repository.

Using ICKDSF

If the device that is being initialized with the ICKDSF INIT command is a space efficient volume, the result is that any allocated space that is associated with this volume is released prior to the initialization.

ICKDSF provides a keyword on its FLASHCPY WITHDRAW request that tells the control unit to release any allocated space that is associated with the target volume of the relationship, if it is space efficient. The keyword to do this is RELATSPACE.

Using the DS CLI

You can release space when you withdraw a FlashCopy SE relationship. The **`rmflash`** command has an option to release space: **`-tgtreleasespace`** (Example 5-14). When you withdraw a FlashCopy SE relationship with this option, the space for the space efficient target volume is released, not immediately, but after a while all data of your space efficient volume is gone.

Example 5-14 Releasing space when withdrawing a relationship

```
dscli> rmflash -tgtreleasespace -quiet 1720:1740
Date/Time: October 24, 2007 5:21:18 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00140I rmflash: FlashCopy pair 1720:1740 successfully removed.
dscli> lssestg -l
Date/Time: October 24, 2007 5:21:25 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
```

```
P3           ckd     Normal    Normal      below                     0           50.0  100.0         8.0         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:21:35 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                     0           50.0  100.0         2.6         37.0

dscli> lssestg -l
Date/Time: October 24, 2007 5:21:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                     0           50.0  100.0         0.0         37.0
```

However, the space efficient volume still exists with the same virtual size, and you can resuse it for another FlashCopy SE relationship.

If the virtual size of the space efficient volume does not match the size of your new source volume, you can dynamically expand the virtual size with the **chckdvol -cap** ***newsize volume*** command. However, you cannot make the virtual size smaller. If you want to make it smaller, you must delete it and recreate it with a different size.

If you did not specify the **-tgtreleasespace** parameter on the **rmflash** command, you can use the **initckdvol -releasespace** ***volume*** command to release space for the specified volume, as we show in Example 5-15.

Example 5-15 Releasing space with the initckdvol command

```
dscli> initckdvol -action releasespace 1740
Date/Time: October 25, 2007 9:35:32 IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00337W initckdvol: Are you sure that you want to submit the command releasespace for
the CKD volume 1740?[Y/N]:y
CMUC00340I initckdvol:: 1740: The command releasespace has completed successfully.
```

Important: Your DS8000 userID needs Administrator rights to use the **initckdvol** command.

After you issue this command for a volume, the volume is empty, and all space is released, but the virtual volume still exists.

Working with the DS GUI

When you want to withdraw a FlashCopy SE relationship, select the **Delete** action in the panel shown in Figure 5-3 on page 51. Another panel is displayed that asks you to confirm your action. This panel also has an option that is selected by default: **Eliminate data and release allocated target space on space efficient target volumes**. Make sure the option is selected.

Archive

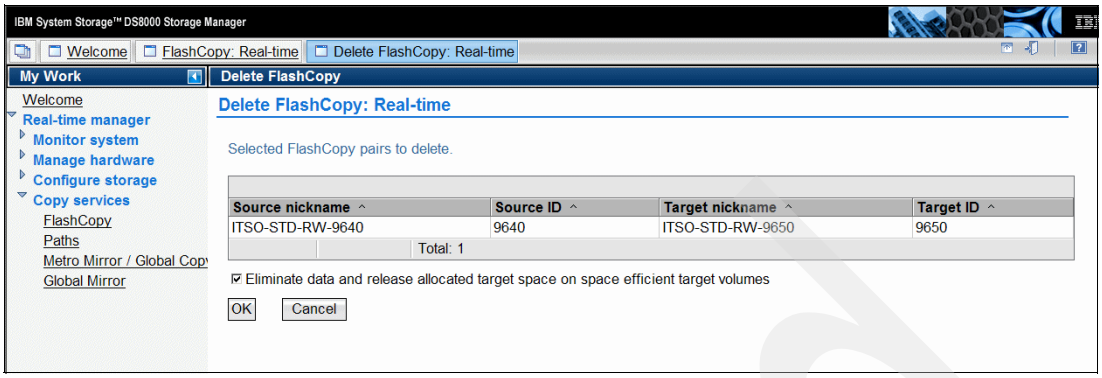


Figure 5-7 Withdrawing a FlashCopy SE relationship with the DS GUI

If you do not select the option, and you do not intend to use the space efficient volume for another FlashCopy right now, and you do not want to delete it either, you can initialize the space efficient volume to release space by selecting the action **Initialize TSE Volume** in the **Real-time Manager** → **Configure Storage** → **Volumes - Open systems** panel after you select the volume you want to initialize, as shown in Figure 5-8.

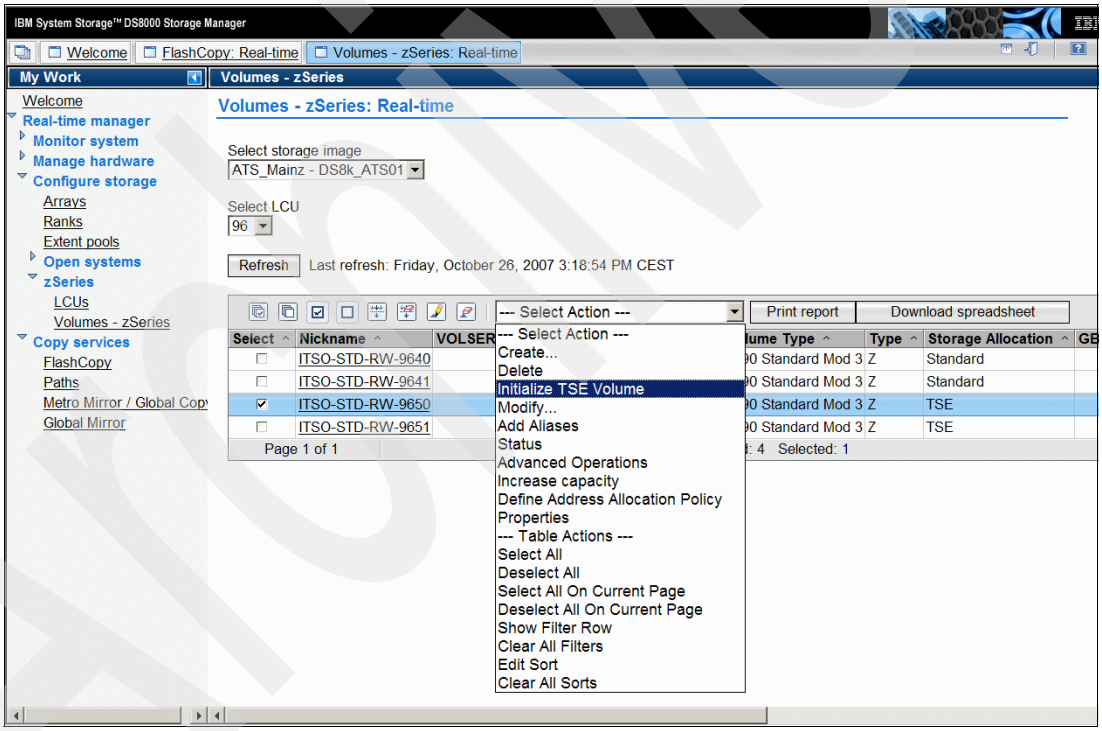


Figure 5-8 Initializing a space efficient volume

5.1.3 Using other FlashCopy SE operations

There are other FlashCopy operations in a standard FlashCopy that we have not yet discussed so far:

- **`commitflash`**
- **`revertflash`**
- **`unfreezeflash`**
- **`setflashrevertible`**

You can use all of these commands with FlashCopy SE in the same way that you use them with the standard FlashCopy without any change. The **setflashrevertible** command has a **-tgtse** option, which you must specify when dealing with space efficient volumes.

The same actions are available in the DS GUI.

You can use the following set of remote FlashCopy commands with FlashCopy SE in the same way that you use them with standard FlashCopy:

- **commitremoteflash**
- **resyncremoteflash**
- **lsremoteflash**
- **mkremoteflash**
- **revertremoteflash**
- **rmremoteflash**
- **setremoteflashrevertible**

For some commands, you must specify the **-tgtse** option if you work with space efficient volumes. We discussed this parameter for the local FlashCopy commands.

For more information about the commands, see *IBM System Storage DS8000: Command-Line Interface User´s Guide*, SC26-7916.

5.1.4 Working with space efficient volumes

You can work with a space efficient volume in the same way that you work with a normal volume. You can vary it online to a host, read from it, write to it, use it in remote copy relations, and so on. Space efficient volumes, however, are not supported for production. Space efficient volumes are supported as FlashCopy target volumes only.

When you have your FlashCopy operations onto space efficient volumes, you want to use these target volumes in some way. You might want to use them as source for your backup jobs or you might want to use them for some tests.

You can use space efficient volumes in the same way that you use normal volumes.

5.1.5 Software Management Services support

Use space efficient volumes only as target volumes for full volume FlashCopy operations. Therefore, we recommend that you do not add space efficient volumes to storage groups, so that they are not selected for Software Management Services (SMS) managed data set allocations, but this is not enforced as a restriction. SMS does, however, check if a volume is space efficient.

Space efficiency is treated as a new volume selection attribute. The relative importance of this attribute is considered *high* with the result that in most situations, space efficient volumes are placed at the bottom of a prioritized list of volumes. Consequently space efficient volumes are not selected until all non-space efficient (normal) volumes are tried.

You can direct allocations to space efficient volumes by putting only space efficient volumes in the storage group or by using guaranteed space.

Archived

5.1.6 Monitoring repository space and out-of-space conditions

Because space efficient volumes can be over-provisioned, which means that the sum of all virtual volume sizes can be larger than the physical repository size, you have to keep an eye on the free space that is available in the repository.

Setting a threshold for a repository

By default there is a warning threshold at 15% and 0% free space left in the repository.You can set your own warning threshold to get informed when the repository reaches a certain filling level. In Example 5-16, we set the threshold to 50% (a more practical threshold is 20% of free space left).

Example 5-16 Setting a notification threshold for the repository

```
dscli> chsestg -repcapthreshold 50 p3
Date/Time: October 25, 2007 3:09:32 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
CMUC00343I chsestg:: The space efficient storage for the extent pool P3 has been modified successfully.

dscli> lssestg -l
Date/Time: October 25, 2007 3:09:38 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
extentpoolID stgtype datastate configstate repcapstatus %repcapthreshold repcap (2^30B) vircap repcapalloc vircapalloc
==========================================================================================================================
P3           ckd     Normal    Normal      below                       50          16.0   60.0         0.0        50.0
```

When the repository fills up and the threshold is reached, a warning is sent out, depending on the options you set in the DS8000. If SNMP notification is configured, you receive a trap, as shown in Example 5-17.

Example 5-17 SNMP alert when repository threshold reached

```
2007/10/25 15:22:26 CEST
Space Efficient Repository or Over-provisioned Volume has reached a warning watermark
UNIT: Mnf Type-Mod SerialNm
      IBM 2107-922 75-03461
Volume Type: 0
Reason code: 0
Extent Pool ID: 3
Percentage Full: 50
```

In z/OS, the notifications are in the form of a unit check, upon the next start I/O with sense information that indicates the ID of the extent pool and the out of space condition that was reached for that repository volume or extent pool. The control unit responds with a unit check to an I/O on one grouped volume in each LSS that has volumes in that extent pool. The control unit has a throttling mechanism that prevents too many notifications from being sent when the allocated space for a repository volume or extent pool wavers at the warning watermark. In response to these notifications, the host issues a message to the console, as we show in Example 5-18.

Example 5-18 Repository warning messages

```
*IEA499E 9100,RS9100,002F,9001,002107.922.IBM.75.000000020781,
 REPOSITORY VOLUME WARNING: AT 15% CAPACITY REMAINING

*IEA499E 9100,RS9100,002F,9001,002107.922.IBM.75.000000020781,
 REPOSITORY VOLUME EXHAUSTED: AT 0% CAPACITY REMAINING
```

Archived

You can also specify a threshold for the repository when you use the DS GUI. You can specify it in the panel that we show in Figure 5-6 on page 52.

Repository full condition

When space on the repository volume is completely exhausted, all space efficient volumes with writes in the repository become unavailable, or *go offline*, which means writes to the volumes are failed with *intervention required* sense. This is true, regardless of whether or not the space efficient volume is currently the target of a FlashCopy relationship.

When space is exhausted while a FlashCopy SE relationship exists, the relationship is put in a failed state, which means that the target copy becomes invalid. Writes will continue to the source volume.

If the space becomes depleted, the relationship continues to exist in a failed state, reads and writes to the source continue to be allowed, but updated tracks are not copied to the target. Also, all reads and writes to the target will fail, which causes any jobs that are running against the target volume to fail, rather than to succeed without data integrity. To clear this condition for the target volume, withdraw the relationship and release the space on the space efficient volume. You have to initialize the target volume with the ICKDSF INIT command.

As space begins to approach depletion on a given repository volume, the control unit begins to delay writes to the space efficient volumes that are backed by that repository volume. This process allows the data in cache to be destaged prior to the space being completely exhausted, which minimizes the amount of data that gets trapped in NVS when the space is finally exhausted. The delay is based on how many updates are occurring and how much space is left.

Example 5-19 shows the state of a FlashCopy SE relationship when the repository is full and the FlashCopy is in a failed state.

Example 5-19 State of a FlashCopy SE relationship when the repository is full

```
dscli> lsflash -l 1720
Date/Time: October 25, 2007 3:56:49 PM CEST IBM DSCLI Version: 5.3.0.991 DS: IBM.2107-7503461
ID        SrcLSS SequenceNum Timeout ActiveCopy Recording Persistent Revertible SourceWriteEnabled
=============================================================================================
1720:1740 -      -           -       -          -         -          -          -


TargetWriteEnabled BackgroundCopy OutOfSyncTracks DateCreated DateSynced State      AllowTgtSE
=============================================================================================
-                  -              -               -           -          Tgt Failed -
```

Figure 5-9 on page 60 shows an example of the state of a FlashCopy SE relationship when the repository is full and the FlashCopy is in a failed state, with the DS GUI.

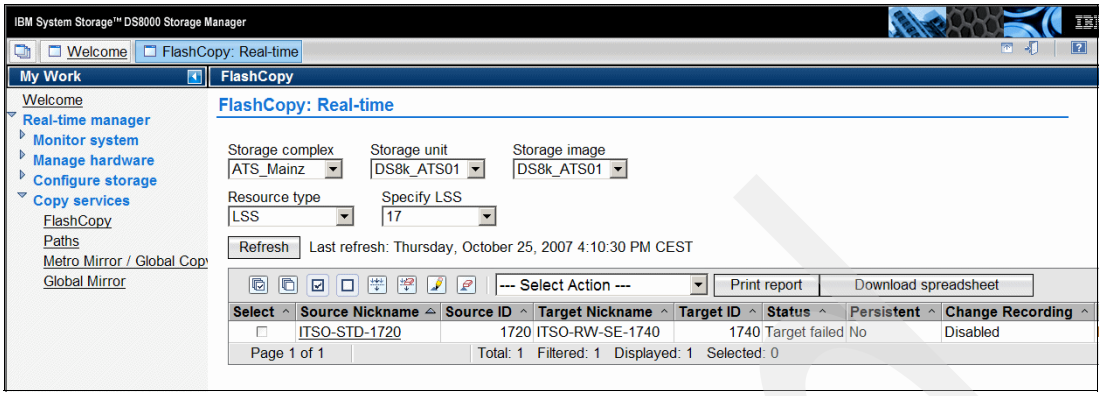


Figure 5-9 Status of a failed FlashCopy SE relation

In a Global Mirror environment, where the FlashCopy target volumes are space efficient volumes, the behavior of a FlashCopy SE relation is different when the repository becomes full. In this case, we want to keep the FlashCopy relation as it represents a consistent state of the Global Mirror target volumes. The FlashCopy *source* volumes are put in a *write-source inhibit* state. Because these FlashCopy source volumes are *target* volumes of Global Mirror, the Global Mirror pairs are suspended at the next mirror write to the remote volume.

Archive

6

Licensing

In this chapter, we briefly describe the licensing functions for Copy Services on the DS8000 Series, with some details regarding the FlashCopy SE function.

Archived

© Copyright IBM Corp. 2008. All rights reserved.

6.1 Licensing

All DS8000 Series machines must have an Operating Environment License (OEL) for the total storage installed, as defined in gross decimal TB. Licenses are also required for the use of Copy Services functions.

Licensed functions require the selection of a DS8000 series feature number (IBM 2107) and the acquisition of DS8000 Series Function Authorization (IBM 2244) feature numbers:

- The 2107 licensed function indicator feature number enables technical activation of the function subject to the client applying an activation code made available by IBM.
- The 2244 function authorization feature numbers establish the extent of IBM authorization for that function on the 2107 machine for which it was acquired.

For the Copy Services, in addition to the basic licences, such as Point-in-Time Copy (for FlashCopy), there are also so-called add-on license features. Add-on license features are cheaper than the complementary basic license feature. An add-on can only be specified when the complementary basic feature exists. The condition for this is that capacity that is licensed by add-on features must not exceed the capacity that is licensed by the corresponding basic feature.

The license for the space efficient FlashCopy and FlashCopy SE (SE), does not require the ordinary FlashCopy (PTC) licence. As with the ordinary FlashCopy, the FlashCopy SE is licensed in tiers by gross amounts of TB installed. FlashCopy (PTC) and FlashCopy SE can be complementary licences: A client who wants to add a 20 TB FlashCopy SE license to a DS8000 that already has a 20 TB FlashCopy license can use the 20 TB FlashCopy SE add-on license (2x #7333) for this.

The following licenses apply to FlashCopy and FlashCopy SE Copy Services:

- **PTC:** Point-in-Time Copy, also known as FlashCopy:
  - PTC - inactive 7200
  - PTC - 1 TB 7201
  - PTC - 5 TB 7202
  - PTC - 10 TB 7203
  - PTC - 25 TB 7204
  - PTC - 50 TB 7205
  - PTC - 100 TB 7210
  - PTC - 200 TB 7215
- **SE:** FlashCopy SE:
  - SE - inactive 7300
  - SE - 1 TB 7301
  - SE - 5 TB 7302
  - SE - 10 TB 7303
  - SE - 25 TB 7304
  - SE - 50 TB 7305
  - SE - 100 TB 7310
  - SE - 200 TB 7315

Archive

6.2 DS Storage Manager GUI support

The DS Storage Manager GUI panels, which are used to define the licensed functions and apply the activation codes, reflect the various licences, as illustrated in Figure 6-1 on page 63, for the Apply Activation codes panel.

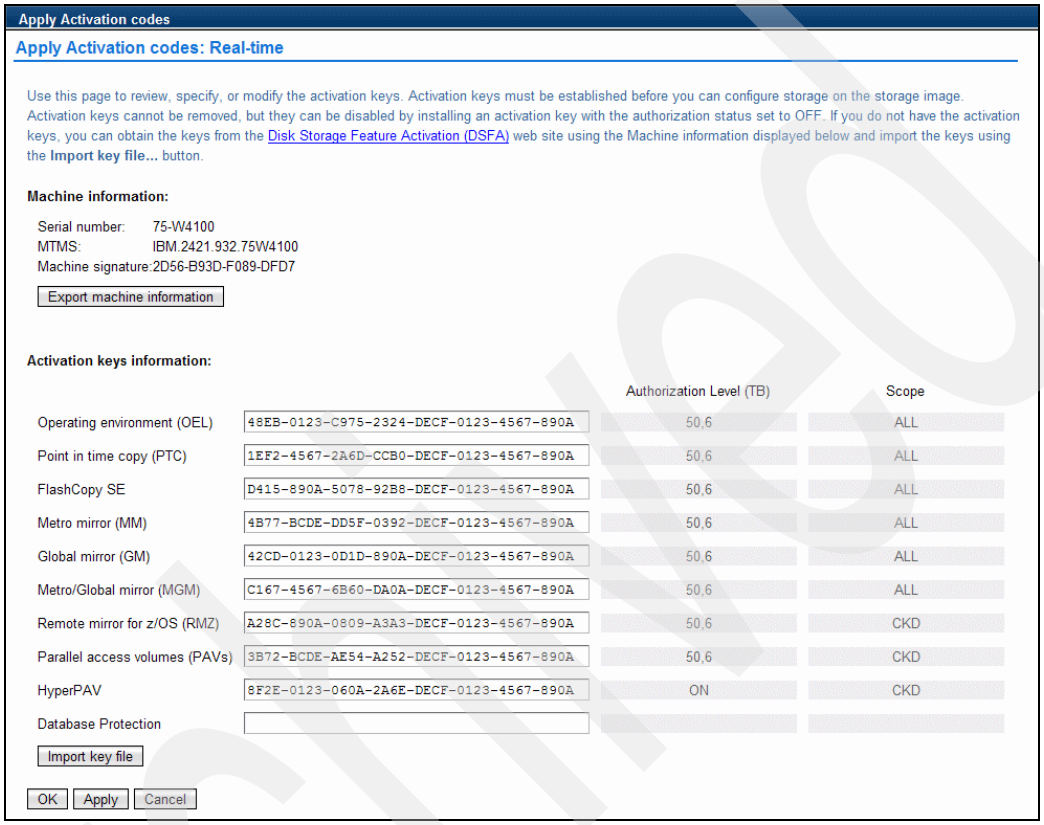


Figure 6-1 Apply Activation Codes panel

6.3 Authorized level

All Copy Services functions require licensing to be activated, which means that you must purchase a license for the appropriate level of storage for each Copy Service function that is required. You then have to install the License Key that is generated using the Disk Storage Feature Activation application (*DSFA*) that is located at the following Web site:

`http://www.ibm.com/storage/dsfa`

Another consideration relates to the authorized level required. In most cases, the total capacity installed must be licensed. This is the total capacity in decimal TB equal to or greater than the actual capacity installed, including all RAID parity disks and hot spares.

An exception might be where a mix of both System z™ and open systems hosts are using the same storage server. In this case, you can acquire Copy Services licenses for just the capacity formatted for CKD or just the capacity formatted for FB storage. This implies that the licensed Copy Services function is required only for open systems hosts or only for System z hosts. If, however, a Copy Services function is required for both CKD and FB, then that Copy Services license must match the total configured capacity of the machine. The authorization level is maintained by the licensed code in the controller and the DSFA application.

Archive

The actual ordered level of any Copy Service license can be any level above that required or installed. Licenses can be added and have their capacities increased, non-disruptively to an installed system.

6.3.1 Charging example

You can choose to purchase any or all DS8000 licenses at an authorization level at or later than the installed raw disk capacity. It may be more cost effective to pre-install authorization to use greater than the currently installed storage capacity.

A simple rule for charges is remembering that the required authorization level depends upon the license scope:

- If the license scope is FB, the authorization level must be equal to or greater than the total amount of physical capacity within the system that will be logically configured as FB.
- If the license scope is CKD, the authorization level must be equal to or greater than the total amount of physical capacity within the system that will be logically configured as CKD.
- If the license scope is ALL, the authorization level must be equal to or greater than the total system physical capacity.

The dual Storage Facility Image (SFI) version of DS8300 (the model 9Bx) uses the same logic, in this case added between the two SFIs. So all of the above apply to each SFI separately. It is possible to license Copy Services for a single *scope* in one SFI.

Tip: It is possible to license Copy Services for a single *scope* in one SFI.

Here is an explanation of this scenario:

- If you activate a licensed function on only SFI 1 with a license scope of FB:

  The authorization level must be equal to or greater than the total amount of physical capacity within SFI 1 that will be logically configured as FB.
- If you activate a licensed function on SFI 1 and SFI 2, both with a license scope of CKD:

  The authorization level must be equal to or greater than the total amount of physical capacity within both SFI 1 and SFI 2 that will be logically configured as CKD.
- If you activate a licensed function on SFI 1 with a license scope of FB and on SFI 2 with a license scope of ALL:

  The authorization level must be equal to or greater than the total amount of physical capacity within SFI 1 that will be logically configured as FB and the total physical capacity within SFI 2.

Archived

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics that we cover in this paper.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 66. Note that some of the documents referenced here may be available in softcopy only:

- *IBM System Storage DS8000 Series: Architecture and Implementation*, SG24-6786.
- *IBM System Storage DS8000 Series: Copy Services in Open Environments*, SG24-6788
- *IBM System Storage DS8000 Series: Copy Services with IBM System z*, SG24-6787

Other publications

These publications are also relevant as further information sources:

- IBM System Storage DS8000 Introduction and Planning Guide, GC35-0515
- *IBM System Storage DS8000: User’s Guide*, SC26-7623
- *IBM System Storage DS8000: Command-Line Interface User´s Guide*, SC26-7916
- *z/OS DFSMS Advanced Copy Services*, SC35-0428

Online resources

The following Web sites are also relevant as further information sources:

- IBM System Storage DS8000 FlashCopy SE Implementation Considerations and Recommendations:

  `http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10617`

- Documentation for the DS8000:

  `http://www.ibm.com/systems/storage/disk/ds8000/index.html`

- The IBM System Storage Interoperation Center:

  `http://www.ibm.com/servers/storage/support/config/ess/index.jsp`

- IBM Disk Storage Feature Activation (DSFA) Web site:

  `https://www.ibm.com/storage/dsfa/index.jsp`

Archived

© Copyright IBM Corp. 2008. All rights reserved.

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Archived

Index

© Copyright IBM Corp. 2008. All rights reserved.

Archived

Archived

Archived

IBM System Storage DS8000 Series: IBM FlashCopy SE

Redpaper™

Concept and design

Implementation and usage examples

Guidelines and recommendations

In October 2007, IBM announced a space efficient FlashCopy capability for the DS8000 platform. We call this new function IBM FlashCopy SE. It is a licensed function of the DS8000 family. It does not replace, but rather complements the standard FlashCopy function.

In this Redpaper, we start with a general overview of FlashCopy SE, where we discuss its concepts and intended usage. We summarize the concepts of the standard FlashCopy function and review its different options in the context of FlashCopy SE. We also have a detailed discussion of the FlashCopy SE design and implementation. As an added benefit, we provide guidance for the definition and usage of FlashCopy SE from both an open system and z/OS environments and provide illustrations using the DS GUI or DSCLI interfaces.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

REDP-4368-00