Redbooks
ibm.com/redbooks

Implementing IBM Real-time Compression on the IBM XIV Storage System

Guenter Rebmann

Albert Verhoeff

Megan Gilge

Cloud

Storage

IBM

Redpaper

International Technical Support Organization

Implementing IBM Real-time Compression on the IBM XIV Storage System

November 2015

REDP-5216-00

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

First Edition (November 2015)

This edition applies to Version 11, Release 6, Modification 0 of IBM XIV Storage System Gen3 software (product number 5639-YYB).

© Copyright International Business Machines Corporation 2015. 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

Notices v
Trademarks vi

Preface vii
Authors vii
Now you can become a published author, too viii
Comments welcome viii
Stay connected to IBM Redbooks viii

Chapter 1. Overview 1
1.1 Current IT challenges 2
1.2 IBM Real Time Compression: Addressing requirements for both capacity reduction and high performance 2
1.3 Common use cases 4
1.3.1 General-purpose volumes 5
1.3.2 Databases 6
1.3.3 Virtualized infrastructures 6

Chapter 2. Compression technology overview 7
2.1 Compression concepts 8
2.1.1 Lossy and lossless data compression 9
2.2 Data efficiency technologies 10
2.2.1 Thin provisioning technology 11
2.2.2 Snapshots 11
2.2.3 Archiving and space management 12
2.2.4 Data deduplication 12

Chapter 3. IBM Real-time Compression 13
3.1 IBM Real-time Compression overview 14
3.2 Random Access Compression Engine (RACE) 14
3.2.1 Traditional data compression in storage systems 14
3.2.2 Comparison of data chunks with traditional methods and RACE 16
3.2.3 Temporal locality 17
3.3 IBM Real-time Compression implementation on the XIV Gen3 storage system 19
3.3.1 XIV software and data distribution 19
3.3.2 Compression architecture 20
3.3.3 Compressing and decompressing existing volumes 22
3.3.4 Mirroring 23

Chapter 4. Planning 25
4.1 Candidate data sets for compression 26
4.2 Requirements 27
4.3 Operational limits 27
4.4 IBM Storage Volume Controller environments 28
4.5 Compression and XIV features 29
4.6 Licensing 29
4.7 The Comprestimator tool 30
4.7.1 Integrated Comprestimator tool 30
4.7.2 Host-based Comprestimator tool 31

© Copyright IBM Corp. 2015. All rights reserved.

4.8 Compression ratio guidelines 32
4.8.1 Guidelines for getting a high compression ratio 32
4.9 Additional space utilization guidance 33

Chapter 5. Configuring compressed volumes 35
5.1 Enabling compression. 36
5.1.1 Enabling IBM Real-time Compression with the GUI 36
5.1.2 Using XCLI commands to enable IBM Real-time Compression 37
5.2 Configuring compressed pools and volumes on XIV 37
5.2.1 Configuring a thin pool for compression 37
5.2.2 XCLI examples for thin pools 40
5.2.3 Using the GUI to change a regular pool to a thin pool 40
5.2.4 Using the GUI to change a regular pool to a thin pool 42
5.2.5 Creating a compressed volume with the GUI 43
5.3 Converting volumes 45
5.3.1 Converting a conventional volume to a compressed volume 45
5.3.2 Compressing single volumes 46
5.3.3 Compressing all volumes in a pool 48
5.3.4 Converting a compressed volume to conventional volume. 51
5.3.5 Decompressing all volumes in a pool 54
5.4 Compressing a mirrored volume 56
5.4.1 Deactivating a mirror relationship 56
5.4.2 Deleting mirrors, snapshots, and destination volumes 58
5.4.3 Compressing the volumes. 60
5.4.4 Creating and activating the mirror. 61
5.5 Disabling compression 64

Chapter 6. Performance 65
6.1 Performance overview 66
6.1.1 Compression and system performance 68
6.1.2 Converting uncompressed volumes 68
6.1.3 Effect of compression on performance for common XIV functions 69
6.2 Appropriate workloads for compression 69
6.3 Application benchmark results 70
6.3.1 Test case 1: Oracle online transaction benchmark. 71
6.3.2 Test case 2: Synthetic workloads 73
6.3.3 Test case 3: VDbench maximum throughput in MBps 75

Chapter 7. Maintenance and troubleshooting 77
7.1 Analyzing IBM Real-time Compression issues 78
7.1.1 Event log 78
7.1.2 GUI log 81
7.2 Involving IBM Support. 81
7.2.1 Information to provide 81
7.2.2 Enhanced Customer Data Repository (ECuRep) 84

Related publications 85
IBM Redbooks 85
Online resources 85
Help from IBM 85

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 grant 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

AIX®	Real-time Compression™	Storwize®
DB2®	Real-time Compression Appliance®	System Storage®
FlashSystem™	Redbooks®	XIV®
IBM®	Redpaper™	
IBM FlashSystem®	Redbooks (logo) ®	

The following terms are trademarks of other companies:

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Linear Tape-Open, LTO, the LTO Logo and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

2015 SUSE LLC. All rights reserved. SUSE and the SUSE logo are registered trademarks of SUSE LLC in the United States and other countries.

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

Preface

IBM® Real-time Compression™ is fully integrated in the IBM XIV® Storage System Gen3 with software version 11.6. Real-time Compression provides the possibility to store 2 - 5 times more data per XIV system, without additional hardware. This technology also expands the storage-replication-related bandwidth, and can significantly decrease the total cost of ownership (TCO).

Using compression for replication and for volume migration with IBM Hyper-Scale Mobility is faster, and requires less bandwidth for the interlink connections between the IBM XIV storage systems, because the data that is transferred through these links is already compressed. IBM Real-time Compression uses patented IBM Random Access Compression Engine (RACE) technology, achieving field-proven compression ratios and performance with compressible data.

This IBM Redpaper™ publication helps administrators understand and implement IBM Real-time Compression on the IBM XIV Gen3 storage system.

Authors

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Guenter Rebmann is a certified XIV Product Field Engineer in Germany. He joined IBM in 1983 as a Customer Engineer for large-system clients. After 10 years of experience with all large system products, he joined the DASD-EPSG (EMEA Product Support Group) in Mainz. In 2009, he became a member of the XIV PFE EMEA-Team. He has more than 30 years of experience providing technical support for past and present high-end DASD products.

Albert Verhoeff is a Client Technical Specialist working in pre-sales for IBM storage solutions. He has a deep technical knowledge of storage solutions. He has 12 years of experience in IT, and has held roles in first-level support for IBM storage software. He has delivered enhanced technical support for enterprise accounts, and worked as a product field engineer for EMEA and CEMEA on IBM N series (NetApp). His areas of expertise include not only storage but also hypervisors (VMware and Hyper-V), cloud orchestrators (OpenStack), Microsoft solutions, and SAN technology.

Megan Gilge is a Project Leader at the IBM International Technical Support Organization. Before joining the ITSO, she was an Information Developer in the IBM Semiconductor Solutions and User Technologies areas.

Special thanks to Rami Elron and Patrick Pollard for their contribution and extensive help in finalizing this document.

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

Wiley Clawson, Bert Dufrasne, Nir Rigai, Tzahi Shahak, Yossi Siles, Steve Solewin
IBM

Now you can become a published author, too

Here’s an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time. Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from 2 - 6 weeks in length, and you can participate either in person or as a remote resident working from your home base.

Learn more about the residency program, browse the residency index, and apply online:

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® publications in one of the following ways:

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

  ibm.com/redbooks

- Send your comments in an email:

  redbooks@us.ibm.com

- Mail your comments:

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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1 Overview

Continued data growth and economic pressures are driving the rapid adoption of data reduction technologies. Although much of the attention regarding data reduction has focused on backup, many businesses are applying data reduction throughout the entire data lifecycle.

Traditionally, data deduplication was generally used with highly redundant data sets found in backup-to-disk, virtual tape, or physical tape library applications. Data deduplication can provide an acceptable solution for sequential access workloads that are less sensitive to performance. However, it has limited use cases, and delivers savings in areas such as virtual desktop infrastructure (VDI) and backup.

Other data that is not repetitive by nature does not typically benefit from deduplication savings. Therefore, it cannot meet the demanding performance requirements of primary storage for random-access, high transaction volumes, and high throughput.

Compression is suitable for most enterprise workloads and use-cases. It is the most common form of data reduction technology for backups and backups to disk, and might be the most common data reduction technology in disk-based systems.

IBM Real-time Compression Software that is embedded in IBM XIV Storage System, IBM System Storage® SAN Volume Controller, IBM Storwize® V7000, and IBM FlashSystem™ V840 addresses all the requirements of primary storage data reduction, including performance. It does so by using a purpose-built technology called IBM Real-time Compression. This publication addresses the key requirements for primary storage data reduction.

This chapter describes the current challenges of data growth, and addresses how to overcome these challenges by using IBM Real-time Compression.

It includes the following sections:

- 1.1, “Current IT challenges” on page 2
- 1.2, “IBM Real Time Compression: Addressing requirements for both capacity reduction and high performance” on page 2
- 1.3, “Common use cases” on page 4

© Copyright IBM Corp. 2015. All rights reserved.

1.1 Current IT challenges

Today's data growth is rapidly expanding. Companies are increasingly using their data to gain insights into the business. Therefore, data is becoming more important, and requirements are set on storage devices. Because performance and up-time are part of these requirements, migration of primary workloads to lower-tiered devices is becoming unacceptable.

Shrinking IT budgets are pressuring IT managers to increase the lifetime of existing storage systems. Traditional methods of cleaningup unneeded data and archiving files to auxiliary storage are time consuming. They shift one resource constraint, physical storage, to another: The human work of storage administrators. Data growth is a factor or a combination of many different sources:

- Business growth of current applications
- New applications for new or business targets
- Big data and business analytics
- Compliance needs
- Mirrors
- Snapshots
- Clones
- Replicas
- Archiving
- Cloud solutions
- Hadoop environments
- Backup of this data

To comply with these requirements, companies are typically forced to install and manage even more storage devices, which typically results in increasing purchase costs in addition to maintenance, space allocation, power consumption, and operational expenses (OPEX).

Companies need to reduce data and maintain performance, which can be challenging with traditional compression methods.

Using IBM Real-time Compression provides an innovative approach that is designed to overcome these challenges.

1.2 IBM Real Time Compression: Addressing requirements for both capacity reduction and high performance

IBM Real-time Compression with the IBM XIV Storage System Gen3 effectively and efficiently answers a key requirement that typically challenges traditional approaches to capacity compression. It reduces capacity while maintaining high performance of the storage system.

Using IBM Real-time Compression reduces the amount of physical storage that is required in your environment. You can reuse free space in the existing storage without archiving or deleting data.

IBM Real-time Compression provides the following benefits:

- It delivers substantial savings across a versatile range of enterprise workloads.
- It uses the IBM Random Access Compression Engine (RACE) technology, which was purpose-built for real-life primary application workloads. IBM Real-time Compression takes advantage of data temporal locality to maximize data savings and system performance.

It is related to the effect of reading in the same (or similar) order to the write order, and is one of the unique RACE technology core pillars (for a detailed explanation of data temporal locality, see "Temporal locality" on page 17). RACE enables a larger number of read I/Os to be served from cache, potentially resulting in faster (application) response time.

- It is easy to use. An administrator simply selects a **Compressed** check box when creating a new compressed volume. For an existing volume, the system can display an accurate estimation of potential compression savings in the XIV GUI.

  Being able to easily assess potential savings before compressing existing data facilitates deployment of IBM Real-time Compression in existing environments. Furthermore, non-disruptive compression, or alternatively, conversion of uncompressed volumes to compressed volumes (and vice versa) for existing volumes can provide an easy way to reclaim capacity and accelerate ROI.
- It benefits from the XIV architecture, because evenly distributed compression load across system resources increases performance and efficiency.
- It benefits from the XIV scaling of performance and the amount of system resources that can be allocated to IBM Real-time Compression.
- It works with primary active data. Due to the system's ability to preserve high performance consistency with compression, IBM Real-time Compression can be used with active primary data. Therefore, it supports workloads that are not candidates for compression in other solutions and also supports compression of existing data.

IBM Real-time Compression uses the reliable, field-proven, and patented IBM Random Access Compression Engine (RACE) technology to achieve a valuable combination of high performance and compression efficiencies:

- It can lower the effective capacity requirements of a volume to 1/5 of the uncompressed capacity.
- No additional hardware is required to use IBM Real-time Compression.
- It can reduce cost for software that is licensed by capacity because less physical storage is required for compressed data.
- It can provide operational expense (OPEX) benefits because it requires no changes to the existing storage environment. It is fully integrated in the XIV Gen3 version 11.6, and allows for non-disruptive compression of volumes.
- It can be enabled without changing your existing storage environment (applications, hosts, networks, fabrics, or external storage systems). The solution is not apparent to hosts, so users and applications continue to work as-is.
- Compression occurs within the XIV system itself. The conversion from non-compressed to compressed is inline and does not require downtime.
- Compressed volumes provide an equivalent level of availability as regular volumes. Compression can be implemented into an existing environment without an impact to service (except for mirroring, which must be temporarily stopped while volumes are converted), and existing data can be compressed transparently while being accessed by users and applications.
- Compressed volumes can be mirrored, thereby minimizing requirements for replication bandwidth and capacity requirements on the target system, and correspondingly maximizing system performance due to the reduction of data to transfer.

  Remote volume copies are always compressed if the source is compressed. This process not only reduces storage requirements but also uses less bandwidth because the data is transferred compressed. Mirroring and Hyper-Scale Mobility are faster and requires less bandwidth because less data is transferred.

1.3 Common use cases

Compression savings are different on every data set. Some data sets can be compressed more than others, resulting in higher capacity savings. For example, image and video data, except for lossless types, are usually already compressed, and therefore IBM Real-time Compression might not provide additional savings.

This section addresses the most common use cases for implementing compression:

- General-purpose volumes
- Databases
- Virtualized infrastructures

Figure 1-1 on page 5 shows typical capacity savings with IBM Real-time Compression.

Understanding compression rates, ratios, and savings:

To clarify the meaning of the terms *compression ratio*, *compression savings rate*, and *compression savings*, consider a use case where the original data physical capacity before compression was 100 TB, and the physical data capacity after compression is 40 TB.

The following values help to clarify these terms:

- Compression rate = 60%
- Compression savings rate = 60%
- Compression savings = 60 TB
- Compression ratio = original size (100 TB) *divided by* the size on disk after compression (40 TB) = 2.5:1

When you consider savings, it is easiest to use the compression rate.

The compression ratio helps in understanding how much effective data you can store on your system. So, when you have a 2.5:1 compression ratio, you will be able to store 250 TB of data on 100 TB of physical capacity.

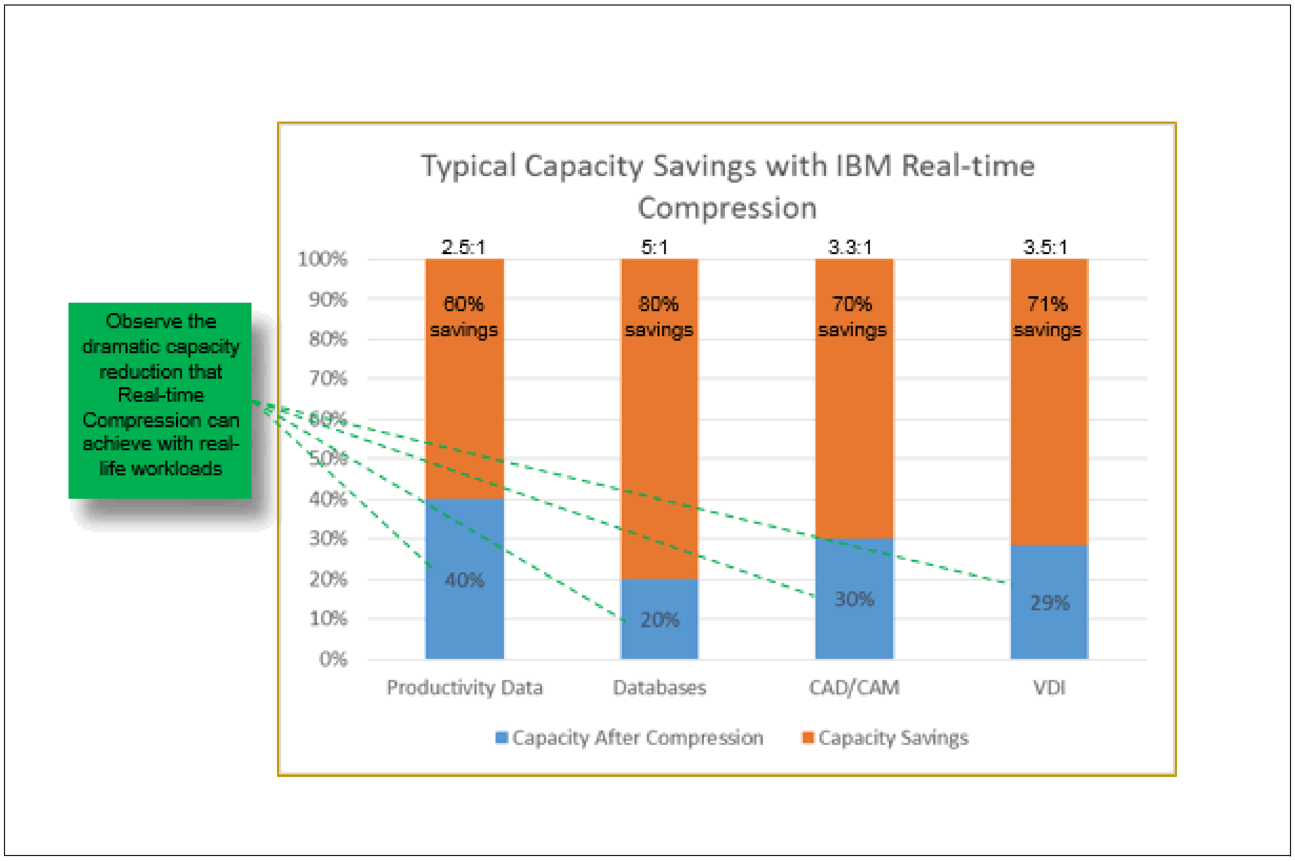


Figure 1-1 Typical capacity savings with IBM Real-time Compression

For more information about compressible data types, see 4.1, "Candidate data sets for compression" on page 26.

1.3.1 General-purpose volumes

Many general-purpose volumes are used for highly compressible data types, such as home directories, CAD/CAM, oil and gas data, and log data. Storing such types of data in compressed volumes can provide immediate capacity reduction to the overall used space. Thus, more space can be provided to users without any change to the environment.

Many file types can be stored in general-purpose volumes. The estimated compression ratios are based on field experience. Expected compression ratios when IBM Real-time Compression is used are between 2:1 and 5:1 (that is, data will use 50 - 20% of the space that it used before compression).

Expected compression ratios: Expected compression ratios refer to typical results when IBM Real-time Compression is used. If no direct assessment can be made, you can generally assume a 2:1 compression ratio.

1.3.2 Databases

Database information is stored in table space files. It is common to observe high compression ratios in relational database volumes. Examples of databases that can greatly benefit from IBM Real-time Compression are IBM DB2®, Oracle, and Microsoft SQL Server. Expected compression ratios are between 2:1 and 5:1 (that is, data will use 50 - 20% of the space that it used before compression).

Tip: Some databases offer optional built-in compression. In some cases, IBM Real-time Compression can provide more compression to save even more space.

1.3.3 Virtualized infrastructures

The proliferation of open systems virtualization in the market has increased the use of storage space, with more virtual servers, virtual desktop infrastructure (VDI), and backups kept online. The use of compression can reduce the storage requirements at the source.

Examples of virtualization solutions that can greatly benefit from IBM Real-time Compression are data stored under VMware, Microsoft Hyper-V, and KVM virtual servers volumes (data stores).

Expected compression ratios are between 1.67:1 and 4:1 (that is, data will use 60 - 25% of the space that it used before compression).

Tip: Virtual machines with file systems that contain already-compressed files are not good candidates for compression.

2

Compression technology overview

This chapter overviews compression concepts and provides a detailed description of IBM Real-time Compression. It includes the following topics:

- 2.1, “Compression concepts” on page 8
- 2.2, “Data efficiency technologies” on page 10

© Copyright IBM Corp. 2015. All rights reserved.

2.1 Compression concepts

In recent years, there have been dramatic changes in the demands placed on application servers and the amount of stored data. Therefore, new ways to optimize the capacity utilization are needed that simultaneously reduce cost and attain high performance. As data requirements have grown, the technologies that are available to reduce the amount of data stored or transported from one place to another have been greatly improved.

One of the first methods for reducing the amount of data was the use of symbols and representations in mathematical format. For example, instead of writing the words “multiplied by,” the related representation used is the asterisk character (*). In the same way, the word “minus” is represented with the dash character (-). In 1836, the invention of Morse code allowed messages to be transmitted quickly over long distances.

Roman letters and Arabic numbers were replaced with symbols formed from lines and dots. To reduce the number of dots or lines used to represent each letter, statistical analysis of the commonality of letters was performed. The most common letters are represented with a shorter combination of dots and lines. The commonality is different for each language.

For example, in the English language, the letter “s” is represented in the Morse code by three dots, whereas the letter “h” is represented by four dots. The representation of “sh” therefore consists of seven dots. However, in some languages “sh” is a common combination, so “sh” is represented by four lines, effectively saving transmission time.

Later in the 20th century, the development of IT technologies raised the need for complex algorithms able to reduce the amount of data. This compression is done by interpreting the information beyond the simple substitution of specific strings or letters.

One of the first techniques of mathematical data compression was proposed by Claude E. Shannon and Robert Fano in 1949. In the Shannon-Fano coding, symbols are sorted from the most probable to the least probable, and then encoded in a growing number of bits. If the source data contains A B C D E, where A is the most common and E is the least common letter, the Shannon-Fano coding is 00-01-10-110-111.

In 1952, a Ph.D. student at MIT named David A. Huffman proposed a more efficient algorithm for mapping source symbols to unique string of bits. In fact, Huffman proved that his coding is the most efficient method for this task, with the smallest average output bits per source symbol.

Later, Abraham Lempel and Jacob Ziv in 1977 proposed a method of replacing repeating words with code words. The method was also applicable to a pattern of text, such as expressions. This was the actual dawn of modern data compression. In 1984, Terry Welch improved the algorithm proposed by Lempel and Ziv (also known as LZ78) and developed a method that is known as LZW.

Today, this algorithm is the basis of modern compression techniques that are used in PKZIP for general file compression, or within GIF and TIFF formats for images. Over time, many data compression algorithms have been developed around the Lempel-Ziv method:

- LZSS (LZ - Storer-Szymanski)
- LZARI (LZ with Arithmetic encoding)
- LZH (LZ + Huffman encoding, which are used by the ARJ utility)

The IBM Real-time Compression Appliance® also uses compression based on LZH. For more information about both algorithms, see the following links.

Details about Lempel-Ziv coding can be found at the following websites:

- Lempel-Ziv explained:

  http://www-math.mit.edu/~shor/PAM/lempel_ziv_notes.pdf

- Lempel-Ziv coding:

  http://www.code.ucsd.edu/~pcosman/NewLempel.pdf

Details about Huffman coding can be found at the following websites:

- Huffman coding explained:

  http://www.cs.cf.ac.uk/Dave/Multimedia/node210.html

- Detailed Huffman coding:

  http://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/handouts/220%20Huffman%20Encoding.pdf

2.1.1 Lossy and lossless data compression

The compression of data has rapidly become a focus for the IT industry. Because of the different types of data and the reasons why data is compressed, two major compression methods are used:

- Lossless data compression: This method allows the information to be rebuilt completely, with no effect on the quantity or quality of the original information.
- Lossy data compression: This method synthesizes the information and keeps only the data that is needed. The original information cannot be rebuilt completely to its original form when the data is extracted.

Examples of lossless data compression include financial data, data of unknown origin, and all data that is always needed in its original format. Tape drives often have a built-in data compression. Tape compression algorithms must always be lossless because the drive does not know the data origin and the value of the data. An example of the need for lossless data compression is a bank account. The transactions on a bank account should all be visible, not just its value at the end of the day.

Examples of lossy data compression are audio, image, video, reports, and graphics that are generated to visualize large amounts of data and statistics. For example, audio compression keeps only information that is noticeable (audible) by the listener. Usually frequencies above 15k Hz are deleted by an audio compression algorithm. However, it is impossible to completely rebuild the original information.

Lossy data compression offers the advantage of higher compression rates and therefore higher storage savings. However, the original data cannot completely be re-created.

Lossless data compression offers the advantage of completely and accurately re-creating the input information. In comparison, the irreversible method offers only some specific information that is related to the original information.

Because of the massive amounts of data and calculations necessary for lossless compressing data, there are two approaches:

- In-line compression. This method processes the data before it is written to the storage device. The key advantage of this approach is that it reduces the storage resources that are required for a data set. If done correctly, the capacity-reduction application preserves the inherent performance of the storage environment. Already optimized data is written to storage, which mitigates the capacity explosion challenge at the point of origin.

It accomplishes this mitigation by eliminating the need to allocate the additional storage capacity that is required by post-processing solutions. Because the primary storage is used, any compression technique must be run in real time and maintain the high availability features of the existing storage system.

- Post-process compression. This method might minimize latency for writing uncompressed data. However, it does not run in real time, which means it does not yield immediate capacity utilization benefits, might not be able to efficiently use conditions such as temporal locality, and it might result in performance challenges later when compressed data needs to be read.

  Extra space is required on the storage system while data is being compressed, and system resources are required to compress data after it is received. Post-process compression actually reduces utilization, rather than capacity requirements. When doing post-process compression, you need enough space to store the original uncompressed data. Then, if you are able to incorporate the post-process into your schedule, you might be able to periodically reduce the utilization.

Over the years, IBM has introduced a series of lossless, in-line compression algorithms and solutions that are used in a wide range of technologies:

- The LTO-DC algorithm that is used in IBM Linear Tape-Open, known as LTO tape drives
- The Streaming Lossless Data Compression (SLDC) algorithm that is used in IBM Enterprise-class TS1130 tape drives
- The Adaptive Lossless Data Compression (ALDC) used by the IBM Information Archive for its disk pool collections
- The Random Access Compression Engine (RACE) used in IBM Storwize V7000, SAN Volume Controller, and IBM FlashSystem V840 and V9000, and in the XIV storage system

2.2 Data efficiency technologies

Data compression technologies can be found in various implementations over the last 15 years. These range from compression at the application level to in-band solutions that can compress data as it is transferred from a host to a storage device. Recently, many new technologies and new concepts have been developed to address the need for optimized storage space and more efficient capacity usage. Some of these technologies are addressed briefly in this chapter to avoid confusion between data compression and other space optimization methods.

Comparison: All of the technologies that are presented in this section are used to optimize the way that the available storage capacity is used. None of those technologies change information, but rather optimize how this information is stored or how much of this information is duplicated within the storage system. Data compression technologies can interpret data. They can provide storage optimization, both regarding the amount of data that is stored and the performance that is needed for the storage system that is used in back-end systems.

The listed technologies arguably place the major significance on a single aspect: maximized capacity utilization. IBM Real-time Compression provides two important benefits: Maximized capacity utilization *and* high performance.

2.2.1 Thin provisioning technology

Thin provisioning enables the storage to present the required capacity to the host while allocating only the actual used capacity in terms of space on the physical storage media. Figure 2-1 shows the difference between a traditional volume, a thin-provisioned volume and a compressed volume.

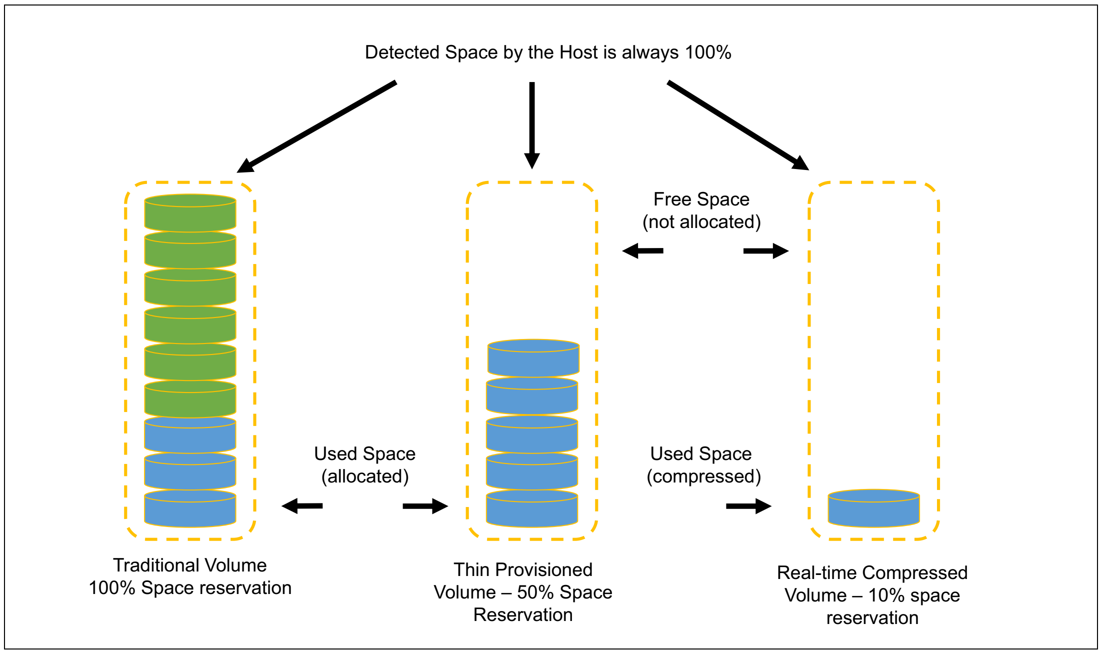


Figure 2-1 Thin provisioning explained

2.2.2 Snapshots

Snapshots is a technique that allows for creation of multiple logical PiT (point-in-time) copies of a volume, without creating multiple physical copies of volume data. This technology allows the copy to depend on the source with all of the data that is in common. When data is changed from the original volume, those changes are available and reflected only in the updated instance. The rest of the data remains as a common foundation.

IBM XIV uses the redirect-on-write technology that is a differential copy instead of a full copy of the original data. This method is similar to copy-on-write, without the double-write penalty, and it offers storage-space and performance-efficient snapshots. New writes to the original volume are redirected to another location set aside for the snapshot. The advantage of redirecting is that only one write takes place, whereas with copy-on-write, two writes occur, one to the copy original data on the storage space, and the other to the changed data.

2.2.3 Archiving and space management

Archiving and space management, also known as information lifecycle management (ILM), is a concept rather than a dedicated technology solution. It can be used in environments where you need long-term storage that optimizes the space that is occupied on different storage types, such as tapes and hard disk drives (HDDs).

The old or rarely used data can be moved out of the production area and placed on much cheaper storage. That is, moving data from enterprise storage systems to low-end storage systems, and from HDDs to tapes. The historical information is usually kept in a database that holds metadata to provide accurate information about the managed data.

Archiving warrants additional storage. The context of archiving within data efficiency might be the potential ability to employ ultra aggressive compression and decompression, or deduplication techniques that might be viable in terms of performance only if the data is expected to be accessed relatively rarely, as might be expected with archived data (to be compressed).

2.2.4 Data deduplication

The data deduplication mechanism identifies identical chunks of data within a storage context and keeps only one copy of each chunk. All the other logically identical chunks are pointed to this chunk. Figure 2-2 shows that only the pointers to the data are stored on the Target storage device. In this example, there are three files that are stored on a storage device that support deduplication. Every file consists of 5 blocks, though some of the blocks are the same. A storage device that supports deduplication can detect these duplicates and stores only one copy of the duplicate blocks. There are various implementations of this method:

- In-line deduplication
- Post-processing

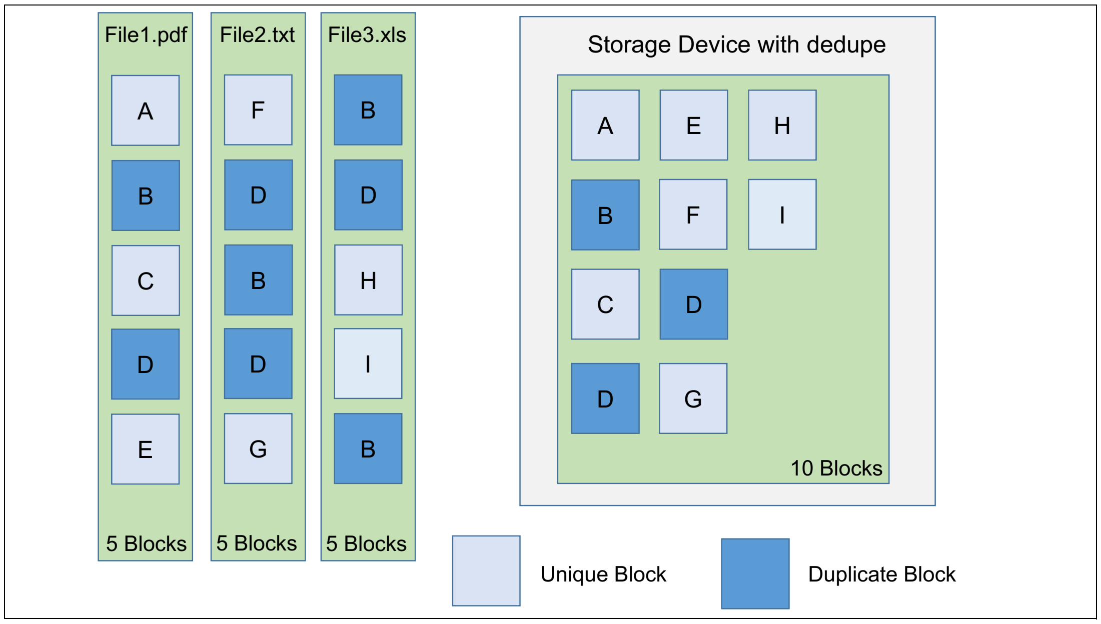


Figure 2-2 Deduplication in action on a storage device

In-line solutions deduplicate information as it is stored. Post-processing solutions deduplicate data at certain time intervals after the information is stored in the original format.

3

IBM Real-time Compression

This chapter introduces the IBM Real-time Compression technology, which is fully embedded into the IBM XIV Gen3 running software level 11.6 and higher. It addresses the basics of the IBM Random Access Compression Engine (RACE) technology. RACE is seamlessly integrated into the existing software stack in a fully transparent way. This integration does not alter the behavior of the system, so previously existing features are supported for compressed volumes.

This chapter includes the following sections:

- 3.1, “IBM Real-time Compression overview” on page 14
- 3.2, “Random Access Compression Engine (RACE)” on page 14
- 3.3, “IBM Real-time Compression implementation on the XIV Gen3 storage system” on page 19

© Copyright IBM Corp. 2015. All rights reserved.

3.1 IBM Real-time Compression overview

To understand the basic design of the IBM Real-time Compression technology, you must understand the basics of modern compression techniques, which include the Lempel-Ziv algorithm and Huffman coding, as explained in Chapter 2, "Compression technology overview" on page 7.

RACE is based on the Lempel-Ziv lossless data compression algorithm that operates in a real-time method. IBM XIV uses RACE *above the cache*, which means that data is compressed before it is written to the cache and is decompressed after it is read from cache. When a host sends a write request, the data is first compressed and then acknowledged by the system, and then written to cache and staged to the storage pool.

For more information about the write and read flow using compressed volumes, see 3.3.2, "Compression architecture" on page 20. For information about performance, see Chapter 6, "Performance" on page 65.

Storage system capacity can be saved because data written to the storage pool is already compressed.

IBM Real-time Compression is flexible because it does not have a rigid approach to workload and implements a flexible approach to compress workloads with varying characteristics. For example, instead of capturing compressed data in identically sized chunks, it varies the size. It is adapted to the workload that runs on the system at any particular moment.

3.2 Random Access Compression Engine (RACE)

RACE technology is based on over 70 patents that are not primarily about compression. Rather they define how to make industry-standard LZ compression of primary storage operate in real time and allow random access.

RACE is based on the Lempel-Ziv lossless data compression algorithm (described in 2.1, "Compression concepts" on page 8) that operates in a real-time method. When a host sends a write request, data passes through the compression engine, and is then sent to the cache. Writes are therefore acknowledged immediately after they are received by the write cache, and are staged to storage pools.

To understand why RACE is unique, you need to consider it in comparison to the traditional compression techniques (described in 2.1, "Compression concepts" on page 8). This description is not about the compression algorithm itself, that is, how the data structure is reduced in size mathematically. Rather, the description is about how it is laid out within the resulting compressed output.

3.2.1 Traditional data compression in storage systems

Compression is probably most known to users because of the widespread use of compression utilities that are used for file compression. At a high level, these utilities take a file as their input, and parse the data by using a sliding window technique. Repetitions of data are detected within the sliding window history, most often 32 KB. Repetitions outside of the window cannot be referenced. Therefore, the file cannot be reduced in size unless data is repeated when the window "slides" to the next 32 KB slot.

Figure 3-1 shows an example of how the data is broken into fixed size chunks (in the upper-left side of the figure). It also shows how each chunk gets compressed independently into chunks with potentially difference sizes (in the upper-right side of the figure). The resulting compressed chunks are stored sequentially in the compressed output.

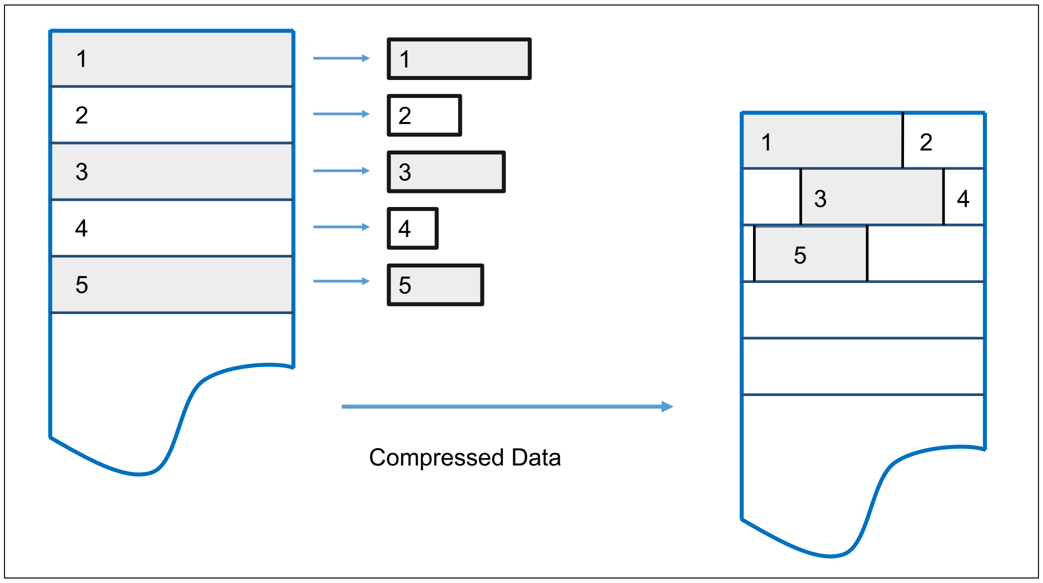


Figure 3-1 Traditional data compression

Compression utilities process files (or objects) in a batch manner, so the utility works on the complete file after it has been created. When this traditional approach is used with storage systems, they may compress the data in-flight, without waiting for the full file (or object) to be written and only then compress it.

However, there are drawbacks to this approach. An update to a chunk requires a read of the chunk followed by a recompression of the chunk to include the update. The larger the chunk size that is chosen, the heaver I/O penalty to recompress the chunks. If a small chunk size is chosen, the compression ratio is potentially reduced, because the repetition detection potential is reduced.

Although this approach is an evolution from compression utilities because it processes data inline instead of batch processing an existing file or object as compression objects do, it is limited to low performance use cases. This limitation is mainly because it does not provide real random access to the data.

The traditional approach that is taken to implement data compression in storage systems is an extension of how compression works in compression utilities (see “Comparison of data chunks with traditional methods and RACE”). Similar to compression utilities, the incoming data is broken into fixed chunks, and then each chunk is compressed and extracted independently.

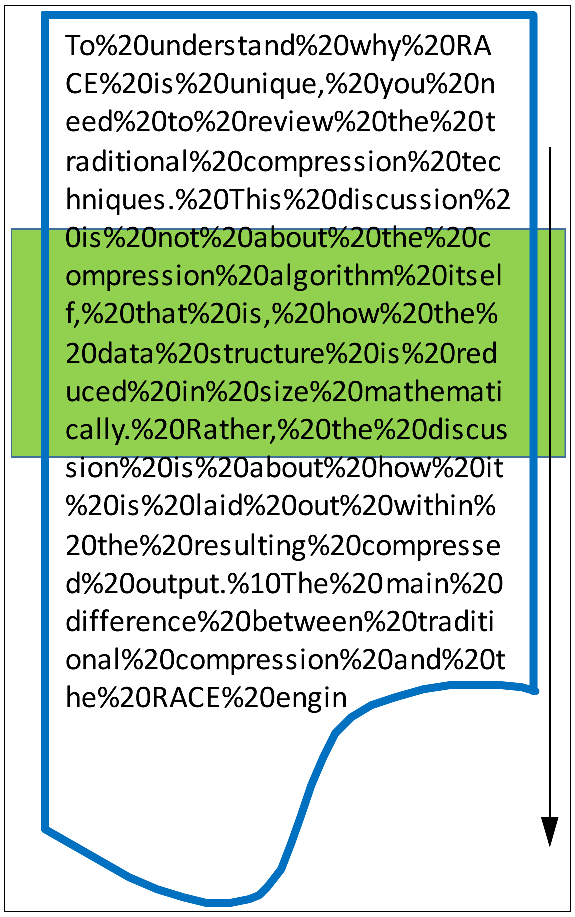
3.2.2 Comparison of data chunks with traditional methods and RACE

A major difference between traditional compression and the RACE is in the size of data chunks that are written to the storage device.

Fixed size writes: Traditional compression writes variable size chunks to the storage device, whereas RACE writes fixed size output chunks. This difference is the basis of many of its benefits.

This method enables an efficient and consistent method to index the compressed data because it is stored in fixed-size containers.

Figure 3-2 shows compression that uses a sliding window.



To%20understand%20why%20RA
CE%20is%20unique,%20you%20n
eed%20to%20review%20the%20t
raditional%20compression%20tec
hniques.%20This%20discussion%2
0is%20not%20about%20the%20c
ompression%20algorithm%20itsel
f,%20that%20is,%20how%20the%
20data%20structure%20is%20red
uced%20in%20size%20mathemati
cally.%20Rather,%20the%20discus
sion%20is%20about%20how%20it
%20is%20laid%20out%20within%
20the%20resulting%20compresse
d%20output.%10The%20main%20
difference%20between%20traditi
onal%20compression%20and%20t
he%20RACE%20engin

Figure 3-2 Compression using a sliding window

Variable size input: The IBM patented Random Access Compression Engine alters the traditional approach to compression. It uses variable-size chunks for the input, and fixed-size chunks for the output.

This method enables an efficient and consistent method to index the compressed data because it is stored in fixed-size containers.

Figure 3-3 shows Random Access Compression with fixed output chunks.

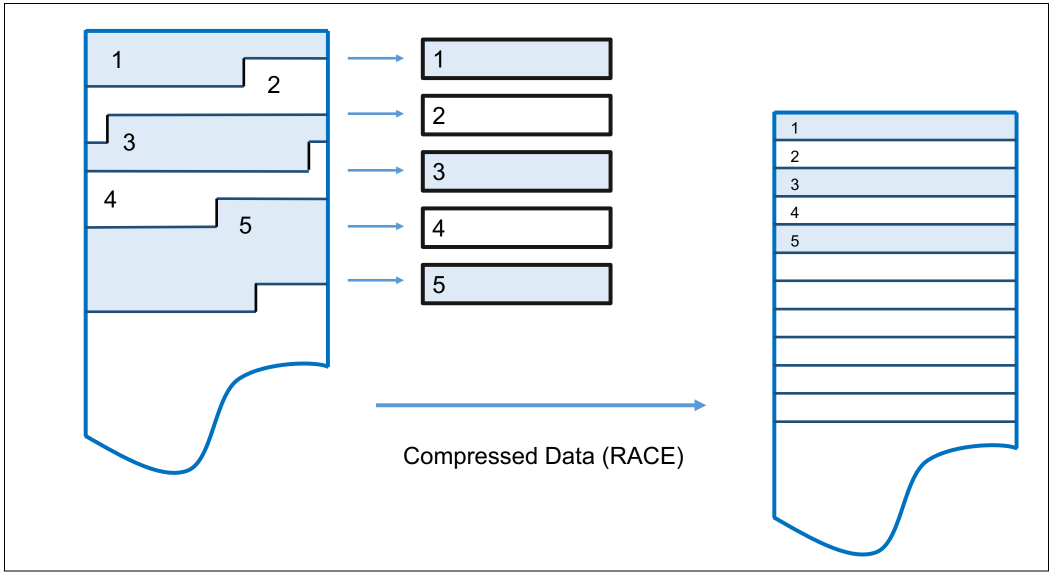


Figure 3-3 Random Access Compression Engine (RACE)

3.2.3 Temporal locality

Both compression utilities and traditional storage systems compression compress data by finding repetitions of bytes within the chunk that is being compressed. The compression ratio of this chunk depends on how many repetitions can be detected within the chunk. The number of repetitions is affected by how much the bytes stored in the chunk are related to each other. The relation between bytes is driven by the format of the object.

For example, an office document might contain textual information, and an embedded drawing (like Figure 3-3). Because the chunking of the file is arbitrary, it has no notion of how the data is laid out within the document. Therefore, a compressed chunk can be a mixture of the textual information and part of the drawing.

This process yields a lower compression ratio, because mixing the different data types together causes a suboptimal dictionary of repetitions. That is, fewer repetitions can be detected because a repetition of bytes in a text object is unlikely to be found in a drawing.

This traditional approach to data compression can also be called location-based compression. The data repetition detection is based on the location of data within the same chunk.

When host writes arrive to RACE, they are compressed and fill up fixed size chunks, also called *compressed blocks*. Multiple compressed writes can be aggregated into a single compressed block. A dictionary of the detected repetitions is stored within the compressed block.

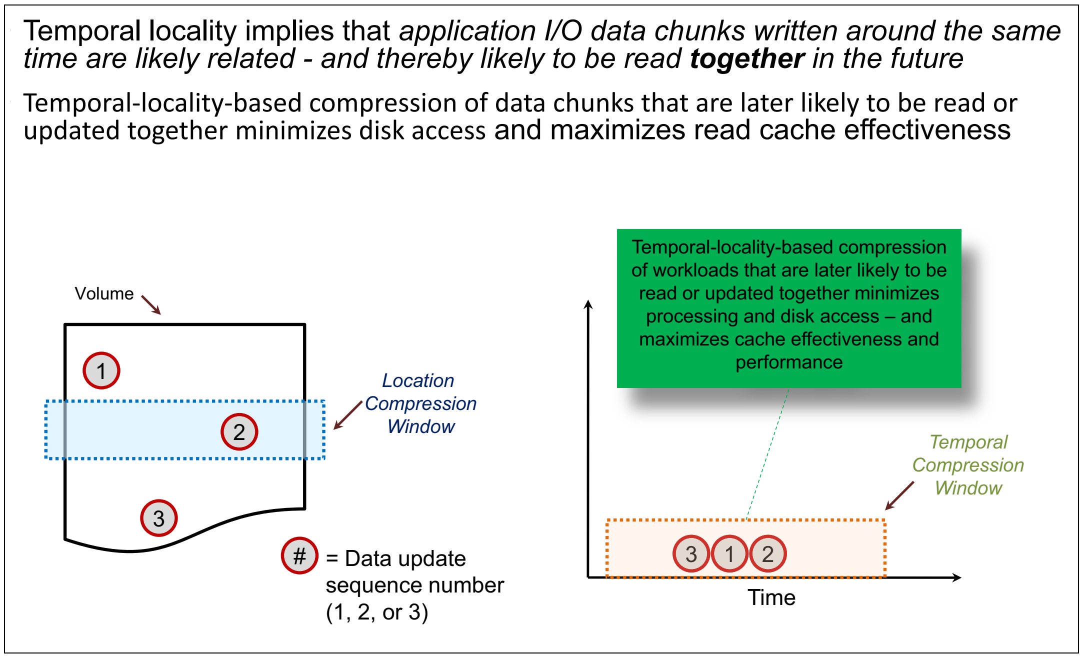
When applications write new data or update existing data, it is typically sent from the host to the storage system as a series of writes. Because these writes are likely to originate from the same application and be of the same data type, more repetitions are usually detected by the compression algorithm.

This type of data compression is called *temporal compression* because the data repetition detection is based on the time the data was written into the same compressed block.

Temporal compression: Temporal compression adds the time dimension that is not available to other compression algorithms. It offers a higher compression ratio because the compressed data in a block represents a more homogeneous input data

Figure 3-4 shows (in the left part of the figure) how three writes (listed in the figure by write number) sent one after the other by a host end up in different chunks. They get compressed in different chunks because their location in the volume is not adjacent. This yields a lower compression ratio because the same data must be compressed non-natively by using three separate dictionaries.

When the same three writes are sent through RACE (in the right part of the figure), the writes are compressed together by using a single dictionary. This yields a higher compression ratio than location-based compression.


Temporal locality implies that *application I/O data chunks written around the same time are likely related - and thereby likely to be read* ***together*** *in the future*

Temporal-locality-based compression of data chunks that are later likely to be read or updated together minimizes disk access and maximizes read cache effectiveness

Temporal-locality-based compression of workloads that are later likely to be read or updated together minimizes processing and disk access – and maximizes cache effectiveness and performance

Volume
1
2
3
Location Compression Window
# = Data update sequence number (1, 2, or 3)
Temporal Compression Window
3 1 2
Time

Figure 3-4 Location-based versus temporal compression

Applications often read those three writes back also with temporal correlation, in which case the read benefits from being written together. That is, when this data is read back, the read stream generally follows the same random (non-contiguous volume logical block address) pattern. Thus, the compression engine reads and extracts the larger chunk of data, which results in the next few random volume I/O reads by the host. This data is read from the data that has already been extracted by extracting the first large chunk. This process therefore results in what is essentially a cache hit in the compression cache memory.

With real-world applications, there is also, in general, no such thing as truly random I/O. The reality is that an application reads and writes objects or groups of data. These groups of I/O requests form a repeatable pattern, with the same group of I/O occurring one after another, even if they are to random locations on disk. IBM has invested heavily in understanding these patterns, and IBM Real-time Compression uses this understanding to maximize compression ratios and return the best performance.

Various application benchmark tools have shown that the compression performance in most cases is as good or better than thin-provisioned performance for an equivalent number of disks.

3.3 IBM Real-time Compression implementation on the XIV Gen3 storage system

This section provides an overview of how IBM Real-time Compression is implemented on the IBM XIV storage system. IBM Real-time Compression and the RACE engine in the IBM XIV Storage System is an exclusively software implementation that takes advantage of the XIV hardware design.

3.3.1 XIV software and data distribution

The software that drives the XIV is adapted to the described hardware design. XIV software comprises software nodes. Each software node is responsible for one or several tasks, and together they form the XIV Storage System.

To help you understand how and where IBM Real-time Compression is implemented, the following subsections provide a brief description of how the XIV Storage System stores the data on the disks, and the general data path for reads and writes.

Data distribution algorithms

Data is distributed across all drives in a pseudo-random fashion. Patented algorithms provide a uniform yet random distribution of data, which is divided into 1 MB partitions across all available disks, to maintain data resilience and redundancy.

All disks in the XIV Storage System are used evenly by partition units, regardless of applications or the size of assigned logical volumes. The storage administrator does not need to worry about data placement impact on performance. After the first volume is configured on the XIV, all read and write operations involve all modules and disks in parallel.

Data path

Reading and writing data from and to the XIV involve different software nodes. See Figure 3-5.

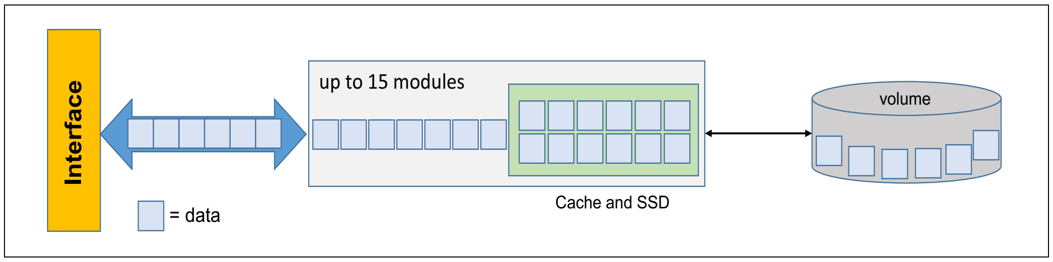


Figure 3-5 Read and write in the XIV

An interface module receives a read or write request from the host. Based on internal mapping tables, it knows on which modules the involved data (and corresponding 1 MB partitions) are located. So, the request (write or read) is forwarded to the modules that hold the required 1 MB partition(s). Writes are acknowledged to the host after the partition is in cache. Destage to disk is done according to sophisticated cache algorithms that are independent of the write requests in the background.

In large part, reads are done from cache or SSD because the XIV Storage System is working with elaborated pre-fetch algorithms, and the SSD is working as a large extension of the read cache. Because cache and SSD is installed in each module, the XIV overall cache and SSD size is very large.

This brief description covers the XIV design points and architecture information that is required to understand IBM Real-time Compression. For more detailed information, see the *IBM XIV Storage System Architecture and Implementation*, SG24-7659 Redbooks publication:

http://www.redbooks.ibm.com/abstracts/sg247659.html?Open

3.3.2 Compression architecture

There are different architectures to implement compression in storage systems. In the XIV Storage System, the key points are the *above cache* architecture and the IBM Random Access Compression Engine (RACE).

Above Cache architecture

Above cache architecture means that data is compressed before it is written to the cache and is decompressed after it is read from cache. All data in cache and SSD (as an extension of read cache) is compressed, so that effectively more data is in cache.

Compression engine RACE

IBM Real-time Compression uses patented IBM Random Access Compression Engine (RACE) technology, achieving field-proven compression ratios and performance with compressible data.

For more detailed information about how RACE compresses data, see 3.2, “Random Access Compression Engine (RACE)” on page 14.

The RACE engine is embedded in a new software node in each module, so the XIV Gen3 can have up to 15 RACE engines, depending on the XIV rack configuration. These RACE engines compress and decompress data in parallel. This implementation allows IBM Real-time Compression to fully participate in XIVs massive parallelism. See Figure 3-6.

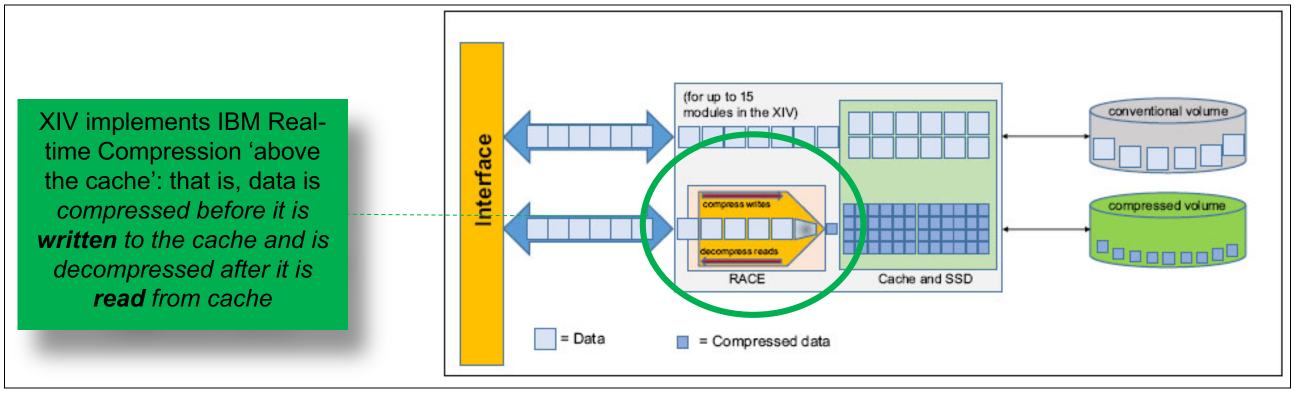

XIV implements IBM Real-time Compression ‘above the cache’: that is, data is *compressed before it is* ***written*** *to the cache and is decompressed after it is* ***read*** *from cache*
Interface
(for up to 15 modules in the XIV)
compress writes
decompress reads
RACE
Cache and SSD
conventional volume
compressed volume
= Data
= Compressed data

Figure 3-6 RACE implementation in the XIV modules

The RACE engine requires 4 GB of RAM when compression is enabled. By default, it is disabled and it must be enabled by using the GUI or the XCLI. See 5.1, “Enabling compression” on page 36.

IBM Real-time Compression is based on *compression objects*. In file system compression, or in other integrations, a *compression object* is a file or volume. On the XIV storage system, with its unique data distribution algorithm, a *section* represents a RACE compression object.

To allow multiple RACE engines to work on a single XIV volume (which is distributed through all modules and disks) in parallel, the volume is divided into *regions*. Several regions are merged to *sections*, and each section is handled by a different compression engine. See Figure 3-7. The graphic only shows a few regions, sections, and compression engines. There can be up to 15 compression engines (one per module).

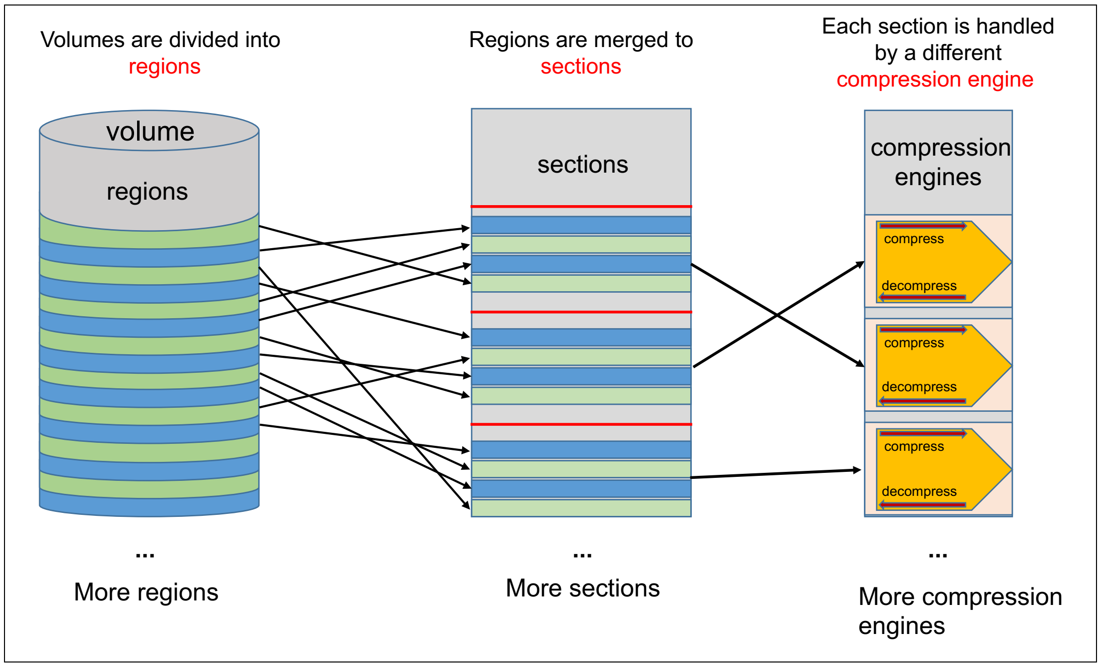


Figure 3-7 Regions and sections

The mapping between sections and the compression engine is maintained in a table. If a module failure or a compression engine failure occurs, this table is used to redistribute and rebalance the affected sections to other RACE engines (failover).

Due to the above cache architecture, data is compressed or decompressed before entering the cache. Data in cache and SSD (as an extension of the read cache) is stored compressed. Therefore, effectively more data is cached.

IBM Real-time Compression and the RACE engine take advantage of the massively parallel XIV design. The additional workload that is generated by IBM Real-time Compression is distributed to all modules. Because of this massive parallel design, IBM Real-time Compression in XIV has almost no adverse effect on the overall performance of the XIV. Indeed, because data is compressed above cache, there is more data in cache and SSD, and compressed volumes can have a better cache hit ratio and less latency.

Writes and reads

IBM Real-time Compression compresses a data stream as it is written. When RACE is used, an incoming write stream turns into a sequential stream of compressed data. This process occurs even if the incoming write stream is random. Thus, any random small block I/O write stream is converged into a single chunk of data to be compressed. This single chunk contains multiple independent writes that are each compressed independently. The compressed data is then written.

In real-life applications, when this data is read back, the compression engine reads the sequential stream of compressed data. Because this stream contains more data than is requested in this I/O request, the cache hit ratio might be effectively higher.

This implementation typically improves system performance because the data is compressed above cache, so you have more available cache. There is also less traffic if you destage down to the physical disk, meaning that XIV is required to handle less I/O.

For real-world applications, I/O is almost never random. The reality is that an application reads and writes objects or groups of data. These groups of I/O requests form a repeatable pattern, with the same group of I/Os occurring one after another, even if they are written to random locations on disk. IBM Real-time Compression analyzes these patterns to provide better compression ratios and return the best performance.

3.3.3 Compressing and decompressing existing volumes

Compressing and decompressing (converting) an existing volume is based on data migration with loop back. XIV creates an internal host to read the source volume (volume to be converted) and write the data to a new volume. During the conversion process, host mapping is set to the new volume.

The conversion is done without impacting normal host read and write operations. Reads will be redirected either to the uncompressed volume or to the compressed volume, depending on whether they are already converted. Writes are always made to the new volume.

Important: Since a new volume is created when compressing or decompressing an existing volume, it implies a new volume id for the converted volume. While this operation is transparent to the host, it may disrupt management tools that rely on the storage volume id to track changes.

Figure 3-8 shows the principle of how the XIV converts a conventional volume to a compressed volume. The reverse conversion from a compressed volume to a conventional volume works the same way, with swapped source and target volumes.

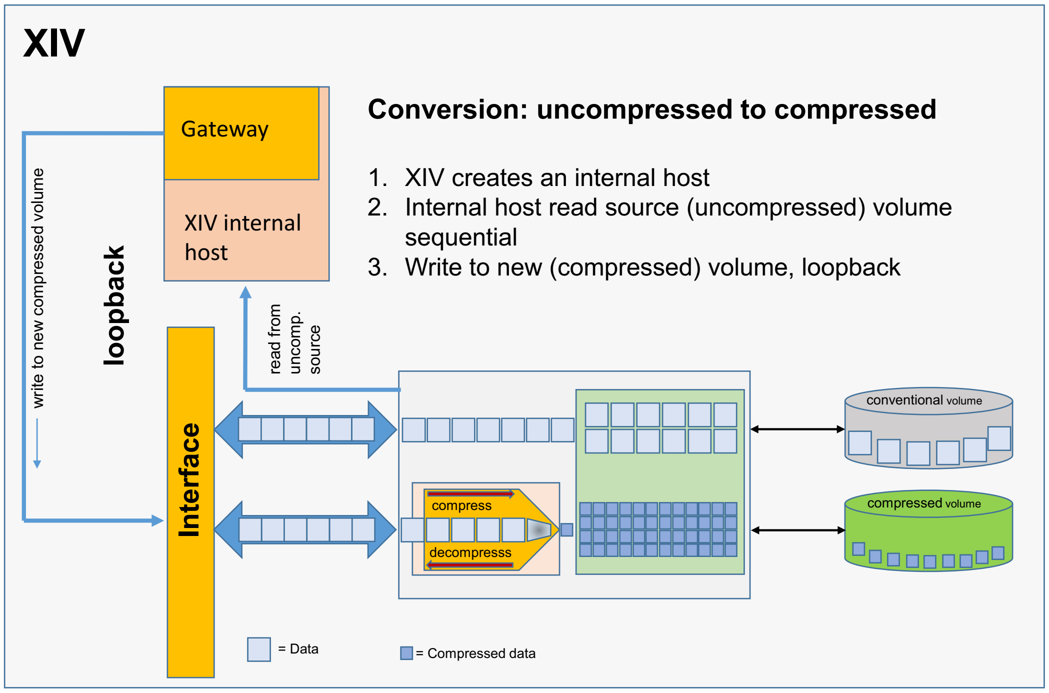

XIV
Gateway
XIV internal host
Conversion: uncompressed to compressed
1. XIV creates an internal host
2. Internal host read source (uncompressed) volume sequential
3. Write to new (compressed) volume, loopback
write to new compressed volume
loopback
read from uncomp. source
Interface
compress
decompresss
conventional volume
compressed volume
= Data
= Compressed data

Figure 3-8 Conversion

The XIV storage system makes a copy of the volume before the conversion, which makes it possible to cancel the operation.

During conversion setup, you can specify whether to keep or delete the copy after the conversion process successfully completes. Conversion is a low-priority task running in the system. Volumes must be converted one volume at a time to ensure that production I/Os are not affected.

There are several limitations for conversion:

- Volumes with snapshots cannot be converted with their snapshots.
- A mirror relationship must be broken before a mirrored volume can be converted.
- Additional space is required for the conversion process until the original volume is removed.

3.3.4 Mirroring

Mirroring (synchronous, asynchronous, and 3-way) benefits when IBM Real-time Compression is used. On the remote site, the same amount of disk space is saved, and compressed data is sent over the mirror link. Asynchronous mirroring, which is based on snapshots, also transfers compressed data to the remote site. This compression reduces the required bandwidth between all involved XIVs. The initialization time when mirrors are activated and the effective RPO can also decrease. See Figure 3-9.

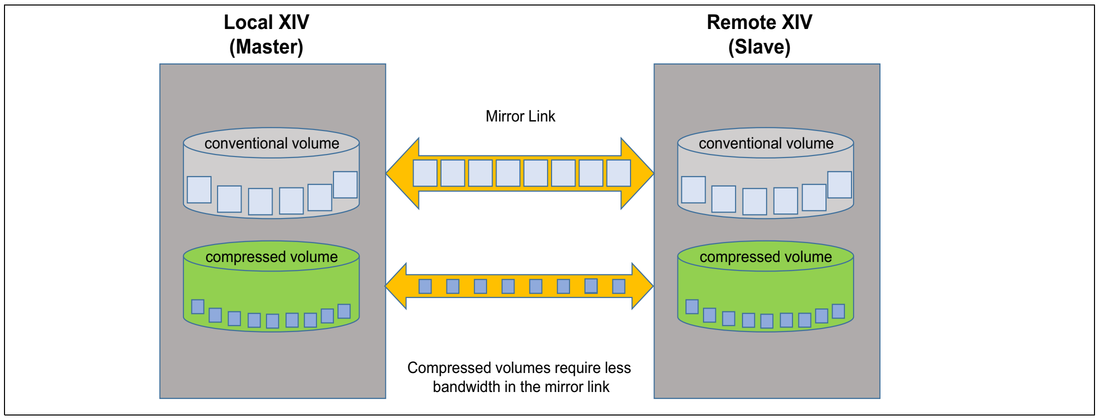


Figure 3-9 Mirror link

Data in mirrored environments is compressed only once, and arrives already compressed on the remote XIV storage system. The remote system detects that the data is already compressed, and it does not go through the compression engine again.

Note: To use compression in a mirrored environment, all included XIV systems must be Gen3 systems with a compression license and must have compression enabled. Both source and target volumes must be either compressed or uncompressed (no intermix).

Volumes in a mirror relationship (synchronous and asynchronous) cannot be compressed without deactivating and deleting the mirror. The steps to compress a mirrored volume are:

1. Deactivate the mirror.
2. Delete the mirror.
3. Make sure that the volume belongs to a thin pool.
4. Compress the volume.
5. Mirror the volume and make sure that the remote pool is a thin pool.

For a detailed procedure, see 5.4, “Compressing a mirrored volume” on page 56.

4

Planning

This chapter describes the path to a successful solution design for compressed volumes. It helps you estimate the compression savings for volumes on the IBM XIV Gen3 when you do not have detailed knowledge of the data. It also describes the limitations that must be considered before you implement IBM Real-time Compression.

This chapter contains the following information:

- 4.1, “Candidate data sets for compression” on page 26
- 4.2, “Requirements” on page 27
- 4.3, “Operational limits” on page 27
- 4.5, “Compression and XIV features” on page 29
- 4.6, “Licensing” on page 29
- 4.7, “The Comprestimator tool” on page 30
- 4.8, “Compression ratio guidelines” on page 32
- 4.9, “Additional space utilization guidance” on page 33

© Copyright IBM Corp. 2015. All rights reserved.

4.1 Candidate data sets for compression

The best candidates for data compression are data types that are not already compressed. Compressed data types are used by many workloads and applications, such as databases, character/ASCII based data, email systems, server virtualization, CAD/CAM, software development systems, and vector data.

The following examples represent workloads and data that are already compressed, and are therefore *not* good candidates for compression:

- Compressed audio, video, and image file formats
  File types such as JPEG, PNG, MP3, medical imaging (DICOM), and MPEG2
- Compressed user productivity file formats
  Microsoft Office 2007 newer formats (`.pptx`, `.docx`, `.xlsx`, and so on), PDF files, Microsoft Windows executable files (`.exe`), and so on
- Compressed file formats
  File types such as `.zip`, `.gzip`, `.rar`, `.cab`, and `.tgz`

IBM Real-time Compression is best suited for data that has the following characteristics:

- Data for which the Comprestimator tool estimates 25% or higher savings:
  - Database applications: Oracle, IBM DB2, Microsoft SQL, and SAP
  - Server/Desktop virtualization: VMware, KVM, and Hyper-V
  - Messaging: IBM Notes and Microsoft Exchange
  - Others: Engineering, collaboration, and seismic

  The Comprestimator tool: The Comprestimator tool estimates how much space can be saved when compression is used. An integrated version is available on the XIV system, and a host-based version can be used for data that is not stored on an XIV system. This tool is described in 4.7, “The Comprestimator tool” on page 30.

- Volumes that contain data that is not already compressed, such as image and video files

Data for which application-based encryption is used or data that is sent encrypted to the XIV. Self Encrypting Drives (SED) are supported.

Table 4-1 shows typical compression rates for common data types (for a correct interpretation of compression rates, refer to the box labeled **Understanding compression rates, ratios, and savings** under 1.3, “Common use cases” on page 4).

Table 4-1 Typical compression rates for common data types

Data types/Applications	Compression rate
Productivity	Up to 75%
Databases	Up to 80%
CAD/CAM	Up to 70%
Virtualization	Up to 75%

4.2 Requirements

Consider the following requirements for creating or converting compressed volumes:

- The system must be an XIV Gen3 model 114 or 214 that is running software version 11.6 or later.
- Compressed volumes can be created only in a thin-provisioned pool.
- A mirrored volume cannot be converted. The mirror relation must be removed and re-created after the conversion is done. See 5.4, “Compressing a mirrored volume” on page 56.
- You cannot convert an uncompressed volume that has snapshots. Snapshots must be deleted first. Alternatively, you can keep the source volume (this option is provided when you compress a volume).
- Space requirements:
  - Prior to enabling compression, the system must have a minimum of 17 GB of free hard space available. Enabling compression reserves 17 GB from the available system hard capacity. It is reserved for internal system use only.
  - Before the compression process, there must be enough space for both compressed and uncompressed versions of the volume.
  - Before decompressing the volume, the target space must be fully allocated.
  - Volumes must be at least 103 GB before compression.

For additional guidance, refer also to 4.9, “Additional space utilization guidance” on page 33.

4.3 Operational limits

At the time of writing, IBM Real-time Compression on the XIV Gen3 Storage System has the following limitations:

- There is no support for VMware VVols at the time of writing.
- Only Gen3 models 114 and 214 are supported.
- Up to 1024 volumes and snapshots can be compressed.
- The following limits apply to compression capacity:
  - System must have a minimum of 17 GB of free hard space to enable IBM Real-time Compression.

    The System will reserve 17 GB of hard space for internal use while IBM Real-time Compression is enabled.
  - Thin pools require a minimum of 17 GB of free hard space available to convert or transform volumes from uncompressed to compressed.
  - Thin pools require a minimum free soft space that is at least as large as the volume size that is being converted from uncompressed to compressed. See space requirements in 4.2, “Requirements” on page 27 for soft size guidelines.
  - When you decompress a compressed volume, you must have both free hard space at least the size of the uncompressed volume, and free soft space.

    It is a good practice to have free soft space at least the size of the uncompressed volume.

  - Minimum size of a volume
    - 17 GB (uncompressed)
    - 103 GB (compressed)
  - Maximum size of a volume
    - 161 TB (uncompressed)
    - 10 TB (compressed)
  - New, with XIV software V11.6, the Storage Admin now has the possibility to extend the system soft capacity. It is only possible through the **`system_soft_capacity_set soft_size=`**`SizeGB` XCLI command.

    For example, **`system_soft_capacity_set soft_size=`**`250000`.

    The soft capacity size of the system can be set to up to 3 times the size of the hard capacity of the system, and as low as the maximum size between the currently allocated soft capacity and the system's hard capacity (whichever is greater). The current hard, soft, or allocated soft capacity can be retrieved using the **`system_capacity_list`** command.

    Tip: Over-provisioning with Real-time Compression is safe, because compression ratios are predictable and stable.

- An uncompressed volume in a thick pool cannot be converted, because compression is supported only for thin volumes.
- Only one conversion process can be active at any time.
- Adding a module, rebuilding a disk, or upgrading the system suspends and then resumes the conversion process.

4.4 IBM Storage Volume Controller environments

Applying compression on both IBM Storage Volume Controller and IBM XIV is redundant and is not recommended. If IBM XIV is planned to be used behind an installation of IBM Storage Volume Controller that runs Real-time Compression, the correct practice is to deploy IBM XIV Gen3 214 without compression enabled.

4.5 Compression and XIV features

The following section shows how compression interacts with other features of IBM XIV Gen3. Table 4-2 shows significant features. It is not an exhaustive list.

Table 4-2 Significant features

Function	Comments
Hot upgrade	Supported.
Scrubbing	Supported.
Multitenancy	Supported.
Consistency Group	Supported.
Hot enable	The system frees RAM space for compression with dynamic cache resize.
Thin provisioning	The volume must reside in a thin pool to allow compression.
Snapshots and volume copy operations	Snapshots and volume copies work the same for compressed and uncompressed volumes. The snapshot or copy of a compressed volume is also compressed. A snapshot or volume copy of an uncompressed volume is also uncompressed.
Mirroring	Supports only when target is the same type as source. From compressed to compressed or uncompressed to uncompressed. No intermix is allowed. Remember that an existing mirror must be stopped before conversion (whether uncompressed to compressed or the opposite).
Offline initialization	Not supported for either sync or async.
Volume Resize	The RACE engine does not support shrinking a compressed volume. Resize up is supported.
Hyper-Scale Mobility	Supported as long as the source and target volumes are both compressed, or both uncompressed.
3-Way mirroring	All participants in a three-way mirror of a compressed volume must support compression.

4.6 Licensing

At the time of writing, enabling the Real-time Compression engine requires a chargeable software feature. This feature can be ordered with a XIV Gen3 system or as an upgrade (Gen3 models 114 and 214) on an existing system. For details, contact your IBM Sales representative.

The RACE engine is included in the V11.6 software and does not require any additional installation. Although no license code is required, you must purchase a license.

At the time of writing, a free trial is available, which allows you to try IBM Real-time Compression without purchasing a license. For more information, see the *IBM XIV Real-time Compression Evaluation User Guide*:

http://public.dhe.ibm.com/common/ssi/ecm/ts/en/tsj03645usen/TSJ03645USEN.PDF

4.7 The Comprestimator tool

It is important to determine which volumes are potentially worth compressing and which are not. The Comprestimator utility can be used to estimate an expected compression ratio by using the IBM Real-time Compression technology. The utility uses advanced mathematical and statistical algorithms to do the sampling and analysis process in a quick and efficient way.

There are two types of Comprestimator: The internal one and the host-based version. The following sections describe both versions.

4.7.1 Integrated Comprestimator tool

The IBM 11.6 software version provides the built-in Comprestimator function. This method is by far the easiest method to determine how well your volumes will compress. The internal XIV Comprestimator function automatically evaluates all uncompressed volumes for the potential savings that XIV Compression feature can provide. The Comprestimator tool is running after you have enabled the compression feature on the XIV system. It can be disabled and re-enabled by using the XCLI, and you can check the status of Comprestimator. Both commands and outputs are shown in Figure 4-1.

```
XIV 1310075 Ansion>>system_comprestimate_get
Command executed successfully.
    periodic_comprestimate_mode=Disabled
XIV 1310075 Ansion>>system_comprestimate_set mode=Enabled
Command executed successfully.
```

Figure 4-1   Enabling and checking the status of Comprestimator

The XIV built-in Comprestimator function runs on all of the volumes that are in a thin pool and that meet the minimum volume size requirements. It runs per volume in the background every 8 minutes in a specific order, as shown in Figure 4-2.

```
XIV 1310075 Ansion>>vol_comprestimate_list
Name                          Compression Ratio (%)   Last estimation time   Status   Position
RDM-CATCAM                    13                      2015-05-12 16:16:34    Idle     16
mirror_1_GR                   91                      2015-05-12 10:16:44    Idle     7
mirror_2_GR                   91                      2015-05-12 10:19:27    Idle     8
GR_async_001                  94                      2015-05-12 10:11:24    Idle     5
GR_async_002                  92                      2015-05-12 10:13:57    Idle     6
vol_145_2                     88                      2015-05-12 10:24:16    Idle     9
vol_145_1                     89                      2015-05-12 10:33:57    Idle     10
RDM_Office                    71                      2015-05-12 10:38:46    Idle     11
RDM-IOMeter                   93                      2015-05-12 10:45:59    Idle     12
RDM-MySQL                     68                      2015-05-12 14:29:14    Idle     13
```

Figure 4-2   Displaying the Comprestimator list

The Position column in the list is showing the order in which the Comprestimator will run. If you manually start Comprestimator against a volume, it will move the volume up in the queue. The command to manually start Comprestimator against a volume and move it to the first position of the queue is shown in Figure 4-3 on page 31.

```
vol_comprestimate vol=[VOLNAME]
```

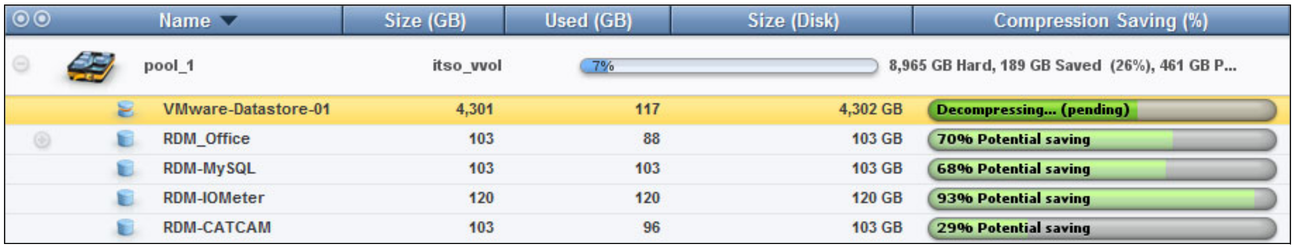
Figure 4-3 Starting Comprestimator against a volume

In addition to starting and stopping Comprestimator, you can exclude a volume from the Comprestimator, as shown in Figure 4-4.

```
vol_comprestimate_stop vol=[volname]
```

Figure 4-4 Excluding a volume from the Comprestimator

The GUI also displays the potential compression savings per Volume in both the Volumes and Snapshots and the Volumes by Pools views. When you move the mouse over the Compression Savings% column for the specific volume of interest, a text pop-up displays the calculated savings, as shown in Figure 4-5.



Name	Size (GB)	Used (GB)	Size (Disk)	Compression Saving (%)
pool_1	itso_vvol	7%		8,965 GB Hard, 189 GB Saved (26%), 461 GB P...
VMware-Datastore-01	4,301	117	4,302 GB	Decompressing... (pending)
RDM_Office	103	88	103 GB	70% Potential saving
RDM-MySQL	103	103	103 GB	68% Potential saving
RDM-IOMeter	120	120	120 GB	93% Potential saving
RDM-CATCAM	103	96	103 GB	29% Potential saving

Figure 4-5 Comprestimator appearing in the GUI

You can stop the Comprestimator from running in the background. You can stop it per volume by using the following XCLI command:

`vol_comprestimate_stop vol=<`*`VolName`*`>`

For more information, see the following topic in the IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/STJTAG/com.ibm.help.xivgen3.doc/Gen3/r_command_2_2_35.dita?lang=en

4.7.2 Host-based Comprestimator tool

The Comprestimator utility is a command-line, host-based utility that can be used to estimate expected compression rates for existing block devices in their native environment. The utility uses advanced mathematical and statistical algorithms to do the device sampling and analysis process efficiently and in a very short time. The utility displays its accuracy level by showing the maximum accuracy range of the expected compression results achieved. The compression accuracy range is 5% based on the statistical algorithms that it uses (Figure 4-6).

```
PS C:\Program Files (x86)\IBM\Comprestimator\Windows> .\Comprestimator.exe -n4 -s XIV
Analysis started at: 13/05/2015 09:16:34.473936

 Sample#      | Device           | Size(GB) | Compressed | Total      | Total      | Storage Efficiency| Compression | Compression
              |  Name            |          |  Size(GB)  | Savings(GB)| Savings(%) |   Savings(%)      | Savings(%)  | Accuracy Range(%)
-------------+-----------------+----------+------------+------------+------------+-------------------+-------------+------------------
3400         |5&1982005&0&0004 |     96.2 |       24.1 |       72.1 |      74.9% |              14.6% |       70.7% |              5.0%
```

Figure 4-6 Host-based Comprestimator output with the RDM_Office from Figure 4-5 f

The utility runs on a host that has access to the devices that will be analyzed, and performs only read operations so it has no effect whatsoever on the data that is stored on the device. It is important to emphasize that because Comprestimator is a host-based utility that analyzes any block device accessible from the host it is running on, it can be used on any device, regardless of the storage system it is stored on. It supports the following hosts:

- Windows 2003 Server, Windows 2008 R2 Server, Windows 2012, Windows 7 and 8
- Red Hat Enterprise Linux Version 5.x, 6.x (x86 64-bit)
- ESXi 4.1, 5.0
- Sun Solaris 10, 11
- IBM AIX® 6.1, 7
- HPUX 11.31
- SUSE® Linux Enterprise Server 11 (x86 64-bit)
- Ubuntu 12 (x86 64-bit)

For better results, use Comprestimator to analyze volumes that contain as much data as possible rather than volumes that are mostly empty. This usage increases accuracy and reduces the risk of analyzing old data that was already deleted, but might still have traces on the volume.

Depending on the environment, in some cases Comprestimator is used on more than one host to analyze more data types that are stored in volumes that are accessible from other hosts. Comprestimator can be downloaded from the IBM website:

http://www14.software.ibm.com/webapp/set2/sas/f/Comprestimator/home.html

4.8 Compression ratio guidelines

Use the threshold for volume compressibility values in Table 4-3 to help you decide whether to compress a volume.

Table 4-3 Recommendations about when to use IBM Real-time Compression

Data compression rate	Guideline
Higher than 25% savings	Use compression.
Below 25% savings	Evaluate a workload with compression.

4.8.1 Guidelines for getting a high compression ratio

Review the following guidelines to achieve the highest compression savings in your environment:

- To achieve a high compression ratio, use compressible data. Table 4-1 on page 26 lists common applications that provide a high compression ratio. Storing these data types on compressed volumes saves disk space and improves the benefits of using compression.
- Avoid using any client, file system, or application compression when using XIV compressed volumes. In most cases, data that is already compressed cannot achieve significant additional savings from compressing it again.
- Avoid using any client, file system, or application encryption when using an XIV compressed volumes. Encrypted data is not compressible.

4.9 Additional space utilization guidance

In an XIV system, *hard capacity* denotes usable, non-compressed capacity, whereas *soft capacity* denotes the nominal capacity that is assigned to volumes, and reported to any hosts mapped to those volumes.

Thin provisioning denotes committing more soft capacity then hard capacity. Hard capacity is assigned at a pool level. In the case of compression, thin-provisioning is obvious: A compressed volume will forever use less hard capacity than soft capacity.

The maximal, non-compressed hard capacity supported in a single XIV frame is 485 TB (Gen3, 15 modules, 6 TB drives). However, an XIV frame can effectively accommodate up to 2 PB of real written data (when the data is compressed).

With very high compression rates, filling a system up to 2 PB of soft capacity may not require a lot of usable capacity.

Illustration: When storing data with a 10:1 compression ratio, the system will exhaust the soft capacity (2 PB) after utilizing only 200 TB (2 PB/10) of usable (hard) capacity. In this case, an XIV system with less than 15 modules would be very attractive. Alternatively, with that data a system with 15 modules will be less than half-utilized (200 TB out of 485 TB, for a 6 TB system).

Considering the compression ratio for typical data profiles on XIV systems, the effective soft capacity leveraged within a 15-module frame will range from 1 PB to 2 PB. To maximize the utilization of XIV hard and soft capacity with compressed data, and avoid oversizing the system, it is important to assess the expected compression ratio for the stored data. The system Comprestimator can be used for that purpose.

Following are select recommendations concerning the setup of hard/soft capacity for storage pools with compressed data:

- For mixed compressed and non compressed volumes in a pool, the pool soft size should be approximately twice as large as the hard size.
- For pools with only compressed volumes, the pool soft size should be approximately 5 times larger than the hard size.

Tip: When defining your thin-provisioned pool, it is also a good practice to group similar application workloads into a pool. In doing so, you can then apply a hard to soft ratio to that pool that is the most aggressive, with the least risk.

5

Configuring compressed volumes

This chapter describes detailed steps describing how to configure, define, and change compressed volumes by using the GUI and the command-line interface (XCLI). It includes the following information:

- 5.1, “Enabling compression” on page 36
- 5.2, “Configuring compressed pools and volumes on XIV” on page 37
- 5.3, “Converting volumes” on page 45
- 5.4, “Compressing a mirrored volume” on page 56
- 5.5, “Disabling compression” on page 64

© Copyright IBM Corp. 2015. All rights reserved.

5.1 Enabling compression

IBM Real-time Compression is disabled by default. To use the XIV Gen3 Storage System for compression, this function must be enabled with either the GUI or XCLI, after the system is upgraded to software 11.6 or later.

To enable IBM Real-time Compression, sign in as an administrator and use one of the following methods:

- 5.1.1, “Enabling IBM Real-time Compression with the GUI” on page 36
- 5.1.2, “Using XCLI commands to enable IBM Real-time Compression” on page 37

5.1.1 Enabling IBM Real-time Compression with the GUI

1. Use the GUI to connect to the system that is eligible to enable compression.
2. Click **Systems** and select **System settings** → **System**. See Figure 5-1.

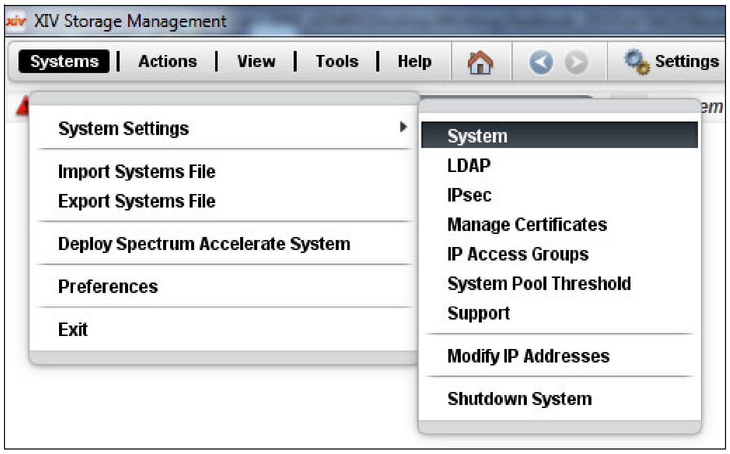
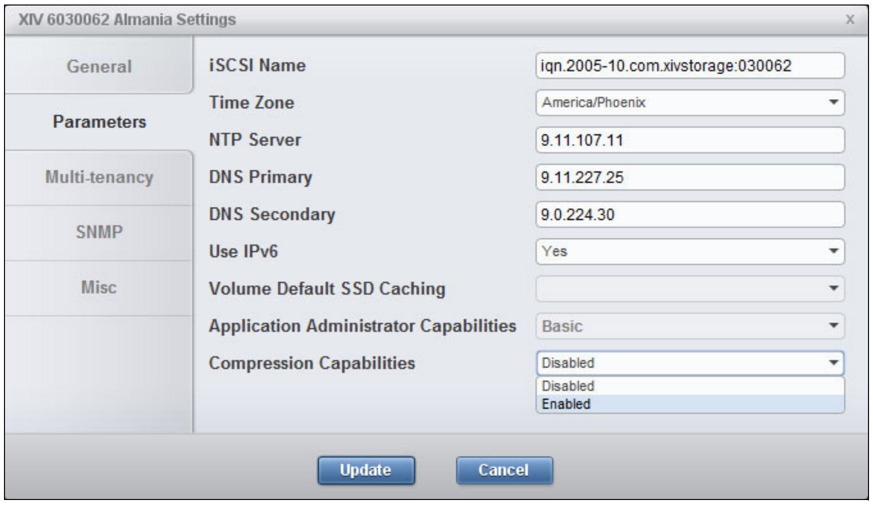


Figure 5-1 System Settings

Compression can be enabled in the Settings window.

3. Select the **Parameters** tab to change the system settings. See Figure 5-2.

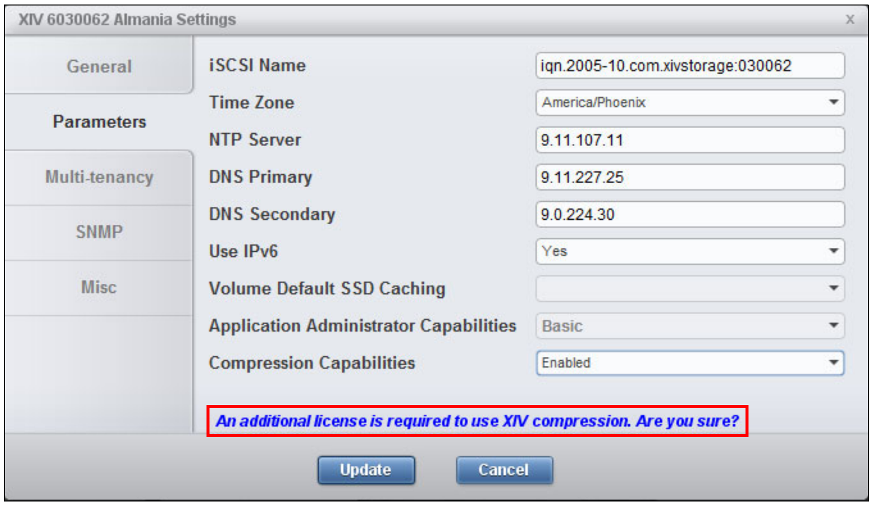


iSCSI Name: iqn.2005-10.com.xivstorage:030062
Time Zone: America/Phoenix
NTP Server: 9.11.107.11
DNS Primary: 9.11.227.25
DNS Secondary: 9.0.224.30
Use IPv6: Yes
Volume Default SSD Caching:
Application Administrator Capabilities: Basic
Compression Capabilities: Disabled (Disabled / Enabled)

Figure 5-2 Parameters

4. Change the Compression Capabilities drop-down to **Enabled**.

   The next window displays a message, as shown in Figure 5-3.



iSCSI Name: iqn.2005-10.com.xivstorage:030062
Time Zone: America/Phoenix
NTP Server: 9.11.107.11
DNS Primary: 9.11.227.25
DNS Secondary: 9.0.224.30
Use IPv6: Yes
Volume Default SSD Caching:
Application Administrator Capabilities: Basic
Compression Capabilities: Enabled

An additional license is required to use XIV compression. Are you sure?

Figure 5-3 A license is required to use compression

5. Click **Update** to confirm that you have purchased the required license, or that you are using IBM Real-time Compression for a trial.

5.1.2 Using XCLI commands to enable IBM Real-time Compression

Example 5-1 shows the XCLI command to enable IBM Real-time Compression. The warning about the license must be confirmed before the change is activated.

Example 5-1 Example system_compression_enable command

```
XIV 6030062 Almania>> system_compression_enable

Warning:   Using XIV compression requires an additional license. Are you sure you
want to enable compression on this system? y/n: y
Command executed successfully.
XIV 6030062 Almania>>
```

5.2 Configuring compressed pools and volumes on XIV

This section describes steps to configure compressed volumes with the XIV GUI. For each step, the related XCLI command example is included.

5.2.1 Configuring a thin pool for compression

Compressed volumes are always thin-provisioned volumes. If no thin pool is configured in XIV, the first step to use IBM Real-time Compression is to create at least one thin pool, or reconfigure an existing regular pool to a thin pool. See 5.2.3, “Using the GUI to change a regular pool to a thin pool” on page 40.

Complete the following steps to configure a thin pool for compression:

1. Click **Actions** to open the drop-down menu and select **Create Pool**, as shown in Figure 5-4.

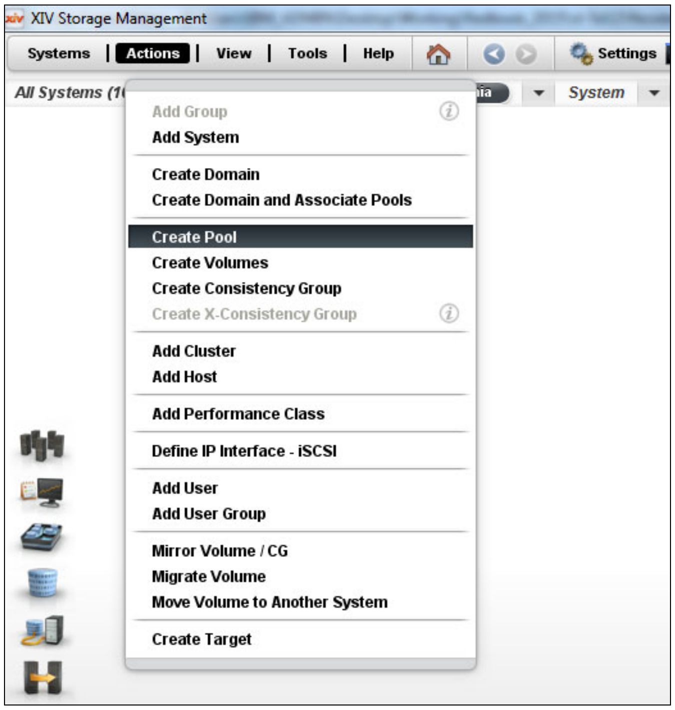
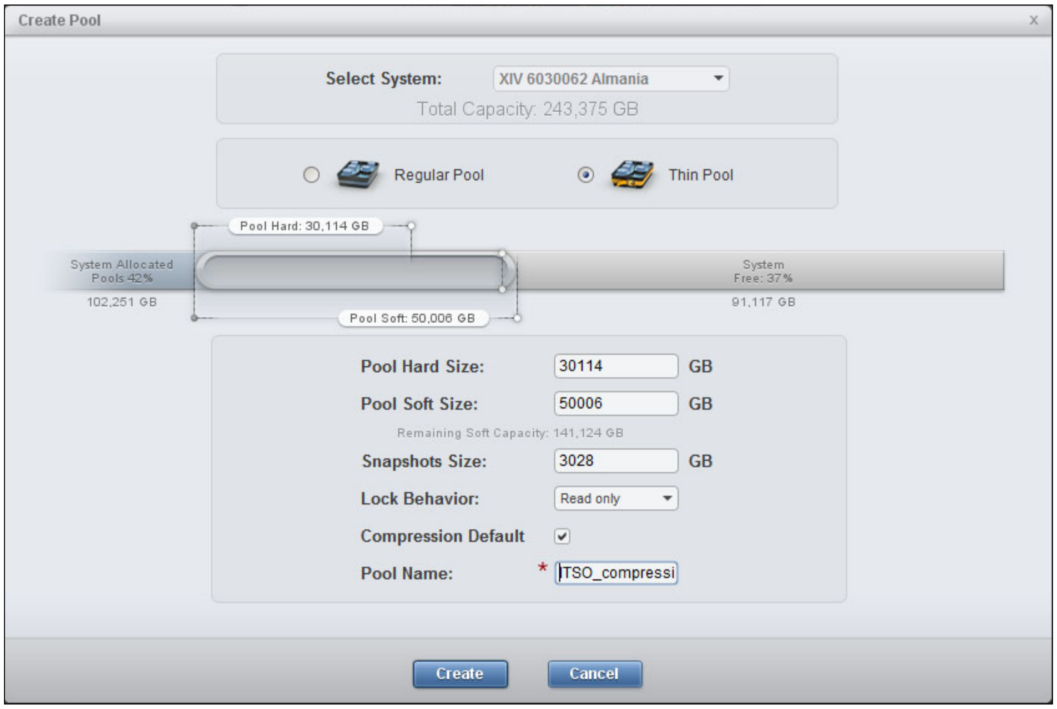


Figure 5-4   Creating a pool

2. In the window that opens, set the pool properties. See Figure 5-5.



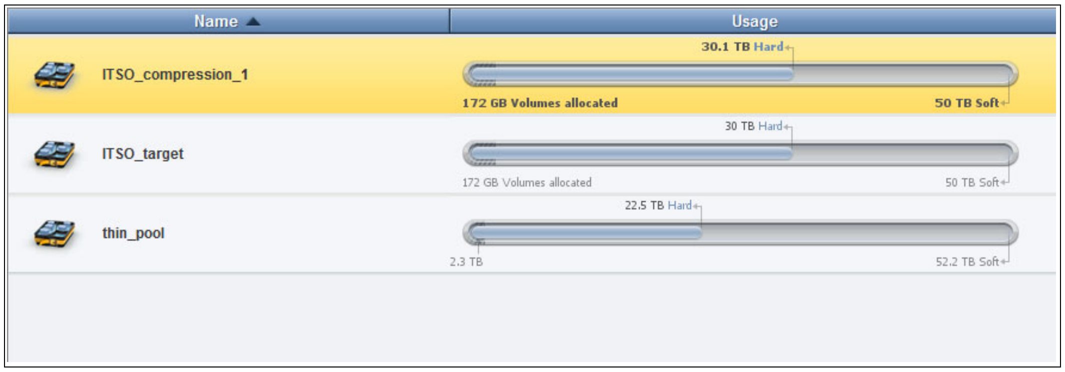
Select System: XIV 6030062 Almania
Total Capacity: 243,375 GB
Thin Pool
Pool Hard Size: 30114 GB
Pool Soft Size: 50006 GB
Remaining Soft Capacity: 141,124 GB
Snapshots Size: 3028 GB
Lock Behavior: Read only
Compression Default ✓
Pool Name: * ITSO_compressi

Figure 5-5   Create Pool window

In addition to the traditional properties, the pool can be configured by enabling **Compression Default**. This setting means that all volumes that are added to the pool will be compressed by default.

3. If you do not want volumes to be compressed, clear this check box before you create the volume in the pool.
4. Click **Create** to create the pool with the selected properties. See Figure 5-6.

   The yellow belt around the pool icon means that it is a thin pool.



Name	Usage
ITSO_compression_1	30.1 TB Hard; 172 GB Volumes allocated; 50 TB Soft
ITSO_target	30 TB Hard; 172 GB Volumes allocated; 50 TB Soft
thin_pool	22.5 TB Hard; 2.3 TB; 52.2 TB Soft

Figure 5-6 Thin pool

5. To change the compression default setting of a thin pool, right-click on the pool and select **Change Compression Default**. See Figure 5-7.

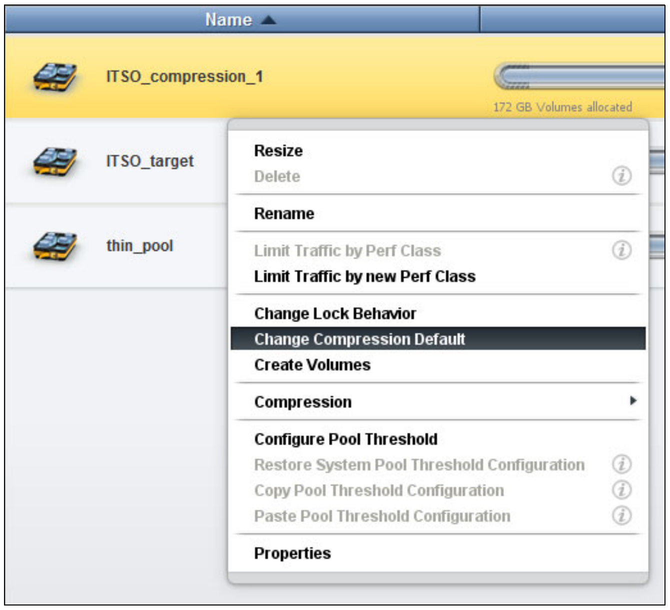

Name
ITSO_compression_1
172 GB Volumes allocated
ITSO_target
thin_pool
Resize
Delete
Rename
Limit Traffic by Perf Class
Limit Traffic by new Perf Class
Change Lock Behavior
Change Compression Default
Create Volumes
Compression
Configure Pool Threshold
Restore System Pool Threshold Configuration
Copy Pool Threshold Configuration
Paste Pool Threshold Configuration
Properties

Figure 5-7 Changing the Compression Default setting

6. On the next screen, as shown in Figure 5-8, set the new Pool Compression Default.

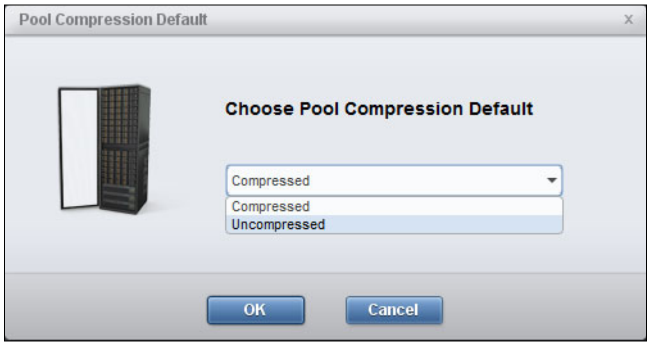


Figure 5-8 Change the compression default setting

5.2.2 XCLI examples for thin pools

Example 5-2 shows how to create a thin pool with compression enabled.

Example 5-2 Creating a thin pool with compression enabled

```
>> pool_create pool=ITSO_compression_2 soft_size=50006 hard_size=30114
snapshot_size=3028 compress=yes
```

Example 5-3 shows how to change the default setting for compression for the pool.

Example 5-3 Changing the pool compression default setting

```
>> pool_change_config pool="test_pool" compress=yes
```

5.2.3 Using the GUI to change a regular pool to a thin pool

To compress volumes that belong to a regular pool, the pool must be converted to a thin pool.

Complete the following steps to convert a pool:

1. To convert a pool, right-click the pool and select **Resize**. See Figure 5-9 on page 41.

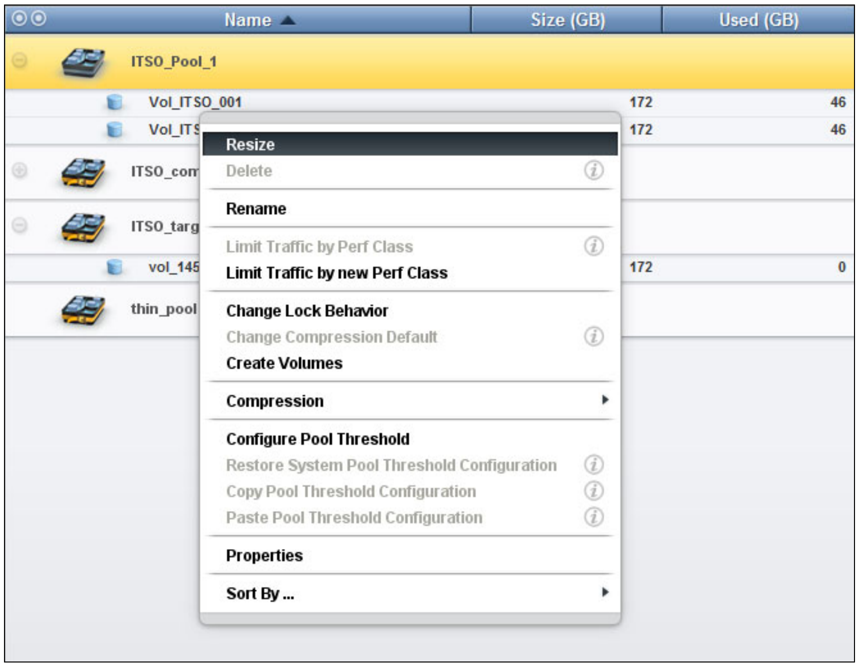
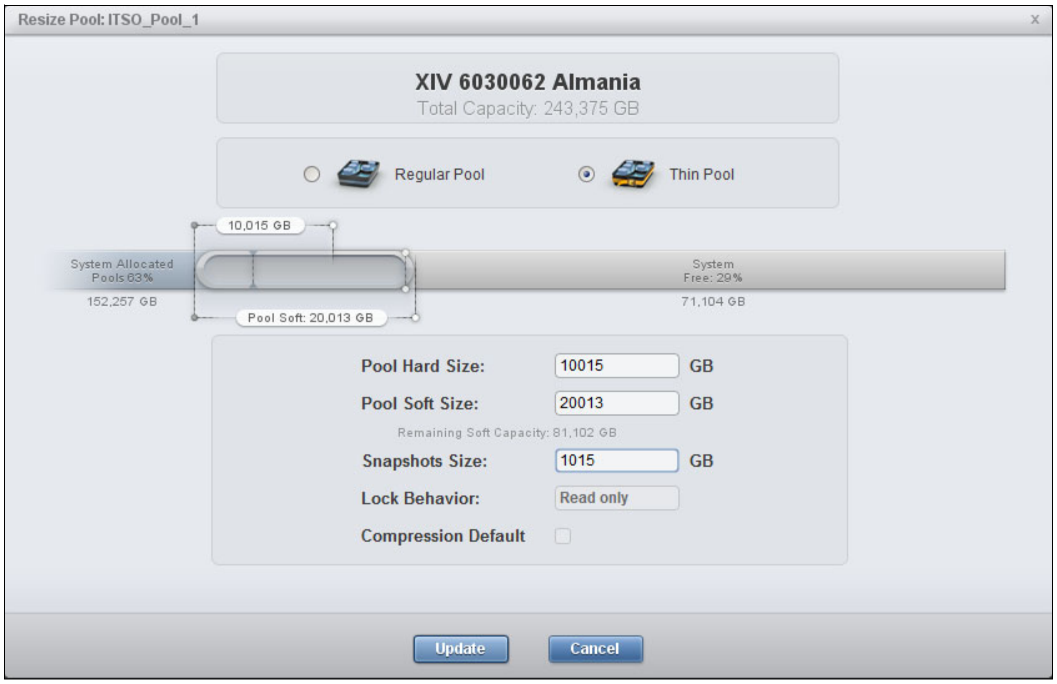


Figure 5-9 Resize a pool and change it to thin

2. In the Resize Pool window, select **Thin Pool** and increase the Pool Soft Size. See Figure 5-10.



Pool Hard Size: 10015 GB
Pool Soft Size: 20013 GB
Snapshots Size: 1015 GB
Lock Behavior: Read only
Compression Default

Figure 5-10 Changing from a regular pool to a thin pool

3. Click **Update** to change the pool.

   A *thin pool* is represented by the yellow belt on the pool symbol.

4. To compress all volumes in a pool, right-click the pool and select **Compression**. See Figure 5-11.

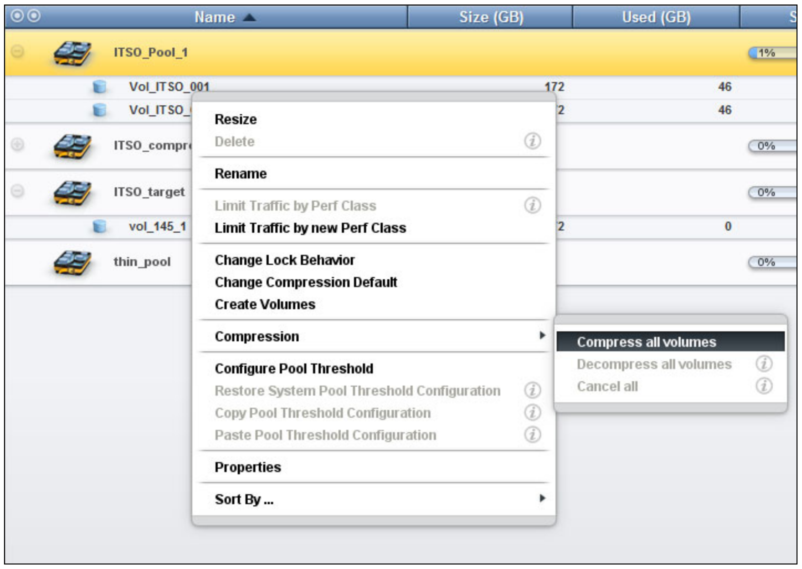


Figure 5-11 Compress all volumes in the pool

5. To compress a single volume, right-click it and select **Compression**. See Figure 5-12.

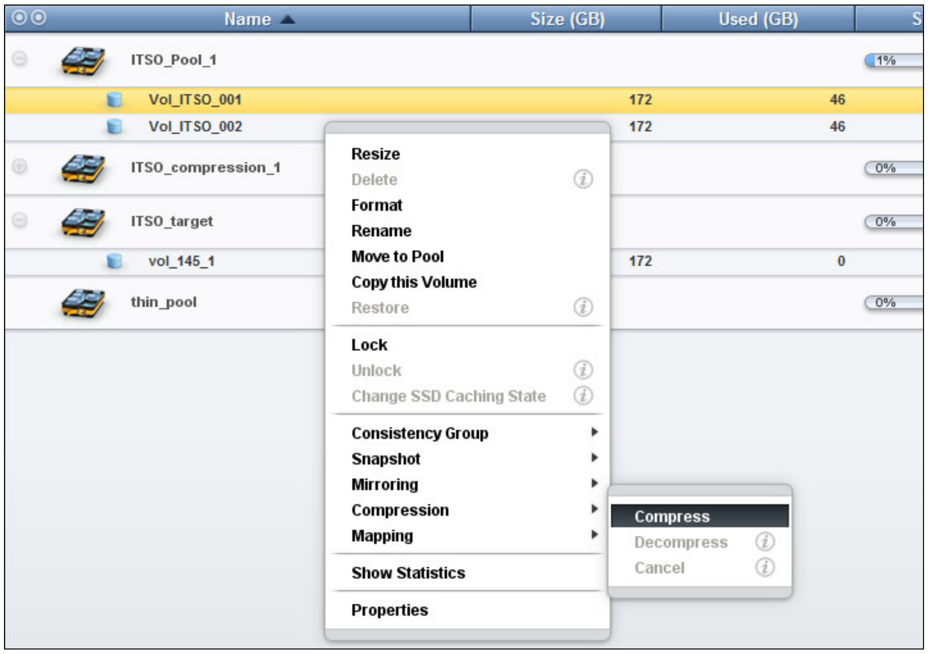


Figure 5-12 Compress individual volumes in the pool

5.2.4 Using the GUI to change a regular pool to a thin pool

Use the **pool_resize** command to change a regular pool to a thin pool. See Example 5-4.

Example 5-4 pool_resize, change regular pool to thin pool

```
>> pool_resize pool="test_pool" hard_size=35552 soft_size=65563
```

5.2.5 Creating a compressed volume with the GUI

Complete the following steps to create a new, compressed volume:

1. Select the system in the GUI, click **Actions**, and select **Create Volumes.** See Figure 5-13.

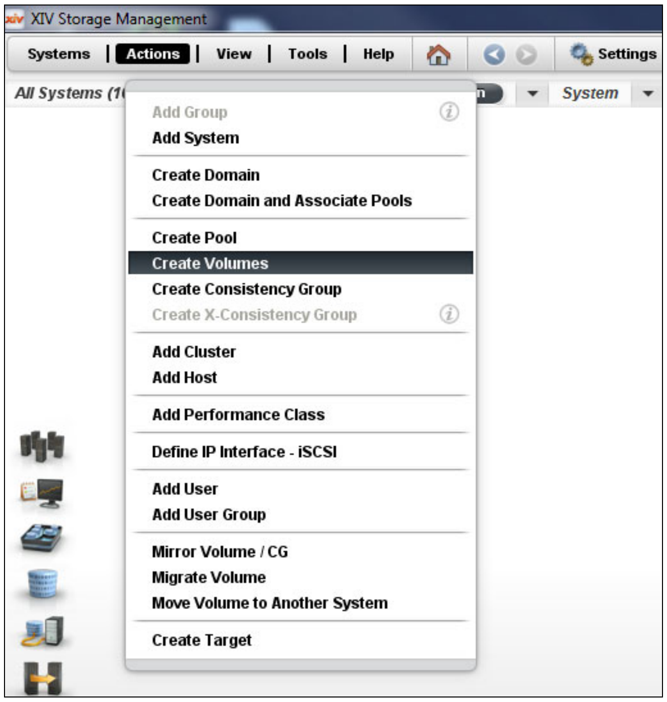
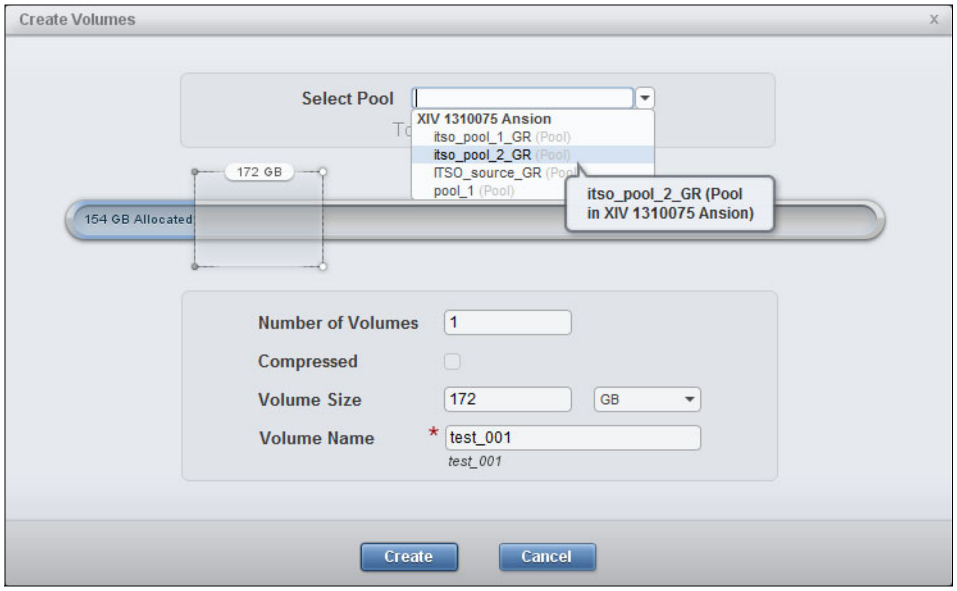


Figure 5-13 Create Volumes

The Create Volumes dialog opens, as shown in Figure 5-14.



Number of Volumes: 1
Compressed: ☐
Volume Size: 172 GB
Volume Name: * test_001
test_001

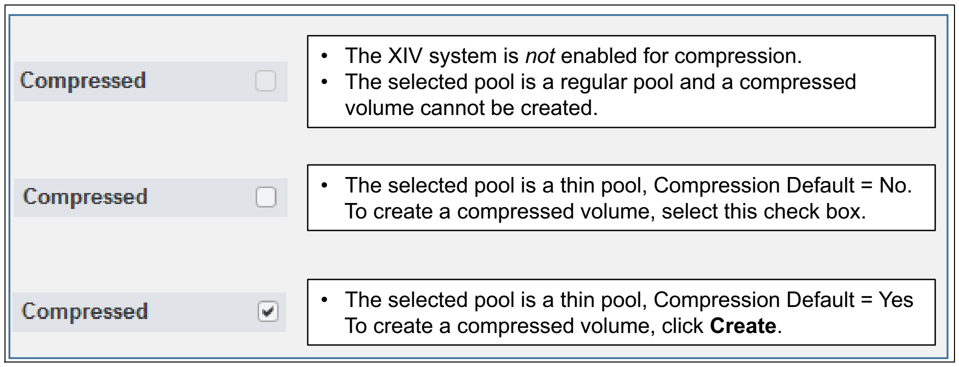
Figure 5-14 Set volumes name and properties

2. Enter the volume definitions.

3. Select the destination pool for the new volume from the drop-down menu shown in Figure 5-14 on page 43, and define Number of Volumes, Volume Size, and Volume Name.

Note: If Create Volumes was started by right-clicking a pool from the Pools view in the GUI, the pool is already preselected.

Depending on the selected pool, the check box for **Compressed** can have a different status and meaning. See Figure 5-15.



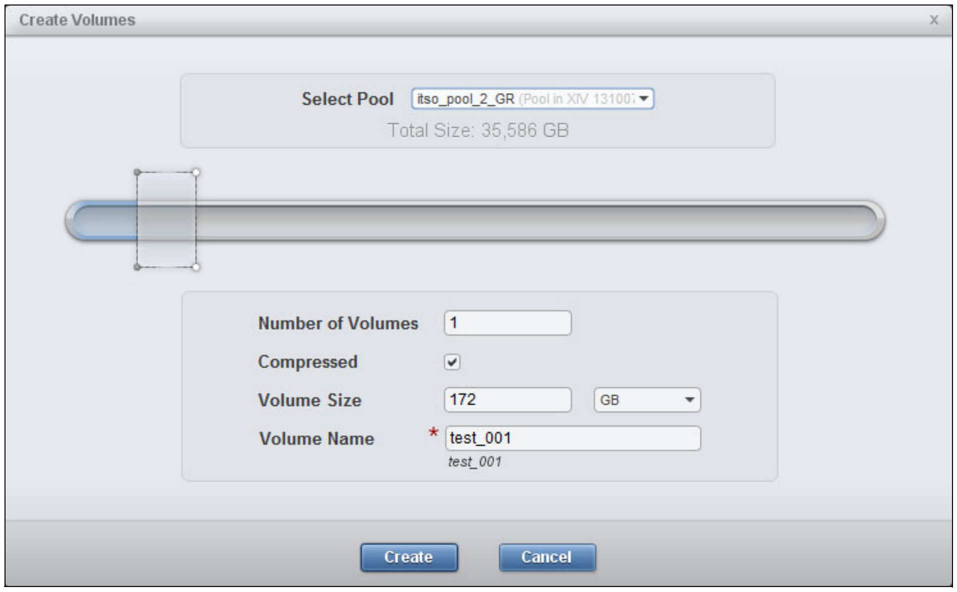
- The XIV system is *not* enabled for compression.
- The selected pool is a regular pool and a compressed volume cannot be created.

- The selected pool is a thin pool, Compression Default = No. To create a compressed volume, select this check box.

- The selected pool is a thin pool, Compression Default = Yes To create a compressed volume, click **Create**.

Figure 5-15 Compressed check box

4. Adjust the Compressed check box for your configuration and click **Create.** See Figure 5-16.

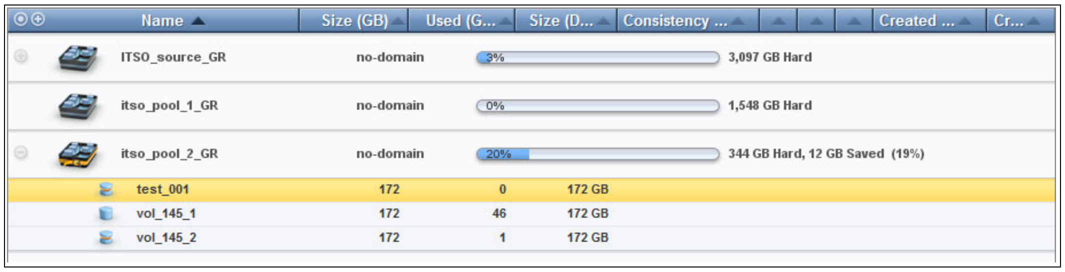


Select Pool: itso_pool_2_GR (Pool in XIV 131007
Total Size: 35,586 GB

Number of Volumes: 1
Compressed: ✓
Volume Size: 172 GB
Volume Name: * test_001
test_001

Figure 5-16 Create compressed volumes

The volume is created. The belt around the volume symbol indicates that it is a compressed volume. See Figure 5-17 on page 45.



Name	Size (GB)	Used (G...	Size (D...	Consistency ...				Created ...	Cr...
ITSO_source_GR	no-domain	3%		3,097 GB Hard					
itso_pool_1_GR	no-domain	0%		1,548 GB Hard					
itso_pool_2_GR	no-domain	20%		344 GB Hard, 12 GB Saved (19%)					
test_001	172	0	172 GB						
vol_145_1	172	46	172 GB						
vol_145_2	172	1	172 GB						

Figure 5-17 New created volume, test_001

You can also use the **`vol_create`** command to create a compressed volume. See Example 5-5.

Example 5-5 Create compressed volume

```
>> vol_create size=172 pool="ITSO_Pool_1_GR" compressed=yes vol="test_1"
```

5.3 Converting volumes

Volumes can be converted from conventional volumes to compressed volumes, and vice versa. This section provides a method to start the conversion.

5.3.1 Converting a conventional volume to a compressed volume

Before you compress an existing volume, make sure that it belongs to a thin pool. A thin pool is identifiable on the yellow belt around the symbol in the GUI. See Figure 5-18.

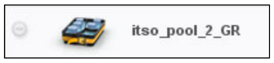


Figure 5-18 Thin pool symbol

Complete the following steps to convert a conventional volume to a compressed volume:

1. From the GUI System view, click **Pools** and select **Volumes by Pools** (Figure 5-19).

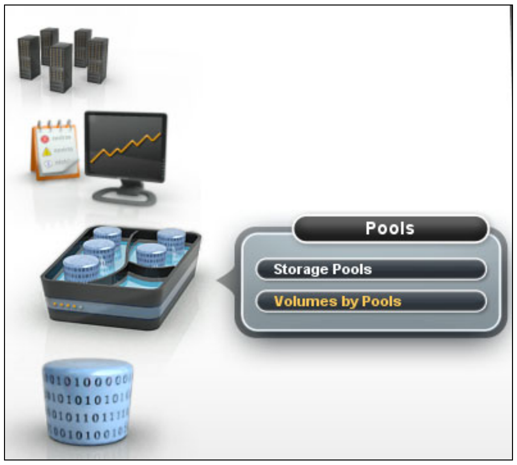
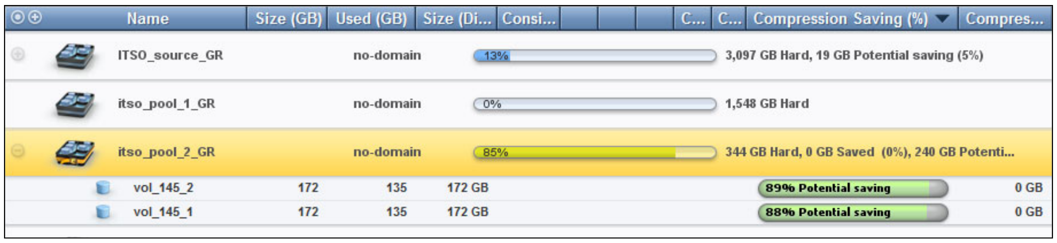


Figure 5-19 Volumes by Pools

The Volumes by Pools view shows a list of configured pools.

2. Click the Plus sign (+) in front of the pool icon to show the configured volumes.

   The result of the Comprestimator tool, which is the potential saving for each volume, is shown on the left side. See Figure 5-20.



```
Name            Size (GB)  Used (GB)  Size (Di...  Consi...     C...  C...  Compression Saving (%)   Compres...
ITSO_source_GR                  no-domain        13%      3,097 GB Hard, 19 GB Potential saving (5%)
itso_pool_1_GR                  no-domain        0%       1,548 GB Hard
itso_pool_2_GR                  no-domain        85%      344 GB Hard, 0 GB Saved  (0%), 240 GB Potenti...
  vol_145_2     172        135        172 GB                            89% Potential saving      0 GB
  vol_145_1     172        135        172 GB                            88% Potential saving      0 GB
```

Figure 5-20 Volumes by pools

From here there are two options to start the compression of volumes:

– Compress a selected volume or volumes.
– Compress all volumes in the pool.

5.3.2 Compressing single volumes

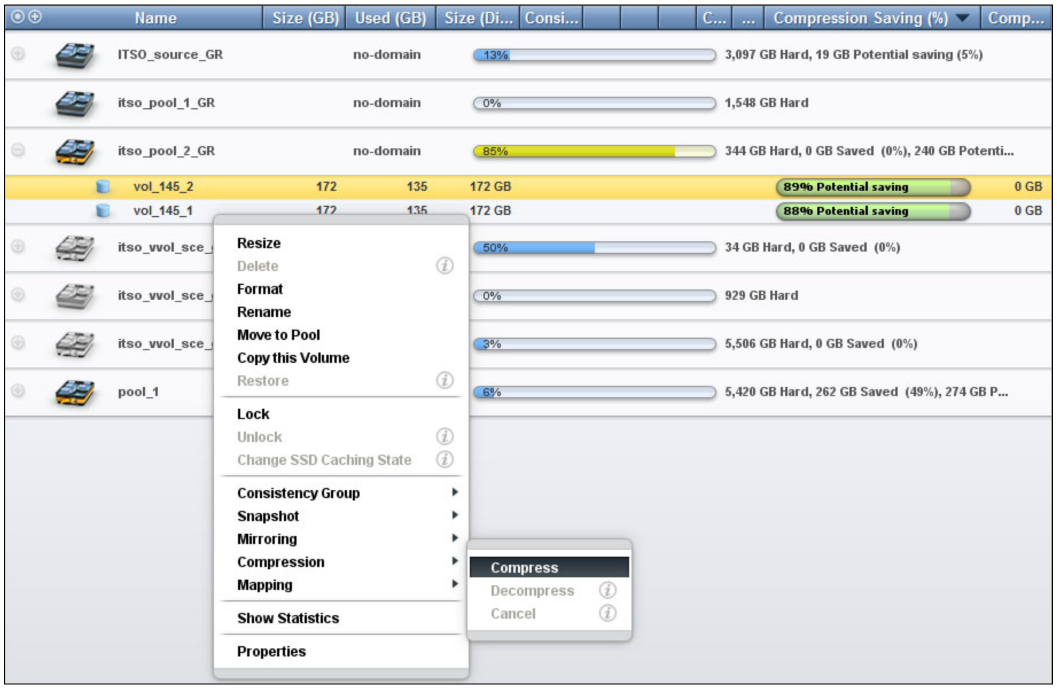
This section explains how to compress single volumes.

Compressing single volumes with the GUI

Complete the following steps to compress a single volume:

Note: Select several volumes for compression by holding the Ctrl key and click volumes that should be compressed.

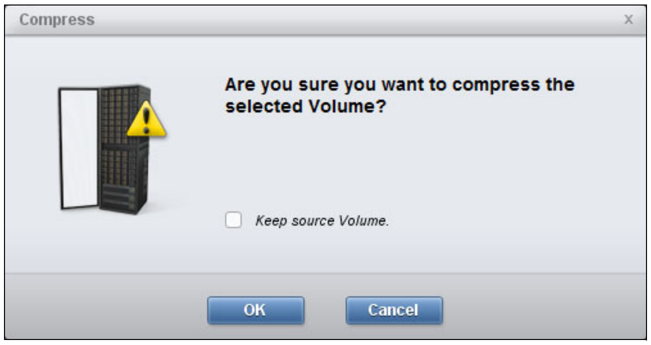
1. Right-click the volume to be compressed and select **Compression** → **Compress**. See Figure 5-21 on page 47.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...				C...	...	Compression Saving (%)	Comp...
ITSO_source_GR		no-domain		13%						3,097 GB Hard, 19 GB Potential saving (5%)	
itso_pool_1_GR		no-domain		0%						1,548 GB Hard	
itso_pool_2_GR		no-domain		85%						344 GB Hard, 0 GB Saved (0%), 240 GB Potenti...	
vol_145_2	172	135	172 GB							89% Potential saving	0 GB
vol_145_1	172	135	172 GB							88% Potential saving	0 GB
itso_vvol_sce_				50%						34 GB Hard, 0 GB Saved (0%)	
itso_vvol_sce_				0%						929 GB Hard	
itso_vvol_sce_				3%						5,506 GB Hard, 0 GB Saved (0%)	
pool_1				6%						5,420 GB Hard, 262 GB Saved (49%), 274 GB P...	

Figure 5-21 Compress all volumes in the pool

2. On the next screen, if you want to keep the source volume, select the **Keep source Volume** check box (Figure 5-22). For more information about **Keep source Volume**, see 5.2, “Configuring compressed pools and volumes on XIV” on page 37.



Compress

Are you sure you want to compress the selected Volume?

☐ *Keep source Volume.*

Figure 5-22 Keep source Volume

3. Click **OK** to confirm and start the compression. A message is displayed that the compression is initiating. See Figure 5-23.

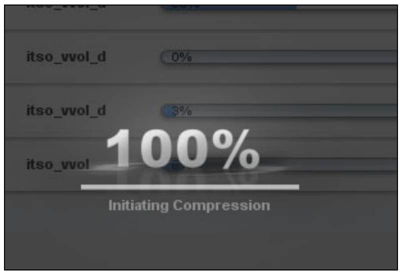
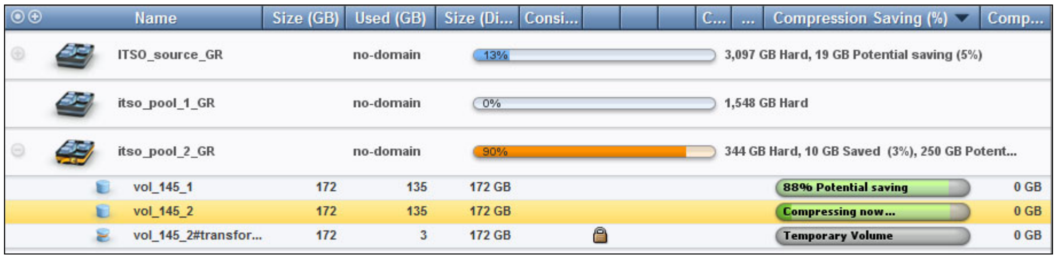


Figure 5-23 Initiating compression

On the Volumes by Pools view (Figure 5-24), you can see the progress of the compression.

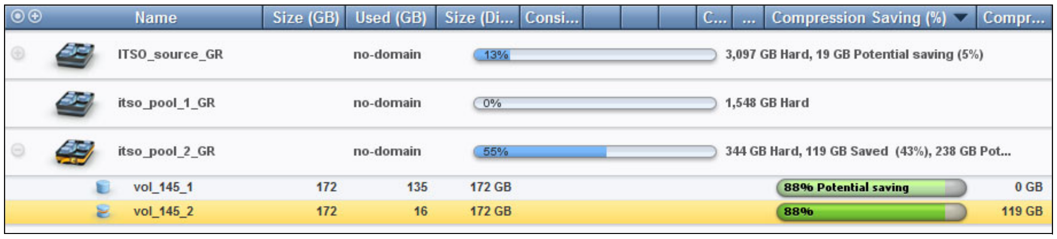


Name	Size (GB)	Used (GB)	Size (Di...	Consi...				C...	...	Compression Saving (%)	Comp...
ITSO_source_GR		no-domain	13%							3,097 GB Hard, 19 GB Potential saving (5%)	
itso_pool_1_GR		no-domain	0%							1,548 GB Hard	
itso_pool_2_GR		no-domain	90%							344 GB Hard, 10 GB Saved (3%), 250 GB Potent...	
vol_145_1	172	135	172 GB							88% Potential saving	0 GB
vol_145_2	172	135	172 GB							Compressing now...	0 GB
vol_145_2#transfor...	172	3	172 GB		🔒					Temporary Volume	0 GB

Figure 5-24 Compression progress

The usage of the pool is increased, and the copy of the volume is created before the compression is started to keep the option to reverse the compression.

After compression is complete, the temporary volume is deleted only if the **Keep source Volume** check box (shown in Figure 5-22 on page 47) was not selected. See Figure 5-25.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...				C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	13%							3,097 GB Hard, 19 GB Potential saving (5%)	
itso_pool_1_GR		no-domain	0%							1,548 GB Hard	
itso_pool_2_GR		no-domain	55%							344 GB Hard, 119 GB Saved (43%), 238 GB Pot...	
vol_145_1	172	135	172 GB							88% Potential saving	0 GB
vol_145_2	172	16	172 GB							88%	119 GB

Figure 5-25 Compression complete

The pool usage, which was 85% at the beginning, increases to 90% during the compression, and is now reduced to 55%. The used volume size is reduced from 135 GB to 16 GB. This reduction provides an 88% saving of used disk space. The yellow belt on the volume icon shows that the volume is compressed.

Compressing single volumes with the XCLI

Use the **`vol_transform_create`** command to compress single volumes with the XCLI, as shown in Example 5-6.

Example 5-6 vol_transform_create

```
>> vol_transform_create delete_source=yes mode=compress  vol="vol_145_1"
```

5.3.3 Compressing all volumes in a pool

You can compress all volumes in a thin pool. The process works in the same way as the process that is shown in 5.3.2, “Compressing single volumes” on page 46.

Complete the following steps to compress all volumes in a pool:

1. Right-click the pool and select **Compression** → **Compress all volumes**, as shown in Figure 5-26 on page 49.

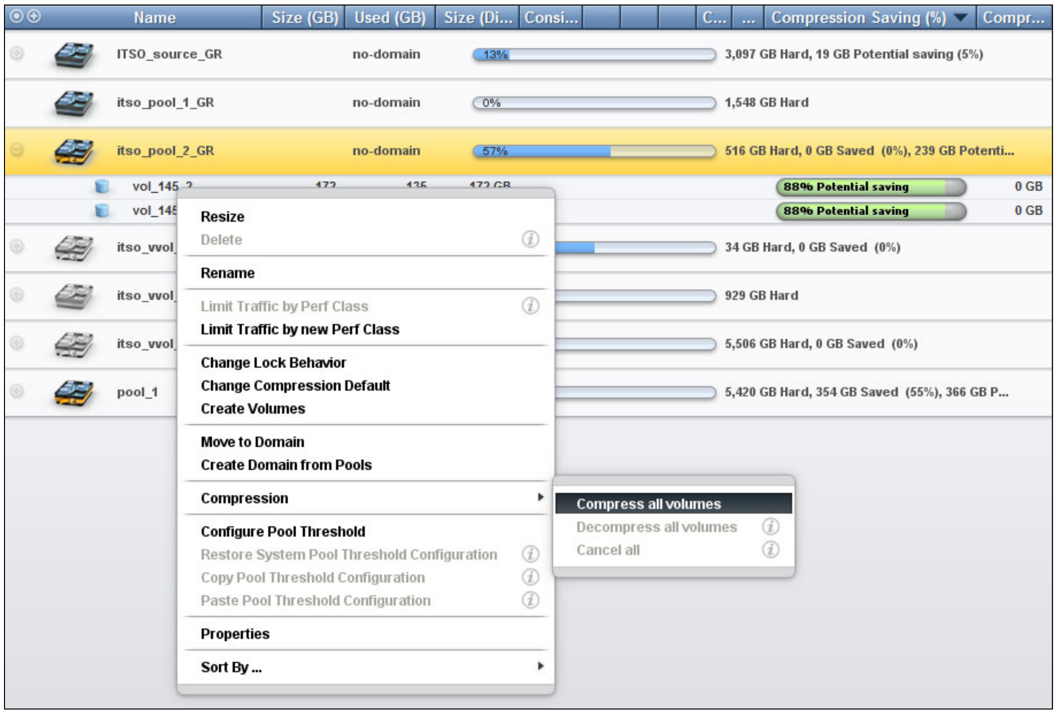
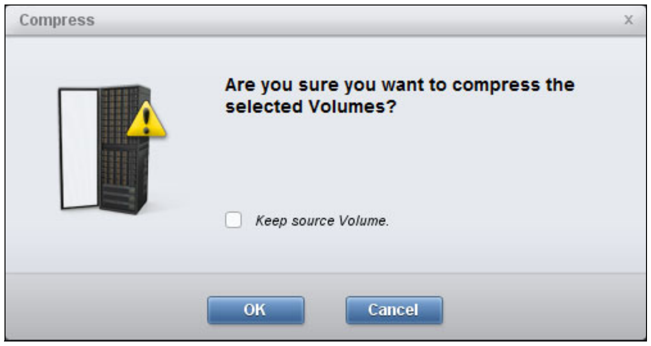


Figure 5-26 Compress all volumes in pool

2. On the next screen, confirm that you want to compress the selected volumes now.

   If you want to keep the source volume, select the check box. See Figure 5-27.



Compress

Are you sure you want to compress the selected Volumes?

☐ *Keep source Volume.*

Figure 5-27 Confirming compression

3. Click **OK** to start the compression. A message is displayed that the compression is initiating. See Figure 5-28.

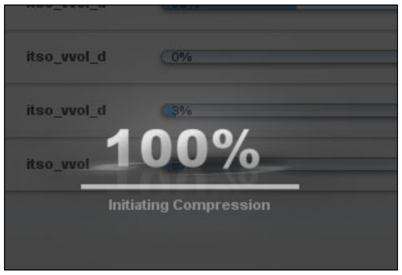
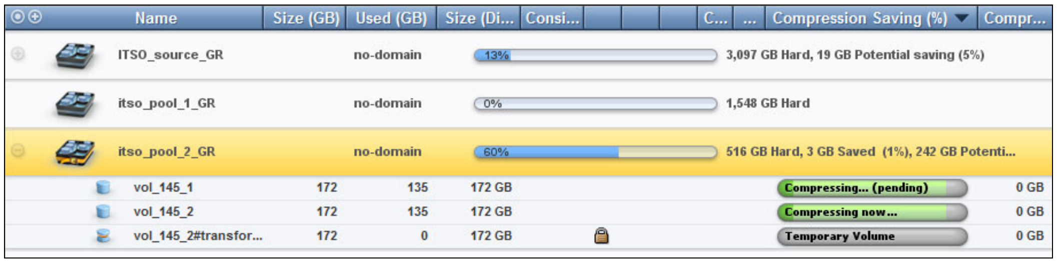


Figure 5-28 Starting compression

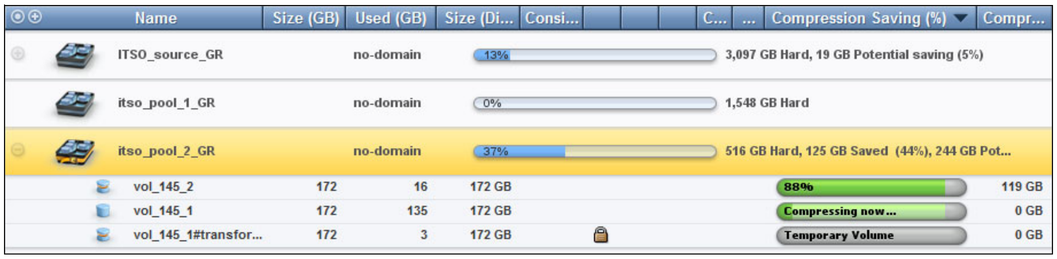
On the Volumes by Pools view (Figure 5-29), you can monitor the progress of compressing the volumes in the pool.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	13%				3,097 GB Hard, 19 GB Potential saving (5%)			
itso_pool_1_GR		no-domain	0%				1,548 GB Hard			
itso_pool_2_GR		no-domain	60%				516 GB Hard, 3 GB Saved (1%), 242 GB Potenti...			
vol_145_1	172	135	172 GB						Compressing... (pending)	0 GB
vol_145_2	172	135	172 GB						Compressing now...	0 GB
vol_145_2#transfor...	172	0	172 GB		🔒				Temporary Volume	0 GB

Figure 5-29 Compressing the first volume

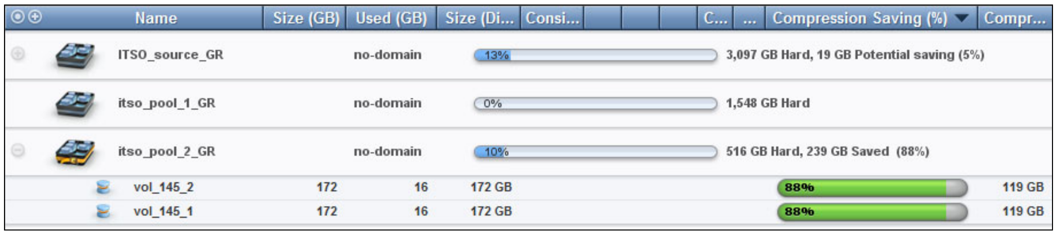
Compression is done volume by volume. After the compression completed for the first volume, the process deletes the temporary volume and starts compressing the next pending volume. See Figure 5-30.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	13%				3,097 GB Hard, 19 GB Potential saving (5%)			
itso_pool_1_GR		no-domain	0%				1,548 GB Hard			
itso_pool_2_GR		no-domain	37%				516 GB Hard, 125 GB Saved (44%), 244 GB Pot...			
vol_145_2	172	16	172 GB						88%	119 GB
vol_145_1	172	135	172 GB						Compressing now...	0 GB
vol_145_1#transfor...	172	3	172 GB		🔒				Temporary Volume	0 GB

Figure 5-30 Compressing the next volume

After compression is complete, the temporary volume is deleted only if the **Keep source Volume** check box (in Figure 5-27 on page 49) was not selected. See Figure 5-31.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	13%				3,097 GB Hard, 19 GB Potential saving (5%)			
itso_pool_1_GR		no-domain	0%				1,548 GB Hard			
itso_pool_2_GR		no-domain	10%				516 GB Hard, 239 GB Saved (88%)			
vol_145_2	172	16	172 GB						88%	119 GB
vol_145_1	172	16	172 GB						88%	119 GB

Figure 5-31 Compression complete

The pool usage, which was at the beginning on 57%, increases to 60% during the compression and is now reduced to 10%. The used volume size is reduced from 270 GB to 32 GB. This reduction is an 88% saving of used disk space. The yellow belt on the volume icon indicates that the volumes are compressed.

Note: There is no XCLI command to compress all volumes in a pool.

5.3.4 Converting a compressed volume to conventional volume

You can convert from compressed volumes to conventional volumes, either for single volumes or for all volumes in a pool.

Decompressing single volumes with the GUI

Complete the following steps to decompress a single volume with the GUI:

1. Click **Pools** and select **Volumes by Pools** (Figure 5-32).

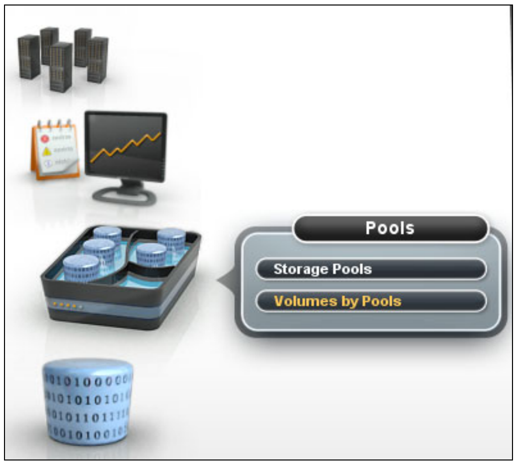
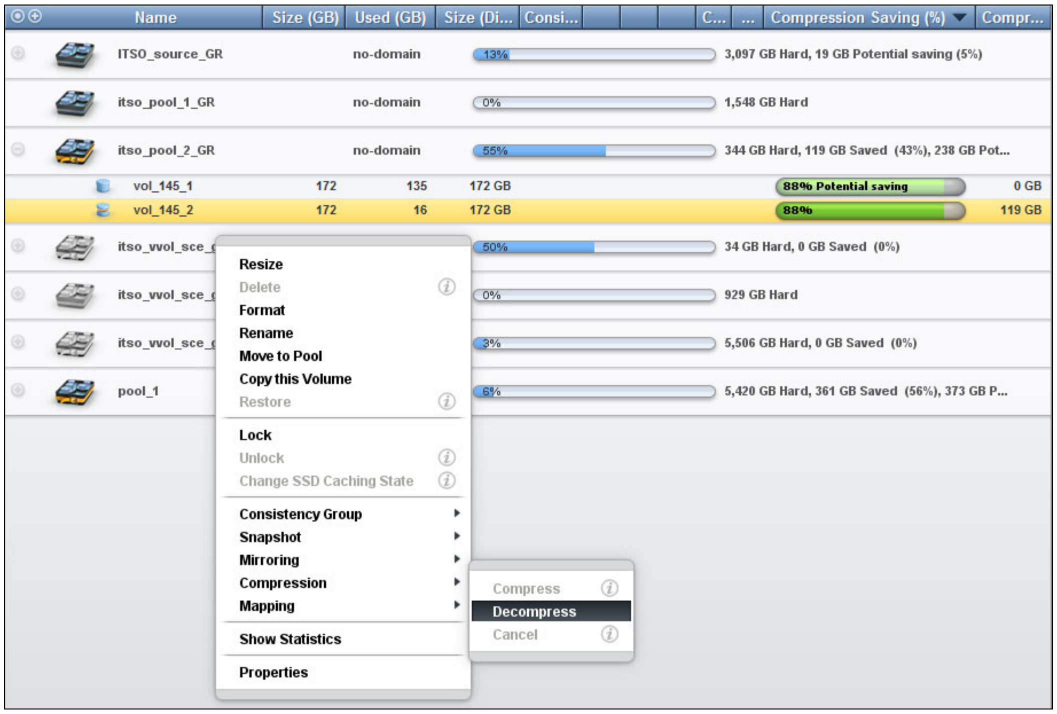


Figure 5-32 Volumes by Pools

The Volumes by Pools view shows a list of configured pools. When you click the Plus sign (+) in front of the pool icon, it also shows the configured volumes. On the left side, you see the result of the Comprestimator tool and the savings for each volume. (See Figure 5-33 on page 52.)

2. Right-click the volume and click **Compression** → **Decompress** to start the process for the selected volume or volumes (Figure 5-33).

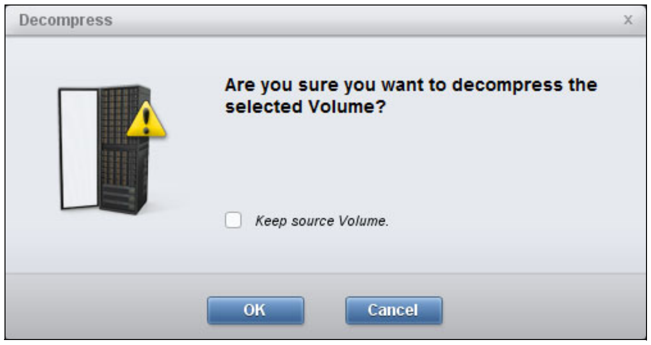
Note: Select several volumes for compression by holding the Ctrl key and clicking the volumes that you want to compress.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain			13% 3,097 GB Hard, 19 GB Potential saving (5%)	
itso_pool_1_GR		no-domain			0% 1,548 GB Hard	
itso_pool_2_GR		no-domain			55% 344 GB Hard, 119 GB Saved (43%), 238 GB Pot...	
vol_145_1	172	135	172 GB		88% Potential saving	0 GB
vol_145_2	172	16	172 GB		88%	119 GB
itso_vvol_sce_					50% 34 GB Hard, 0 GB Saved (0%)	
itso_vvol_sce_					0% 929 GB Hard	
itso_vvol_sce_					3% 5,506 GB Hard, 0 GB Saved (0%)	
pool_1					6% 5,420 GB Hard, 361 GB Saved (56%), 373 GB P...	

Figure 5-33 Decompress a volume

3. On the next panel, confirm that you want to compress the selected volume now. If you want to keep the source volume, select the check box (Figure 5-34). For more information about **Keep source Volume**, see 5.4, “Compressing a mirrored volume” on page 56.



Decompress

Are you sure you want to decompress the selected Volume?

☐ *Keep source Volume.*

Figure 5-34 Confirm decompression

4. Click **OK** to start the decompression. A message is displayed that the compression is starting (see Figure 5-35 on page 53).

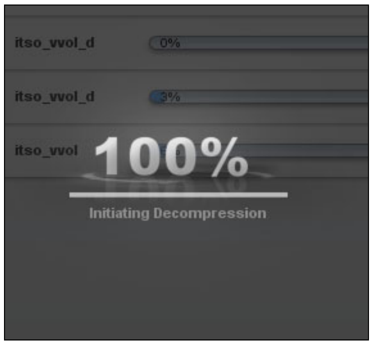
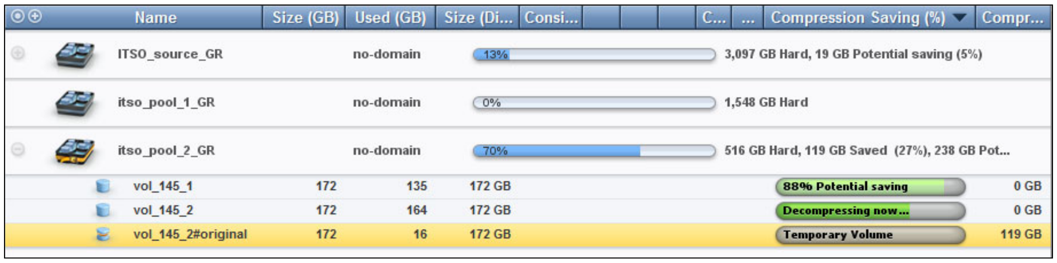


Figure 5-35   Initiating the decompression

On the Volumes by Pools view (Figure 5-36), you can see the progress of the decompression.

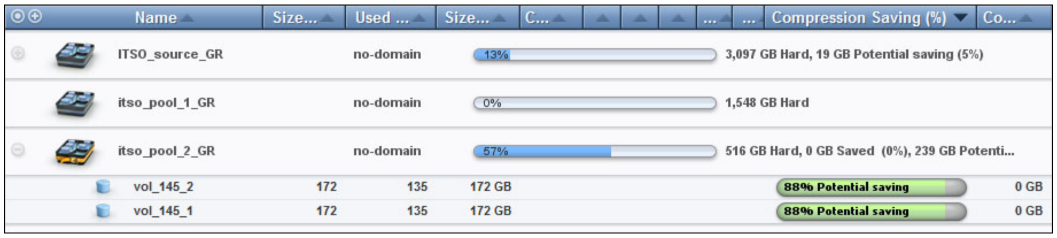


Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	13%				3,097 GB Hard, 19 GB Potential saving (5%)			
itso_pool_1_GR		no-domain	0%				1,548 GB Hard			
itso_pool_2_GR		no-domain	70%				516 GB Hard, 119 GB Saved (27%), 238 GB Pot...			
vol_145_1	172	135	172 GB						88% Potential saving	0 GB
vol_145_2	172	164	172 GB						Decompressing now...	0 GB
vol_145_2#original	172	16	172 GB						Temporary Volume	119 GB

Figure 5-36   Decompression progress

Notice increased usage of the pool and the temporary volume copy that is created before the decompression is started, which provides the option to reverse the decompression.

After the decompression is complete, the temporary volume is deleted if the **Keep source Volume** check box (Figure 5-34 on page 52) was not selected. See Figure 5-37.



Name	Size...	Used ...	Size...	C...				...	...	Compression Saving (%)	Co...
ITSO_source_GR		no-domain	13%					3,097 GB Hard, 19 GB Potential saving (5%)			
itso_pool_1_GR		no-domain	0%					1,548 GB Hard			
itso_pool_2_GR		no-domain	57%					516 GB Hard, 0 GB Saved (0%), 239 GB Potenti...			
vol_145_2	172	135	172 GB							88% Potential saving	0 GB
vol_145_1	172	135	172 GB							88% Potential saving	0 GB

Figure 5-37   Decompression complete

The pool usage, which was 55% at the beginning, increased to 70% during decompression. After the task is complete, the pool usage is 57% and the volume size increased from 16 GB to 135 GB. The yellow belt from the volume icon is removed to visualize that it is now a conventional volume.

Decompressing single volumes with the XCLI

Use the **`vol_transform_create`** command to decompress single volumes with the XCLI interface. See Example 5-7.

Example 5-7   Decompress

```
>> vol_transform_create delete_source=yes mode=decompress  vol="vol_145_1"
```

5.3.5 Decompressing all volumes in a pool

The method to convert all volumes in a pool works similarly to the method described in 5.3.4, “Converting a compressed volume to conventional volume” on page 51.

1. Right-click the pool and select **Compression** → **Decompress** all volumes. See Figure 5-38.

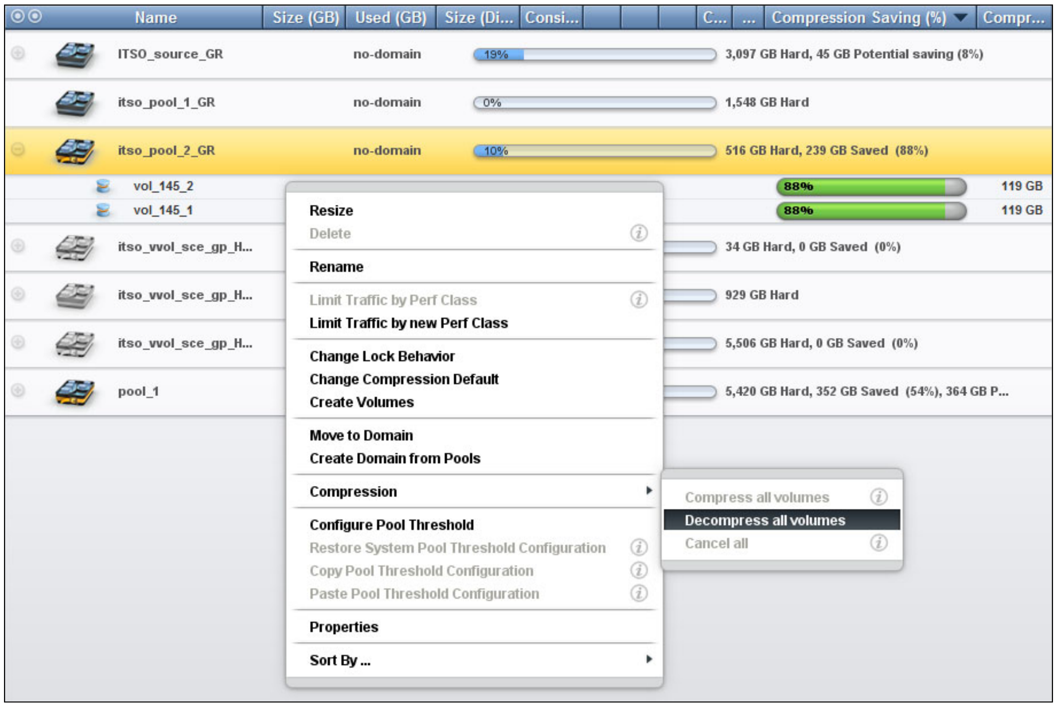
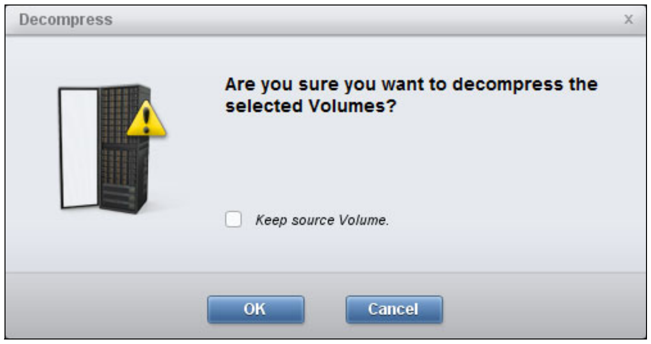


Figure 5-38 Decompress all volumes

2. On the next screen, confirm that you want to decompress the selected volumes now. If you want to keep the source volume, select the check box (Figure 5-39).



Are you sure you want to decompress the selected Volumes?

☐ Keep source Volume.

Figure 5-39 Confirming to decompress a volume

3. Click **OK** to start the decompression. A message that the decompression is initiating is displayed. See Figure 5-40 on page 55.

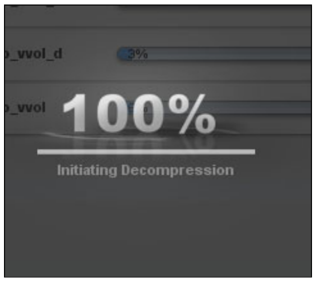
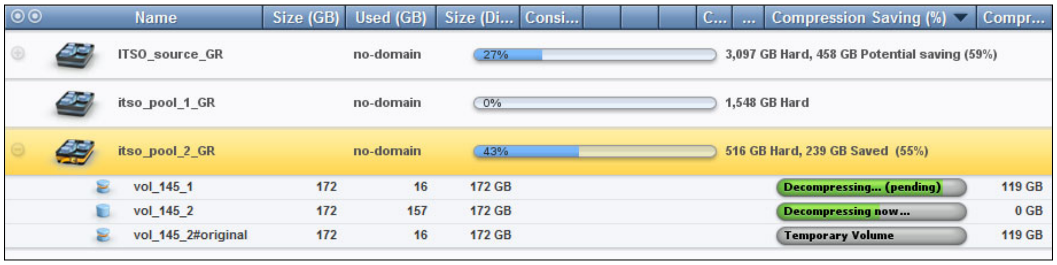


Figure 5-40 Initiating Decompression

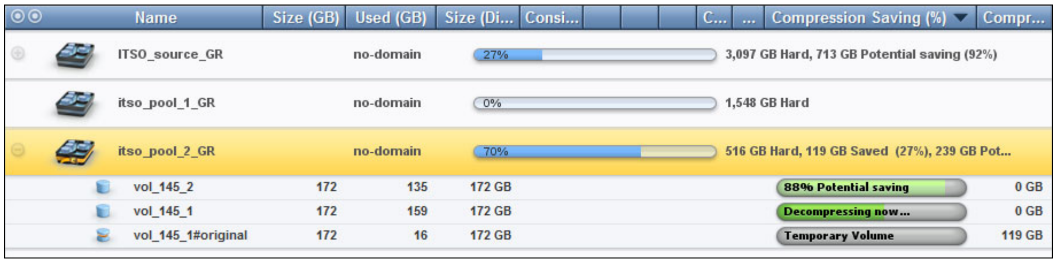
On the Volumes by Pools view (Figure 5-41), you can monitor the progress of decompression.



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	27%						3,097 GB Hard, 458 GB Potential saving (59%)	
itso_pool_1_GR		no-domain	0%						1,548 GB Hard	
itso_pool_2_GR		no-domain	43%						516 GB Hard, 239 GB Saved (55%)	
vol_145_1	172	16	172 GB						Decompressing... (pending)	119 GB
vol_145_2	172	157	172 GB						Decompressing now...	0 GB
vol_145_2#original	172	16	172 GB						Temporary Volume	119 GB

Figure 5-41 Decompress the first volume

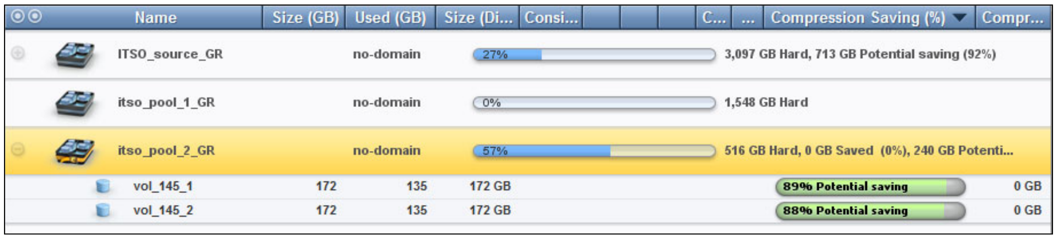
Decompression is done volume by volume. After the process finishes the first volume, the temporary volume is deleted and the system starts decompressing the next, pending volume (Figure 5-42).



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	27%						3,097 GB Hard, 713 GB Potential saving (92%)	
itso_pool_1_GR		no-domain	0%						1,548 GB Hard	
itso_pool_2_GR		no-domain	70%						516 GB Hard, 119 GB Saved (27%), 239 GB Pot...	
vol_145_2	172	135	172 GB						88% Potential saving	0 GB
vol_145_1	172	159	172 GB						Decompressing now...	0 GB
vol_145_1#original	172	16	172 GB						Temporary Volume	119 GB

Figure 5-42 Decompress the next pending volume

After decompression is complete for all volumes in the pool, the temporary volume is deleted when the process is initiated without selecting **Keep source Volume** (Figure 5-43).



Name	Size (GB)	Used (GB)	Size (Di...	Consi...			C...	...	Compression Saving (%)	Compr...
ITSO_source_GR		no-domain	27%						3,097 GB Hard, 713 GB Potential saving (92%)	
itso_pool_1_GR		no-domain	0%						1,548 GB Hard	
itso_pool_2_GR		no-domain	57%						516 GB Hard, 0 GB Saved (0%), 240 GB Potenti...	
vol_145_1	172	135	172 GB						89% Potential saving	0 GB
vol_145_2	172	135	172 GB						88% Potential saving	0 GB

Figure 5-43 Decompression complete

The pool usage, which was 10% at the beginning, increased during decompression to 70%, and returned to 57% after decompression was done. The used volume size increased from 16 GB to 135 GB, and the yellow belt is removed to show that the volumes are no longer compressed.

Note: There is no XCLI command to decompress all volumes in a pool.

5.4 Compressing a mirrored volume

As described in 3.3.4, “Mirroring” on page 23, volumes in a mirror relation cannot be compressed. Before the volumes are compressed, the mirror must be deactivated and deleted. After the volumes are compressed, the mirror relationship must be set up again.

Complete the following tasks to compress mirrored volumes:

- 5.4.1, “Deactivating a mirror relationship” on page 56
- 5.4.2, “Deleting mirrors, snapshots, and destination volumes” on page 58
- 5.4.3, “Compressing the volumes” on page 60
- 5.4.4, “Creating and activating the mirror” on page 61

5.4.1 Deactivating a mirror relationship

Complete the following steps to deactivate the mirrored volumes that you want to compress:

1. Hover over the Remote icon in the GUI and click **Mirroring**. See Figure 5-44.

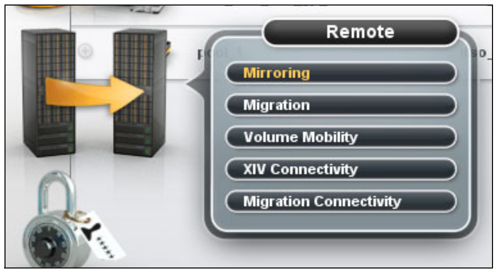
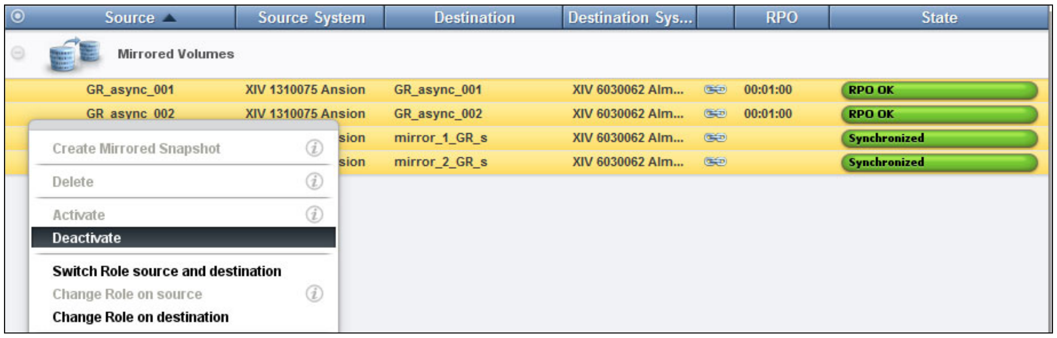


Figure 5-44 Remote - Mirroring

2. On next screen (Figure 5-45 on page 57), click the Plus (**+**) sign in front of Mirrored Volumes to list mirrored volumes.
3. Select all volumes that need to be deactivated by right-clicking them. Hold the Ctrl key to select multiple volumes.
4. After all volumes are selected, click **Deactivate**.



Source	Source System	Destination	Destination Sys...		RPO	State
Mirrored Volumes						
GR_async_001	XIV 1310075 Ansion	GR_async_001	XIV 6030062 Alm...		00:01:00	RPO OK
GR_async_002	XIV 1310075 Ansion	GR_async_002	XIV 6030062 Alm...		00:01:00	RPO OK
	sion	mirror_1_GR_s	XIV 6030062 Alm...			Synchronized
	sion	mirror_2_GR_s	XIV 6030062 Alm...			Synchronized

Figure 5-45 Deactivate

The selected mirrors will be deactivated. See Figure 5-46.

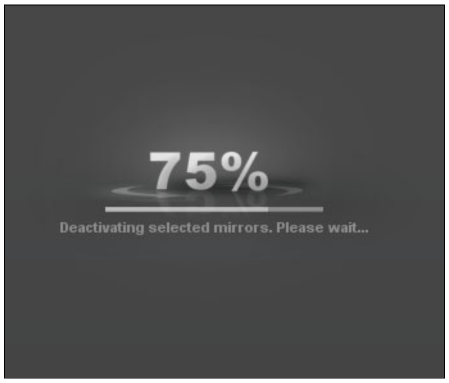
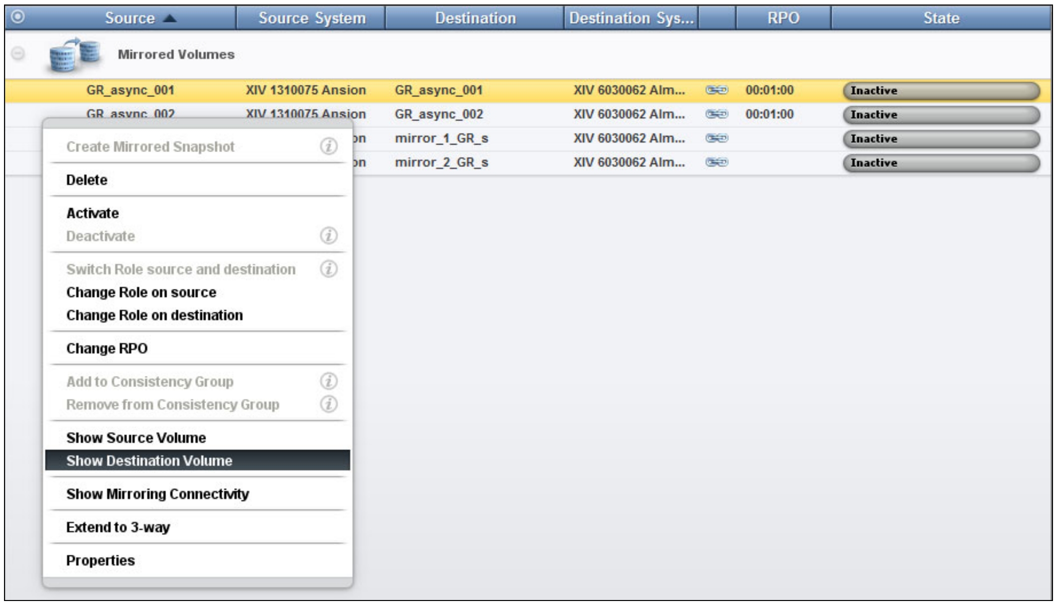


Figure 5-46 Deactivating mirrors

5. The state is changed to Inactive for selected volumes. Right-click and select **Show Destination Volume** to see the state of the remote volume. See Figure 5-47.



Source	Source System	Destination	Destination Sys...		RPO	State
Mirrored Volumes						
GR_async_001	XIV 1310075 Ansion	GR_async_001	XIV 6030062 Alm...		00:01:00	Inactive
GR_async_002	XIV 1310075 Ansion	GR_async_002	XIV 6030062 Alm...		00:01:00	Inactive
	on	mirror_1_GR_s	XIV 6030062 Alm...			Inactive
	on	mirror_2_GR_s	XIV 6030062 Alm...			Inactive

Figure 5-47 Mirror inactive

You can use the **mirror_deactivate** command to deactivate the mirror, as shown in Example 5-8.

Example 5-8 mirror_deactivate

```
>> mirror_deactivate vol="mirror_1_GR" target="XIV 6030062 Almania"
```

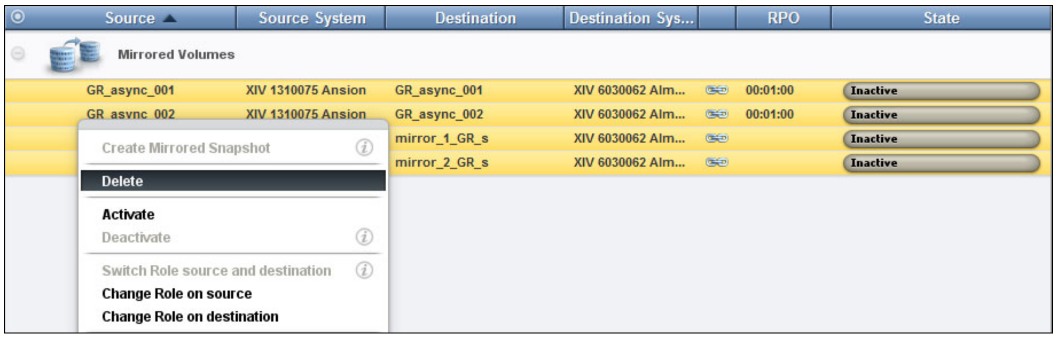
5.4.2 Deleting mirrors, snapshots, and destination volumes

This section describes how to delete mirrors, snapshots, and destination volumes.

Deleting mirrors

Complete the following steps to delete the mirrors:

1. Select all mirrored volumes to be deleted by right-clicking. Hold the Ctrl key to select multiple volumes. After all volumes are selected, click **Delete**. See Figure 5-48.

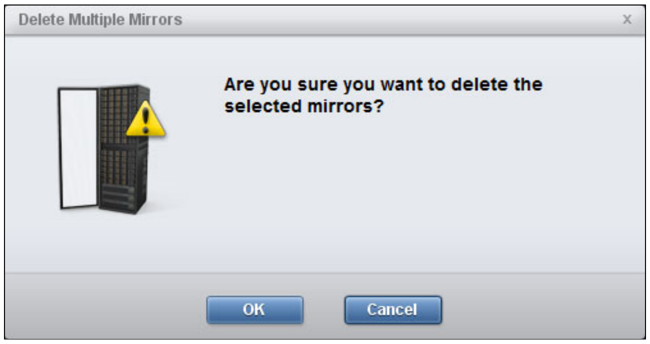


Mirrored Volumes

Source	Source System	Destination	Destination Sys...		RPO	State
GR_async_001	XIV 1310075 Ansion	GR_async_001	XIV 6030062 Alm...		00:01:00	Inactive
GR_async_002	XIV 1310075 Ansion	GR_async_002	XIV 6030062 Alm...		00:01:00	Inactive
		mirror_1_GR_s	XIV 6030062 Alm...			Inactive
		mirror_2_GR_s	XIV 6030062 Alm...			Inactive

Figure 5-48 Delete mirrors

2. Click **OK** to confirm that you want to delete the mirrors. See Figure 5-49.



Delete Multiple Mirrors

Are you sure you want to delete the selected mirrors?

Figure 5-49 Confirm the mirror delete

The selected mirrors are deleted and disappear from the Mirroring view.

You can use the **mirror_delete** XCLI command to delete mirrors. See Example 5-9.

Example 5-9 mirror_delete

```
>> mirror_delete vol="GR_async_001" target="XIV 6030062 Almania"
```

Deleting snapshots for asynchronous mirrors

Complete the following steps for asynchronous mirrors:

1. Go to the Volumes by Pools view and select the last-replicated snapshots from the deleted asynchronous mirrors by right-clicking them. Hold the Ctrl key to select multiple snapshots. See Figure 5-50.

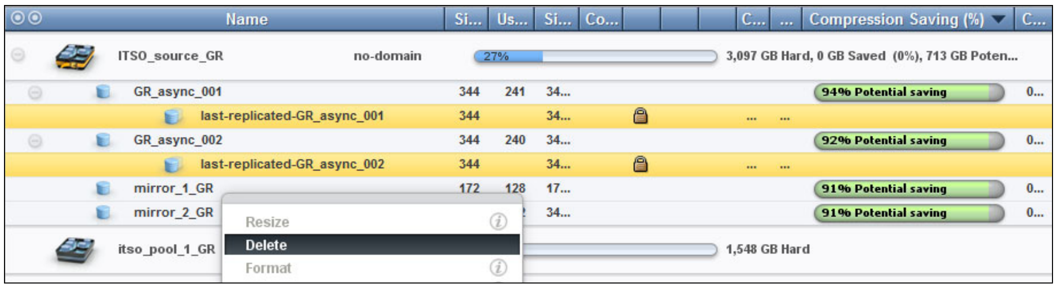
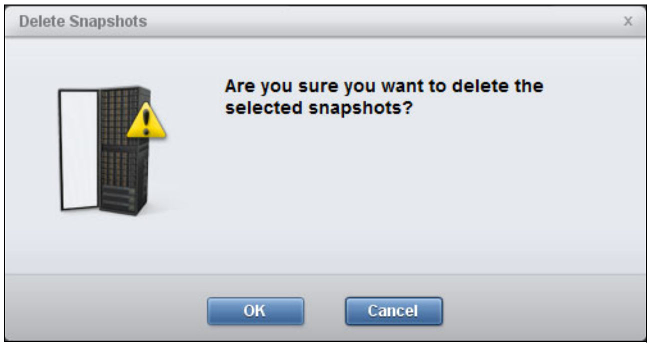


Figure 5-50 Delete last-replicated snapshots

2. Confirm the snapshot deletion by clicking **OK**. See Figure 5-51.



Are you sure you want to delete the selected snapshots?

Figure 5-51 Confirm snapshot deletion

You can use the **`snapshot_delete`** command to delete snapshots with the XCLI. See Example 5-10.

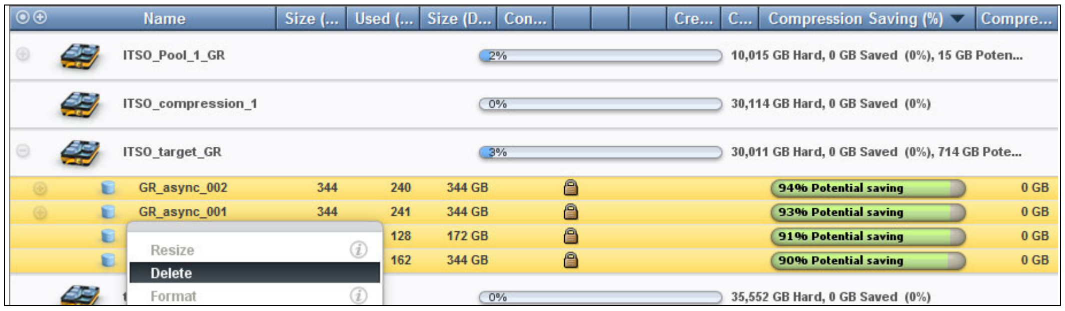
Example 5-10 Example snapshot_delete command

```
>> snapshot_delete snapshot="last-replicated-GR_async_001"
```

Completing mirror cleanup

Perform the following steps to complete the mirror cleanup:

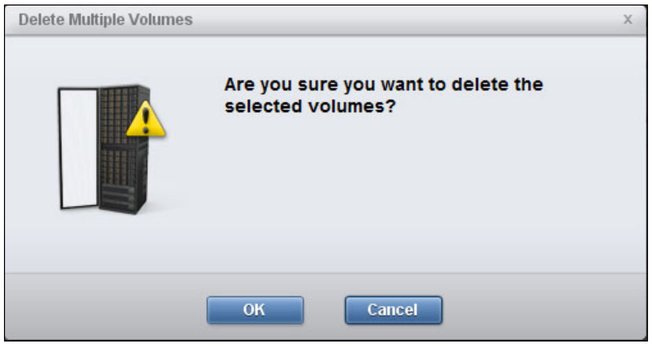
1. Switch to the remote XIV system, go to the Volumes by Pools view, and delete the target volumes and last-replicated snapshots. Select target volumes to delete by right-clicking. Hold the Ctrl key to select several volumes.
2. Click **Delete**. See Figure 5-52.



Name	Size (...	Used (...	Size (D...	Con...				Cre...	C...	Compression Saving (%)	Compre...
ITSO_Pool_1_GR				2%						10,015 GB Hard, 0 GB Saved (0%), 15 GB Poten...	
ITSO_compression_1				0%						30,114 GB Hard, 0 GB Saved (0%)	
ITSO_target_GR				3%						30,011 GB Hard, 0 GB Saved (0%), 714 GB Pote...	
GR_async_002	344	240	344 GB	🔒						94% Potential saving	0 GB
GR_async_001	344	241	344 GB	🔒						93% Potential saving	0 GB
		128	172 GB	🔒						91% Potential saving	0 GB
		162	344 GB	🔒						90% Potential saving	0 GB
				0%						35,552 GB Hard, 0 GB Saved (0%)	

Figure 5-52   Delete target volumes

3. Click **OK** to confirm deletion (Figure 5-53).



Delete Multiple Volumes

Are you sure you want to delete the selected volumes?

Figure 5-53   Confirm deletion

You an use the **`vol_delete`** command to delete volumes with the XCLI. See Example 5-11.

Example 5-11   vol_delete

```
>> vol_delete vol="GR_async_002"
```

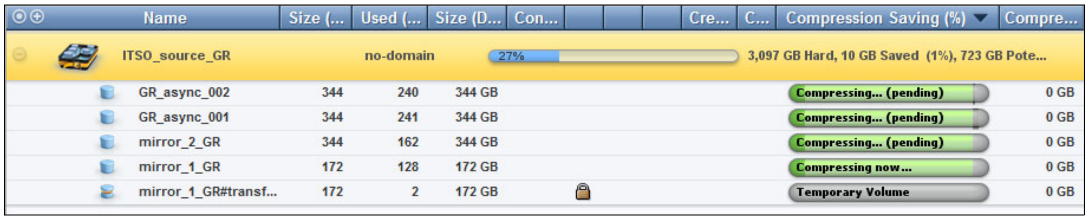
5.4.3 Compressing the volumes

Before you compress the volumes, verify the following things:

- The source and target pools must be thin pools. See “Using the GUI to change a regular pool to a thin pool” on page 40.
- The source pool must have enough free capacity.

To compress the volumes, follow the steps for either 5.3.2, “Compressing single volumes” on page 46, or for 5.3.3, “Compressing all volumes in a pool” on page 48.

Figure 5-54 on page 61 shows that compression is occurring. The volumes are compressed one at a time, **Keep source Volume** was *not* selected, and the Temporary Volumes are deleted when the compression is done.



Name	Size (...	Used (...	Size (D...	Con...	Cre...	C...	Compression Saving (%)	Compre...
ITSO_source_GR		no-domain		27%			3,097 GB Hard, 10 GB Saved (1%), 723 GB Pote...	
GR_async_002	344	240	344 GB				Compressing... (pending)	0 GB
GR_async_001	344	241	344 GB				Compressing... (pending)	0 GB
mirror_2_GR	344	162	344 GB				Compressing... (pending)	0 GB
mirror_1_GR	172	128	172 GB				Compressing now...	0 GB
mirror_1_GR#transf...	172	2	172 GB	🔒			Temporary Volume	0 GB

Figure 5-54 Compress the volumes

You can use the **vol_transform_create** XCLI command to compress volumes. See Example 5-12.

Example 5-12 Example vol_transform_create command to compress a volume

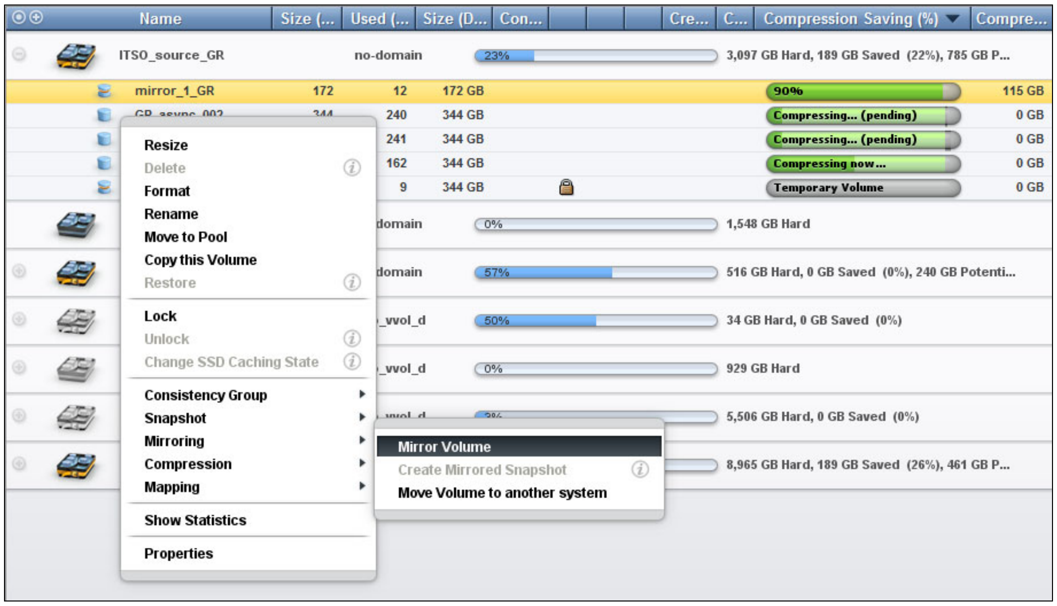
```
>> vol_transform_create delete_source=yes mode=compress vol="mirror_1_GR"
```

5.4.4 Creating and activating the mirror

As soon as the compression is complete, the mirror can be set up again.

Complete the following steps to create and activate the mirror:

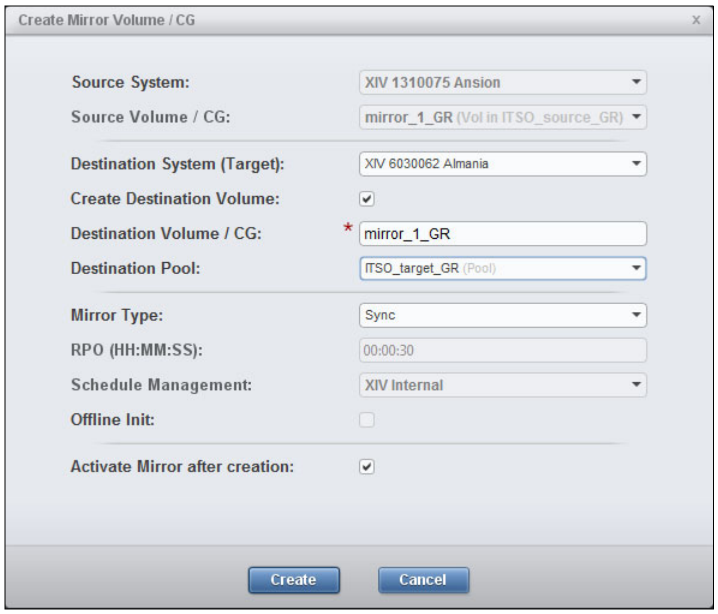
1. Right-click the volume in the Volumes by Pools view.
2. Select **Mirroring** → **Mirror Volume**. See Figure 5-55.



Name	Size (...	Used (...	Size (D...	Con...	Cre...	C...	Compression Saving (%)	Compre...
ITSO_source_GR		no-domain		23%			3,097 GB Hard, 189 GB Saved (22%), 785 GB P...	
mirror_1_GR	172	12	172 GB				90%	115 GB
GR_async_002	344	240	344 GB				Compressing... (pending)	0 GB
		241	344 GB				Compressing... (pending)	0 GB
		162	344 GB				Compressing now...	0 GB
		9	344 GB	🔒			Temporary Volume	0 GB
		domain		0%			1,548 GB Hard	
		domain		57%			516 GB Hard, 0 GB Saved (0%), 240 GB Potenti...	
		_vvol_d		50%			34 GB Hard, 0 GB Saved (0%)	
		_vvol_d		0%			929 GB Hard	
		vvol_d		3%			5,506 GB Hard, 0 GB Saved (0%)	
							8,965 GB Hard, 189 GB Saved (26%), 461 GB P...	

Figure 5-55 Mirror Volume

3. On the next panel, enter the required information to create and activate the mirror. See Figure 5-56.


Create Mirror Volume / CG

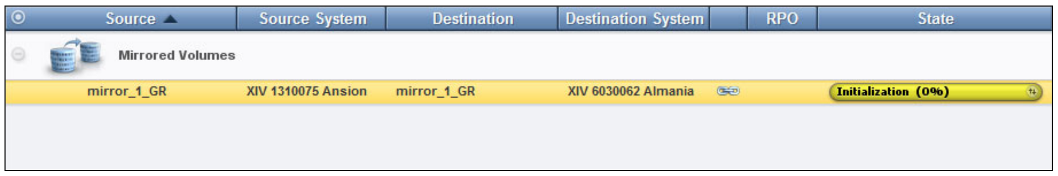
Source System:	XIV 1310075 Ansion
Source Volume / CG:	mirror_1_GR (Vol in ITSO_source_GR)
Destination System (Target):	XIV 6030062 Almania
Create Destination Volume:	✓
Destination Volume / CG: *	mirror_1_GR
Destination Pool:	ITSO_target_GR (Pool)
Mirror Type:	Sync
RPO (HH:MM:SS):	00:00:30
Schedule Management:	XIV Internal
Offline Init:	
Activate Mirror after creation:	✓

Figure 5-56 Create mirror

Enter the following information in the Create Mirror Volume / CG panel shown in Figure 5-56:

- **Destination System (Target):** Select the target XIV for the destination volume.
- **Create Destination Volume:** Select to let the XIV create the target volume.
- **Destination Volume / CG:** Specify the name of the volume or CG on the target XIV.
- **Destination Pool:** Select the destination pool.
- **Mirror Type:** Decide whether the mirror is asynchronous or synchronous.
- **Activate Mirror after creation:** Select to activate the mirror immediately after creation, otherwise the mirror needs to be activated manually.

4. Click **Create** to create and activate the mirror. Because Activate Mirror after creation was selected, XIV creates and activates the mirror in one step, as shown in Figure 5-57.

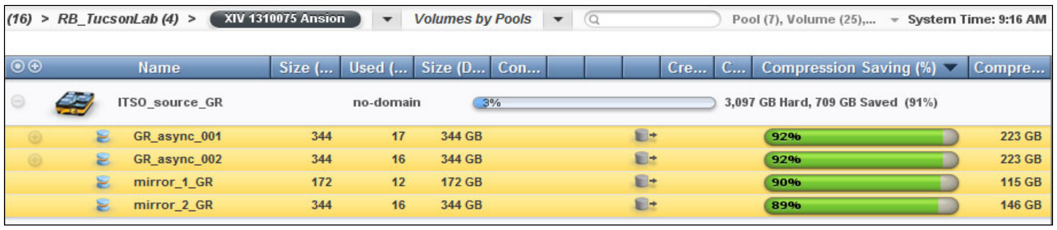


Source	Source System	Destination	Destination System	RPO	State
Mirrored Volumes					
mirror_1_GR	XIV 1310075 Ansion	mirror_1_GR	XIV 6030062 Almania		Initialization (0%)

Figure 5-57 Create and activate mirror

5. Repeat step 4 for all compressed volumes to restore the mirrors. The initialization is much faster than for conventional volumes, because less data needs to be synchronized.

   The conversion for mirrored volumes is complete. Using compression in a mirrored environment is very effective, because disk space is saved. Figure 5-58 on page 63 shows the source.

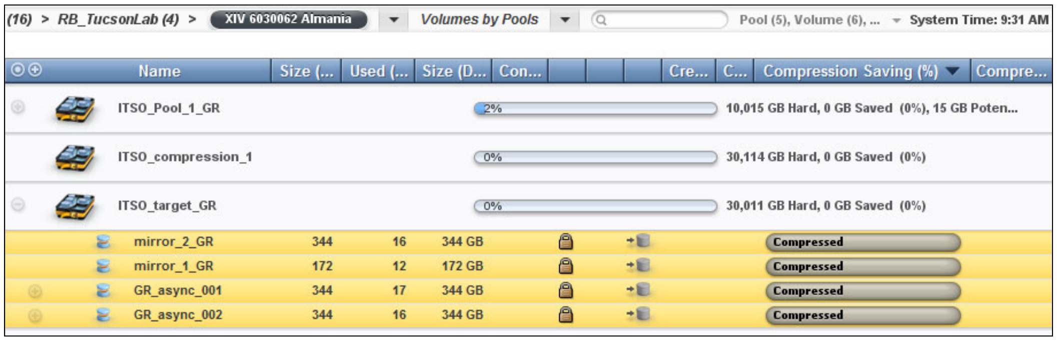


(16) > RB_TucsonLab (4) > XIV 1310075 Ansion | Volumes by Pools | Pool (7), Volume (25),... | System Time: 9:16 AM

Name	Size (...	Used (...	Size (D...	Con...				Cre...	C...	Compression Saving (%)	Compre...
ITSO_source_GR		no-domain	3%							3,097 GB Hard, 709 GB Saved (91%)	
GR_async_001	344	17	344 GB							92%	223 GB
GR_async_002	344	16	344 GB							92%	223 GB
mirror_1_GR	172	12	172 GB							90%	115 GB
mirror_2_GR	344	16	344 GB							89%	146 GB

Figure 5-58 Compressed mirror, source

Figure 5-59 shows the target.



(16) > RB_TucsonLab (4) > XIV 6030062 Almania | Volumes by Pools | Pool (5), Volume (6), ... | System Time: 9:31 AM

Name	Size (...	Used (...	Size (D...	Con...				Cre...	C...	Compression Saving (%)	Compre...
ITSO_Pool_1_GR			2%							10,015 GB Hard, 0 GB Saved (0%), 15 GB Poten...	
ITSO_compression_1			0%							30,114 GB Hard, 0 GB Saved (0%)	
ITSO_target_GR			0%							30,011 GB Hard, 0 GB Saved (0%)	
mirror_2_GR	344	16	344 GB							Compressed	
mirror_1_GR	172	12	172 GB							Compressed	
GR_async_001	344	17	344 GB							Compressed	
GR_async_002	344	16	344 GB							Compressed	

Figure 5-59 Compressed mirror, target

You can also use the **`mirror_create_target`** and **`mirror_activate`** XCLI commands to create and activate a mirror.

Example 5-13 shows a synchronous mirror.

Example 5-13 Example commands to create and activate a synchronous mirror

```
>> mirror_create_target="XIV 6030062 Almania" vol="mirror_2_GR"
slave_vol="mirror_2_GR" remote_pool="ITSO_target_GR" create_slave=yes
>> mirror_activate vol="mirror_2_GR" target="XIV 6030062 Almania"
```

Example 5-14 shows an asynchronous mirror.

Example 5-14 Example commands to create and activate mirror an asynchronous mirror

```
>> mirror_create target="XIV 6030062 Almania" vol="GR_async_001"
slave_vol="GR_async_001" remote_pool="ITSO_target_GR" create_slave=yes
type=ASYNC_INTERVAL rpo=60 remote_rpo=60
>> mirror_activate vol="GR_async_001" target="XIV 6030062 Almania"
```

5.5 Disabling compression

Before compression can be disabled, all volumes must be decompressed. See 5.3.4, “Converting a compressed volume to conventional volume” on page 51. As an administrator, complete the following steps to disable compression:

1. Connect with the GUI to System. Click **Systems** → **System Settings** → **System**. See Figure 5-60.

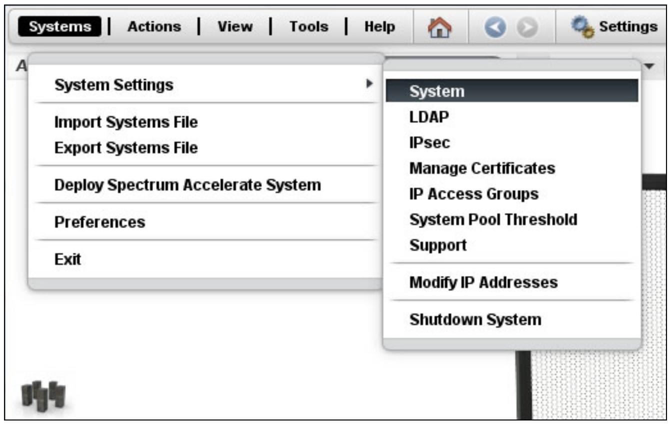
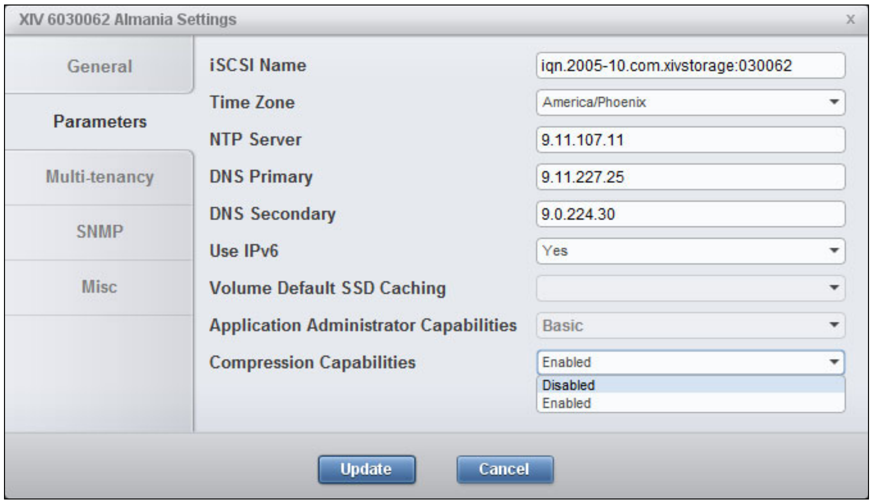


Figure 5-60 System Settings

2. In the Settings window, select the Parameters tab to change the system settings. See Figure 5-61.



iSCSI Name	iqn.2005-10.com.xivstorage:030062
Time Zone	America/Phoenix
NTP Server	9.11.107.11
DNS Primary	9.11.227.25
DNS Secondary	9.0.224.30
Use IPv6	Yes
Volume Default SSD Caching	
Application Administrator Capabilities	Basic
Compression Capabilities	Enabled (Disabled / Enabled)

Figure 5-61 Settings, Parameters

3. Change the Compression Capabilities to **Disabled** and click **Update**.

You can also use the **system_compression_disable** XCLI command to disable IBM Real-time Compression. See Example 5-15.

Example 5-15 Disable compression

```
XIV 6030062 Almania>> system_compression_disable
Command executed successfully.
```

6 Performance

This chapter provides general information about performance when using IBM Real-time Compression in the IBM XIV Gen3 Storage System. In addition, it provides results of performance tests and information about how to correctly assess performance.

It contains the following information:

- 6.1, “Performance overview” on page 66
- 6.2, “Appropriate workloads for compression” on page 69
- 6.3, “Application benchmark results” on page 70

© Copyright IBM Corp. 2015. All rights reserved.

6.1 Performance overview

Traditional implementations of compression technologies in storage systems do not yield real-time capacity savings or high performance for compressed primary workloads.

IBM XIV with IBM Real-time Compression can provide great, real-time capacity savings *and* consistent high performance for real-life application workloads.

One of the most striking benefits of XIV with Real-time Compression is the ability to display a similar level of performance for both compressed volumes and uncompressed volumes in many typical workload scenarios.

The *maximum* possible system throughput for uncompressed volumes may be higher than for compressed volumes, but differences in performance will typically be noticeable only if the system is highly taxed, and at performance levels that are arguably much higher than typically observed customer demands. See Figure 6-1.

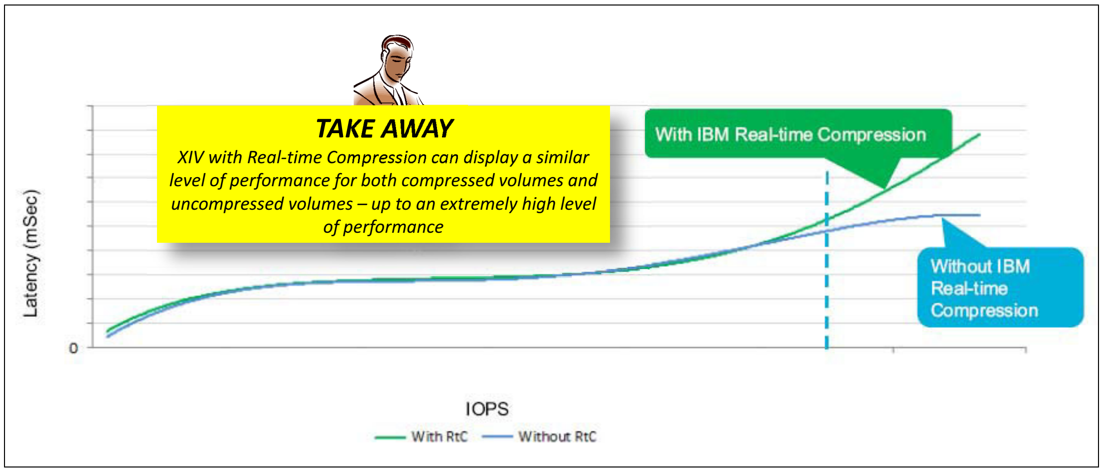


Figure 6-1 Similar level of performance for both compressed volumes and uncompressed volumes

IBM Real-time Compression on the IBM XIV Storage System can attain high performance while delivering a high compression ratio.

Example: Understanding compression rates, ratios, and savings

To clarify the meaning of the terms *compression ratio*, *compression savings rate*, and *compression savings*, consider a use case where the original data physical capacity before compression was 100 TB, and the physical data capacity after compression is 40 TB.

The following values help to clarify these terms:

- Compression rate = 60%
- Compression savings rate = 60%
- Compression savings = 60 TB
- Compression ratio = original size (100 TB) *divided by* the size on disk after compression (40 TB) = 2.5:1

When you consider savings, it is easiest to use the compression rate.

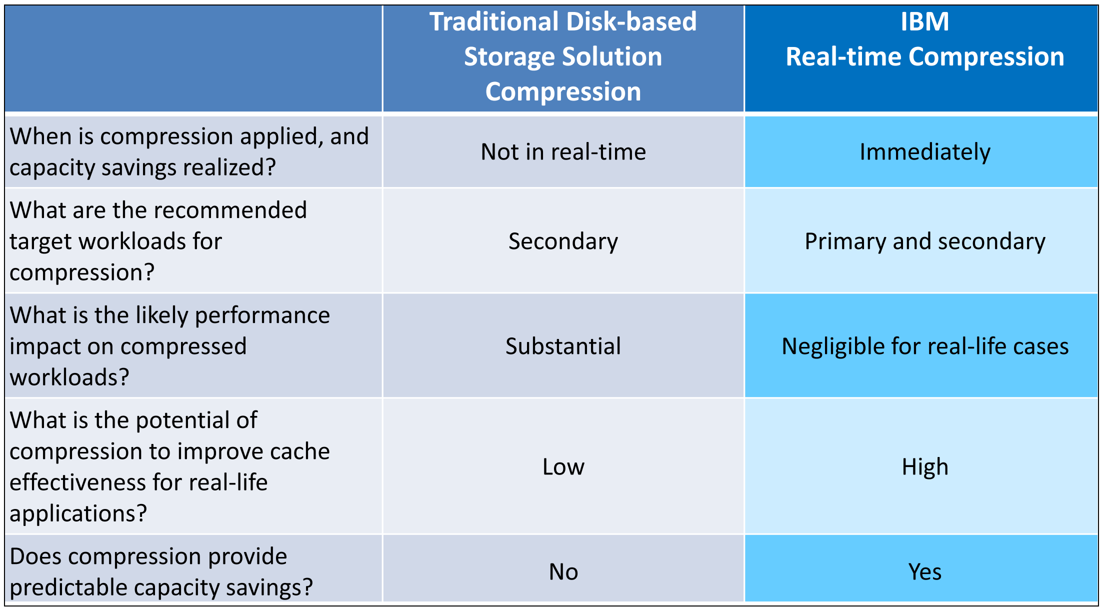
The compression ratio helps in understanding how much effective data you can store on your system. So, when you have a 2.5:1 compression ratio, you will be able to store 250 TB of data on 100 TB of physical capacity.

IBM Real-time Compression on the IBM XIV Storage System has the following characteristics:

- It is a technology that can deliver great compression savings for a range of application workloads.
- IBM Real-time Compression uses IBM RACE technology, which was purpose-built for real-life application workloads. Among other things, RACE can use temporal locality of data to realize maximized compression benefits.

  Temporal locality for data denotes the degree to which chunks of data that is written around the same time are related, and so likely to be read together in the future. Therefore, IBM Real-time Compression can minimize disk access and compression/decompression resource use, and maximize capacity utilization and compression performance benefits for data that is characterized by high temporal locality.
- With XIV, neither the system nor the administrator needs to optimize performance. The system can provide an estimation of potential compression savings for volumes, which is displayed through the UI. This facilitates assessment of potential savings before compressing existing data, and maximizes deployment value. Furthermore, IBM Real-time Compression supports non-disruptive conversion of existing volumes, yielding an easy way to reclaim capacity and accelerate ROI.

Figure 6-2 shows key benefits of IBM Real-time Compression.



	Traditional Disk-based Storage Solution Compression	IBM Real-time Compression
When is compression applied, and capacity savings realized?	Not in real-time	Immediately
What are the recommended target workloads for compression?	Secondary	Primary and secondary
What is the likely performance impact on compressed workloads?	Substantial	Negligible for real-life cases
What is the potential of compression to improve cache effectiveness for real-life applications?	Low	High
Does compression provide predictable capacity savings?	No	Yes

Figure 6-2 Key benefits of IBM Real-time Compression

This section provides the following information about performance for IBM Real-time Compression on the XIV:

- 6.1.1, "Compression and system performance" on page 68
- 6.1.2, "Converting uncompressed volumes" on page 68
- 6.1.3, "Effect of compression on performance for common XIV functions" on page 69

6.1.1 Compression and system performance

The IBM Random Access Compression Engine (RACE) uses a journal structure to manage the user data. This implementation helps optimize the I/O service for applications (OLTP, VMware applications, and so on).

IBM Real-time Compression is designed to attain both high compression and storage performance in real-life scenarios. In most cases where a compression of 30% or higher is attained and high temporal locality is observed (real-life applications), if the compression hardware resources (CPU and memory) are not fully loaded, negative performance impact is not expected.

Adding flash will typically improve both IOPS and latency (up to 800% improvement).

6.1.2 Converting uncompressed volumes

Compression is set by default to operate at a 300 MBps rate, but that rate can be modified. Increasing the rate might affect overall system performance, depending on the bandwidth requirements of active workloads and processes (for example, application storage I/O, rebuild, and remote replications).

Example of converting an uncompressed volume to a compressed volume:

Consider an XIV system with 3 volumes (a, b, and c) with 100 TB of data. In this example, all three volumes are requested to be compressed.

Compression of the three volumes will be done sequentially (1 by 1). The user can run a command to compress multiple volumes, but the volumes will be compressed one at a time.

The default compression rate was set to maintain a good performance level while minimizing the effect on active system processes and services. Changing the rate might affect overall system performance, depending on the bandwidth requirements of active workloads and processes.

No functions will be disabled during the process.

6.1.3 Effect of compression on performance for common XIV functions

This section describes the effect of compression on the performance for the following XIV functions:

- Remote replication (mirroring)
- Snapshots

Remote replication (mirroring)

IBM Real-time Compression can realize a high compression ratio in real time. Consider a request to replicate data of a compressed volume to a remote location (remote replication). The effective compression that is realized in real time with IBM Real-time Compression will result in minimized capacity to replicate, and a potentially shorter effective RPO for asynchronous replication of a volume. IBM Real-time Compression might nonetheless affect the observed latency with synchronous replication of compressed data, subject to the workload, though the effect will be lower as the compression ratio is higher.

Snapshots

Snapshot restore performance will not be affected by compression.

6.2 Appropriate workloads for compression

Do not compress data or data types that are already compressed or encrypted (that is, compressed or encrypted before the data reaches the storage system). Selecting such data to be compressed provides little or no savings, yet uses processor resources and in some cases might even generate more I/Os.

Evaluate workloads that are extremely sensitive to I/O latency before using compression. In cases where there is little or no correlation between the data write order and the read order (that is, no temporal locality benefits), compression might deliver suboptimal performance.

When to use compression:

Use compression for data with compression savings of 25% or higher (that is, 1.33:1 compression ratio).

In this context, 25% savings means that 100 (uncompressed) becomes 75 (compressed). That is, 100/75 = 1.33 compression ratio.

It is important to understand compression performance before evaluating IBM Real-time Compression performance.

If the system CPU, disk utilization, or both are overloaded, enabling compression will affect the overall service level. However, if there is a low cache hit ratio, migrating volumes with a good compression ratio might possibly increase the cache hit and potentially improve performance.

Workloads that are best suited to compression

The following workloads are best suited to compression:

- General purpose volumes
- OLTP Databases
- Virtualized workloads
- Sequential workloads
- VDI
- Backup

Workloads that are not recommended for compression

The following workloads are not recommended for compression, because they will not result in compression capacity savings, or might result in non-negligible performance impact:

- Workloads with less than 25% savings (that is, a 1.33:1 compression ratio) should not be compressed, unless these specific savings are significant and make business sense to the client, and the client may be willing to compromise performance with these volumes.
- Workloads using application encrypted data, where the data is encrypted before it reaches the storage system, should not be compressed.

6.3 Application benchmark results

This section provides information and observations concerning performance tests that have been run for XIV and IBM Real-time Compression. The tests covered a broad spectrum of workloads, including OLTP, sequential, and pure synthetic profiles. This section expands on each test environment used, including its pertinent system configuration, and portrays how IBM real-time Compression can maintain high performance while delivering high compression ratios with varied workload profiles.

This section presents the following results:

- 6.3.1, “Test case 1: Oracle online transaction benchmark” on page 71
- 6.3.2, “Test case 2: Synthetic workloads” on page 73
- 6.3.3, “Test case 3: VDbench maximum throughput in MBps” on page 75

6.3.1 Test case 1: Oracle online transaction benchmark

This test demonstrates how XIV performance with IBM Real-time Compression maintains high performance with typical OLTP workloads in application environments.

Target workload

This benchmark simulates an order-entry application by running a mixture of read-only and update-intensive transactions typical of online transaction processing (OLTP) environments. The benchmark incorporated five transaction types (for example, New Order, Delivery, and Payment). Throughput is measured in transactions per minute. The benchmark also reports the response time per transaction, which is broken out by transaction type. The benchmark ran these transactions against the database:

- STOCK LEVEL: Checking the stock level
- DELIVERY: Processing a batch of 10 orders
- ORDER STATUS: Monitoring the status of orders
- PAYMENT: Processing a payment
- NEW ORDER: Entering a complete order

The benchmark measures transactions per minute (tpmC), which indicates new order transactions run per minute and provides a measure of business throughput. The benchmark also measures response time, which is the average time that a user got a response for each transaction.

Test configuration

The test used 3 IBM X3550 M4 hosts with the following configuration:

- Memory: 128 GB
- CPU: Xeon E5-2680 at 2.70 GHz
- Fibre Channel HBA: QLE2562 QLogic 8 Gb FC Dual-port

The 3 hosts were set up as a Real Application Cluster (RAC).

The following configuration for the TPC-C benchmark was used:

- Quest benchmark factory (v 6.6.1)
- Users: 15,000 up to Max
- Oracle 11.2.03
- Scale 50,000
- 5 TB total Oracle database
- 75% CR compression rate (that is, a reduction of 75%, or 100 GB of data is compressed to 25 GB)
- Oracle memory (SGA) 30 GB

Table 6-1 lists the XIV Gen 3 configurations that were used for the benchmarks.

Table 6-1 XIV Gen3 configurations

Component	**XIV 114 System**	**XIV 214 System**
Number of modules	15	15
RAM	24 GB x 15	48 GB x 15
Disks	180 x 3 TB 7.2K rpm disks	180 x 4 Tb 7.2K RPM Disks
Total Fibre Channel ports	12	12
SSDs	15 x 512 GB	15 x 800 GB

Test results

This section shows the results of the tests run.

Table 6-2 shows the results of the Oracle benchmarks.

Table 6-2 Oracle TPC-C benchmark results

Transaction	XIV Gen3 model 114, 5 TB	XIV Gen3 model 214, 5 TB
Stock Level	40.33 sec	29.16 sec
Delivery	4.82 sec	17.56 sec
Order status	0.55 sec	0.37 sec
Payment	0.68 sec	0.59 sec
New Order	2.09 sec	1.65 sec
Average Response Time in Seconds	3.06 sec	2.88 sec
tpmC (Throughput)	30,459	50,525
Average IOPS Achieved	99,097	144,532

Note: *Average Response Time in Seconds* is not the statistical/mathematical average of the response time, but rather an average time that the benchmark delivers based on the weight of transactions and their rate per minute. It is also worth mentioning that a single TPC-C transaction can be composed of tens or hundreds of SQL queries, which may generate thousands or even more I/Os to complete a single business transaction.

Figure 6-3 shows Oracle TPC-C benchmark results with a 4:1 compression ratio.

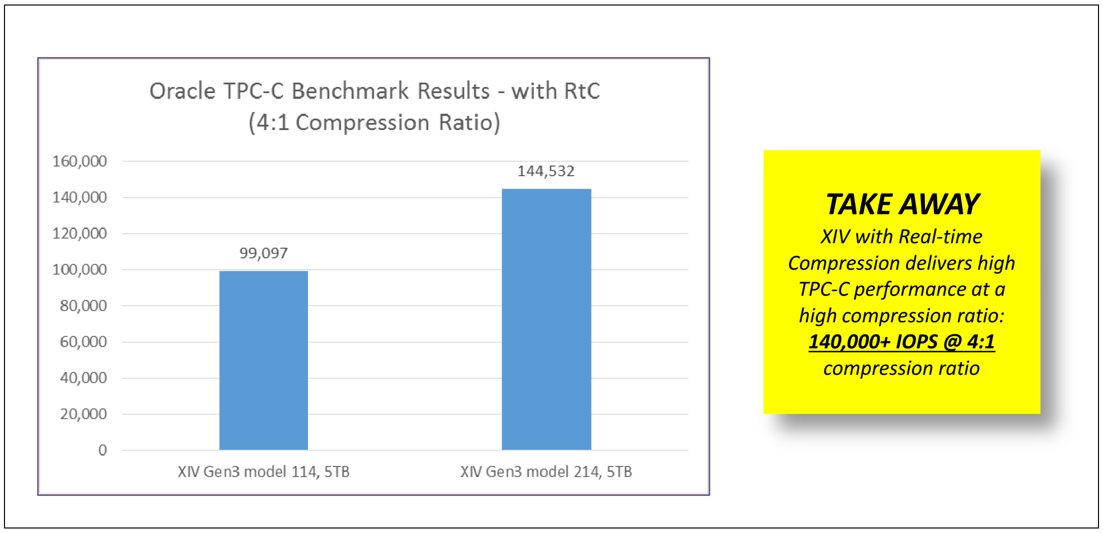


Figure 6-3 Oracle TPC-C benchmark results with a 4:1 compression ratio

Explanation of results

The system attains > 144,000 IOPS for a 4:1 compression ratio.

Note: With IBM Real-time Compression, higher compression ratios (for example, 5:1) will likely result in even higher performance.

6.3.2 Test case 2: Synthetic workloads

This test showcases the maximum IOPS and bandwidth delivered by XIV with Real time Compression enabled. It also demonstrates performance of XIV with IBM Real-time Compression under impractically low temporal locality conditions.

Important: Synthetic benchmarks do not correctly reflect the relative performance between compressed and uncompressed configurations. Furthermore, the ratio between maximum performance with compressed and uncompressed configuration does not apply to performance for lower-than-max workloads. Synthetic benchmarks model non-real-life workloads which feature low, or no temporal locality characteristics.

With benchmarks modeling real life applications, observed differences in performance between an XIV system with compression enabled and an XIV system with compression disabled are likely to be small for a large number of typical profiles.

These tools generate synthetic workloads that do not have any temporal locality. Data is not read back in the same (or similar) order in which it was written. It is therefore not useful to estimate what your performance will look like for an application with these tools.

Consider what data a benchmark application uses. If the data is already compressed, or is all binary zero data, the differences that are measured are artificially bad, or good, based on the compressibility of the data. The more compressible the data, the better the performance is.

Target workload

To demonstrate the variability between workloads that demonstrate zero and 100% temporal locality, you must modify traditional block benchmark tools to enable repeatable random workloads. This process results in best case and worst case raw block performance workloads when using compressed volumes. These tests were run with a known compressibility of data blocks by using a 1 MB pattern file that has a 65% compression ratio.

Test configuration

The following configuration was used for 6 IBM X3550 M4 hosts:

- Memory: 128 GB
- CPU: Xeon E5-2650 at 2.00 GHz
- Fibre Channel HBA: QLE2562 - QLogic 8 Gb FC Dual-port
- HBA queue depth: 64

The following 4K I/O benchmark configuration was used:

- Data set: 2.4 TB (24 Vdisks x 103 GB)
- Compression ratio: 65% (that is, a reduction of 65%, or 100 GB is compressed to 35 GB)

Table 6-3 lists the XIV Gen 3 configurations that were used for the benchmarks.

Table 6-3 XIV Gen3 configurations

Component	XIV 214 System
Number of modules	15
RAM	48 GB x 15
Disks	180 x 4 Tb 7.2K RPM Disks
Total Fibre Channel ports	12
SSDs	15 x 800 GB

Test results

Table 6-4 shows these performance results, which can be used as a generic framework for upper-limit expected performance for several models of XIV Gen3 using compression.

Table 6-4 Performance results for the 4K I/O benchmark

	XIV Gen3 model 214, 2.4 TB	
Operation	**Worst case**	**Best case**
Read 4 KB Random IOPS	133,935	343,694
70/30 4 KB Random IOPS	141,762	274,068
Write 4 KB Random IOPS	96,670	193,154

Figure 6-4 shows a visual representation of best case (100% temporal locality) and worst case (0% temporal locality) performance results.

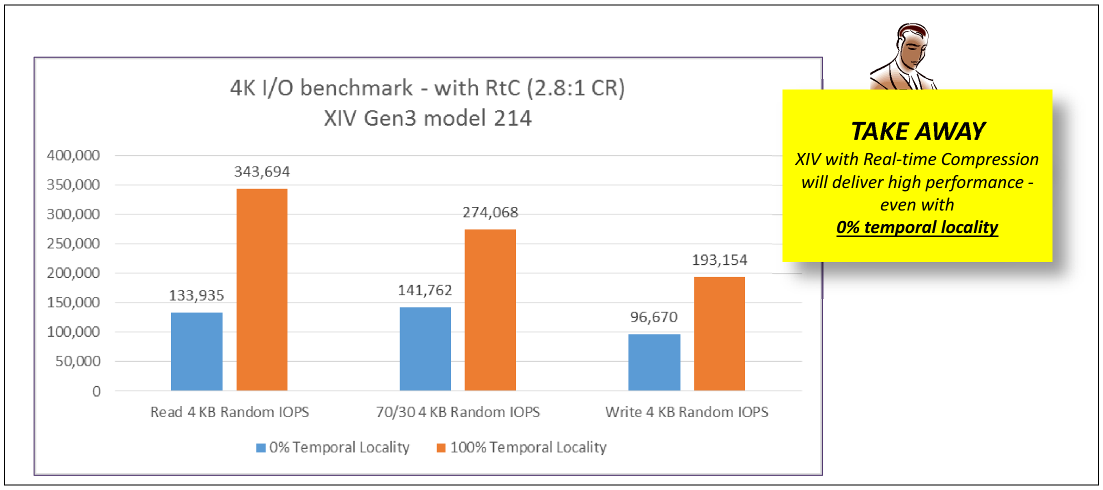


Figure 6-4 Performance results measured in IOPS for best and worst case workloads

Explanation of results

Synthetic benchmark results reflect theoretic workloads and do not correctly reflect real life performance with Real-time Compression. The reason they are presented here is to highlight that, even when tested against a workload profile that has 0% temporal locality, the system can attain very high performance with compression.

A best case reflects a repeatable workload, where the order in which blocks were "pseudo-randomly" written is repeated when reading back in the same "pseudo-random" order. A worst case scenario reflects 0% temporal locality, a workload with no correlation between write and read I/O patterns. In such a case, the system must read and extract a large chunk of data for every host (volume) 4 KB I/O.

Even when subject to a synthetic benchmark (for example, at a proof of concept) at 0% temporal locality (the worst-case scenario), the system can attain > 96,000 IOPS.

6.3.3 Test case 3: VDbench maximum throughput in MBps

This test shows system sequential performance with IBM Real-time Compression for varying compression ratios (CR). IBM Real-time Compression is optimized for application workloads that are more random in nature, and have a mixture of read and write I/O. Writing sequentially to a few target compressed volumes, or to a narrow area in a single compressed volume, provides lower throughput.

Similarly, sequential read streaming is governed by the decompression performance per core. The VDbench tool is used to demonstrate the maximal bandwidth (in MBps) with varying compression ratios (CR). For more information, see the following link:

http://www.oracle.com/technetwork/server-storage/vdbench-downloads-1901681.html

Target workload

This test demonstrates the maximal bandwidth (in MBps) with varying CRs.

Test configuration

The following configuration was used for 6 IBM X3550 M4 hosts:

- Memory: 128 GB
- CPU: Xeon E5-2650 at 2 GHz
- Fibre Channel HBA: QLE2562 - QLogic 8 Gb FC Dual-port

The following configuration was used for the VDbench benchmark:

- Threads: 4
- Data Set: 600 x 430 GB (258 TB), Write: 6 TB, Read: 3 TB
- Compression rate: 30%, 50%, 65%, 75%, 93% (for example, a 30% compression ratio means a reduction of 30%, so that 100 GB of data now takes 70 GB, or 70% of the space used by the uncompressed data)
- I/O Size: 64 KB

Table 6-5 lists the XIV Gen 3 configurations that were used for the benchmarks.

Table 6-5 XIV Gen3 configurations

Component	**XIV 214 System**
Number of modules	15
RAM	48 GB x 15
Disks	180 x 4 Tb 7.2K RPM Disks
Total Fibre Channel ports	12
SSDs	15 x 800 GB

Test results

Table 6-6 shows the maximum throughput results.

Table 6-6 System performance for VDbench maximum throughput in MBps

	XIV Gen3 model 214 for 2.4 TB	
Block size	**64 KB**	
Compression rate	Compressed write	Compressed read
30%	4,092 MBps	6,212 MBps
50%	4,874 MBps	7,877 MBps
65%	5,831 MBps	8,630 MBps
75%	7,384 MBps	8,792 MBps
93%	8,014 MBps	9,161 MBps

Figure 6-5 shows the results.

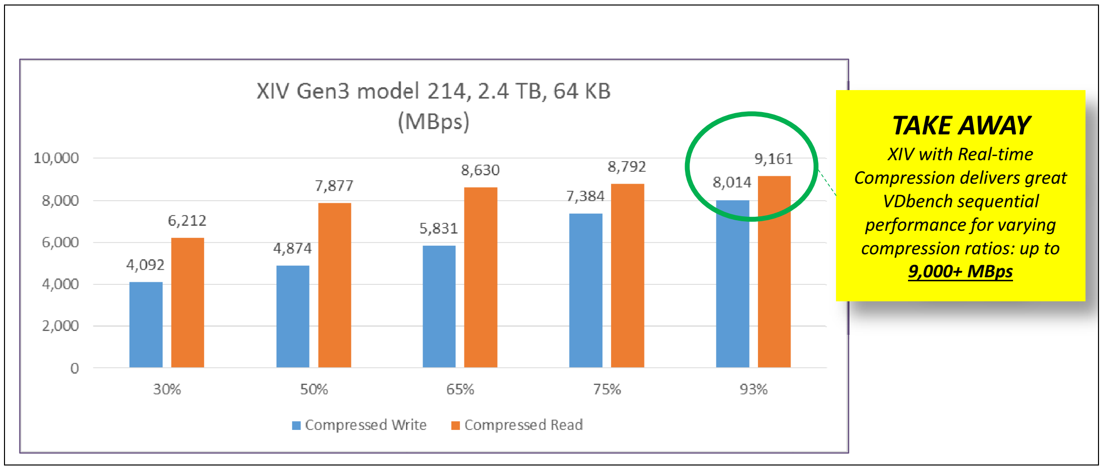


Figure 6-5 Sequential performance with IBM Real-time Compression for varying compression rates

Explanation of results

The system attains > 9,000 MBps.

Even with compression, the system can sustain a large number of IOPS. The more CPU power you have, the more efficient the compression will be.

7

Maintenance and troubleshooting

This chapter describes possibilities to troubleshoot problems that are related to IBM Real-time Compression. It provides some guidance about how to analyze problems, where to find log files, and which data needs to be provided if a problem is reported to IBM Support.

It includes the following sections:

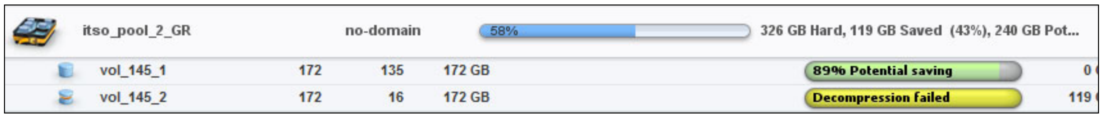
- 7.1, "Analyzing IBM Real-time Compression issues" on page 78
- 7.2, "Involving IBM Support" on page 81

7.1 Analyzing IBM Real-time Compression issues

This section describes how to analyze IBM Real-time Compression issues.

7.1.1 Event log

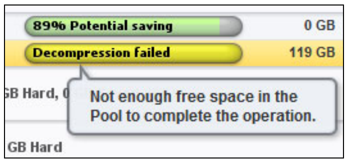
If a problem is encountered during compression or decompression, the process fails and an event is generated in the event log. This event is indicated in the Compression Saving (%) column in the Volumes by Pools view. See Figure 7-1.



itso_pool_2_GR			no-domain	58%	326 GB Hard, 119 GB Saved (43%), 240 GB Pot...	
vol_145_1	172	135	172 GB		89% Potential saving	0
vol_145_2	172	16	172 GB		Decompression failed	119

Figure 7-1 Decompression failed

To obtain details about why the process is failing, hover over the yellow shaded message. A pop-up window shows additional information about the issue. See Figure 7-2.



89% Potential saving 0 GB
Decompression failed 119 GB

Not enough free space in the Pool to complete the operation.

Figure 7-2 More information

Complete event and description can be reviewed in the event log. There are two GUI options to display the events. The first method is to open the **Monitor** icon and click **Events** (Figure 7-3).

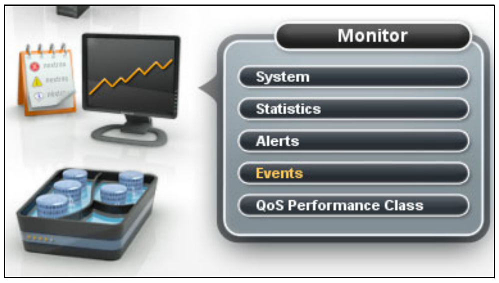


Figure 7-3 Monitor → Events

The second method is to click **View** and select **Monitor** → **Events** (Figure 7-4).

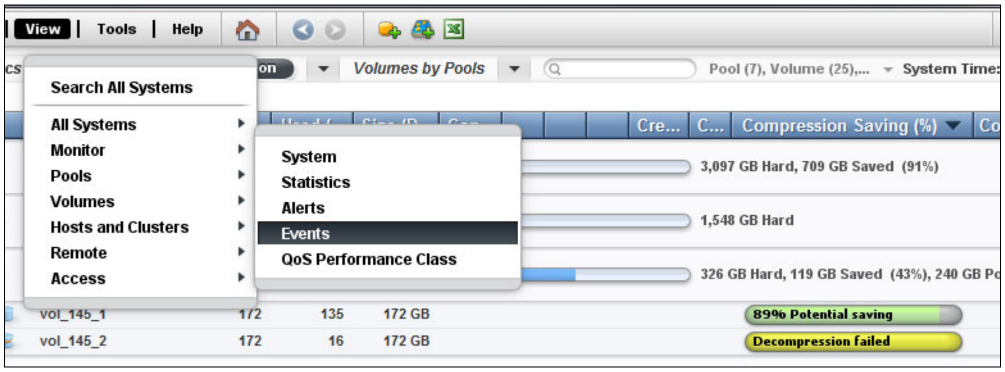
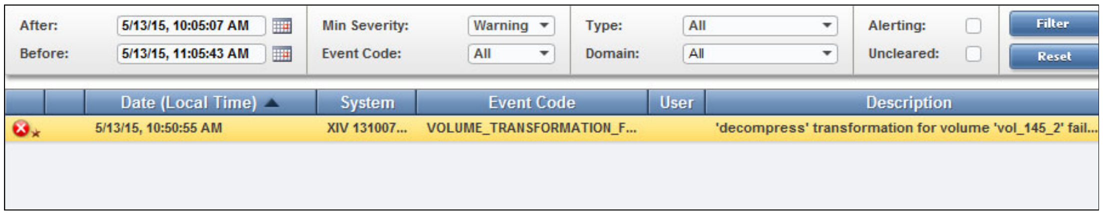


Figure 7-4 View → Monitor → Events

Setting a filter in the event log simplifies finding the corresponding event. Set a one-hour timeframe and filter on Min Severity to reduce the number of events to a manageable view. See Figure 7-5.



After:	5/13/15, 10:05:07 AM	Min Severity: Warning	Type: All	Alerting:	☐	Filter
Before:	5/13/15, 11:05:43 AM	Event Code: All	Domain: All	Uncleared:	☐	Reset

Date (Local Time)	System	Event Code	User	Description
5/13/15, 10:50:55 AM	XIV 131007...	VOLUME_TRANSFORMATION_F...		'decompress' transformation for volume 'vol_145_2' fail...

Figure 7-5 Related event in the log

The event description can be read in the event properties (Figure 7-7 on page 80). To display the properties, right-click the event and select **Event Properties** (Figure 7-6), or double-click the event.

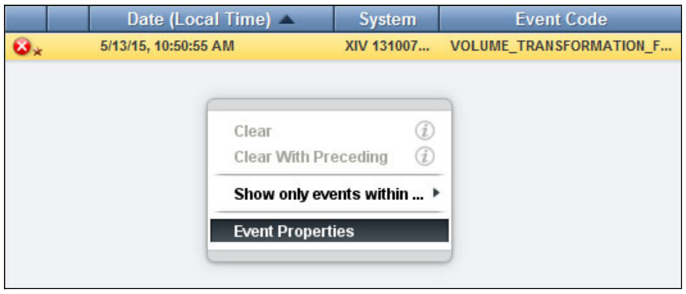
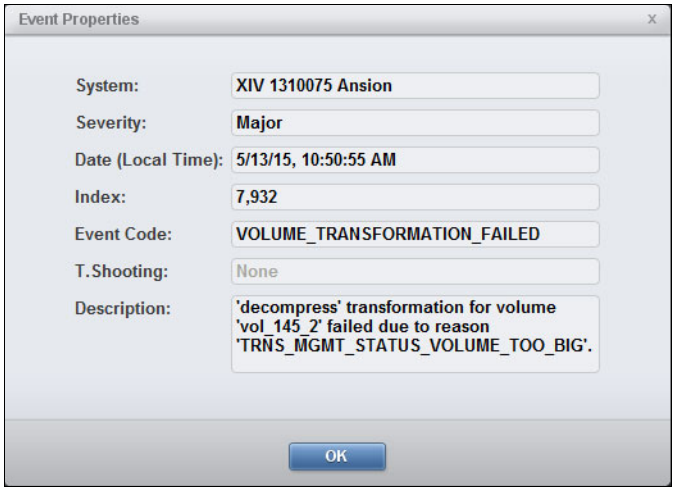


Figure 7-6 View Event Properties

Figure 7-7 shows the event properties.



System:	XIV 1310075 Ansion
Severity:	Major
Date (Local Time):	5/13/15, 10:50:55 AM
Index:	7,932
Event Code:	VOLUME_TRANSFORMATION_FAILED
T.Shooting:	None
Description:	'decompress' transformation for volume 'vol_145_2' failed due to reason 'TRNS_MGMT_STATUS_VOLUME_TOO_BIG'.

Figure 7-7 Event Properties

Decompression failed due to "Not enough space in the pool". To fix this issue, the pool must be resized or the decompression must be canceled.

Back on the Volumes by Pools view, there are two options to go ahead. Right-click the volume that failed the process and select **Compression**, as shown in Figure 7-8.

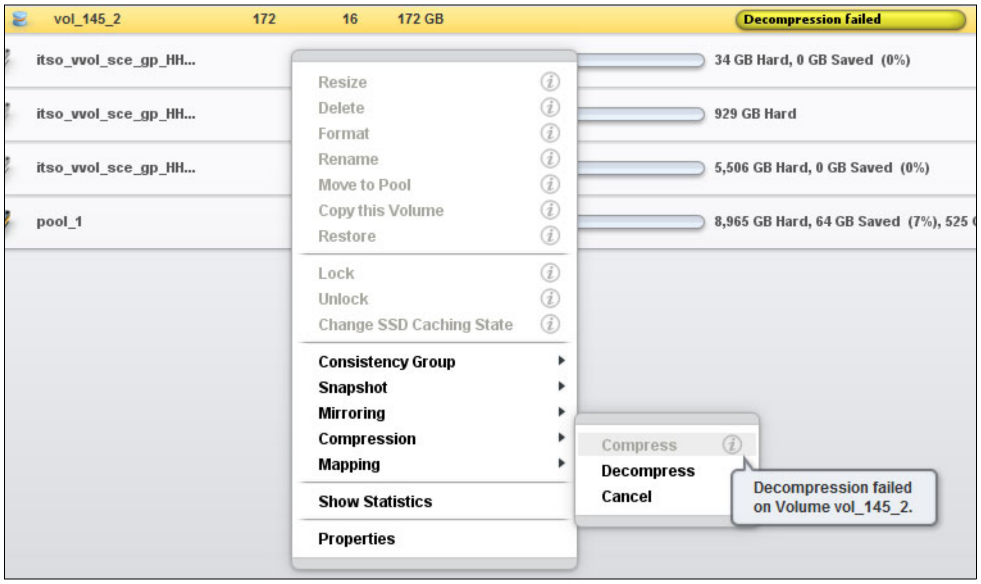


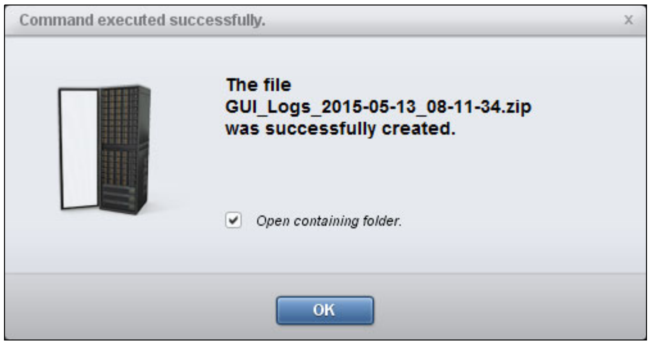
Figure 7-8 Decompression failed

If the problem is corrected, and the pool is resized, you can click **Decompress** to start the process again. Otherwise, the process must be canceled if the pool cannot be resized.

7.1.2 GUI log

More GUI logs are available for review if a problem must be investigated. To generate a compressed file that includes the logs, you must to be logged on to the GUI. Complete the following steps to collect GUI logs:

1. Press Alt+F1 to generate the `.zip` file and provide the option to browse to the folder that contains the file. See Figure 7-9.



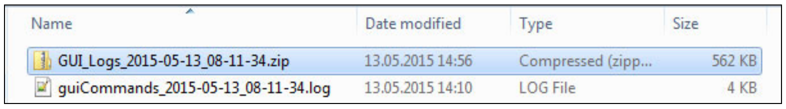
Command executed successfully.

The file GUI_Logs_2015-05-13_08-11-34.zip was successfully created.

☑ Open containing folder.

Figure 7-9 Generate GUI_Logs.zip

2. Click **OK** to open the folder on your local system that contains the generated file. This file contains the GUI `.log` files that can be analyzed or, most probably, sent to IBM Support. See Figure 7-10.



Name	Date modified	Type	Size
GUI_Logs_2015-05-13_08-11-34.zip	13.05.2015 14:56	Compressed (zipp...	562 KB
guiCommands_2015-05-13_08-11-34.log	13.05.2015 14:10	LOG File	4 KB

Figure 7-10 GUI_Logs

7.2 Involving IBM Support

This section describes the information that can be useful to IBM Support.

7.2.1 Information to provide

If IBM Support must be involved to help with a problem, make sure that the following information and data are provided:

- Meaningful problem description
- GUI Logs
- XIV Support Logs (xray)
- XIV_diag, when HAK is installed
- VMware support logs, if VMware is involved

Problem description

Describe the problem as clearly as possible:

- What is not working or is going wrong
- Which steps were done before the problem occurred or started
- Whether the problem is permanent and can be re-created
- General changes that were made before the issue started

GUI logs

See 7.1.2, “GUI log” on page 81 for information about how to create a `GUI_Logs.zip` file.

XIV support logs (xray)

Complete the following steps to collect and send XIV support logs:

1. In the GUI, click **Tools** and select **Collect Support Logs** (Figure 7-11).

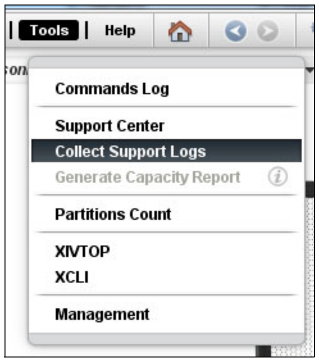
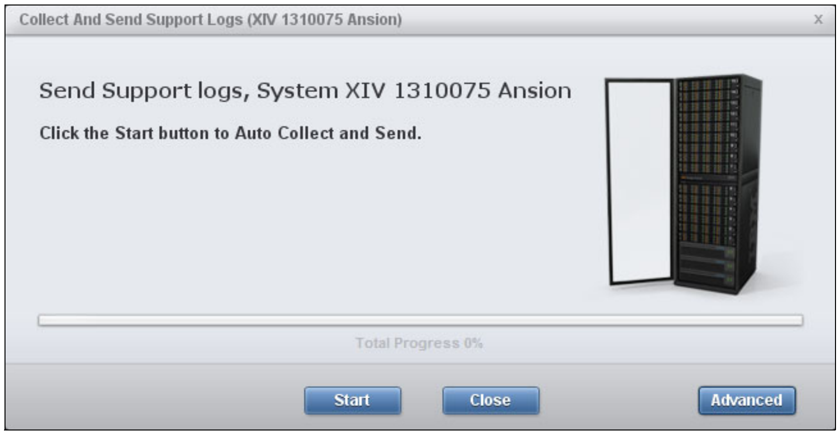


Figure 7-11 Collect Support Logs

2. This step opens the Collect Support Logs dialog box. See Figure 7-12.


Collect And Send Support Logs (XIV 1310075 Ansion)

Send Support logs, System XIV 1310075 Ansion

Click the Start button to Auto Collect and Send.

Total Progress 0%

Figure 7-12 Start support logs

XIV_diag (if HAK is installed)

Use the host diagnostics utility, which is part of the Host Attachment Kit, to generate an `xiv_diag` file.

VMware system logs when VMware is involved

Complete the following steps to collect VMware system logs:

1. In the vSphere Client, click **Administration** and select **Export System Logs**. See Figure 7-13.

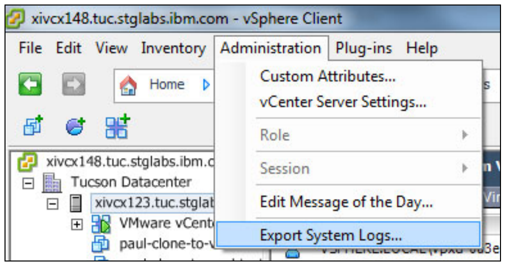
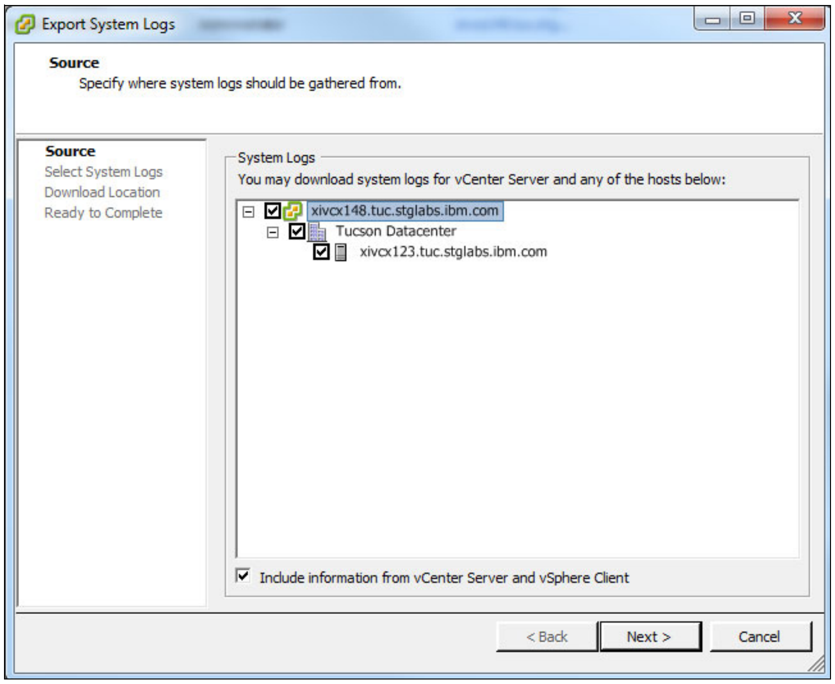


Figure 7-13 Export System Logs

This step starts the dialog to collect VMware logs.

2. Specify which logs to collect (Figure 7-14).



Export System Logs

Source
Specify where system logs should be gathered from.

System Logs
You may download system logs for vCenter Server and any of the hosts below:

☑ xivcx148.tuc.stglabs.ibm.com
☑ Tucson Datacenter
☑ xivcx123.tuc.stglabs.ibm.com

☑ Include information from vCenter Server and vSphere Client

Figure 7-14 Collect VMware logs

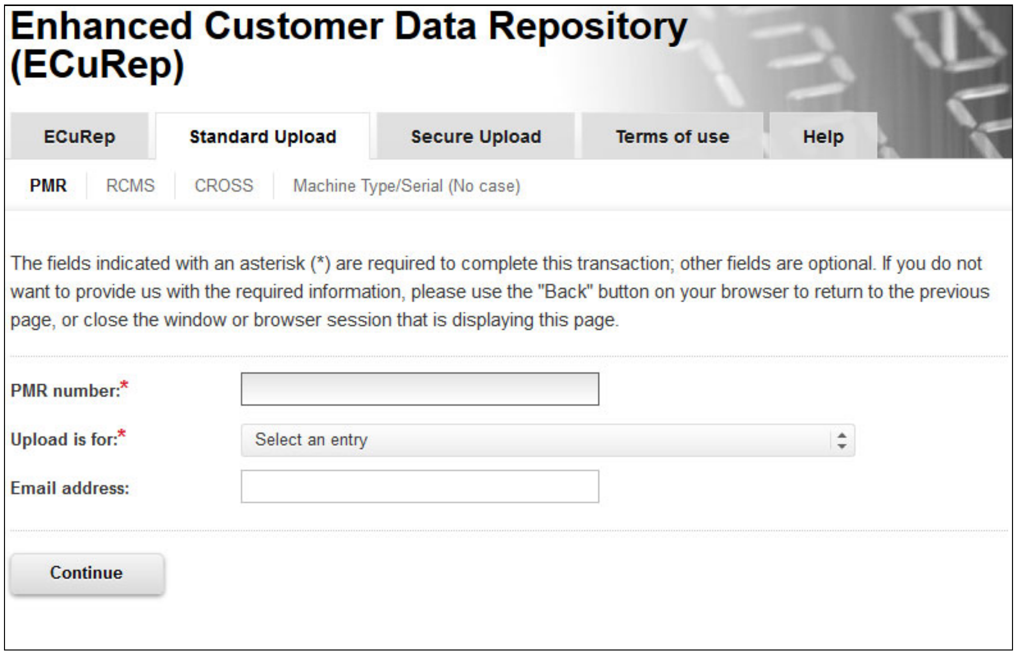
3. Click **Next** to continue.

7.2.2 Enhanced Customer Data Repository (ECuRep)

After all required data is collected, use ECuRep to send it to IBM Support:

http://www.ecurep.ibm.com/app/upload

Make sure to use either a PMR or R CMS number, so that the data can be related to the correct problem report without any needless delay. See Figure 7-15.



The fields indicated with an asterisk (*) are required to complete this transaction; other fields are optional. If you do not want to provide us with the required information, please use the "Back" button on your browser to return to the previous page, or close the window or browser session that is displaying this page.

Figure 7-15 ECuRep

Complete the requested information and click **Continue** to select the files that are ready for upload on your local server.

Related publications

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

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *IBM Real-time Compression on the IBM XIV Storage System*, REDP-5215
- *IBM XIV Storage System Architecture and Implementation*, SG24-7659

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, drafts, and additional materials, from the following website:

ibm.com/redbooks

Online resources

These websites are also relevant as further information sources:

- IBM XIV Storage System Documentation

  http://www.ibm.com/support/knowledgecenter/STJTAG/com.ibm.help.xivgen3.doc/xiv_kcwelcomepage.html
- *IBM XIV Real-time Compression Evaluation User Guide*

  http://public.dhe.ibm.com/common/ssi/ecm/ts/en/tsj03645usen/TSJ03645USEN.PDF

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

REDP-5216-00

ISBN 0738454699

Printed in U.S.A.

Get connected

Redbooks

ibm.com/redbooks