IBM System Storage DS8700 Disk Encryption

- Benefit from a robust and sophisticated key management
- Learn about recovery key and dual platform key servers
- Encrypt data at rest with no performance degradation

Bertrand Dufrasne
Antonio Rainero
Roland Wolf

ibm.com/redbooks

Redpaper

International Technical Support Organization

IBM System Storage DS8700 Disk Encryption

August 2010

REDP-4500-02

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

Third Edition (August 2010)

This edition applies to the IBM SystemStorage DS8700 with DS8000 Licensed Machine Code (LMC) level 6.5.1.xx (bundle version 75.1.xx.xx).

This document created or updated on February 4, 2011.

© Copyright International Business Machines Corporation 2010. 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 . . . vii
Trademarks . . . viii

Preface . . . ix
The team who wrote this paper . . . ix
Now you can become a published author, too! . . . x
Comments welcome . . . xi
Stay connected to IBM Redbooks . . . xi

Summary of changes . . . xiii
August 2010, Third Edition . . . xiii

Chapter 1. Encryption overview . . . 1
1.1 Business context . . . 2
1.1.1 Threats and security challenges . . . 2
1.1.2 Need for encryption . . . 3
1.2 Encryption concepts and terminology . . . 4
1.2.1 Symmetric key encryption . . . 4
1.2.2 Asymmetric key encryption . . . 5
1.2.3 Hybrid encryption . . . 9
1.3 Encryption challenges . . . 9
1.4 Tivoli Key Lifecycle Manager . . . 10
1.4.1 Tivoli Key Lifecycle Manager components . . . 11
1.4.2 Tivoli Key Lifecycle Manager resources . . . 12

Chapter 2. DS8000 encryption mechanism . . . 13
2.1 DS8700 disk encryption . . . 14
2.2 Encryption key management . . . 16
2.3 Encryption deadlock . . . 27
2.4 Working with a recovery key . . . 28
2.4.1 Recovery key management is enabled . . . 29
2.4.2 Disabling or enabling a recovery key . . . 33
2.5 Dual key server support . . . 34

Chapter 3. Planning and guidelines for DS8700 encryption . . . 37
3.1 Planning and implementation process flow . . . 38
3.2 Encryption-capable DS8700 ordering and configuration . . . 39
3.3 Requirements for encrypting storage . . . 40
3.4 Best practices for encrypting storage environments . . . 41
3.4.1 Security . . . 41
3.4.2 Availability . . . 41
3.4.3 Encryption deadlock prevention . . . 42
3.5 Dual HMC and redundancy . . . 44
3.5.1 Dual HMC advantages . . . 44
3.5.2 Redundant HMC configurations . . . 45
3.6 Multiple Tivoli Key Lifecycle Managers for redundancy . . . 47
3.6.1 Setting up Tivoli Key Lifecycle Manager servers . . . 47

Chapter 4. DS8700 encryption implementation . . . 49

© Copyright IBM Corp. 2010. All rights reserved.

4.1 Tivoli Key Lifecycle Manager configuration . . . 50
4.1.1 Log in to Tivoli Key Lifecycle Manager console . . . 50
4.1.2 Creating SSL certificate for secure communication . . . 52
4.1.3 Configuring the DS8700 Storage Facility Image . . . 58
4.2 DS8700 GUI configuration for encryption . . . 62
4.2.1 Configuring Tivoli Key Lifecycle Manager server connection to DS8700 . . . 62
4.2.2 Assigning storage and security administrators . . . 64
4.2.3 Managing recovery key . . . 67
4.2.4 Creating the encryption group . . . 73
4.2.5 Applying the encryption activation key . . . 74
4.2.6 Configuring and administering encrypted arrays . . . 77
4.2.7 Configuring and administering encrypted ranks . . . 79
4.2.8 Configuring and administering encrypted extent pools . . . 82
4.3 DS8700 DS CLI configuration for encryption . . . 85
4.3.1 Configuring the Tivoli Key Lifecycle Manager server connection . . . 85
4.3.2 Managing recovery key . . . 87
4.3.3 Configuring and administering the encryption group . . . 89
4.3.4 Applying encryption activation key . . . 89
4.3.5 Creating encrypted arrays . . . 90
4.3.6 Creating encrypted ranks . . . 91
4.3.7 Creating encrypted extent pools . . . 92
4.4 Encryption and Copy Services functions . . . 93

Chapter 5. Maintaining the DS8700 encryption environment . . . 95
5.1 Backup and restore . . . 96
5.1.1 Categories of data in a backup file . . . 96
5.1.2 Backup file security . . . 97
5.1.3 Tivoli Storage Manager as a backup repository . . . 97
5.1.4 Backup and restore runtime requirements . . . 97
5.1.5 Backing up critical files . . . 98
5.1.6 Restoring a backup file . . . 99
5.1.7 Deleting a backup file . . . 101
5.2 Starting and stopping Tivoli Key Lifecycle Manager server . . . 101
5.3 Key exporting and importing tasks . . . 105
5.3.1 Exporting keys . . . 106
5.3.2 Importing keys . . . 106
5.4 Tivoli Key Lifecycle Manager dual platform support implementation . . . 107
5.4.1 Creating the certificates . . . 108
5.4.2 Exchanging the public keys . . . 110
5.4.3 Adding DS8700 storage image . . . 116
5.5 Rekey data key . . . 118
5.6 Recovery key usage and maintenance . . . 119
5.6.1 Validate recovery key . . . 120
5.6.2 Initiate recovery key . . . 121
5.6.3 Re-key recovery key . . . 128
5.6.4 Delete recovery key . . . 132
5.6.5 Recovery key enabling . . . 135
5.6.6 Recovery key state summary . . . 137

Chapter 6. Integrating encryption devices . . . 139
6.1 Integrate with existing tape encryption installations . . . 140
6.1.1 Migrating from EKM to Tivoli Key Lifecycle Manager . . . 140
6.2 Multiple encrypted disk or tape devices . . . 141

Appendix A. Tivoli Key Lifecycle Manager Installation 143
Hardware requirements 144
Operating system requirement and software prerequisites 145
Java runtime environment requirements. 145
Database authority and requirements. 145
Installing Tivoli Key Lifecycle Manager on a Windows platform. 146
GUI mode installation 146

Related publications 153
IBM Redbooks 153
Other publications 153
Online resources 153
How to get Redbooks. 154
Help from IBM 154

Index 155

Notices

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

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

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

© Copyright IBM Corp. 2010. All rights reserved.

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®
DB2®
DS8000®
FlashCopy®
IBM®
RACF®
Redbooks®
Redpaper™
Redbooks (logo) ®
S/390®
System Storage®
System x®
System z®
Tivoli®
WebSphere®
XIV®
z/OS®

The following terms are trademarks of other companies:

SUSE, the Novell logo, and the N logo are registered trademarks of Novell, Inc. in the United States and other countries.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Intel Xeon, Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

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

Preface

IBM® recognizes the requirement for data protection, both from hardware or software failures, and also from physical relocation of hardware, theft, and re-tasking of existing hardware.

The IBM DS8000® Series offers encrypted Fibre Channel drives. These Full Disk Encryption drive sets are used with key management services provided by IBM Tivoli® Key Lifecycle Manager software to allow encryption for data at rest on a DS8700 Storage System. Use of encryption technology has a number of considerations that are critical for you to understand to maintain the security and accessibility of encrypted data.

This IBM Redpaper™ publication contains information that can help customers and storage administrators plan for disk encryption. It also explains how to install and manage the encrypted storage.

More important, this paper documents how to comply with IBM requirements for using the IBM DS8000 encrypted storage.

Important: Failure to follow these requirements can result in an *encryption deadlock*.

This edition applies specifically to the IBM System Storage DS8700 with DS8000 Licensed Machine Code (LMC) level 6.5.1.xx (bundle version 75.1.xx.xx).

The team who wrote this paper

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), San Jose Center.

Bertrand Dufrasne is an IBM Certified Consulting IT Specialist and Project Leader for IBM System Storage® disk products at the ITSO, San Jose Center. He has worked at IBM in various IT areas. Bertrand has written many IBM Redbooks® publications and has also developed and taught technical workshops. Before joining the ITSO, he worked for IBM Global Services as an Application Architect in the retail, banking, telecommunication, and healthcare industries. He holds a Master's degree in Electrical Engineering from the Polytechnic Faculty of Mons (Belgium).

Antonio Rainero is a Certified IT Specialist working for Integrated Technology Services organization at IBM in Italy. He joined IBM in 1998 and has more than 10 years of experience in the delivery of storage services both for z/OS® and Open Systems customers. His areas of expertise include storage subsystems implementation, performance analysis, storage area networks, storage virtualization, disaster recovery, and high availability solutions. Antonio holds a degree in Computer Science from University of Udine, Italy.

Roland Wolf is a Certified IT Specialist in Germany. He has worked for IBM for 24 years and has extensive experience with high-end disk-storage hardware in S/390® and Open Systems environments. He works in Field Technical Sales Support for storage systems. His areas of expertise include performance analysis and disaster recovery solutions in enterprises using the unique capabilities and features of the IBM disk storage servers, DS8000, and IBM XIV®. He has contributed to various IBM Redbooks publications including, ESS, DS80000 Architecture, and DS8000 Copy Services. He holds a Ph.D. in Theoretical Physics.

© Copyright IBM Corp. 2010. All rights reserved.

Special thanks to **Rick Ripberger** for his input and advice in preparation of this paper.

Thanks to the authors of the previous editions of this paper, Kerstin Blum, Uwe Dubberke, Marcus Gorzellik, Gabor Penzes.

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 two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

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

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

Summary of changes

This section describes the technical changes made in this edition of the paper and in previous editions. This edition may also include minor corrections and editorial changes that are not identified.

Summary of Changes
for IBM System Storage DS8700 Disk Encryption
as created or updated on February 4, 2011.

August 2010, Third Edition

This revision reflects the addition, deletion, or modification of new and changed information.

New information

The following new features are covered in this edition:

- Disable recovery key

  With security administrator access, encryption group recovery key is optionally disabled, or configured, before establishing initial logical configuration.

- Rekey data keys of an encryption group

  With security administrator access, users can rekey the data keys of their encryption group.

Important: This edition applies specifically to the IBM System Storage DS8700 with Licensed Machine Code (LMC) level 6.5.1.xx (bundle version 75.1.xx.xx).

© Copyright IBM Corp. 2010. All rights reserved.

1

Encryption overview

Strong security is not a luxury anymore in today's round-the-clock, global business environment. Ensuring the protection and security of an organization's information is the foundation of any successful business.

Encrypting data at rest is a key element when addressing these concerns. The IBM DS8700 offers a self-encrypting disk solution that uses IBM Full Disk Encryption (FDE) disks and flexible key manager software. The DS8700 solution not only secures data at rest but also offers a simple, cost effective solution for securely erasing (cryptographic erasure) any disk drive that is being retired or repurposed.

Encryption, however, must not be deployed without careful planning and a thorough understanding of encryption techniques and encryption management products.

Important: Improper handling or implementation can result in a *permanent encryption deadlock*, which is mostly equivalent to the permanent loss of all key-server managed encrypted data. Refer to 2.3, “Encryption deadlock” on page 27.

To gain access to data, even in a deadlock situation, the DS8700 offers a recovery key implementation. The recovery key can only be set as the *first activity* when setting up a DS8700.

Starting with LMC level 6.5.1.xx, the recovery key can be configured as *disabled* in those environments where the customer does not want to maintain a recovery key.

This chapter contains the following topics:

- Business context for encryption
- Encryption concepts and terminology:
  - Symmetric key encryption
  - Asymmetric key encryption
  - Digital certificate
  - Hybrid encryption
- Encryption infrastructure and management with the Tivoli Key Lifecycle Manager

© Copyright IBM Corp. 2010. All rights reserved.

1.1 Business context

Businesses today need tools to protect against the known threats, but also guard against as yet unknown threats. Effective threat and vulnerability management must be proactive rather than reactive, preventing problems rather than responding to them. To be efficient and effective, businesses must address prevention, detection, and compliance in an integrated way.

1.1.1 Threats and security challenges

Figure 1-1 illustrates how threats and challenges add to the complexity, hence cost of running your business.

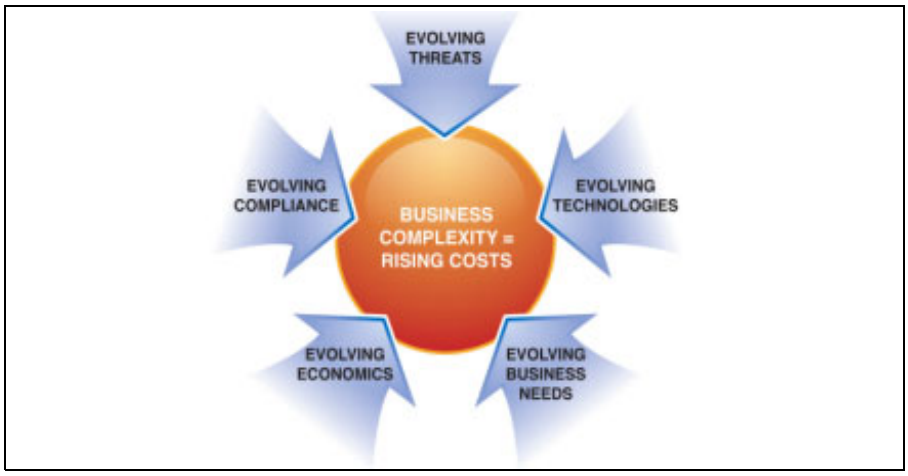


Figure 1-1 Business complexity

Companies face certain threats and security challenges:

- Increasing number and sophistication of threats. Businesses face more than just viruses and worms. You have to be able to defend against all threats rather than just respond to intrusions.
- Preventing data breaches and inappropriate data disclosure, while ensuring no impact on business and productivity.
- Intrusions that affect the bottom line in both customer confidence and business productivity. Security breaches can destroy your brand image and affect your critical business processes.
- Growing demand for regulatory compliance and reporting. You must be able to meet a growing number of compliance initiatives without diverting resources from core activities.
- Protecting your data and maintaining appropriate levels of access.
- Security issues are both internal *and* external. How do you protect against the well-intentioned employee who mishandles information, and the malicious outsider?
- Having your business comply with a growing number of corporate standards and government regulations; you must have tools that can document the status of your application security.
- Growing number of regulatory mandates. You have to prove that your physical assets are secure.

1.1.2 Need for encryption

In particular, organizations experience a continued push to minimize the risks of data breaches. There is a new focus on privacy management tools with the capability to mask data. This focus reinforces the need for cryptography, and subsequent demand to simplify the complexity of the key-based algorithms and management of keys throughout the life cycle.

A big concern is often when disk drives leave the company premises, which usually happens when a disk drive fails and the IBM technician replaces it with a new drive. Often, the drive is not really damaged and data can still be accessed. Of course, IBM has a procedure to delete all data on the drive. However, this task is no longer under the control of the customer. Some customers buy the drives back and destroy them themselves. This can be quite expensive. Another concern is when the whole DS8000 is going to be returned to IBM. The IBM technician will erase all data, but this is not sufficient for some customers. IBM offers a service (IBM Certified Secure Data Overwrite) to erase all data (several passes) in compliance with the American Department of Defense regulations (DoD 5220.20-M).

All these concerns become obsolete when data on the drives is encrypted. Without a decryption key the data is unreadable.

What should you encrypt, and just as important, what should you not encrypt? Simply encrypt everything that you can encrypt and still be able to recover data in event of a disaster. As long as system data can be separated from application data, encrypting everything with no performance impact is easier than choosing which data falls into which legislation for encryption, and trying to keep current on the dynamic privacy rights rules and regulations.

Before using any encryption technology, understanding the encryption concepts and the requirements to maintain the security and the accessibility of the encrypted data is absolutely important.

Indeed, you do not want the encryption solution to negatively affect your storage environment and the applications that depend on it. You want an encryption solution that will not degrade application performance or jeopardize your disaster recovery plan. You also need the assurance that encryption will not cause any data loss and that all the appropriate measures have been taken to protect and safeguard the encryption keys.

To address these concerns, the DS8700 encryption solution approach use disks that have encryption hardware, and can perform symmetric encryption and decryption of data at full disk speed with no impact on performance. The disk-based encryption is combined with an enterprise-scale key management infrastructure. That infrastructure is based on the IBM Tivoli Key Lifecycle Manager and life cycle management tools to help organizations efficiently deploy, back up, restore, and delete keys and certificates in a secure and consistent fashion. The DS8000 solution is further described in Chapter 2, “DS8000 encryption mechanism” on page 13.

Important: The DS8700 provides disk-based encryption, for data at rest on disk. If encryption over the network is required, additional encryption services have to be investigated and deployed as appropriate.

For a successful deployment, following the instructions and guidelines outlined in this document is also imperative.

For more information about IBM security solutions in general, refer to the IBM security site:

http://www.ibm.com/security/index.html

1.2 Encryption concepts and terminology

Encryption transforms data that is unprotected, or *plain text*, into encrypted data, or *ciphertext,* using a *key*. Without knowledge of the encryption key, the ciphertext cannot be converted back to plain text.

Computer technology has enabled increasingly sophisticated encryption algorithms. Working with the U.S. Government National Institute of Standards and Technology (NIST), IBM invented one of the first computer-based algorithms, Data Encryption Standard (DES), in 1974. Today, several widely used encryption algorithms exist, including triple DES (TDES) and Advanced Encryption Standard (AES).

1.2.1 Symmetric key encryption

Early encryption methods used the same key to encrypt plain text to generate ciphertext, and to decrypt the ciphertext to regenerate the plain text. Because the same key is used for both encryption and decryption, this method is called *symmetric encryption*. All of the encryption algorithms previously mentioned use symmetric encryption.

Everyone who gets knowledge of the key can transform the ciphertext back to plain text. If you want to preserve confidentiality, you must protect your key and keep it a secret. Therefore, symmetric encryption is also called *private* or *secret key encryption*, which is not to be confused with the private key in an asymmetric key system.

In Figure 1-2, we show a sample encryption and decryption data flow path. Here, we use the symmetric key AES_256_ITSO to encrypt plain text using the AES encryption algorithm, which yields encrypted data. The decryption of the enciphered text uses the same AES_256_ITSO symmetric key and the AES algorithm to decrypt the data back to its plain text format.

Symmetric key encryption algorithms are significantly faster than asymmetric encryption algorithms, which makes symmetric encryption an ideal candidate for encrypting large amounts of data.

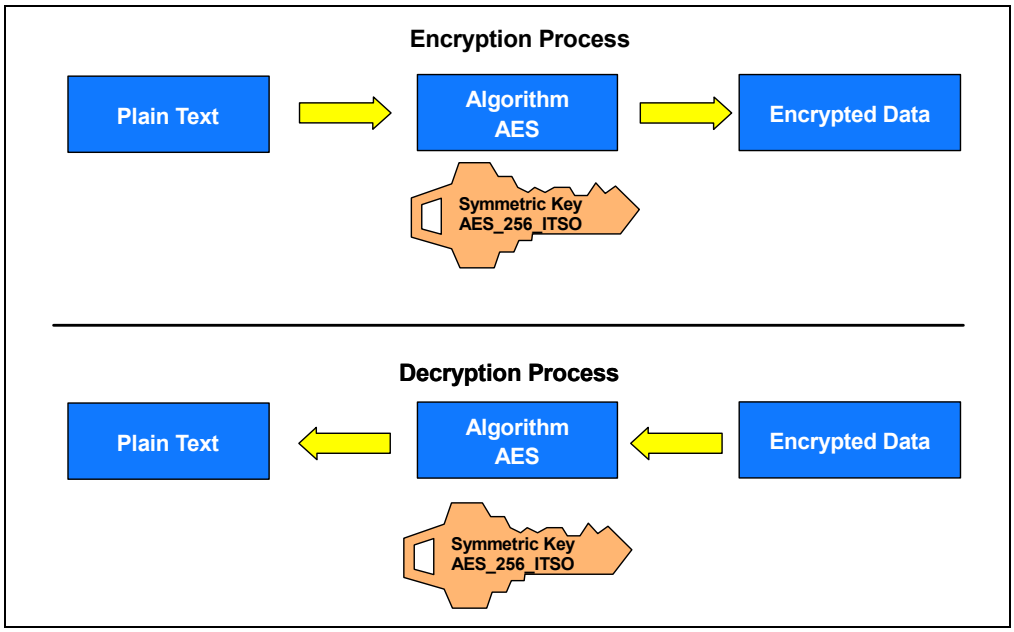


Figure 1-2 Symmetric key encryption

1.2.2 Asymmetric key encryption

It was only in the 1970s that cryptographers invented asymmetric key algorithms for encryption and decryption. Encryption methods using separate keys for encryption and decryption are called *asymmetric encryption.* Asymmetric encryption addresses certain drawbacks of symmetric encryption, which became more important with computer-based cryptography.

Asymmetric key encryption uses one key for encrypting (*public key*) and one key (*private key*) for decrypting data. Because the key used for encrypting a message cannot be used for decrypting, this key does not have to be kept a secret. It can be widely shared and is therefore called a *public key.*

Anyone who wants to send secure data to an organization can use its public key. The receiving organization then uses its *private key* to decrypt the data. The private key must always be kept a secret. Because asymmetric encryption uses public/private key pairs, it is also called *public/private key encryption* or *public key encryption.*

Public/private key encryption is widely used on the Internet today to secure transactions, including Secure Sockets Layer (SSL).

To encrypt data requires an algorithm. Today, the *RSA* algorithm is the most widely used public key technique. It is named after the surnames of the three developers, Ronald L. Rivest, Adi Shamir, and Leonard Adleman. who developed this algorithm in 1977.

The advantage of asymmetric key encryption is the ability to share secret data without sharing the same encryption key. But disadvantages exist too. Asymmetric key encryption is computationally more intensive and therefore significantly slower than symmetric key encryption.

In practice, you will often use a combination of symmetric and asymmetric encryption. We describe this method in 1.2.3, “Hybrid encryption” on page 9. With the DS8000, the IBM Encryption solution uses a combination of symmetric and asymmetric encryption methods.This combination of symmetric and asymmetric encryption algorithms (*hybrid encryption*) is prevalent in many security solutions.

Important: The IBM Full Disk Encryption (FDE) solution utilizes the asymmetric RSA algorithm only to encrypt symmetric AES keys used for data encryption.

Figure 1-3 shows an encryption and decryption data path when using public key encryption algorithms.

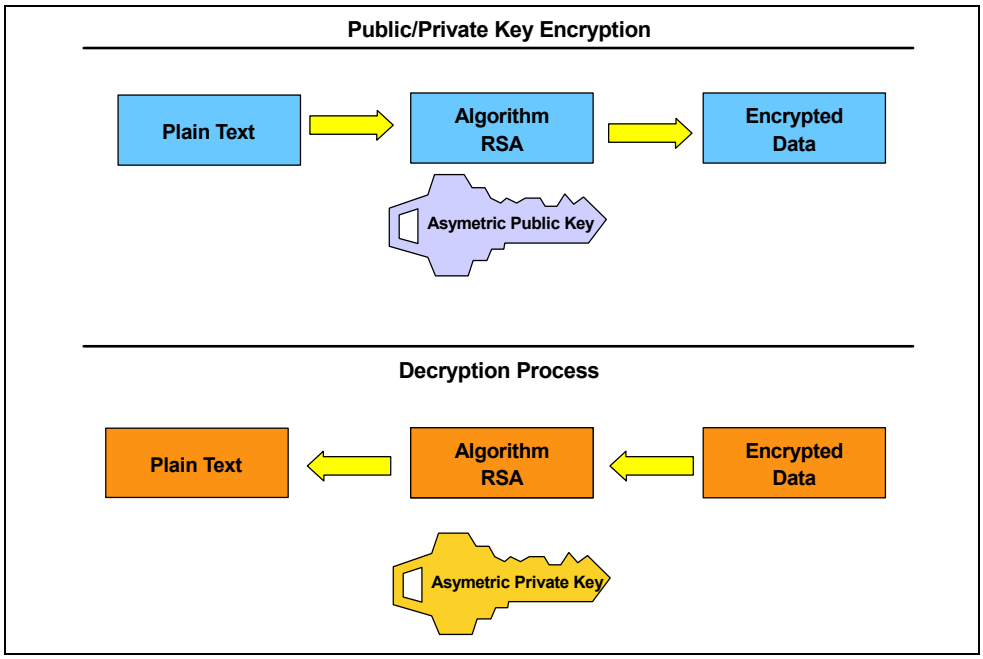


Figure 1-3 Public/private key encryption

Digital signature

You can use public/private key pairs to protect the content of a message, and also to digitally sign a message. When a digitally signed message is sent, the receiver can be sure that the sender has sent it because the receiver can prove it by using the public key from the sender. In practice, predominantly for efficiency reasons, a hash value of the message is signed rather than the whole message, but the overall procedure is the same.

In Figure 1-4 on page 7, we show how the digital signature is used in the communication between the DS8000 and Tivoli Key Lifecycle Manager, using an asymmetric key pair. It illustrates a mechanism used as part of the DS8000 encryption process. The DS8000 has a private key, and the Tivoli Key Lifecycle Manager has a copy of DS8000 public key. The DS8000 sends the Tivoli Key Lifecycle Manager a message that is encrypted with DS8000's private key. The Tivoli Key Lifecycle Manager then uses the public key to validate the message sent from the DS8000. The Tivoli Key Lifecycle Manager cannot use the public key for decryption of encrypted data, but the Tivoli Key Lifecycle Manager is able, with the DS8000 public key, to validate that the message was encrypted with the DS8000 private key. This approach proves to the Tivoli Key Lifecycle Manager that it is in fact communicating with the DS8000, because only the DS8000 has a copy of its private key. Then, the Tivoli Key Lifecycle Manager uses the DS8000 public key to encrypt the data that it wants to protect and sends the data to the DS8000. The DS8000 can use his private key to decrypt the data.

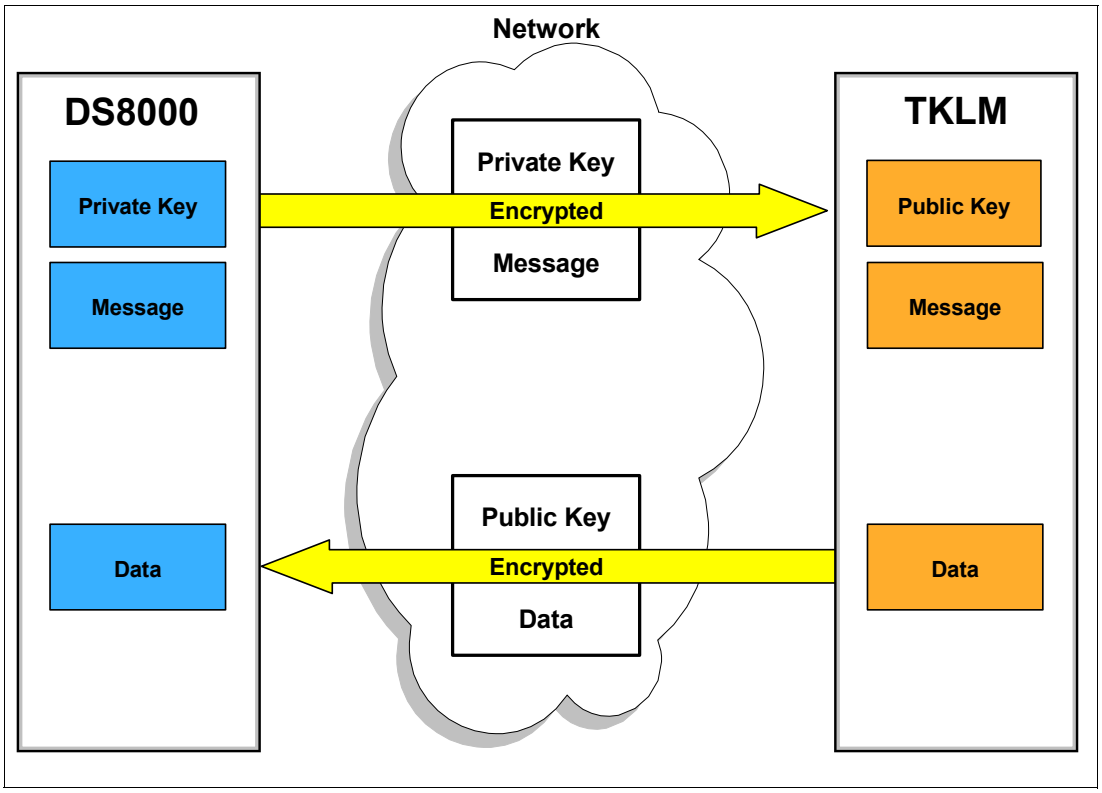


Figure 1-4 Identity verification using public/private key encryption

Figure 1-5 on page 8 shows a more detailed flow of actions. A hash of the message is created, encrypted with the sender's private key and attached to the message. Both the message and the digital certificate (the encrypted hash) are encrypted with the receiver's public key and transmitted to the receiver.

The receiver uses the receiver's private key to decrypt the message and digital certificate. Note, that the digital certificate is the hash that is encrypted with the *sender's private key*. The receiver cannot decrypt this, but the receiver can also produce the hash and encrypt it, this time, however with the *sender's public key*. If both match, the receiver can be sure of the identity of the sender.

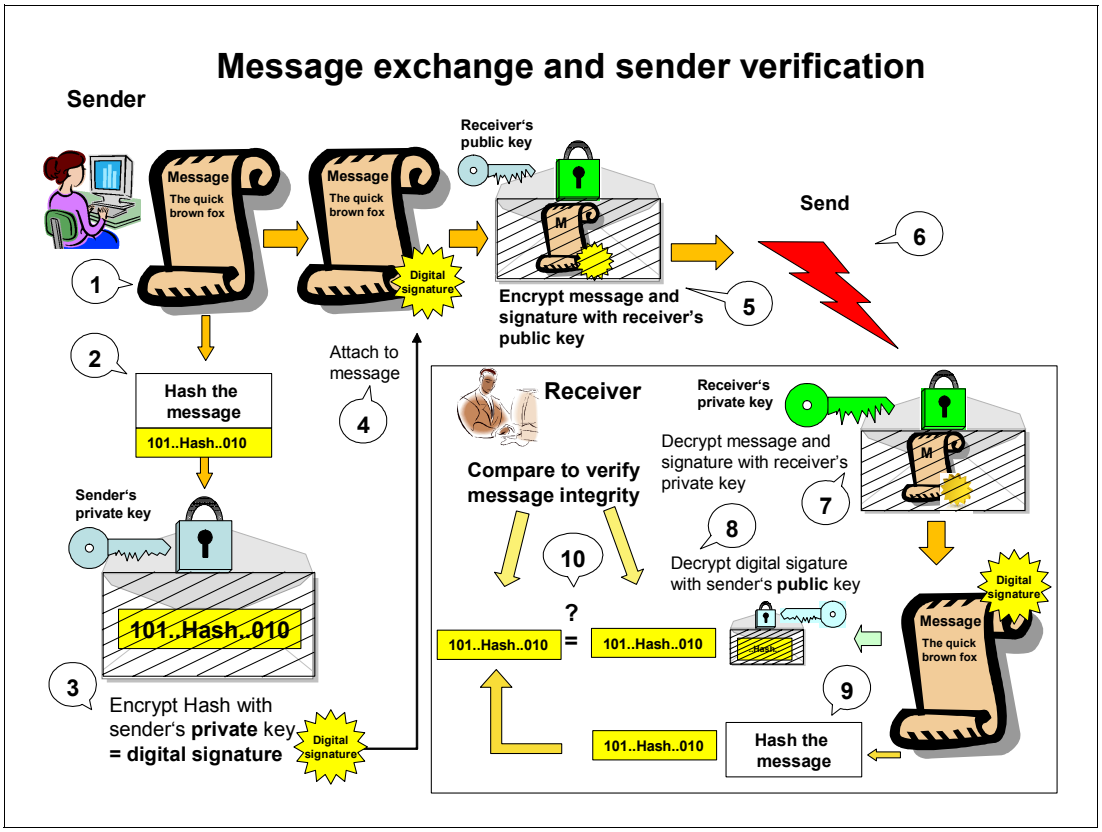


Figure 1-5 Verification of identity of sender

Digital certificates

Another possibility is to make sure that the sender can trust the receiver by using a *certificate*, which is signed by a *certificate authority* (CA).

Digital certificates are a way to bind public key information with an identity. The certificates are signed by a CA. If users trust the CA and can verify the CA's signature, then they can also verify that a certain public key does indeed belong to whomever (person or entity) is identified in the certificate.

Part of the information that is stored in a digital certificate includes the following items:

- Name of the issuer
- Subject Distinguished Name (DN)
- Public key belonging to the owner
- Validity date for the public key
- Serial number of the digital certificate
- Digital signature of the issuer

Note: For the DS8700, digital certificates are created and set by manufacturing for each Storage Facility Image.

Both asymmetric and symmetric key encryption schemes are powerful ways to protect and secure data. In 2.2, “Encryption key management” on page 16, we discuss in details their use with the IBM System Storage DS8000 Series family that provides an extremely secure way of protecting data.

1.2.3 Hybrid encryption

In practice, encryption methods often combine symmetric and asymmetric encryption. Thus, they can take advantage of fast encryption with symmetric encryption and still securely exchange keys using asymmetric encryption.

Hybrid methods use a symmetric data key to actually encrypt and decrypt data. They do not transfer this symmetric data key in the clear, but use public/private key encryption to encrypt the data key. The recipient is able to decrypt the encrypted data key and use the data key to encrypt or decrypt a message.

Hybrid encryption methods allow you to combine secure and convenient key exchange with fast and efficient encryption of large amounts of data.

The IBM Full Disk Encryption (FDE) solution uses a symmetric AES data key to encrypt and decrypt data. This data key is protected by the asymmetric RSA algorithm and is not available in plaintext when storage device and the Tivoli Key Lifecycle Manager communicate. For details, refer to 1.4, “Tivoli Key Lifecycle Manager” on page 10

1.3 Encryption challenges

Encryption, as we have seen, is dependent upon encryption keys. Those keys have to be, at the same time, kept secure and available, and responsibilities have to be split:

- Keys security

  To preserve the security of encryption keys, the implementation must be such that no one individual (system or person) has access to all the information required to determine the encryption key. In a system-based solution, the encryption data keys are encrypted with a wrapping key (that is another key to encrypt/decrypt the data keys). This wrapped key method is used with the DS8000 by separating the storage of a wrapped data key stored on the disk from the storage of the wrap/unwrap keys within a key server.

- Key availability

  More than one individual (person or system) has access to any single piece of information necessary to determine the encryption key. In a system-based solution, redundancy is provided by having multiple isolated key servers. Additionally, backups of key server’s data are maintained.

- Separation of responsibilities

  The DS8700 offers a recovery key to get access to data if none of the key servers are available. To prevent one person from gaining access to the data, the handling of a recovery key requires two people (separate roles): a security administrator and a storage administrator. Starting with DS8000 Licensed Machine Code (LMC) level 6.5.1.xx, you also have the possibility to disable the recovery key.

The sensitivity of possessing and maintaining encryption keys and the complexity of managing the number of encryption keys in a typical environment results in a customer requirement for a key server. A key server is integrated with encrypting storage products to resolve most of the security and usability issues associated with key management for encrypted storage.

Note: IBM offers an enterprise-scale key management infrastructure through IBM Tivoli Key Lifecycle Manager and life cycle management tools to help organizations efficiently deploy, back up, restore and delete keys and certificates in a secure and consistent fashion.

However, the customer must still be sufficiently aware of how these products interact to be able to provide appropriate management of the IT environment. Even with a key server, generally at least one encryption key (the overall key that manages access to all other encryption keys, or a key that encrypts the data used by the key server) or a recovery key must be maintained manually.

On critical consideration with a key server implementation is that all code and data objects required to make the key server operational must not be stored on storage that is dependent on any key server to be accessed.

A situation where all key servers cannot become operational because there is data or code that cannot be accessed without an operational key server is referred to as an *encryption deadlock*. It is analogous to having a bank vault that is unlocked with a combination and the only copy of the combination is locked inside the vault.

This situation, and the policies and mechanisms required to avoid it, are discussed more fully in Chapter 2, “DS8000 encryption mechanism” on page 13.

1.4 Tivoli Key Lifecycle Manager

The IBM approach to key management revolves around IBM Tivoli Key Lifecycle Manager that is enhanced in phases. From an initial focus on key management for tape and disk encryption, IBM is expanding Tivoli Key Lifecycle Manager into a centralized key management facility for managing encryption across a range of deployments.

In your enterprise, a large number of symmetric keys, asymmetric keys, and certificates can exist. All of these keys and certificates have to be managed and can be handled by Tivoli Key Lifecycle Manager.

The Tivoli Key Lifecycle Manager application performs key management tasks for IBM encryption-enabled hardware such as the IBM System Storage DS8000 Series family and IBM encryption-enabled tape drives (TS1130 and TS1040). Tivoli Key Lifecycle Manager provides, protects, stores, and maintains encryption keys that are used to encrypt information being written to, and decrypt information being read from, an encryption-enabled disk. Tivoli Key Lifecycle Manager operates on a variety of operating systems. Currently, the supported operating systems are as follows:

- AIX® 5.3 and AIX 6.1 (64 bit)
- Red Hat AS 4.0 x86 (32 bit)
- SUSE Linux® Enterprise Server (SLES) 9.0 and 10 x86 (32 bit)
- Solaris 10 Sparc (64 bit)
- Windows® Server 2003 (32 bit)
- IBM z/OS V1.9 and V1.10 (Tivoli Key Lifecycle Manager hosted in the System Service Runtime Environment for z/OS)

Tivoli Key Lifecycle Manager is designed to be a shared resource, deployed in several locations within an enterprise. It is capable of serving numerous IBM encrypting-enabled types of hardware regardless of where those devices reside.

Note: For the DS8700, an isolated primary Tivoli Key Lifecycle Manager key server is required and must be deployed on an IBM System x® running SLES 9.0 with storage that is *not* provisioned on the DS8700. Additionally, secondary key servers can be deployed on any of the previously mentioned platforms.

1.4.1 Tivoli Key Lifecycle Manager components

With the DS8700, Tivoli Key Lifecycle Manager is used to handle serving keys to the encrypting disk drives. In addition to the key-serving function, the Tivoli Key Lifecycle Manager offers the following functions, which can also be used for IBM encryption-enabled tape drives:

- Lifecycle functions
  - Notification of certificate expiration through the Tivoli Integrated Portal
  - Automated rotation of certificates
  - Automated rotation of groups of keys
- Usability features
  - Provides a graphical user interface (GUI)
  - Initial configuration wizards
  - Migration wizards
- Integrated backup and restore of Tivoli Key Lifecycle Manager files
  - One button to create and restore a single backup packaged as a `.jar` file

To perform these tasks, Tivoli Key Lifecycle Manager relies on external components. The Tivoli Key Lifecycle Manager solution includes the Tivoli Key Lifecycle Manager server, an IBM embedded WebSphere® Application Server, and a database server (IBM DB2®).

The solution also incorporates the *Tivoli Integrated Portal* installation manager which provides simple to use installation for Windows, Linux, AIX, and Solaris.

In Tivoli Key Lifecycle Manager, the Drive Table, LTO Key Group, and metadata are all kept in DB2 tables. The Tivoli Key Lifecycle Manager DB2 tables enable the user to search and query that information much easier. Note that the keystore, configuration file, audit log, and debug log are still flat files.

1.4.2 Tivoli Key Lifecycle Manager resources

Tivoli Key Lifecycle Manager also relies on the following resources:

- Configuration file

  Tivoli Key Lifecycle Manager has a an editable configuration file with additional configuration parameters that is not offered in the GUI. The file can be text-edited, however the preferred method is modifying the file through the Tivoli Key Lifecycle Manager command-line interface (CLI).

  We discuss installation and configuration in Appendix A, “Tivoli Key Lifecycle Manager Installation” on page 143, and also describe a set of configuration options.

- Java™ security keystore

  The keystore is defined as part of the Java Cryptography Extension (JCE) and an element of the Java Security components, which are, in turn, part of the Java Runtime Environment. A *keystore* holds the certificates and keys (or pointers to the certificates and keys) used by Tivoli Key Lifecycle Manager to perform cryptographic operations. A keystore can be either hardware-based or software-based. Tivoli Key Lifecycle Manager supports several types of Java keystores, offering a variety of operational characteristics to meet your needs.

  Tivoli Key Lifecycle Manager on open systems supports the JCEKS keystore. This keystore supports both CLEAR key symmetric keys, and CLEAR key asymmetric keys. Symmetric keys are used for LTO 4 encryption drives, and asymmetric keys are used for DS8000 and TS1100 tape drives.

- Cryptographic Services

  Tivoli Key Lifecycle Manager uses the IBM Java Security components for its cryptographic capabilities. Tivoli Key Lifecycle Manager does not provide cryptographic capabilities and therefore does not require, nor is allowed to obtain, FIPS 140-2 certification. However, Tivoli Key Lifecycle Manager takes advantage of the cryptographic capabilities of the IBM Java Virtual Machine in the IBM Java Cryptographic Extension component and allows the selection and use of the IBMJCEFIPS cryptographic provider, which has a FIPS 140-2 level 1 certification.

  By setting the FIPS configuration parameter to `ON` in the Configuration Properties file either through text editing or by using the Tivoli Key Lifecycle Manager CLI, you can make Tivoli Key Lifecycle Manager use the IBMJCEFIPS provider for all cryptographic functions.

  > **Important:** Tivoli Key Lifecycle Manager takes advantage of the cryptographic capabilities of the IBM Java virtual machine in the IBM Java Cryptographic Extension component and allows the selection and use of the IBMJCEFIPS cryptographic provider

  You can find more information about the IBMJCEFIPS provider, its selection, and its use at the following website:

  http://www.ibm.com/developerworks/java/jdk/security/50/FIPShowto.html

2

DS8000 encryption mechanism

This chapter provides information about the DS8000 disk encryption mechanisms.

This chapter contains the following topics:

- DS8000 disk encryption
- Encryption key management
- Encryption deadlock
- Recovery key
- Dual key server support

© Copyright IBM Corp. 2010. All rights reserved.

2.1 DS8700 disk encryption

The DS8700 disk subsystem supports data encryption with the IBM Full Disk Encryption (FDE) drives. These drives are available in 300 GB, 450 GB, and 600 GB capacity, with a rotational speed of 15,000 RPM. All disks in the DS8700 must be FDE drives, no intermix is allowed.

These disks have encryption hardware, and can perform symmetric encryption and decryption of data at full disk speed with no impact on performance.

The disk encryption hardware is used in conjunction with Tivoli Key Lifecycle Manager. Tivoli Key Lifecycle Manager and the DS8700 use asymmetric encryption to encrypt and decrypt the data key. When connected to the DS8700, Tivoli Key Lifecycle Manager generates encryption and decryption keys that are used to lock each FDE drive.

Without these keys managed by Tivoli Key Lifecycle Manager, the customer can no longer decrypt the data on disk.

Note: If all copies of the decryption key are lost (whether intentionally or accidentally), then no feasible way exists to decrypt the associated ciphertext, and the data contained in the ciphertext is said to have been *cryptographically erased*. The data is lost, because it cannot be decrypted without the key.

For more details about the encryption key management, see 2.2, “Encryption key management” on page 16.

To be able to use data encryption, the DS8700 must be ordered from manufacturing with FDE drives (replacing regular FC drives with FDE drives in a an existing DS8700 is not supported). Details about the ordering process are given in 4.1, “Tivoli Key Lifecycle Manager configuration” on page 50.

Currently the DS8700 does not support intermix of FDE and non-FDE drives, so any additional disks must be consistent with the drives that are already installed. A DS8700 with FDE drives is referred to as being *encryption-capable*. An encryption-capable DS8700 can be configured to either enable or disable encryption for all data that is stored on customer disks.

Attention: Enabling encryption cryptographically erases all data on the drives. Therefore, encryption must be enabled directly at the beginning, not when data is already stored in the DS8700.

The DS8700 must be configured to communicate with *at least two* Tivoli Key Lifecycle Manager key servers to enable encryption. Two Tivoli Key Lifecycle Manager key servers are required for redundancy. After the DS8700 powers on, it must be able to communicate to at least one of the Tivoli Key Lifecycle Manager servers to get the unlock keys. The communication between the DS8700 and the Tivoli Key Lifecycle Manager key server is done through the Hardware Management Console (HMC). Therefore, having two HMCs to also provide redundancy on the storage-device side is important. For details, refer to 3.5, “Dual HMC and redundancy” on page 44.

The physical connection between the DS8700 HMC and the key server is through a TCP/IP network, as depicted in Figure 2-1 on page 15.

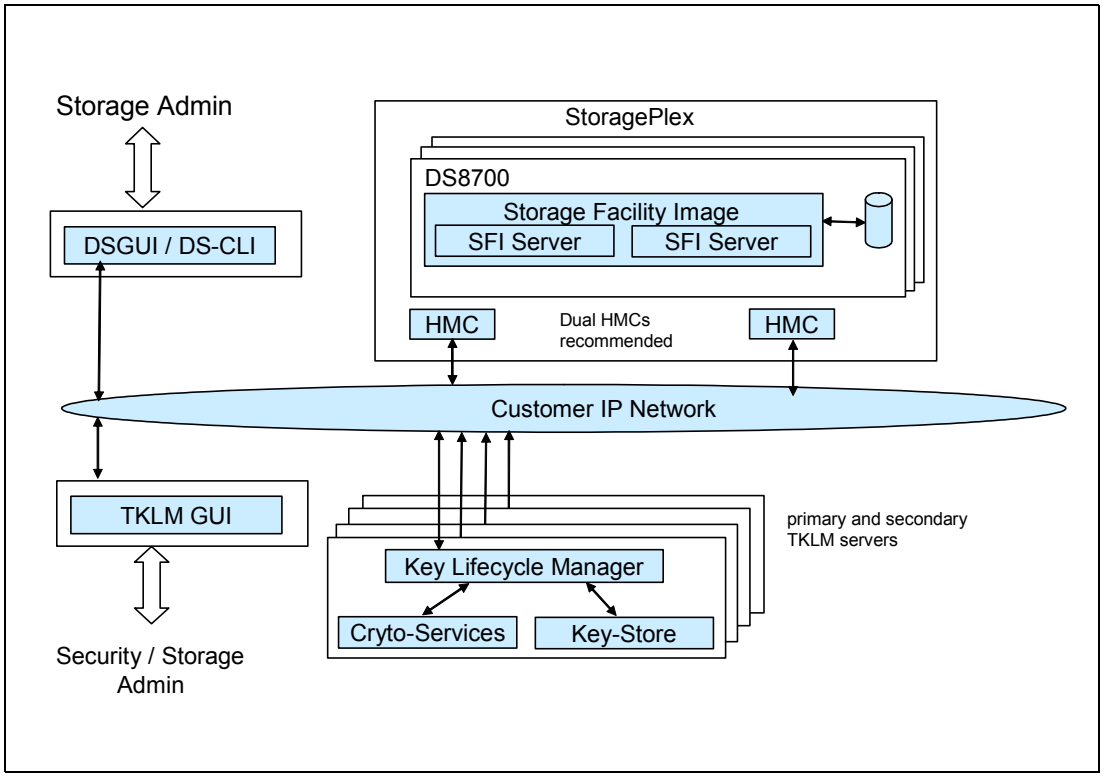


Figure 2-1 Connection between DS8000 HMC and Tivoli Key Lifecycle Manager

Before we explain the various keys used by DS8700 and Tivoli Key Lifecycle Manager for encryption, we discuss how messages can be exchanged between two systems in a secure way. We discuss the concept of digital signatures.

Digital signatures are used to authenticate a sender. The digital signatures are generated using the private and public keys. Figure 2-2 on page 16 explains the following steps:

1. The Sender writes its message.
2. According to a mathematical formula, a digital string, usually of a fixed length, is derived from the message. This is called a *hash*. Although a hash is derived from and uniquely linked to the data, deriving the data from the hash is not possible.
3. The hash is encrypted with the *sender's private key.* The encrypted hash is called a *digital signature*.
4. The digital signature is attached to the message.
5. Both message and digital signature are encrypted with the receiver's public key.
6. The encrypted message is sent to the receiver.
7. The receiver decrypts the message and signature combination.

   Now the sender reproduces the message hash in two ways:

   – The receiver decrypts the digital signature with the sender's public key to get the original hash.

   – The receiver calculates the hash from the received message.
8. If both hashes match, the receiver has good reason to trust the message.

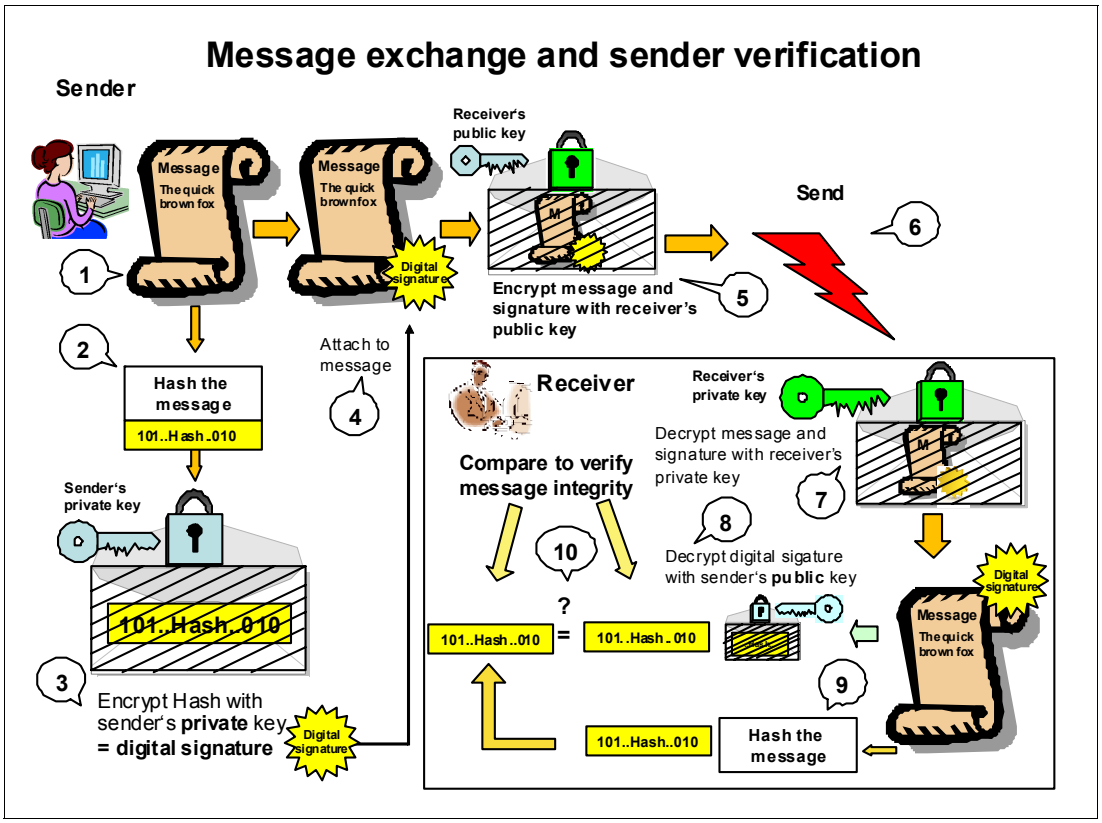


Figure 2-2 Authentication with digital signatures

2.2 Encryption key management

In this section, we provide details about how the Tivoli Key Lifecycle Manager key server manages and creates the encryption keys used by the DS8700 during key label, encryption group, rank creation, and at DS8700 power-on time.

> **Important:** Key negotiation and authentication between the Tivoli Key Lifecycle Manager and DS8700 takes place at DS8700 power-on time only. In other words, there is no traffic overhead in an encrypted DS8700 at run time that is created by key negotiation.

Tivoli Key Lifecycle Manager uses the wrapped key method to serve keys to encryption-enabled DS8700. The wrap and unwrap keys on Tivoli Key Lifecycle Manager are a public/private asymmetric key pair referred to as the public key encrypting key (KEK) and the private key encrypting key (KEK'), respectively.

The configuration processes on Tivoli Key Lifecycle Manager and the storage device (DS8700) define one or more key labels. Refer to 4.1, "Tivoli Key Lifecycle Manager configuration" on page 50.

The key label is a user-specified text string that is associated with the asymmetric key label pair (KEK/KEK'), generated by Tivoli Key Lifecycle Manager (TKLM) when the key label is configured (see Figure 2-3 on page 17). The key generation and propagation processes on the Tivoli Key Lifecycle Manager associates a key label with each wrap/unwrap key pair. This key label is a user-specified text string that is retained with each wrap/unwrap key pair. The key encrypting key-pair key is kept secret by Tivoli Key Lifecycle Manager in a keystore.

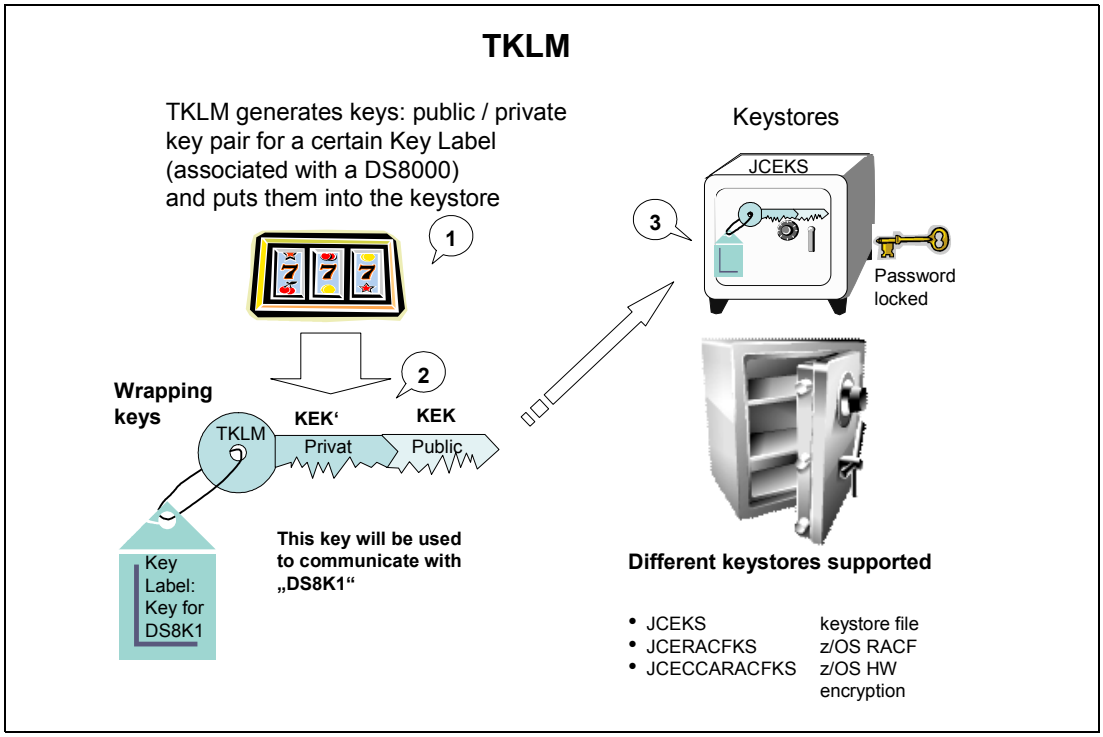


Figure 2-3 Configure Tivoli Key Lifecycle Manager key label

Note: The Licensed Machine Code (LMC) level 6.5.1.xx (bundle version 75.1.xx.xx) enables the rekey data key feature. This feature allows a user to change the data key labels (see 5.5, "Rekey data key" on page 118).

Now the user (storage administrator) can use the DS8000 GUI to register the key server on the DS8700. Next, still using the DS8000 GUI, an encryption group is created. For details, refer to 4.2.1, "Configuring Tivoli Key Lifecycle Manager server connection to DS8700" on page 62).

As part of creating the encryption group, you must specify the key label that was set when configuring the Tivoli Key Lifecycle Manager server, which was configured for a certain DS8000.

Note: Currently, the DS8700 has only one encryption group.

While creating the encryption group, the DS8700, which we are referring to as DS8K1, generates a device session key pair (device session public key/device session private key, respectively noted as DSK/DSK') from a random number. The public/private key pair is associated with a key label. The device session private key (DSK') is kept secret by the DS8700.

The key label, device session public key (DSK), and the DS8700 storage facility certificate (which was set and stored on the DS8700 by manufacturing) are sent to Tivoli Key Lifecycle Manager to request a data key (see Figure 2-4).

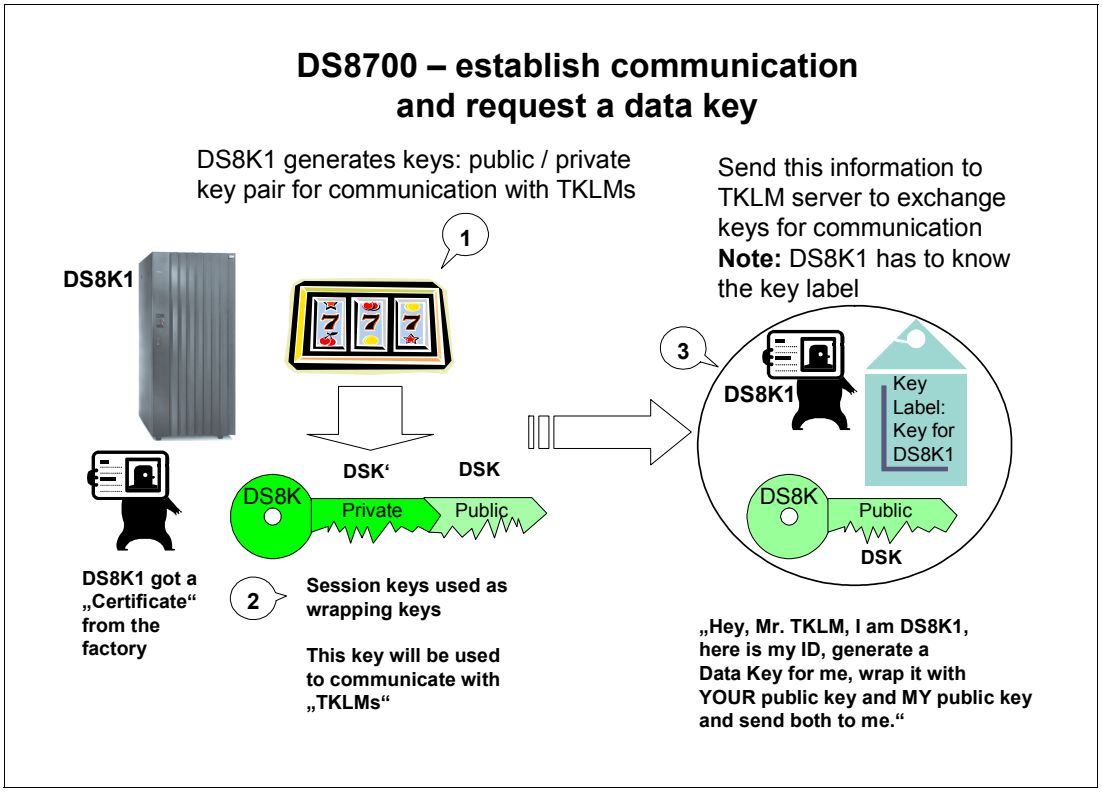


Figure 2-4 DS8000 creates session keys and requests a data key

Upon reception of these elements, Tivoli Key Lifecycle Manager carries out the following steps (see Figure 2-5 on page 19):

1. It validates the DS8700 certificate (its ID).
2. It generates the data key (DK).
3. The data key (DK) is wrapped with DS8700's device session public key (DSK) and stored in a structure referred to as the session encrypted data key (SEDK).
4. From the key label, Tivoli Key Lifecycle Manager retrieves the key pair (KEK/KEK') for the specified key label. The data key (DK) is wrapped with the key-label public key (or public key encrypting key, KEK) and stored in a structure referred to as the externally encrypted data key (EEDK).

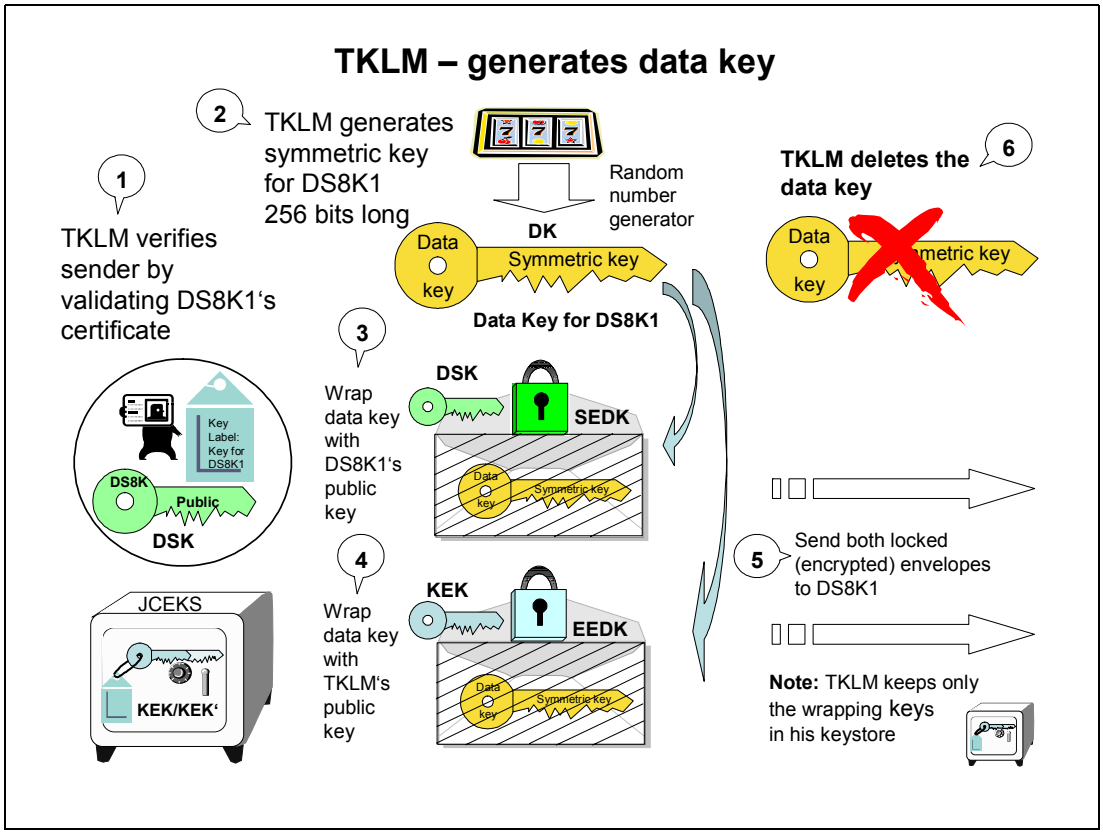


Figure 2-5 Tivoli Key Lifecycle Manager generates data key

Now, Tivoli Key Lifecycle Manager transfers the SEDK key and EEDK key to the DS8000 and the following steps are carried out at the DS8700:

1. The DS8000 receives the encrypted structures with the data key in it.
2. To recreate the data key (DK) at the DS8700, the session encryption data key (SEDK) is unwrapped with DS8000's device session private key (DSK'). The DS8700 holds the data key (DK) in memory. See Figure 2-6 on page 20.
3. The encrypted data key EEDK is stored in DS8700's keystore. Note, that the DS8700 does not have the key to unlock this structure.
4. The DS8000 generates a random 256-bit group key (GK) for the encryption group. See Figure 2-7 on page 21.
5. The group key (GK) is wrapped with the data key and stored in a structure referred as the encrypted group key (EGK).
6. The EGK is persistently stored on the system disk in the key repository. Both the externally encrypted data key (EEDK) and the EGK are stored in multiple places for reliability.

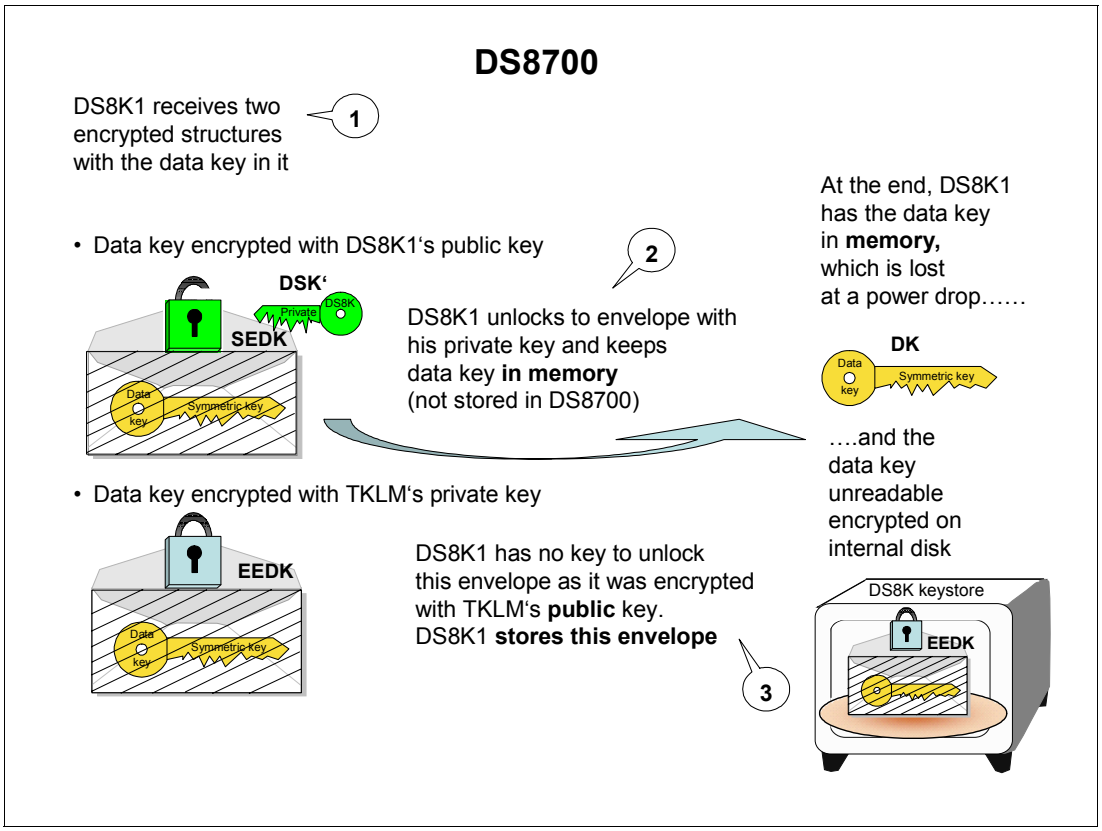


Figure 2-6 DS8700 unwraps data key and stores encrypted data key

This dual control, (from DS8700 and Tivoli Key Lifecycle Manager) improves security: the DS8000 does not maintain a persistent copy of the DK in the clear and is thus unable to encrypt or decrypt data without access to Tivoli Key Lifecycle Manager.

Note that the DK is *erased* by the DS8700 at power off, such that each time it is powered on, the DS8700 must communicate with Tivoli Key Lifecycle Manager to obtain the DK again.

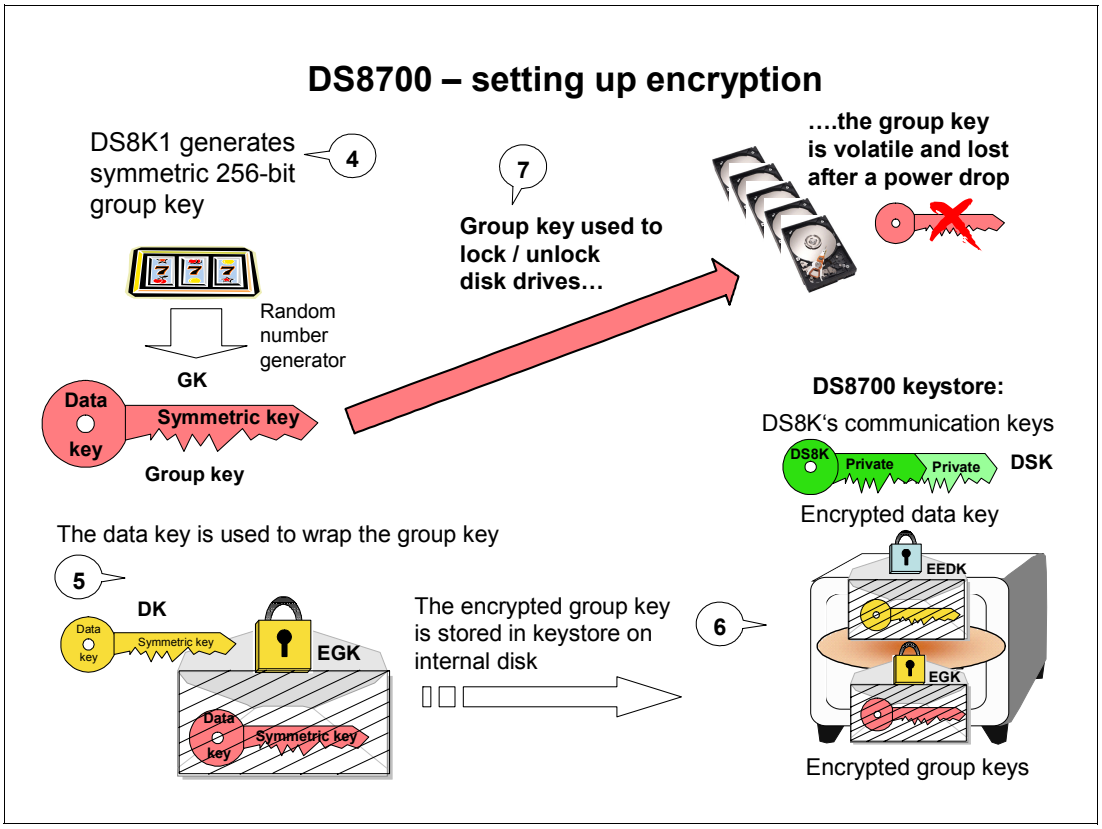


Figure 2-7 Setting up encryption

When the user configures a rank, the DS8700 creates for each DDM in this rank, an access credential to lock the drive. See Figure 2-8 on page 22. The following steps occur during configuration of the rank:

1. The DS8700 reads the serial number of each disk.
2. The serial number is hashed with the group key to create the access credential.
3. The access credential is sent to the drive.
4. In the drive the encryption key is wrapped with the access credential. A hash of the access credential is also stored on the drive.

The drives are locked now. This means after a power off and a power on the drives will only grant access to data when the encrypted encryption key that is stored on the drives is unlocked by providing access credentials and an unlock key. See Figure 2-9 on page 23.

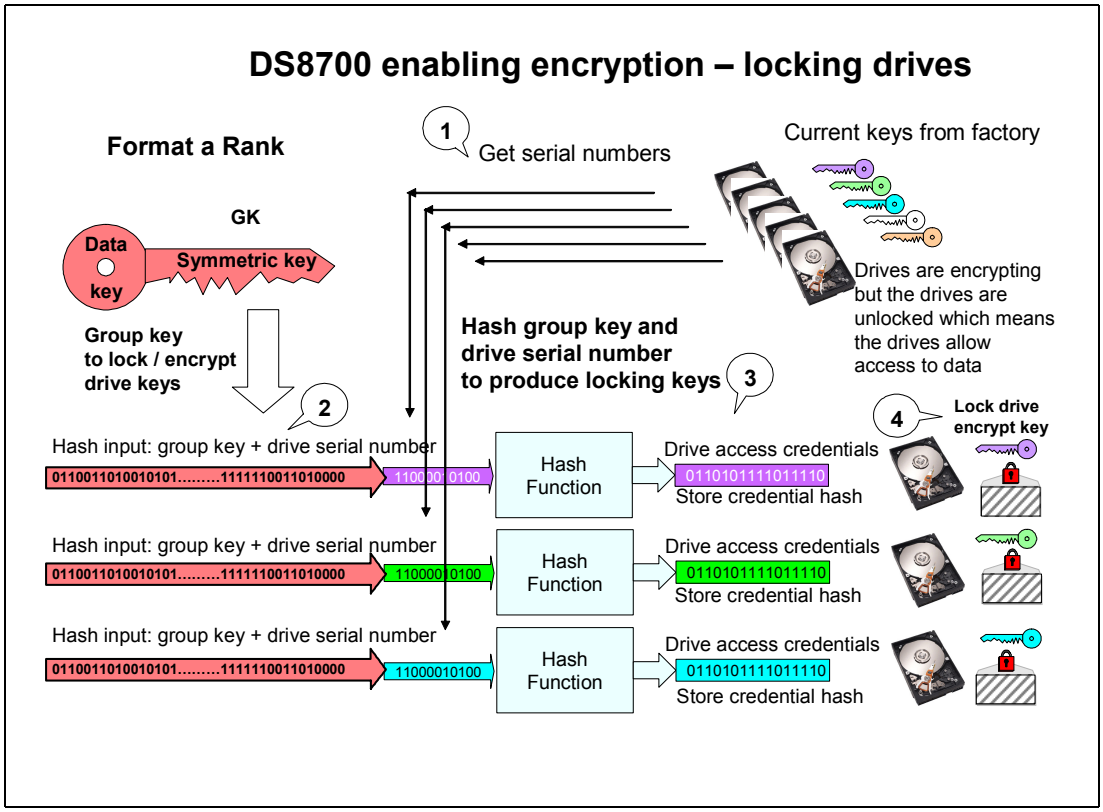


Figure 2-8 Setting up encryption: locking the drives

Disk encryption details

Each FDE drive has an encryption key for the area of the drive that contains customer data (Band 1). As shown in Figure 2-9 on page 23, a Band 0 is for internal global data, which is also encrypted.

When the customer data area is *unlocked*, the FDE drive still encrypts/decrypts the data with a data encryption key and this data encryption key is also wrapped (encrypted) with access credentials. Here, a default encryption key is used to encrypt the data encryption key, but it is done transparently to the initiator (DS8700). If someone, however, takes the disk plate without the interface, and somehow tries to read from the disks, it would be impossible because the data is encrypted.

The data encryption key for the data area is *wrapped* (encrypted) with an access credential produced with the group key. This access credential is converted to a secure hash and stored on the disk. At that stage, the customer data area is *locked*.

After a disk power loss, the read/write access to the data on a locked area is blocked until the DS8000 has authenticated by supplying the currently active access credential, the group key See Figure 2-9 on page 23. (The DS8700 must first get unlock keys for the group key from the Tivoli Key Lifecycle Manager server). The following steps occur:

1. The disk drives verifies the access credentials (containing the group key).
2. The drive validates the access credentials with the one stored on the disk drive.
3. The drive reads the stored encrypted data key.
4. The encrypted data key is decrypted utilizing access credentials (group key).

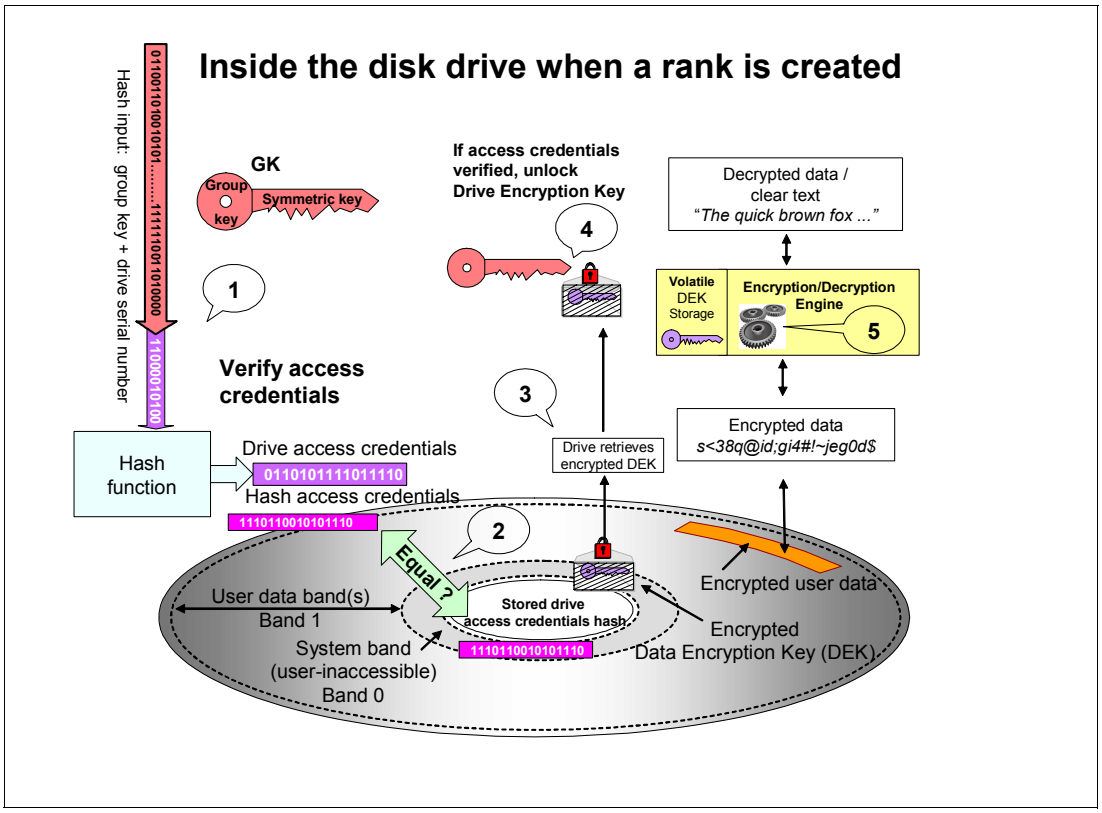


Figure 2-9 Unlocking drives

An FDE drive that is made a member of an encryption-enabled rank is locked. The FDE drive is unlocked when it is unassigned, is a spare, or is a member of an encryption-disabled rank. Locking occurs when an FDE drive is added to an encryption-enabled rank either during rank creation or sparing. Unlocking occurs when an encryption-enabled rank is deleted or a member of an encryption-enabled rank is reused as a spare. Unlocking always results in a cryptographic erasure of a FDE drive (the disk resets its own encryption key). This also happens when an encryption-disabled rank is deleted.

In a cryptographic erasure, a new data encryption key is generated in each disk drive. See Figure 2-10 on page 24. The new key is encrypted with default access credentials, both the access credentials and the encrypted data encryption key are stored on band 0 of the drive. Now the drive is unlocked. If someone were to try to read the old data, nonsense data would be returned because decryption now uses another key, which does not decrypt the data any longer.

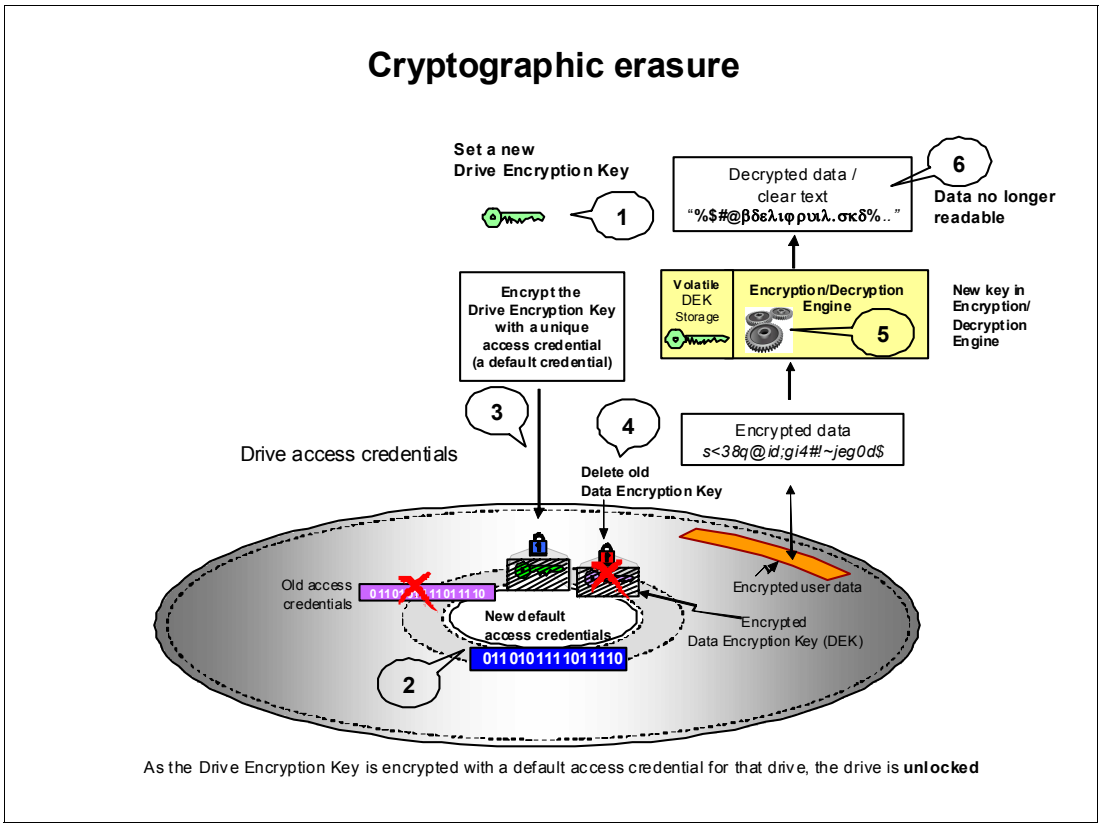


Figure 2-10 Cryptographic erasure

FDE drives are *not* cryptographically erased when a drive fails. In this case, there is no guarantee that the device adapter can communicate with the disk. More specifically, the device adapter intentionally fences the failing drive from the device interface as soon as possible to prevent it from causing other problems on the interface. However, because the currently active encryption key is encrypted, the data is not readable.

Getting access to data after a power on

After powering off and powering on, the DS8700 no longer has a data key or a group key in the clear as shown in Figure 2-11 on page 25. The data encryption keys in the drives are encrypted, the group key to unlock the drives is encrypted and the data key to get access to the group key keystore is encrypted. But the DS8700 does not have access to all these keys. It must first get a key to unlock the data key from Tivoli Key Lifecycle Manager.

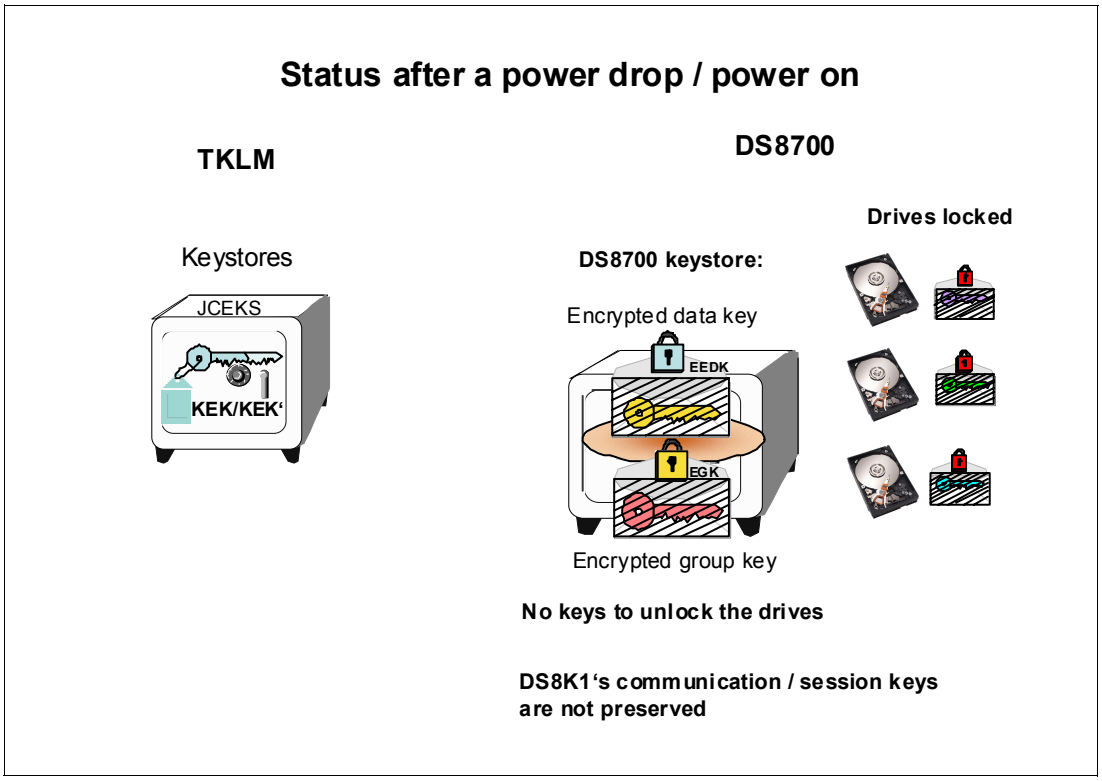


Figure 2-11 DS8700 status after a power off/on

The DS8700 must perform the following steps, also shown in Figure 2-12 on page 26, to regain access to locked drives and data at power-on:

> **Important:** The DS8700 must be able to communicate with at least one Tivoli Key Lifecycle Manager server at power-on.

1. The DS8700 generates a new session key pair (private and public) to communicate with Tivoli Key Lifecycle Manager.
2. DS8700 gets the encrypted data key (EEDK) from his keystore.
3. The DS8700 requests Tivoli Key Lifecycle Manager to unwrap an existing wrapped data key by sending the request to Tivoli Key Lifecycle Manager with the saved externally encrypted data key (EEDK), the session public key (DSK) and DS8700's certificate.
4. Tivoli Key Lifecycle Manager unwraps the EEDK with the key-label private key to obtain the data key (DK).
5. The DK is wrapped with DS8700's session public key (DSK) to create the session encrypted data key (SEDK).
6. The SEDK is returned to the DS8700.
7. The SEDK is decrypted with DS8700's session private key (DSK') to obtain the data key (DK).
8. The DK is then used to unwrap the encrypted group key (EGK) to get the group key (GK).
9. The serial number of the disk is read and hashed with the GK to obtain the access credential. The hashed access credential is sent to disk and the validity of the access credential is verified. If the access credential is valid, the disk encrypted data key is unwrapped to gain access to the data.

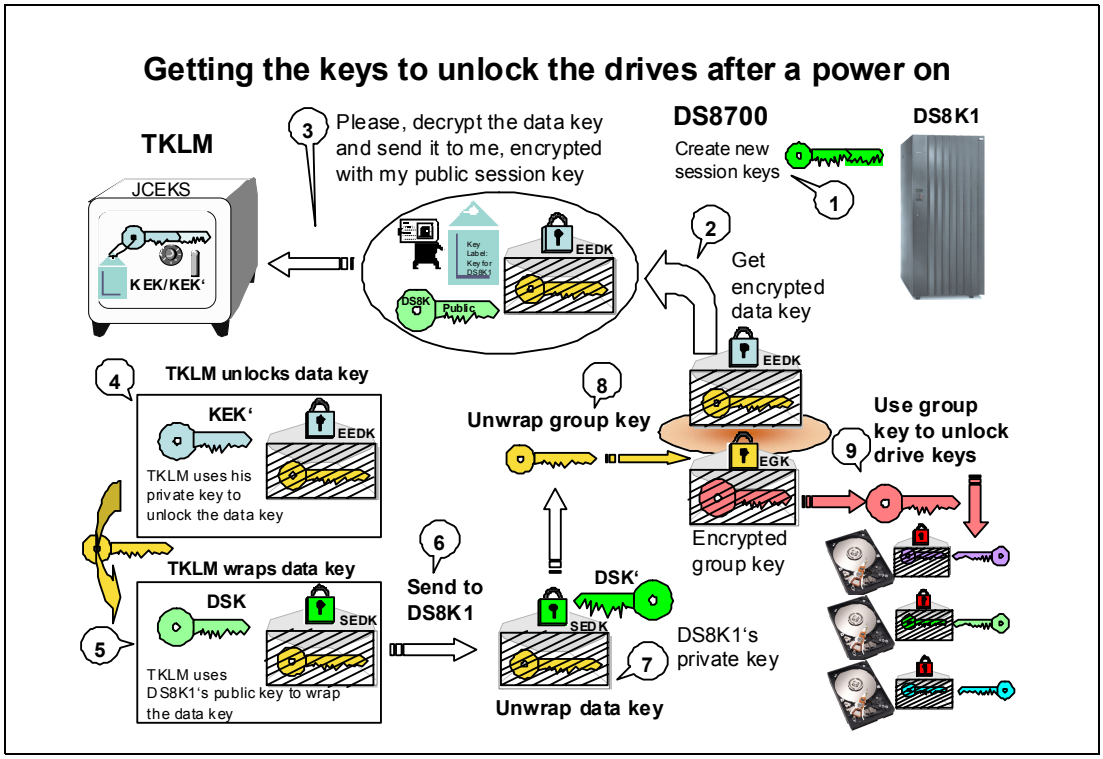


Figure 2-12 Steps to regain access to data after a power-on

Figure 2-13 again summarizes all key management mechanisms.

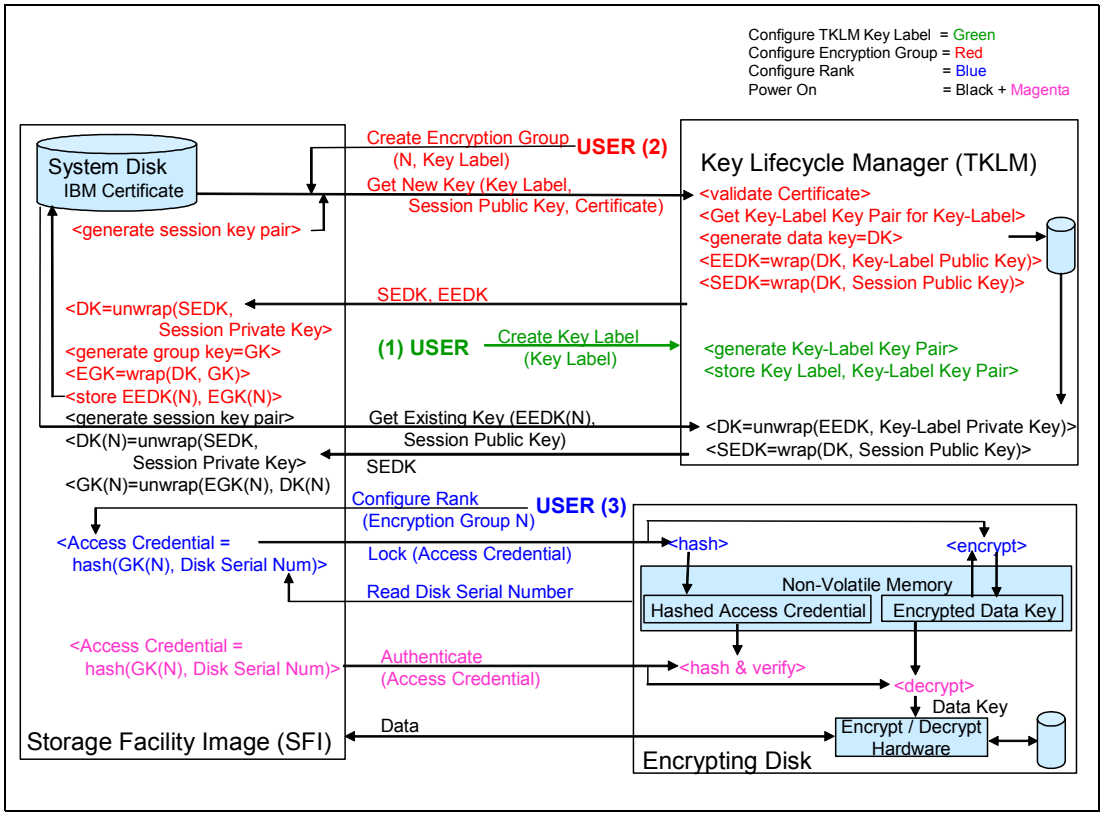


Figure 2-13 Encryption Key Management

2.3 Encryption deadlock

The key server platform provides the operating environment for the key server application to run in, to access its keystore on persistent storage, and to interface with client storage devices such as the DS8700 that require key server services.

The keystore data is accessed by the key server application through a password specified by the customer. As such, the keystore data is encrypted at rest, independently of where it is stored. However, any online data that is required to initiate the key server must not be stored on a storage server that has a dependency on the key server to enable access. If this constraint is not met, the key server is not able to complete its initial program load (IPL) and does not become operational.

This required data includes the boot image for the operating system that runs on the key server, and any other data that is required by that operating system and its associated software stack to run the key server application, to allow the key sever to access its keystore, and to allow the key server to communicate with its storage device clients. Similarly, any backups of the keystore must not be stored on storage that has a dependency on a key server to access data.

Not strictly following these implementation requirements might result in the situation where the encrypted data can no longer be accessed either temporarily, or worse, permanently. This situation is referred to as *encryption deadlock*.

Important: Any data required to make Tivoli Key Lifecycle Manager key server operational must *not* be stored on an encrypted storage device that is managed by this particular key server. Again, this situation is referred as an *encryption deadlock.* This situation is similar to having a bank vault that is unlocked with a combination and the only copy of the combination is locked inside the vault.

The differences between a temporary encryption deadlock and a permanent encryption deadlock are as follows:

- Temporary encryption deadlock

  The temporary encryption deadlock indicates a situation where the DS8000 cannot access its disk devices because Tivoli Key Lifecycle Manager servers are not online, the network is down, or for other temporary hardware-related errors. This temporary failure can be fixed at the customer site.

- Permanent encryption deadlock

  This permanent encryption deadlock is the worse case. Here, all Tivoli Key Lifecycle Manager servers that manage some set of data cannot be made operational because they have a dependency on inaccessible encrypted storage; or all encrypted online and offline data managed by the set of Tivoli Key Lifecycle Managers is, in effect, cryptographically erased and for all practical purposes permanently lost.

When considering encryption in your environment, consider the following factors:

- As the availability of encryption-capable devices becomes more pervasive, more and more data will be migrated from non-encrypted storage to encrypted storage. Even if the key servers are initially configured correctly, it is possible that a storage administrator might accidentally migrate some data required by the key server from non-encrypted to encrypted storage.
- Generally, a number of layers of virtualization in the I/O stack hierarchy can cause difficulties for the customer to maintain an awareness of where all the files (necessary to

make the key server, and its associated keystore, available) are stored. The key server may access its data through a database that runs on a file system that runs on a logical volume manager, which communicates with a storage subsystem that provisions logical volumes with capacity obtained from other subordinate storage arrays. The data required by the key server might end up provisioned over various storage devices, each of which can be independently encryption-capable or encryption-enabled.

- Consolidation of servers and storage tends to drive data migration and tends to move increasingly more data under a generalized shared storage environment. This storage environment becomes encryption-capable as time goes on.
- All IBM server platforms support fabric-attached boot devices and storage. Some servers do not support internal boot devices. Therefore, boot devices are commonly present within the generalized storage environment.These storage devices are accessible to generalized storage management tools that support data management and relocation.

To mitigate the risk of an encryption deadlock, a stand-alone Tivoli Key Lifecycle Manager server is mandatory and the customer must be directly involved in managing the encryption environment. Refer to Chapter 4, “DS8700 encryption implementation” on page 49 and Chapter 3, “Planning and guidelines for DS8700 encryption” on page 37.

2.4 Working with a recovery key

To get out of a deadlock situation or, as a safety belt, if all Tivoli Key Lifecycle Managers are destroyed and unrecoverable, the DS8700 allows you to create a *recovery key*. With a recovery key, a security administrator can unlock a DSS8000 without involvement of a Tivoli Key Lifecycle Manager server. With the Licensed Internal Code level 65.10.xx a user can also *disable* the recovery key management.

Important: A recovery key can only be created if encryption is not yet enabled. You cannot create a recovery key when a DS8700 is already configured. Similarly disabling the recovery key management is allowed only for an unconfigured DS8700. Creating or disabling a recovery key must be one of the first actions when setting up a DS8700.

Managing the recovery key requires two people (roles): a storage administrator and a security administrator. The security administrator is a new role for DS8700 users. A storage administrator cannot create a security administrator user on a DS8700 and vice versa. The security administrator maintains the recovery key and keeps it safe; the storage administrator has to approve every action of the security administrator.

Customer responsibility: The DS8700 supports the two roles, storage administrator and security administrator, but the customer really is responsible to effectively assign these roles to two *separate* individuals.

2.4.1 Recovery key management is enabled

The following sections summarize the actions allowed in a recovery key-enabled scenario.

Creating a recovery key

Setting up a recovery key involves the following steps, shown in Figure 2-14 on page 30:

1. The security administrator user requests the creation of a recovery key. This can be done with the DS CLI or the GUI. Note that this request function is not available to other users.
2. At some stage in the process, the storage administrator must approve the action that the security administrator is going to perform.
3. Having obtained the request to generate a recovery key, the DS8700 generates a random 256-bit recovery key (RK).
4. The DS8700 generates a secure hash of the RK producing the recovery key signature.
5. The storage facility generates an asymmetric key pair from a random 2048 bit number. The private key is referred to as the primary recovery key (PRK) and the public key is referred to as the secondary recovery key (SRK).
6. Next, the DS8700 wraps the PRK with the RK producing the encrypted primary recovery key (EPRK).
7. The EPRK, SRK, and the recovery signature (RS) are stored in multiple places within the storage facility for reliability.
8. The storage facility provides the RK to the security administrator. The system follows a verification process, which is not further detailed here (the security administrator has to re-input the RK).
9. The DS8700 deletes the PRK and the RK.

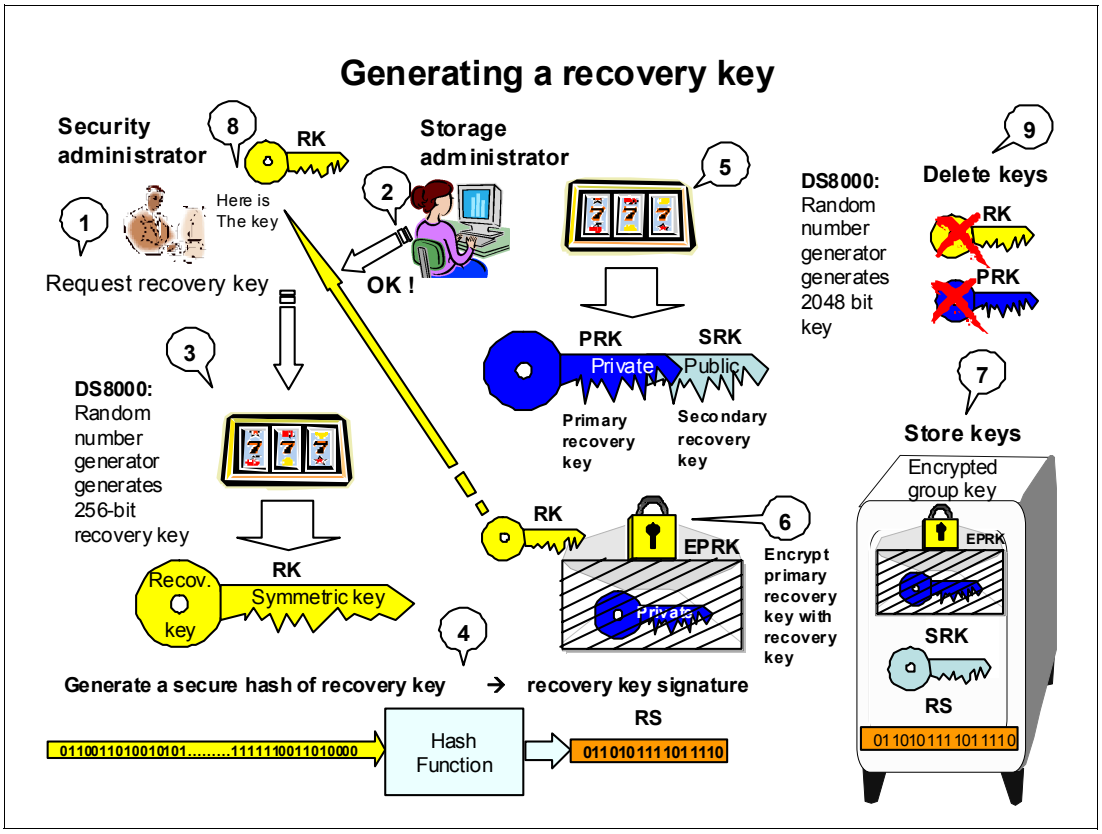


Figure 2-14 Generating a recovery key

When configuring an encryption group with a recovery key defined, additional steps to those shown in Figure 2-7 on page 21 are carried out (see Figure 2-15 on page 31):

1. The storage facility also wraps the group key with the secondary recovery key (SRK) producing the encrypted group recovery key (EGRK).
2. The EGRK is stored together with the encrypted primary recovery key (EPRK) and the other encrypted keys (EEDK and EGK) in the DS8700 keystore.

After the encryption group is configured, ranks may be created and assigned to an encryption group.

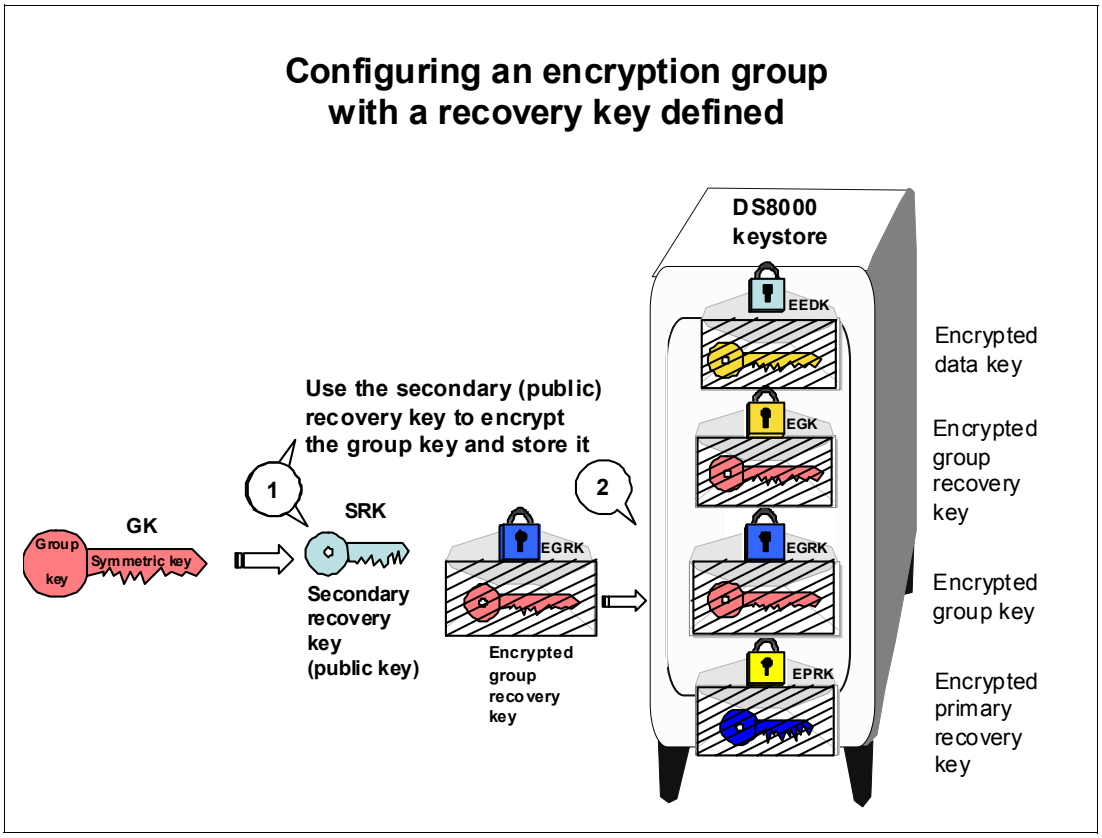


Figure 2-15 Setting up encryption with a recovery key defined

Using a recovery key to unlock a DS8700

If the DS8000, after a power-off and power-on, is unable to obtain the required data key (DK) from a key server, it attempts to contact all other configured Tivoli Key Lifecycle Manager servers to obtain the required key.

On a DS8700 with a recovery key configured, an option exists to let a security administrator input the recovery key (RK).

If the security administrator provides the RK and the storage administrator approves this operation, the DS8700 uses the RK to unwrap the encrypted primary recovery key (EPRK) to obtain the PRK, as follows (see Figure 2-16 on page 32):

1. DS8000 cannot communicate with any Tivoli Key Lifecycle Manager server.
2. DS8700 can ask for a recovery key.
3. The security administrator enters the recovery key.
4. The storage administrator approves the action.
5. The recovery key is used to unlock the primary recovery key.
6. The primary recovery key is used to unlock the group key.
7. The group key is used to unlock the drives.

Now, access to data is restored.

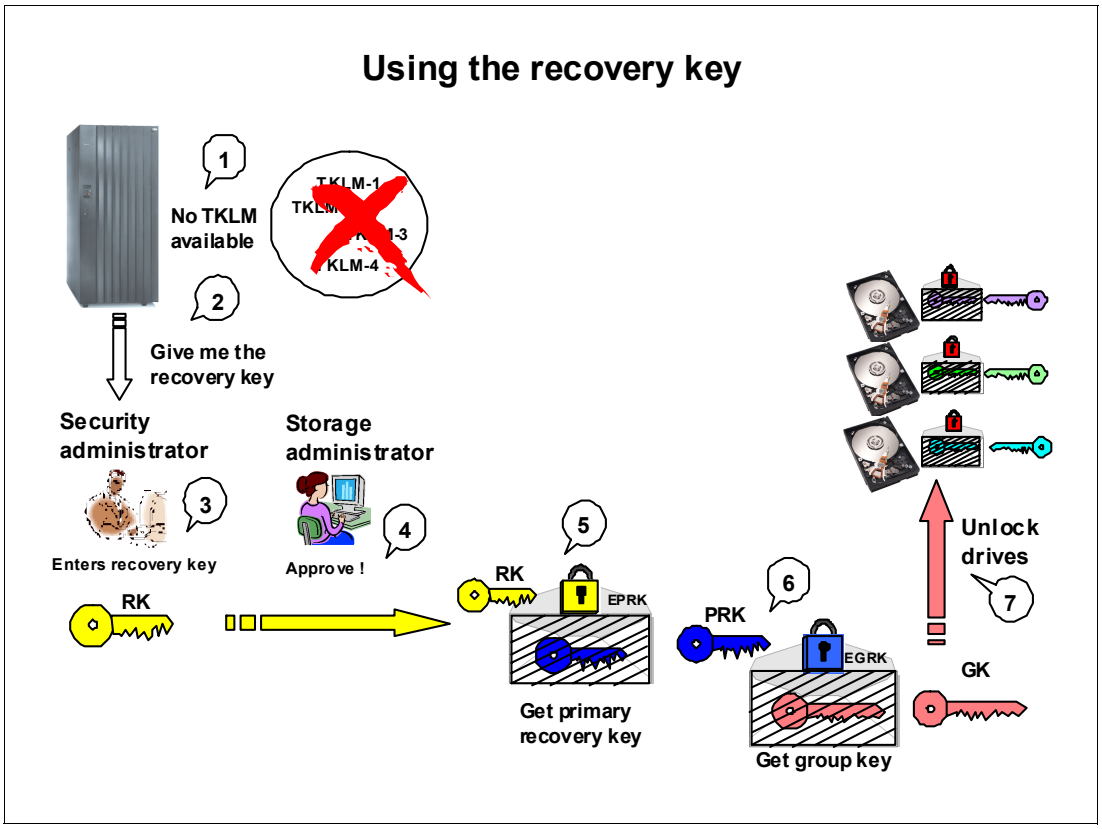


Figure 2-16 Using the recovery key

Changing the recovery key

The DS8700 also supports functions to re-key, verify, and de-configure a recovery key.

The re-key and verify recovery key functions can be performed at anytime while the recovery key is configured and a Tivoli Key Lifecycle Manager server is available. The presence of a Tivoli Key Lifecycle Manager server is required. It allows the DS8700 to verify that it is in the correct environment. Only when the Tivoli Key Lifecycle Manager can decrypt the data key, the DS8700 can be sure that it is in the same environment (see Figure 2-17 on page 33). Only then, it will generate a new recovery key. For example, on a DS8700 that was stolen and put in a separate environment, re-keying the recovery key is not possible.

During a re-key operation, the following steps are performed:

1. DS8700 sends the externally encrypted data key (EEDK) and its public key to Tivoli Key Lifecycle Manager and requests a re-key validation.
2. Tivoli Key Lifecycle Manager tries to decrypt the data key (DK).
3. If Tivoli Key Lifecycle Manager can decrypt the DK, it signals the DS8700 that it can proceed to generate a new recovery key.
4. DS8700 generates a new recovery key.

Changing the recovery key does not erase the data.

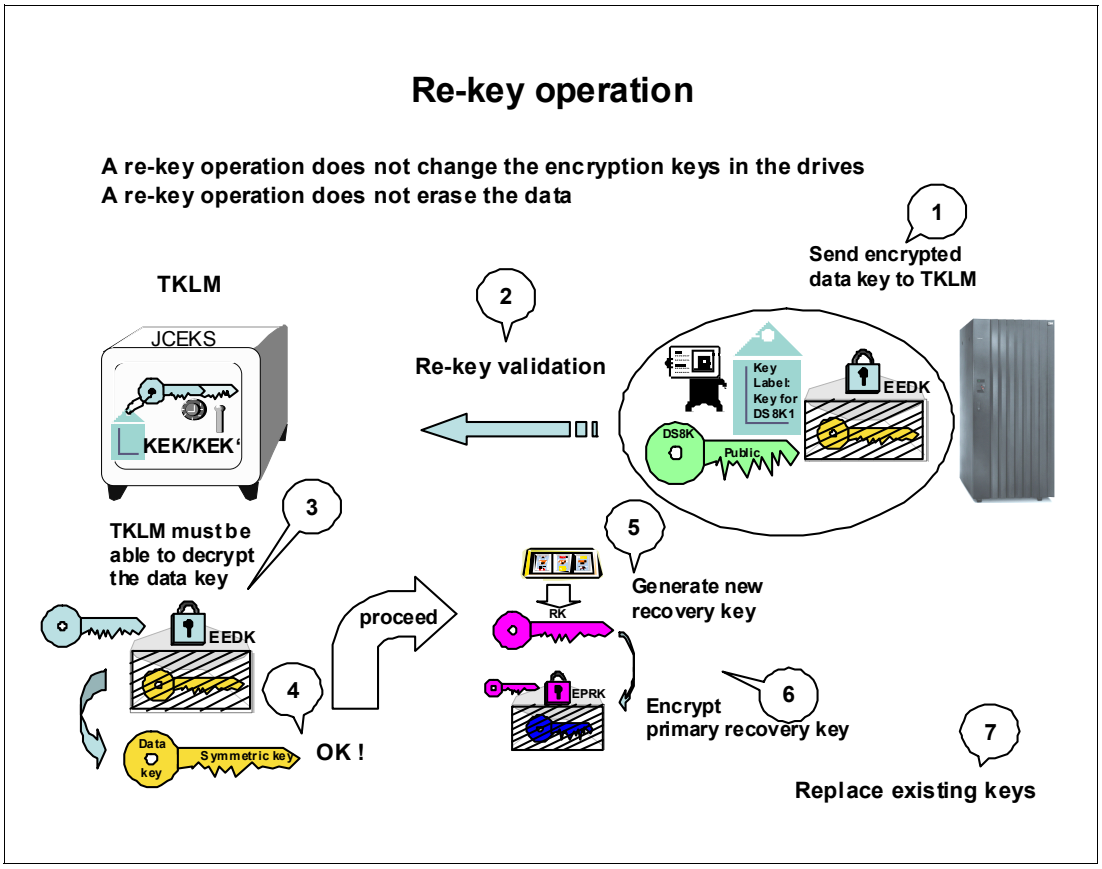


Figure 2-17 Re-key the recovery key

The deconfigure function is available only when no encryption group is defined.

2.4.2 Disabling or enabling a recovery key

The recovery key can be optionally disabled, or configured, before establishing an initial logical configuration. This section describes the actions required for disabling or enabling the recovery key.

Disabling a recovery key

If you do not want to manage a recovery key in your environment, it can be disabled. However, this must be done as your first action, before defining the encryption group. The process of disabling the recovery key is as follows:

1. The security administrator user requests the disabling of a recovery key. This can be done with the DS CLI or the GUI. Note that this request function is available only to a user with security administrator role.
2. The storage administrator approves the disabling status.
3. The recovery key goes into the *disable* state.

The encryption group can now be defined.

Refer to section 4.2, “DS8700 GUI configuration for encryption” on page 62 and 4.3, “DS8700 DS CLI configuration for encryption” on page 85 for an illustration of these actions.

Enabling a recovery key

When a recovery key has been disabled, it can later be re-enabled. Note that this action is disruptive: all data (on the DS8700) must be erased as a prerequisite.

Enabling the recovery key management involves the following steps:

1. The security administrator user requests the enabling of a disabled recovery key. This can be done with the DS CLI or the GUI. Note that this request function is not available to other users.
2. The storage administrator approves the enabling status.

The recovery key can now be created as described in “Creating a recovery key” on page 29.

Refer also to section 4.2, “DS8700 GUI configuration for encryption” on page 62 and 4.3, “DS8700 DS CLI configuration for encryption” on page 85 for an illustration of these actions.

2.5 Dual key server support

DS8700 supports the configuration of either one or two key labels for the encryption group:

When all key server platforms operate their key stores in clear-key mode or when only a single host platform is used for all key servers, a single key label is typically sufficient to allow all key severs to interoperate with the DS8700. In this case, it is possible for the asymmetric key pair maintained for the key label by the Tivoli Key Lifecycle Manager to be propagated across all supporting key servers so that each key server has the necessary keys to wrap and unwrap the one EEDK maintained on the DS8700.

When there are two key server platforms and at least one of the key server platforms is operating in secure key mode (which is available on the z/OS platform), a second key label is typically required. A key server operating in secure key mode typically does not support the export of any private keys outside of the key server platform. In this case, the following actions are performed to synchronize keys between key servers (see Figure 2-18 on page 35):

- Key label 1 (with public and private key) is configured on platform UNIX®, for example.
- Key label 2 (with public and private key) is configured on platform z/OS, for example.
- The public key from key label 1 is exported to platform Tivoli Key Lifecycle Manager for z/OS (abbreviated as TKLM-z/OS in the figures).
- The public key from key label 2 is exported to platform Tivoli Key Lifecycle Manager for UNIX (abbreviated as TKLM-UNIX in the figures).

At this point, all key servers on both platforms have the *public keys for both key labels* and the *private of one or the other key label*. The DS8700 can request a new key from *any* key server and store an EEDK associated with each key label. Therefore, the DS8700 has two separate EEDKs now.

Each Tivoli Key Lifecycle Manager has a public key of each key label so it can generate two EEDKs.

Let us summarize the steps of an example shown in Figure 2-18 on page 35:

1. TKLM-UNIX creates public/private key pair for key label 1.
2. TKLM-z/OS creates public/private key pair for key label 2.
3. Both Tivoli Key Lifecycle Managers exchange their public keys.

4. A DS8700 can request a data key from any Tivoli Key Lifecycle Manager. For our example, let us assume it requests the data key from TKLM-z/OS.
5. TKLM-z/OS generates the data key DK, wraps it with DS8700's public key to produce the SEDK and wraps the DK with his own public key to produce EEDK-z and wraps the DK with TKLM-UNIX's public key to produce EEDK-Ux. Then, SEDK, EEDK-z, and EEDK-Ux are sent to the DS8700.

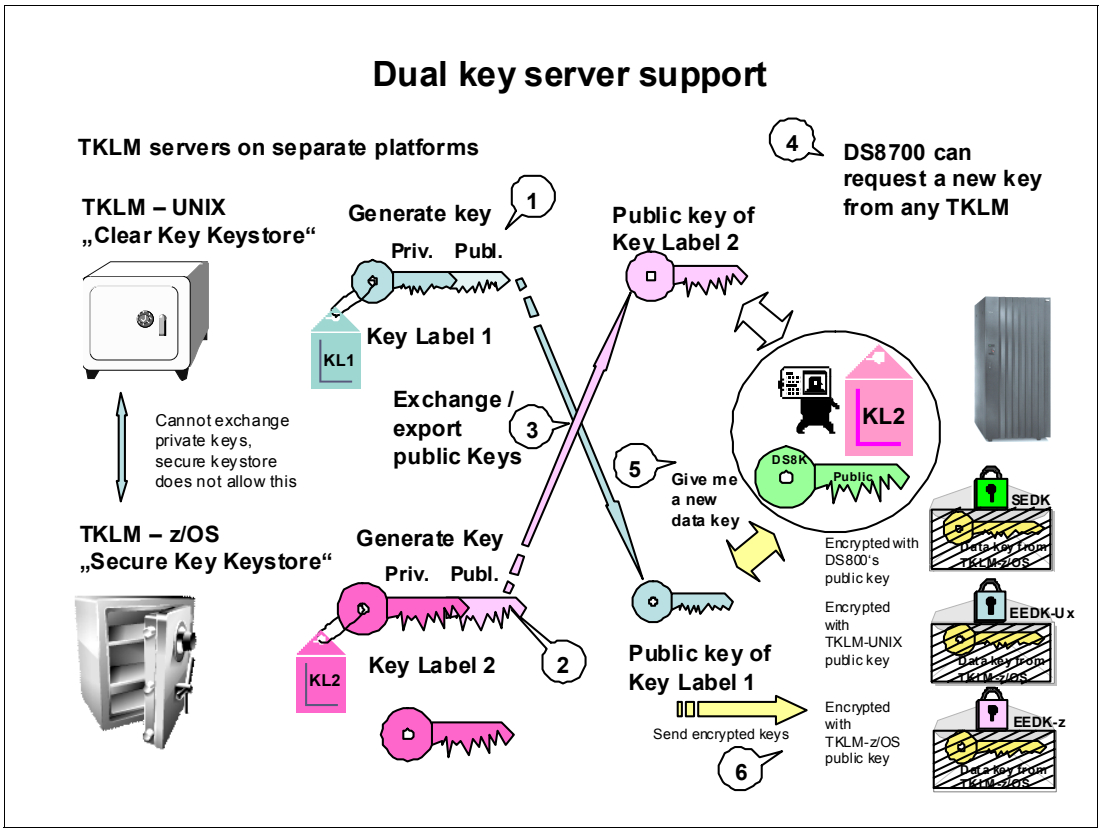


Figure 2-18 Dual key server support

The DS8700 can request the EEDKs to be unwrapped by any key sever because the request will contain both EEDKs (see Figure 2-19 on page 36), and any key server has the private key for at least one of the two EEDKs in the request. Secure key mode operation is maintained during the exporting of secure keys because only the public key is exported.

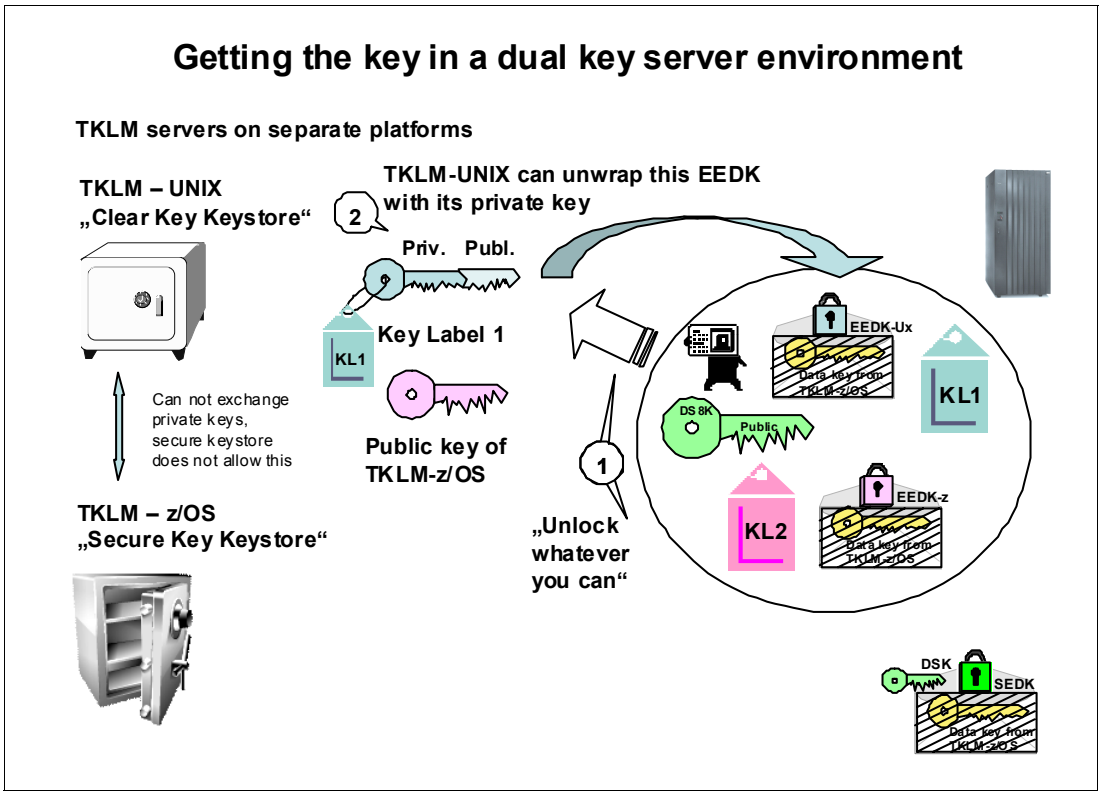


Figure 2-19 Getting the recovery key in a dual server environment

In the example shown in Figure 2-19, the DS8700 sends its request to decrypt the data key to TKLM-UNIX with both key labels and both EEDKs. TKLM-UNIX can decrypt EEDK-Ux associated with key label 1.

TKLM-UNIX can now send the encrypted data key back to the DS8700.

3

Planning and guidelines for DS8700 encryption

This chapter provides information when planning for a DS8000 encryption capable storage system.

This chapter contains the following topics:

- Planning and implementation process flow
- Encryption-capable DS8700 ordering and configuration process
- Best practices for encrypting storage environments
- Dual HMC and redundancy
- Multiple Tivoli Key Lifecycle Manager for redundancy
- Overall configuration process

Important: For up-to-date considerations and best practices regarding DS8700 encryption, refer to IBM Encrypted Storage Overview and Customer Requirements at:

http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101479

The website also includes the *IBM Notice for Storage Encryption* that must be read by all customers acquiring an IBM storage device that includes encryption technology.

© Copyright IBM Corp. 2010. All rights reserved.

3.1 Planning and implementation process flow

The diagram in Figure 3-1 shows the planning and implementation process for an encryption-capable DS8700. The details for this process are discussed in subsequent sections of this chapter. This diagram shows the overall decision flow and outcomes.

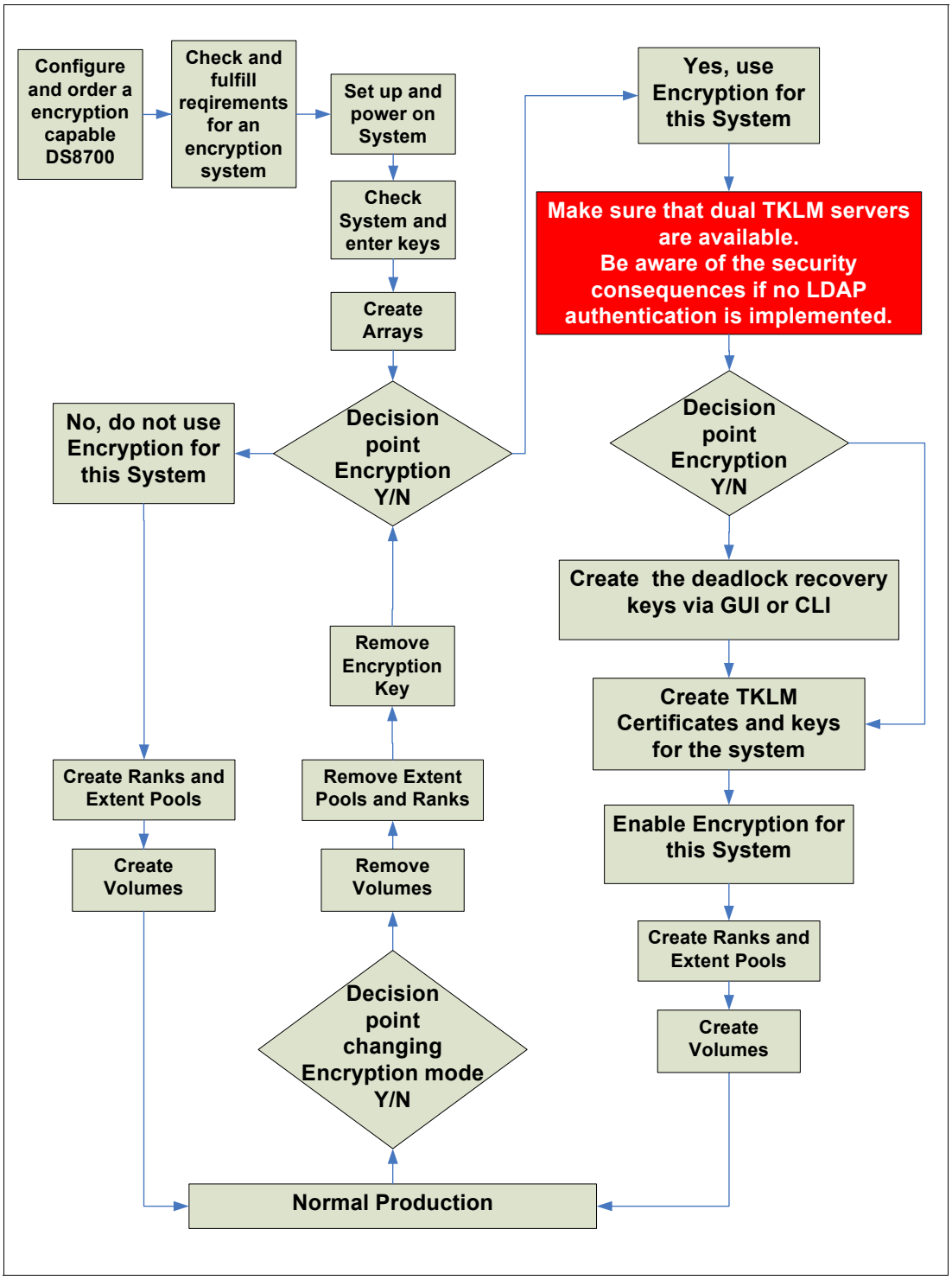


Figure 3-1 Encryption implementation planning flow

3.2 Encryption-capable DS8700 ordering and configuration

The Full Disk Encryption support feature is available only as plant order. Plant-configured encryption systems allow on-site installation of additional drive sets. Intermixing drives is not supported: the entire subsystem is either using encrypted drives (#5xxx features) or intermixed devices of Fibre Channel, SATA, and SSD devices (#2xxx and #6xxx features). The order of Laptop Management Console with feature numbers 912x is required when ordering the Full Disk Encryption disk drive sets.

The following steps summarize the sequence for the ordering, activating, and configuring an encryption-capable DS8700:

1. The IBM sales representative or IBM Business Partner places a customer order for a DS8700 with encryption-capable Disk Drive Modules (DDMs). This step automatically engages additional IBM support resources to ensure successful implementation of encryption.
2. The additional IBM support resources carry out the following steps in parallel:
   - A system assurance call is established
   - A manufacturing order is submitted
   - A request to enable encryption is submitted
3. Customers must complete an environment verification process to ensure best practice configuration of the encryption solution. This verification can be requested from IBM Lab Based Services, or completed by customers, but is a prerequisite of the encryption solution activation process.

   Note: In certain countries, customers might have to sign an “Import Agreement” to be able to import or export FDE drives.

4. After the ordering and verification process has been completed, IBM delivers the DS8700, the IBM service representative installs the DS8700, and IBM provides the customer with an Encryption Authorization Licensed Internal Code feature key (or keys) for the Storage Facility Image (or images) on the DS8700. Each Licensed Internal Code feature key is unique to the SFI for which it is generated. The key cannot be obtained from the Disk Storage Feature Activation (DSFA) website. IBM Lab Services or the customer installs the Licensed Internal Code feature key on the SFI.
5. After the storage system is made available to the customer, the deadlock recovery key must be created, using the GUI or DS CLI.
6. Now that the system can be recovered at any time by using the recovery key, the Tivoli Key Lifecycle Manager connection must be set up and made functional.

   After this step is completed, the system is fully enabled and activated for encryption.
7. The customer may now configure ranks or extent pools.

   Refer to 4.3.7, “Creating encrypted extent pools” on page 92.

Notes:

- All ranks and extent pools on a given encryption-capable DS8700 must be configured with the same encryption group attribute. The first rank or encryption group that is configured determines how the remaining objects must be configured. A value of zero indicates encryption-disabled. A value of non-zero indicates encryption-enabled.
- To change between encryption-enabled and encryption-disabled, all ranks and extent pools must be unconfigured. Unconfiguring an encryption-enabled rank causes any data that was stored on the rank to be cryptographically erased and, subsequently, overwritten to re-initialize the rank.

3.3 Requirements for encrypting storage

The following requirements must be strictly respected to deploy an encryption-capable DS8700:

- Configuring the deadlock recovery key is the mandatory first step (if you do not want to set a Recovery Key, it must be disabled at this time, refer to 2.4.1, “Recovery key management is enabled” on page 29 and 2.4.2, “Disabling or enabling a recovery key” on page 33). The key is generated from the DS GUI or DS CLI. The DS8700 *secadmin* and *admin* users must cooperate to establish the encryption deadlock recovery key.
- Any DS8700 that is encryption-enabled must be configured to at least one isolated key server.

  This key server can be configured to serve keys to any Tivoli Key Lifecycle Manager supported device, including other encryption enabled DS8700 or supported IBM tape drives.

  The isolated key server is a separately purchased hardware product (using System Storage Productivity Center, Machine type 2805 Model MC4) for the DS8700 and is required for all DS8700 encryption-enabled sites.

  The isolated Tivoli Key Lifecycle Manager key server consists of the following equipment:

  – IBM System x3650 with L5420 processor:
    - Quad-core Intel® Xeon processor X5420 (2.5 GHz, 12 MB L2, 1.0 GHz FSB, 50 W)
    - 6 GB memory
    - 146 GB SAS RAID 1 storage
    - SUSE Linux 10 with Service Pack 3
    - Dual Gigabit Ethernet ports (standard)
    - Power supply

  – Tivoli Key Lifecycle Manager V1:
    - Includes DB2 9.1 FB4

  Isolated key servers ordered with feature number 0021 (Machine type 2805 Model MC4) have a pre-installed Linux operating system and Tivoli Key Lifecycle Manager software. Customers have to acquire a Tivoli Key Lifecycle Manager license for use of the Tivoli Key Lifecycle Manager software, ordered separately from the stand-alone server hardware.
- An encryption-enabled DS8700 requires at least two key servers (that is, the isolated server and a backup server) to be configured. The key servers can be shared by more than one DS8700.

3.4 Best practices for encrypting storage environments

The following information can help you find the best practices for encrypting storage environments. It includes key techniques for mitigating the risk of an encryption deadlock.

3.4.1 Security

Considerations and best practices are as follows:

- General

  Ideally, a good practice is to manage the physical security of access to hardware through an LDAP implementation. This allows a close monitoring of who, when, and what actions were taken by monitoring the auditlogs of the DS8700. With a basic security policy, having a single person handling the *admin* and *secadmin* role of a DS8700 is still possible. With LDAP, a policy can be set up that does not allow having the same user ID for both roles in the DS8700.

- Keystore

  During the setup of the Tivoli Key Lifecycle Manager key server, a password is specified that is used to access the keystore. Customers must decide whether the Tivoli Key Lifecycle Manager password will be provided manually or whether a mechanism is in place to automatically provide the password to the Tivoli Key Lifecycle Manager. If a startup script containing the password is used at the Tivoli Key Lifecycle Manager key server, the script file must have access controls to prevent unauthorized access to the file and password.

3.4.2 Availability

Considerations and best practices are as follows:

- DS8700

  The DS8700 should be configured with the dual HMC option to provide redundant access to the customer network. Refer to 3.5, “Dual HMC and redundancy” on page 44. The inability of a DS8700 to communicate with a Tivoli Key Lifecycle Manager key server when it powers on will prevent access to encrypted storage on the DS8700.

- Tivoli Key Lifecycle Manager key server

  Note the following information:

  - Configure redundant key servers to each encrypting storage device. The customer should have independent and redundant key servers on each site.
  - To initiate the Tivoli Key Lifecycle Manager key server operation after power on, without human intervention, the key server should be set up to automatically power on when power is available and to automatically initiate the key server application. The application should be configured to automatically boot.
  - Configure redundant network fabrics between the Tivoli Key Lifecycle Manager key servers and encrypting DS8700 (see Figure 3-2 on page 45). Providing independent network paths through independent key servers prevents any single point of failure.

3.4.3 Encryption deadlock prevention

Considerations and best practices are as follows:

- General

  Note the following information:

  - The change management processes at the customer installation should cover any procedures required to ensure adherence to guidelines required to ensure proper configuration of key servers, encrypted storage, and data placement of data related to key servers.
  - All personnel who have any of the following assignments or capabilities should be required to review a customer document that describes these risks and the processes adopted to mitigate them, at least annually:
    - Responsibility for the implementation of Tivoli Key Lifecycle Manager key servers or encrypted storage products
    - Responsibility to manage the placement or relocation of data related to, or required by, any Tivoli Key Lifecycle Manager key server
    - Access authority to configure Tivoli Key Lifecycle Manager key servers or encrypted storage products
    - Responsibility to re-key the deadlock recovery key of the DS8700, if used
  - The customer should implement automated monitoring of the availability of any equipment associated with management of key services and take appropriate action to keep them operational. This equipment can include but is not limited to key servers, SNMP masters, domain name servers, and DS8700 HMCs.
  - The customer should pay particular attention to disaster recovery plans and scenarios, and consider the availability of key servers, key server backups, and key server synchronization. A good practice is to establish the independence of each recovery site from the other recovery site.
  - If the recovery key management is enabled, the customer must have a documented process to handle and maintain the deadlock recovery keys of each DS8700. This key is the last resort to unlock the DS8700 if the Tivoli Key Lifecycle Manager environment is destroyed or totally inaccessible. Note that the deadlock recovery key is not used while Tivoli Key Lifecycle Manager remains available.

- Tivoli Key Lifecycle Manager key server

  Note the following information:

  - Configuration of redundant key servers (at least two) is required. Redundancy implies independent servers and independent storage devices. For key servers operating in LPARs, do not use data sharing techniques that result in one copy of the data being shared by multiple instances of the key server.
  - Configuration of one key server with dedicated hardware and non-encrypted storage resources at each recovery site is required. The key server is referred to as *isolated key server*.

    Note: IBM DS8700 requires at least one isolated key server to be configured, but a good practice is to use two for redundancy.

The objective of this requirement is to avoid encryption deadlock by the following tasks:

- Implementing a key server environment that is independent of all non-key server applications so that management of the key server can be restricted to those personnel specifically authorized to manage key servers
- Implementing a key server that is physically and logically isolated from other applications that may require access to encrypting storage so that the key server environment does not need to be configured with access to any encrypting storage
- Implementing a key server that is physically and logically isolated from encrypting storage so that the risk storing (initially or through data migration) code and data objects required by the key server on encrypting storage is eliminated
- Ensuring that a recovery site can operate independently of any other sites by configuring a key server that is not subject to encryption deadlock because of the characteristics of an isolated key server

– Configuration of additional key servers on generalized server hardware and generalized storage is allowed. The customer should establish appropriate procedures and controls to prevent these key servers from having their data access compromised by storing the data on key server managed encrypting storage. These key servers are referred to as *general key servers*.

– Configuration of key servers at independent sites is a good practice and provides additional immunity to encryption deadlocks because it reduces the probability that all key servers will experience a simultaneous power loss.

– Customers must ensure that all key servers, which a given storage device is configured to communicate with, have a consistent keystore content relative to any wrapping keys that will be used by the storage device. Failure to synchronize the keystores effectively eliminates one or more key servers from the set of redundant key servers for a storage device that uses the keys that are not synchronized.

– Customers should back up key server data after it is updated. The backups should not be stored on encrypted storage media that is dependent on a key server. Refer to 5.1, “Backup and restore” on page 96.

– Customers should periodically audit to ensure that all online and backup data required to make each key server operational is stored on storage or media that is not dependent on a key server to access the data.

– Customers should not delete keys on the key server under normal circumstances. Deletion of all copies of a key is a cryptographic erase of all encrypted data that is encrypted under this key.

► DS8700

Note the following information:

– Before any Tivoli Key Lifecycle Manager server is connected to the DS8700, run the deadlock recovery key generation process.

– Manually configure DS8700 devices on the Tivoli Key Lifecycle Manager key server (this is a suggestion). The option to automatically configure them may be used, but increases the risk that an unauthorized DS8700 might gain access to a key server.

– The DS8700 supports up to four Tivoli Key Lifecycle Manager key server ports. A requirement is that at least one port is assigned to one isolated key server. A good practice is to assign two ports to isolated key servers. To use key servers at the local site should be preferred to improve reliability.

– When the DS8700 is configured to enable encryption the DS8700 verifies that at least two Tivoli Key Lifecycle Manager key serves are configured and accessible to the machine.

– The DS8700 will reject the creation of ranks and extent pools with non-zero encryption group specified if the encryption has not been activated.

– The DS8700 monitors all configured Tivoli Key Lifecycle Manager key servers. Customer notification is provided through the DS8700 customer notification mechanism (SNMP traps, email, or both, when configured) when loss of access to the key servers is detected. Key server-related errors are provided through the same mechanism. Set up monitoring for these indications and take corrective actions when a condition is detected. This reflects a degraded key server environment.

  The following conditions are monitored and reported:

  - If the DS8700 cannot receive a required data key, during power-on, for a configured encryption group from the key servers, it reports the error condition to the customer and to IBM. In this case, logical volumes associated with the encryption group are inaccessible to attached hosts. If the DS8700 is able to obtain the required data key from a key server, after reporting the error, it reports the condition to the customer and to IBM and makes the associated logical volume accessible.
  - DS8700 access to each configured key server is verified at five-minute intervals. Loss of access is reported to the customer.
  - The ability of each key server to unwrap data keys configured on the DS8700 is verified at eight-hour intervals. Loss of the ability to unwrap a configured data key is reported to the customer and to IBM.
  - The DS8700 detects if fewer than two key servers are configured, or if fewer than two key servers are available, or if fewer than two key servers can unwrap data keys configured on the DS8700 at eight-hour intervals. If detected, this condition is reported to the customer and to IBM.

3.5 Dual HMC and redundancy

A dual Hardware Management Console (HMC) is a redundant management system that provides flexibility and high availability. When two HMCs manage one system, they are peers, and each can be used to control the managed system. One HMC can manage multiple managed systems, and each managed system can have two HMCs.

The dual HMC implementation is also the only way to get more than one Ethernet cable being used to access the DS8700. Therefore, having dual HMCs in a Tivoli Key Lifecycle Manager environment to prevent a single point of failure is a good practice.

3.5.1 Dual HMC advantages

Having an external DS HMC provides a number of advantages:

- Enhanced maintenance capability

  Because the DS HMC is the only interface available for service personnel, a second DS HMC greatly enhances maintenance operational capabilities if the primary DS HMC fails.

- Improved remote support

  In many environments, the DS8700 and internal HMC are secured behind a firewall in a user's internal LAN. In this case, for IBM to provide remote support can be difficult. An external DS HMC can be configured in such a way that it is able to communicate with both the DS8700 and IBM. Thus, the dual HMC configuration can greatly enhance remote support capabilities.

- High availability for configuration operations

  In open systems environments, all configuration commands must go through the HMC. This is true regardless of whether you use the DS CLI, the DS Storage Manager, or the DS Open API. An external DS HMC will allow these operations to continue to work despite a failure of the internal DS HMC.

- Dual Ethernet cable connection to the HMC

  The second Ethernet cable must be connected with a separate subnet to allow access to the DS8700, even during network maintenance.

- High availability for Advanced Copy Services operations

  In open systems environments, all Advanced Copy Services commands must also go through the HMC. This approach is true regardless of whether you use the DS CLI, the DS Storage Manager, or the DS Open API. An external DS HMC will allow these operations to continue to work despite a failure of the internal DS HMC.

- High availability for Tivoli Key Lifecycle Manager server

  Refer to 3.5.2, “Redundant HMC configurations” on page 45.

3.5.2 Redundant HMC configurations

You can have a configuration in which dual HMC servers are connected to the DS8700 Storage Server and to the Tivoli Key Lifecycle Manager servers.

Using a redundant HMC configuration with your Tivoli Key Lifecycle Manager server and DS8700 Storage Server setup requires a specific port configuration, as shown in Figure 3-2. In this configuration, each Tivoli Key Lifecycle Manager server is connected to two network switches and each of the switches is connected to one HMC. Each HMC is connected to the DS8700 Storage Server. The network switches connected to the HMC must remain in the power-on state. Any Ethernet switch or hub can be used to connect the Tivoli Key Lifecycle Manager server and HMC.

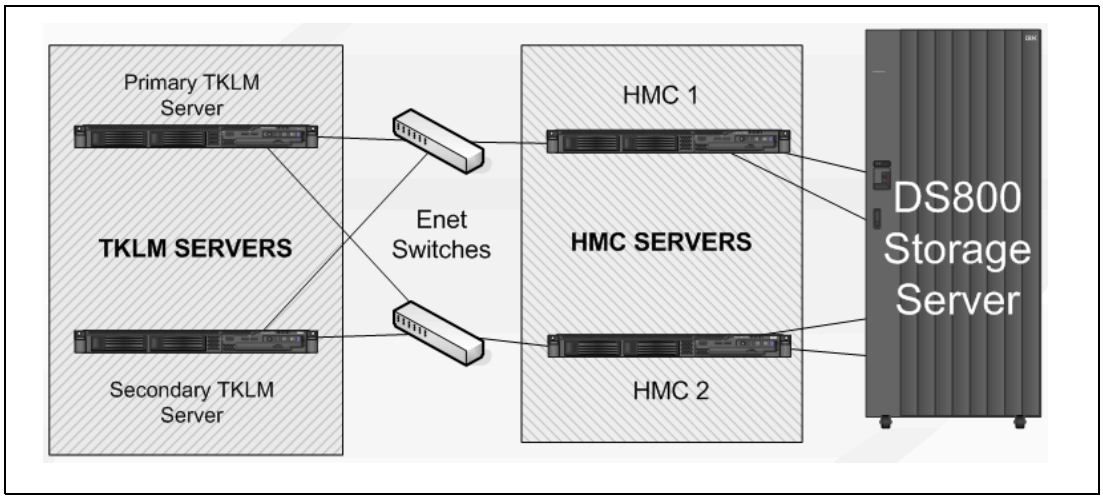


Figure 3-2 Redundant HMC configuration: normal operation

In a normal operation, the DS8700 Storage Server connects to the primary Tivoli Key Lifecycle Manager server across the Primary HMC, as shown in Figure 3-3 on page 46.

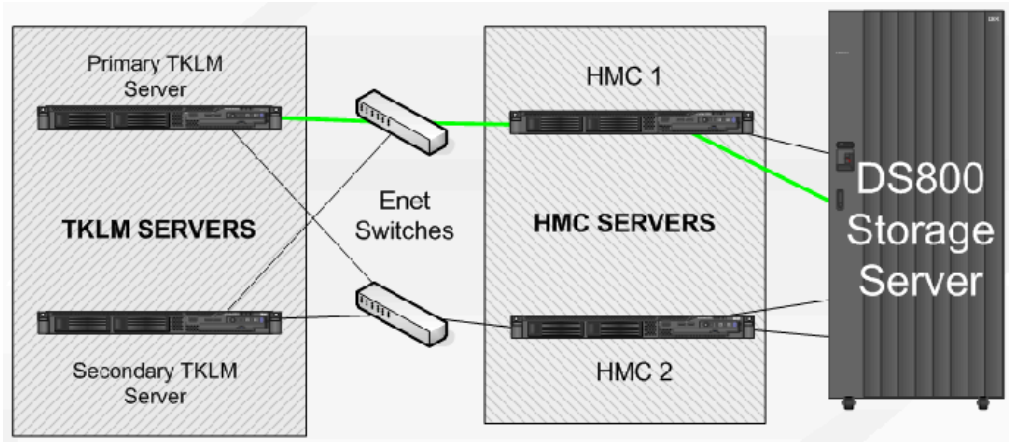


Figure 3-3 Communication path between DS8700 and Tivoli Key Lifecycle Manager server

In case of network failure or maintenance issues on the primary HMC, the DS8700 Storage Server can connect to the Primary Tivoli Key Lifecycle Manager Server over the secondary HMC, as shown in Figure 3-4.

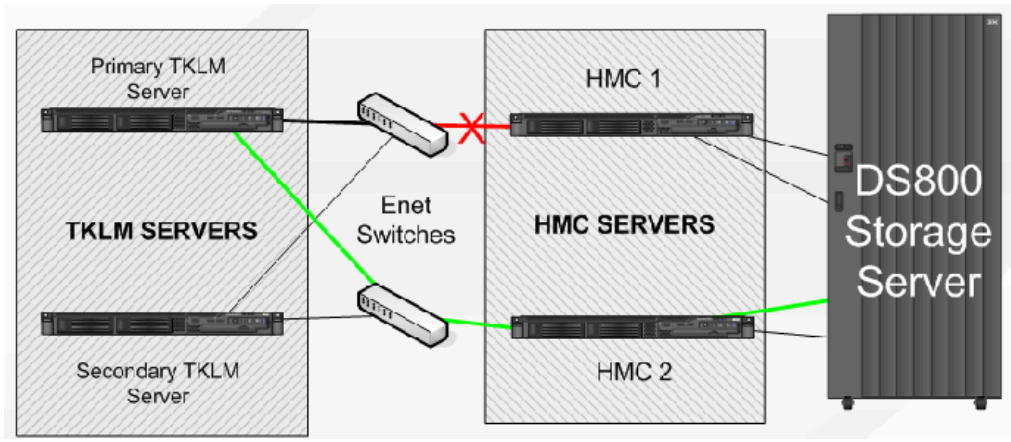


Figure 3-4 Failover communication between DS8700 and Tivoli Key Lifecycle Manager server

In a redundant HMC configuration, *both* HMCs and Tivoli Key Lifecycle Manager servers are *fully active, highly available*, and *accessible* at *all times*, enabling you to perform management tasks from either HMC at *any time*.

3.6 Multiple Tivoli Key Lifecycle Managers for redundancy

To ensure continuous key and certificate availability to encrypting devices, configure a primary and a replica Tivoli Key Lifecycle Manager server for your enterprise, and then provide repeated backup/restore or import/export actions to protect critical data.

On Windows systems and other systems such as Linux or AIX, both computers must have the required memory, speed, and available disk space to meet the workload. The operating system, middleware components, and Tivoli Key Lifecycle Manager application directory layout must be identical on both computers.

Note that this is not a failover or clustered server from a Tivoli Key Lifecycle Manager point of view. The redundancy is managed by setting up multiple key manager destinations at the DS8700 Storage Server.

Synchronization is achieved by backing up one server and restoring the backup configuration on the other server, assuming both servers have the same operating system in this case. If you have servers with different operating systems you have to use the export/import function. Plan to do this backup/restore or export/import process when the following events take place:

- Initial configuration
- Adding keys or devices
- Key or certificate replacement intervals
- Certificate authority (CA) requests

3.6.1 Setting up Tivoli Key Lifecycle Manager servers

In our experimentation environment, we were running two SUSE Linux 9 hosts. This allowed us to easily match the environment seen by Tivoli Key Lifecycle Manager and its middleware.

A good practice is to fill out the installation worksheets found in the *Tivoli Key Lifecycle Manager Installation and Configuration Guide*, SC23-9977.

For detailed information, refer to the following resources:

- “Installing Tivoli Key Lifecycle Manager on a Windows platform” on page 146
- *IBM Tivoli Key Lifecycle Manager Installation and Configuration Guide,* SC23-9977
- The Tivoli Integrated Portal information center:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.tip.doc/ctip_install_overview.html

4

DS8700 encryption implementation

This chapter reviews the sequence of tasks involved in the deployment of an encryption-capable DS8700, from ordering to installation and usage.

This chapter contains the following topics:

- Configuring Tivoli Key Lifecycle Manager
- Disk encryption configuration for the DS8700 Storage Server with the GUI
- Disk encryption configuration for the DS8700 Storage Server with the DS CLI
- Copy Services functions considerations

© Copyright IBM Corp. 2010. All rights reserved.

4.1 Tivoli Key Lifecycle Manager configuration

In this section, we describe, step by step, the procedure required to prepare Tivoli Key Lifecycle Manager (on open systems) to serve an encryption-enabled DS8700 storage system. We assume Tivoli Key Lifecycle Manager servers are installed and ready to be configured. For more information, refer to the following resources:

- Detailed installation procedures on System z® (z/OS) can be found in *IBM Tivoli Key Lifecycle Manager for z/OS*, REDP-4472
- For more complete information, refer to *IBM Tivoli Key Lifecycle Manager Installation and Configuration Guide,* SC23-9977.
- Additionally, refer to the Tivoli Integrated Portal:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.tip.doc/welcome_tip_ic.htm

4.1.1 Log in to Tivoli Key Lifecycle Manager console

The Tivoli Key Lifecycle Manager solution for open systems platforms incorporates the *Tivoli Integrated Portal* installation manager, which provides simple to use installation and a management console for AIX, Windows, Linux, and Solaris. To manage and configure Tivoli Key Lifecycle Manager, y log in to the Tivoli Integrated Portal as explained in this section.

Unlike open systems, the access interface to the Tivoli Key Lifecycle Manager installed on System z (z/OS) differs slightly. Tivoli Key Lifecycle Manager on z/OS runs within the System Services Runtime Environment (SSRE). SSRE has its *Integrated Solutions Console (ISC)*; logging in to ISC is required to gain access to the Tivoli Key Lifecycle Manager application.

However, after you are logged in to Integrated Solutions Console or Tivoli Integrated Portal, the Tivoli Key Lifecycle Manager menu is identical, regardless of the platform on which Tivoli Key Lifecycle Manager server is installed.

Log in to Tivoli Integrated Portal

Open a web browser and point it at Tivoli Key Lifecycle Manager, specifying its `IP-Address` and `IP-Port`, using the following web address format:

```
https://<IP-Address>:<IP-Port>/ibm/console
```

The default Tivoli Key Lifecycle Manager installation secures the HTTPS transport with a self-signed certificate. Depending on what browser you use, you might get an exception and will then have to accept that certificate as a trusted certificate. The Tivoli Integrated Portal login window opens; see Figure 4-1 on page 51. Specify a User ID and Password and click **Log in**.

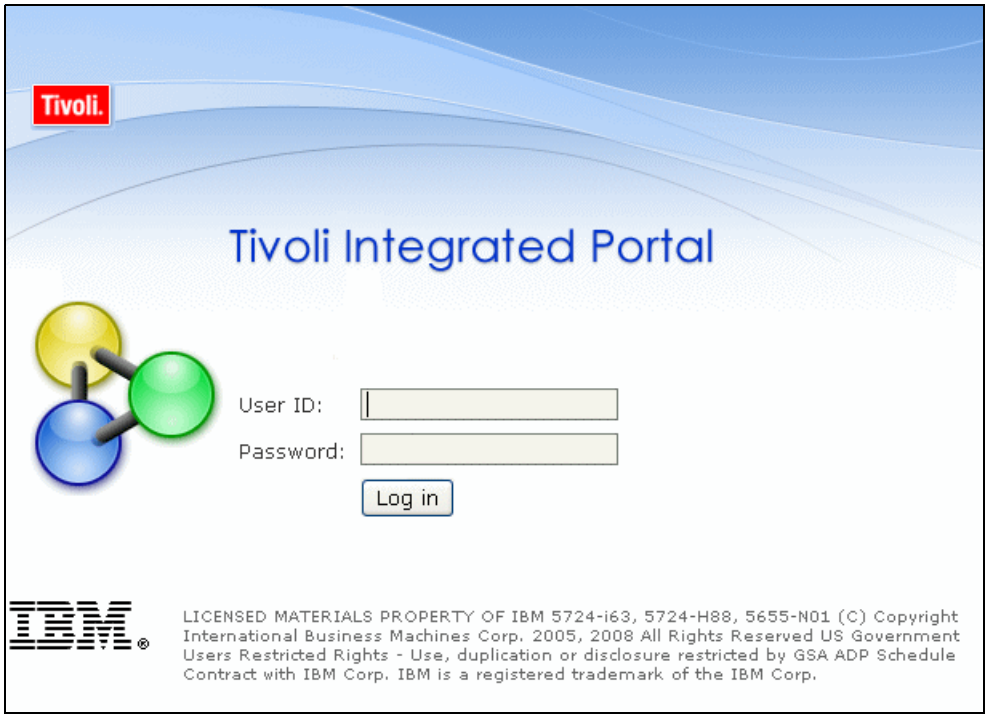


Figure 4-1 Tivoli Integrated Portal login window

The welcome window opens; you are successfully logged in to Tivoli Integrated Portal, as shown in Figure 4-2.

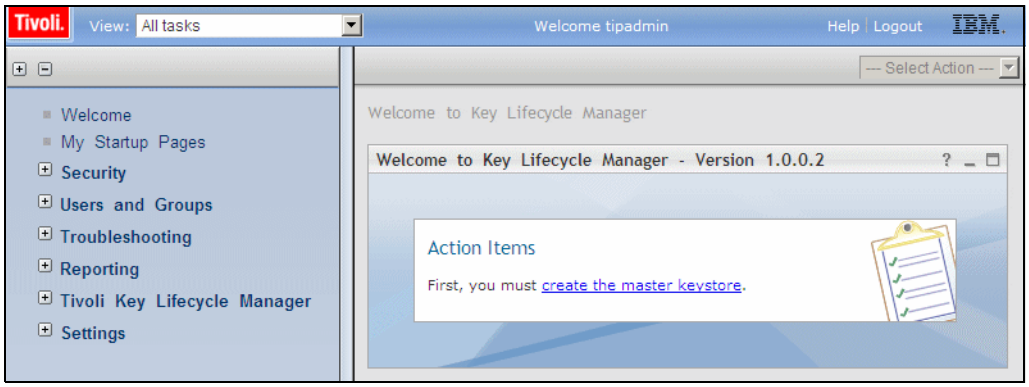


Figure 4-2 Tivoli Integrated Portal welcome window

The actual version number is visible in the title of the welcome window. In this case it is 1.0.0.2, which indicates that the fixpack 2 has been installed on the 1.0 base level.

Log in to Integrated Solutions Console

If the Tivoli Key Lifecycle Manager server is installed on System z, open a web browser and direct it to the following default location for SSRE Integrated Solutions Console:

```
https://<IP-Address or hostname>:32211/ibm/console
```

The Integrated Solutions Console login window opens as shown in Figure 4-1. Specify a User ID and Password and click **Log in**.

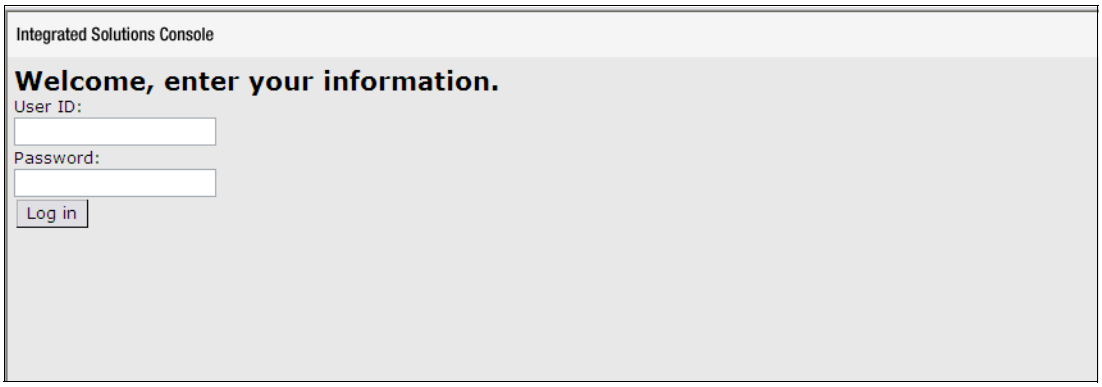


Figure 4-3 Integrated Solutions Console login window

The Welcome panel shown in Figure 4-4 opens.

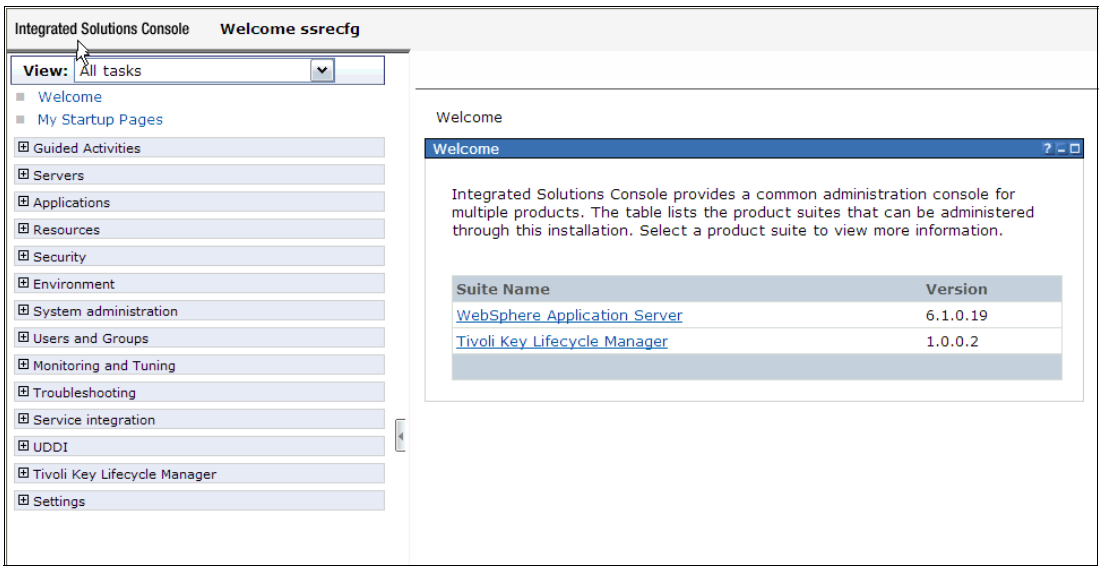


Figure 4-4 Integrated Solutions Console welcome panel

The Tivoli Key Lifecycle Manager version number is visible in the Suite Name section of the Welcome panel. In this case, it is 1.0.0.2, which indicates that the fixpack 2 has been installed on the 1.0 base level.

In the Suite Name section, click **Tivoli Key Lifecycle Manager** to start configuration.

Further in this chapter we access Tivoli Key Lifecycle Manager server mainly with Tivoli Integrated Portal interface.

4.1.2 Creating SSL certificate for secure communication

The first time you access Tivoli Key Lifecycle Manager, a link is available to create the master keystore. This link is in the Action Items section of the Welcome panel, shown in Figure 4-2 on page 51. This keystore is used to hold all keys and certificates managed by Tivoli Key Lifecycle Manager (open systems). Perform the following steps:

1. Click **create the master keystore** link, which then opens the Keystore window shown in Figure 4-5 on page 53. Select the Keystore type **JCEKS**, which is actually the only

possible choice when the Tivoli Key Lifecycle Manager server is installed on open systems platform. The JCEKS software keystore type supports asymmetric and symmetric keys.

In our example, we enter `ITSO Sample Keystore` as keystore name and `/opt/TKLM/Keystores` as keystore path and file name.

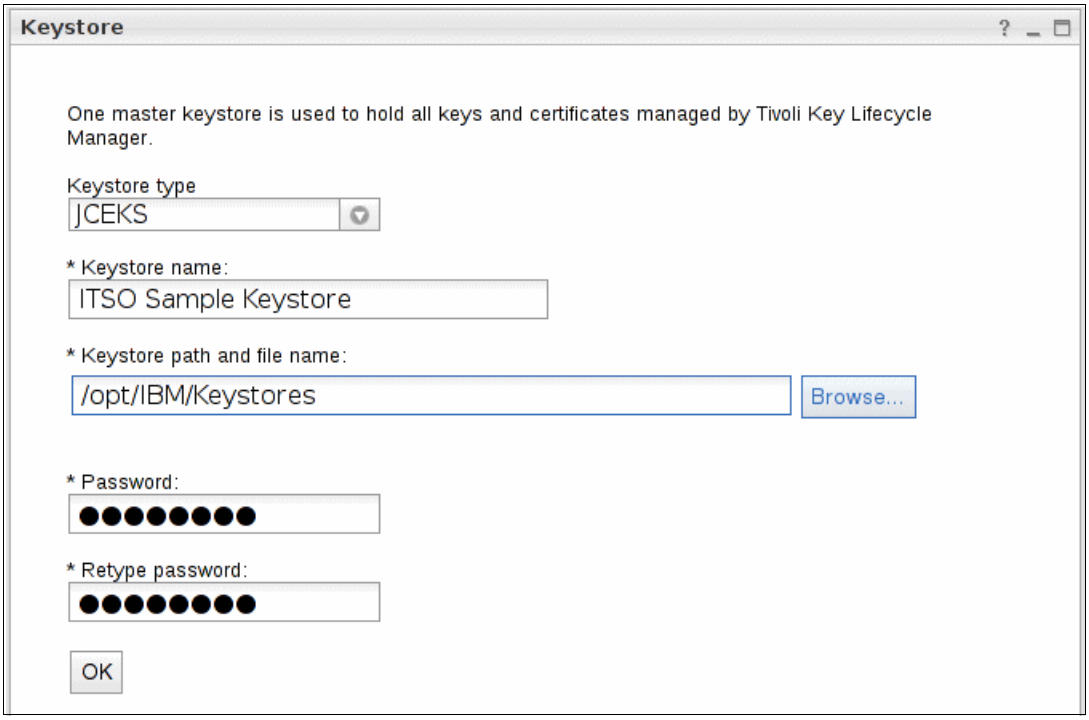


Figure 4-5 Create keystore name and location for Tivoli Key Lifecycle Manager server on open systems platform

2. Specify a password. This password is set as the master keystore password. This password is also referred to as the Tivoli Key Lifecycle Manager Master Key. It is the key to all other keys that are maintained by the Tivoli Key Lifecycle Manager keystore.

   Note: Losing the password results in not being able to transfer any certificate from this Tivoli Key Lifecycle Manager server to another Tivoli Key Lifecycle Manager server.

   If the Tivoli Key Lifecycle Manager server is installed on System z (z/OS) platform, it provides support for the following keystore types:

   – JCERACFKS (JCE software provider)

     The JCERACFKS keystore makes use of all the security advantages of IBM RACF® by storing keys and certificates in a RACF database structure referred to as a RACF keyring. Use this keystore type to store key material in your RACF keyring that is not using Integrated Cryptographic Services Facility (ICSF). Note that this keystore does not support symmetric keys, thus it is not used with LTO tape drives. It supports all DS8700 family of disk storage systems and TS1100 family enterprise tape drive.

   – JCECCAKS (IBMJCECCA provider)

     JCECCAKS is a file-based keystore. Both symmetric and asymmetric keys can be stored in JCECCAKS, thus it can be used with the DS8700 family of disk storage systems, TS1100 family of tape drives and LTO tape drives. Use this keystore type when you desire a file-based keystore that has the added security benefit of being hardware protected. By leveraging Integrated Cryptographic Services Facility (ICSF),

your flat file JCECCAKS keystore will reside in a restricted area of the file system on the Tivoli Key Lifecycle Manager system. JCECCAKS is primarily intended to store asymmetric keys that are to be used with the cryptographic hardware coprocessors. It can also be used to access symmetric keys kept in the ICSF CKDS. Note that to use hardware protection you must have Crypto Express2 cards in your processor.

– JCECCARACFKS (IBMJCECCA provider)

  The JCECCARACFKS keystore makes use of all the security advantages of RACF and hardware protection by storing a PKDS label in a RACF database entry, referred to as a key ring. The actual keys and certificates are stored encrypted with the coprocessor Master Key in the ICSF PKDS. Only asymmetric keys can be stored in JCECCARACFKS, thus it can be used with the DS8700 family of disk storage systems and TS1100 family of tape drives and DS8700 Turbo drives. Use this keystore type when you need a RACF protected keystore that has the added security benefit of being hardware-protected and you do not have LTO drives that are encrypting data. Note that hardware protection requires a Crypto Express2 card in your processor.

As shown in Figure 4-6, for Tivoli Key Lifecycle Manager server running on System z, select the appropriate keystore according to your requirements.

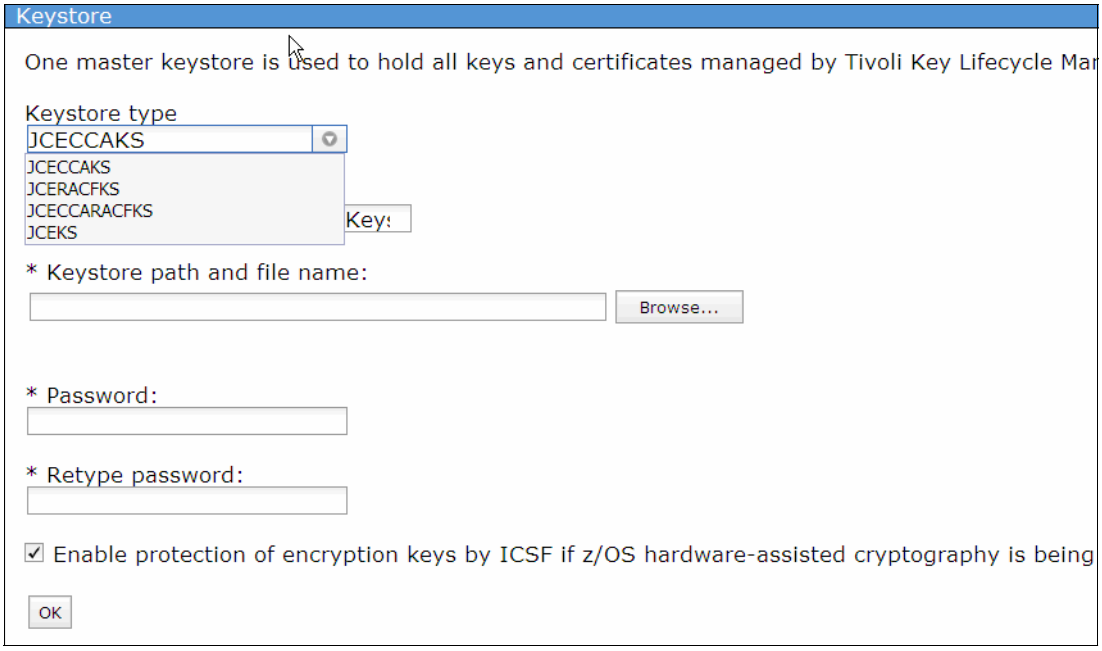


Figure 4-6 Create keystore name and location for Tivoli Key Lifecycle Manager server on System z (z/OS)

For additional information related to Tivoli Key Lifecycle Manager keystore planning and considerations on a z/OS platform, refer to *IBM Tivoli Key Lifecycle Manager for z/OS*, REDP-4472.

3. After you select a keystore type and provide all other required information, click **OK**. The window shown in Figure 4-7 on page 55 opens.

Figure 4-7 Keystore created successfully

4. In this window, under Next Steps, click **Configure the product to use specific communication protocol(s)**. In this step, we are creating the certificate that will be used to encrypt data for secure communication over SSL. This certificate is not to be confused with a certificate associated with the DS8700 Storage Facility Image, which is created later (see 4.1.3, “Configuring the DS8700 Storage Facility Image” on page 58)

   The window shown in Figure 4-8 opens.

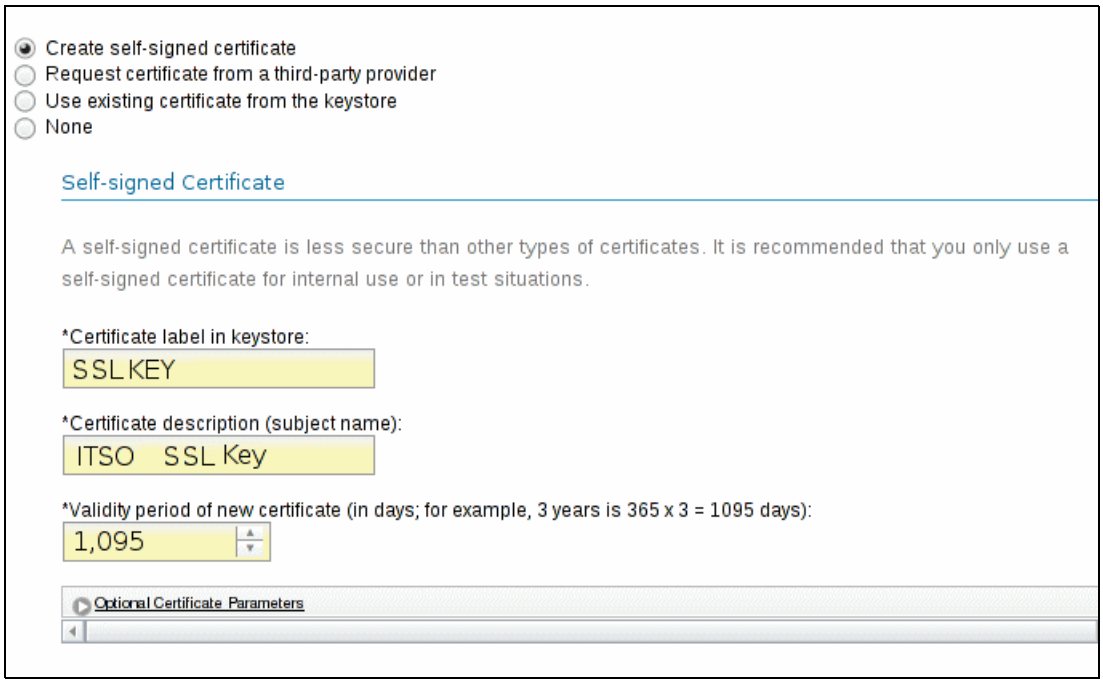


Figure 4-8 Create certificate

5. Select **Create self-signed certificate**. Using a third-party signed certificate is also supported. We could use an existing certificate from the keystore, but using a certificate that is used for encrypting disk data to also protect the communication protocol is *not* good practice. Choose a descriptive label and a certificate expiration that follows your security guidelines. Optional certificate parameters can also be entered.
6. Click **OK**.

   The Configuration window shown in Figure 4-9 opens.

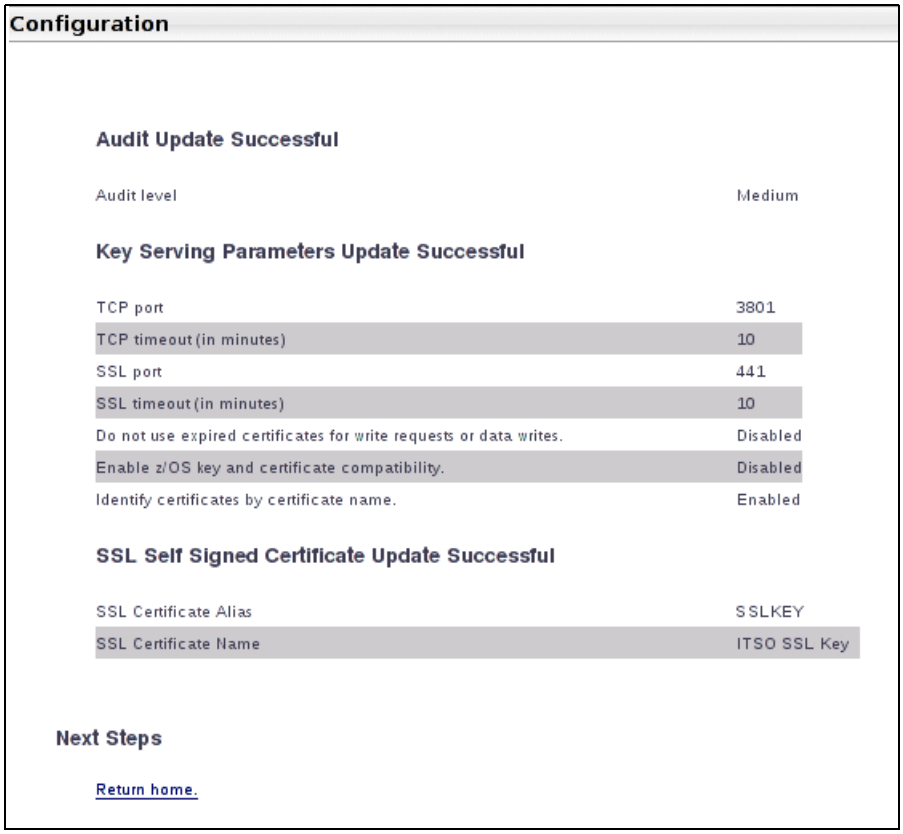


Figure 4-9 Key serving parameters update successful

   As indicated, the SSL certificate creation was successful.

7. Click **Return home**.

You are returned to the Welcome window (Figure 4-10).

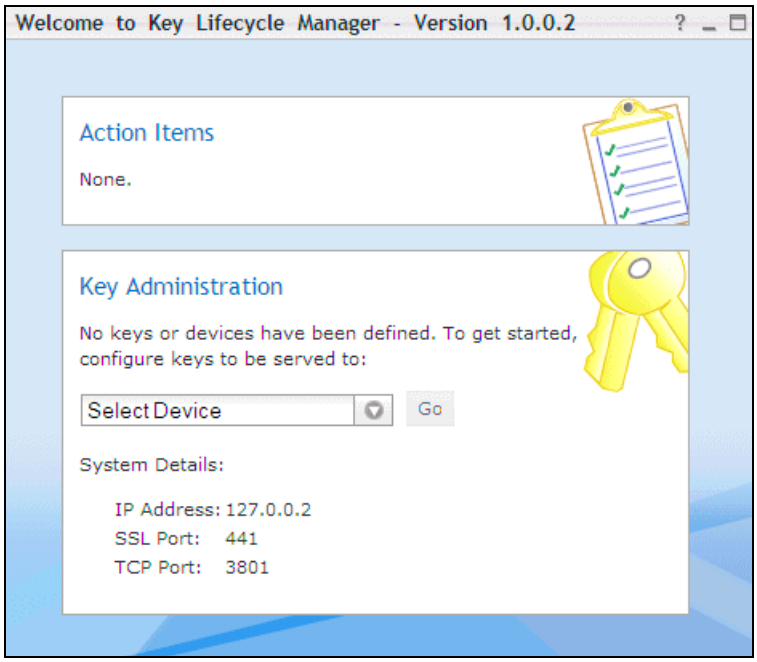


Figure 4-10 Welcome Key Lifecycle Manager screen

The Tivoli Key Lifecycle Manager master keystore and the SSL certificate have now been created.

4.1.3 Configuring the DS8700 Storage Facility Image

The next step in the configuration sequence is to create the image certificates necessary for associating with the Storage Facility Images (SFIs) that you are about to identify:

1. Under Key Administration, select **DS8000** in the drop-down menu, as shown in Figure 4-11.

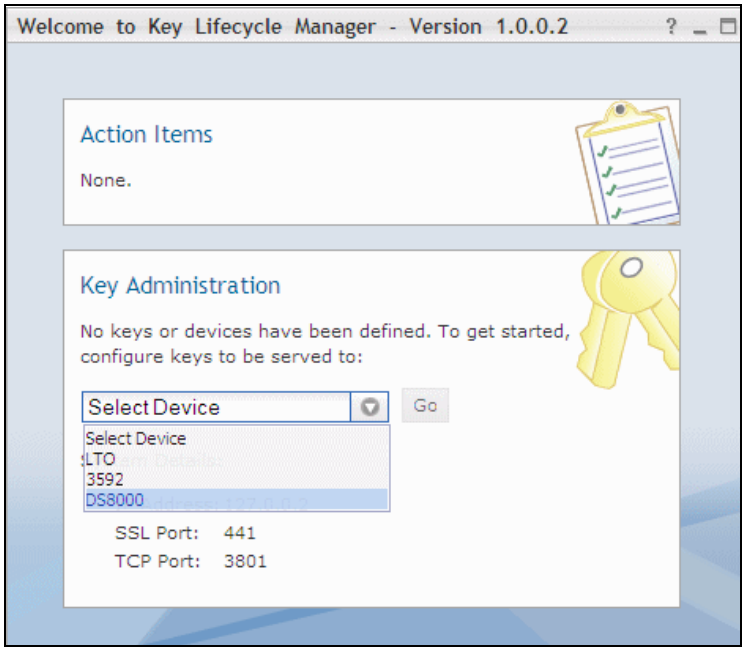


Figure 4-11 Configure SFI: Select DS8000

2. Click **Go**.

   The DS8000 Drive page opens to Step 1, as shown in Figure 4-12 on page 59. It provides a guided set of configuration steps. (Note that DS8000 Drive is a misnomer; it is not a drive but actually the DS8000 Storage Facility Image that is in context here.)

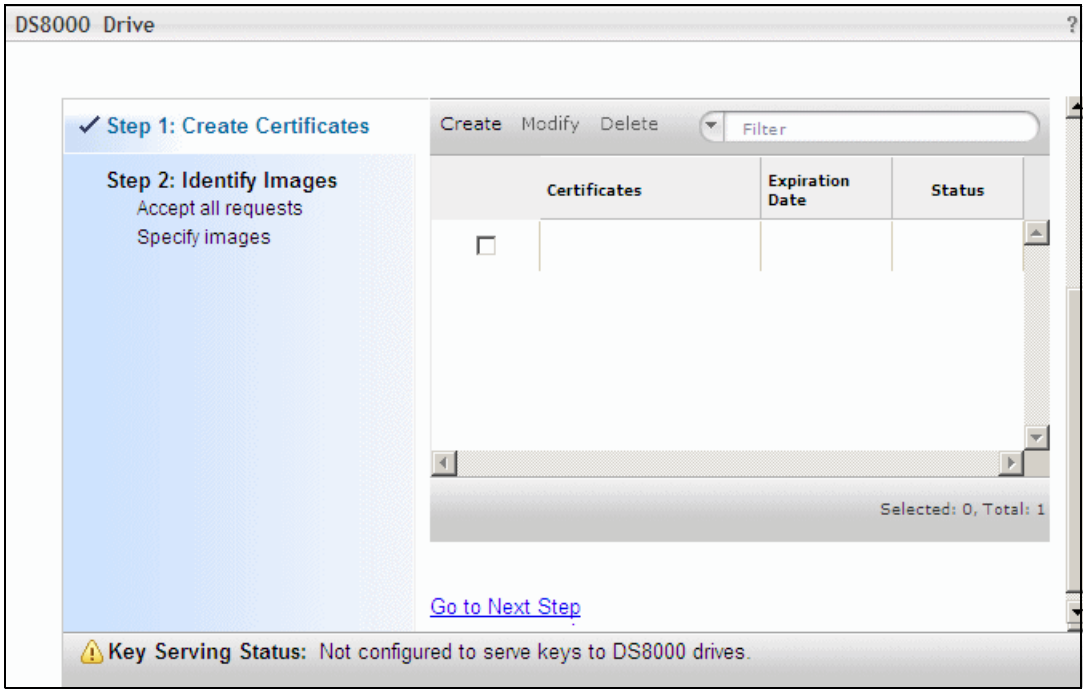


Figure 4-12 Create Certificate, Step 1

3. Under Step 1 (Create Certificates), click **Create** to create the certificate for your DS8700. The Create Certificate dialog shown in Figure 4-13 is displayed. Fill in the fields as appropriate and click **OK** (not shown in the figure).

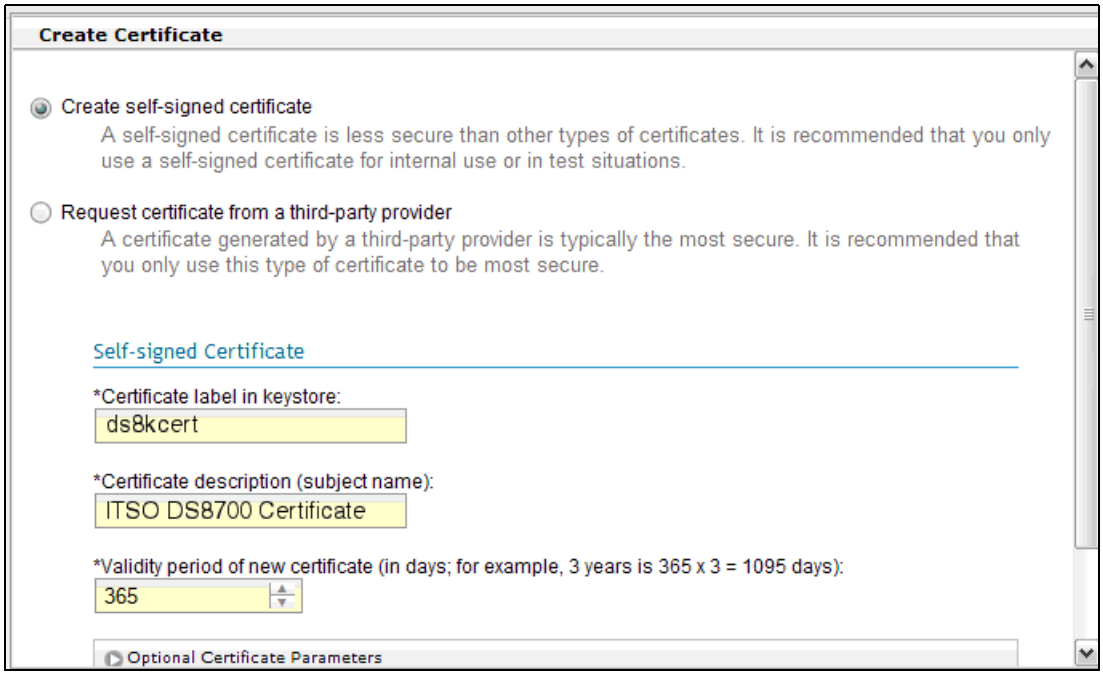


Figure 4-13 Create DS800 certificate

4. You are back to the dialog shown in Figure 4-12. Continue by clicking either **Go to Next Step** or **Step 2: Identify Images**.

The Step 2 window shown in Figure 4-14 opens.

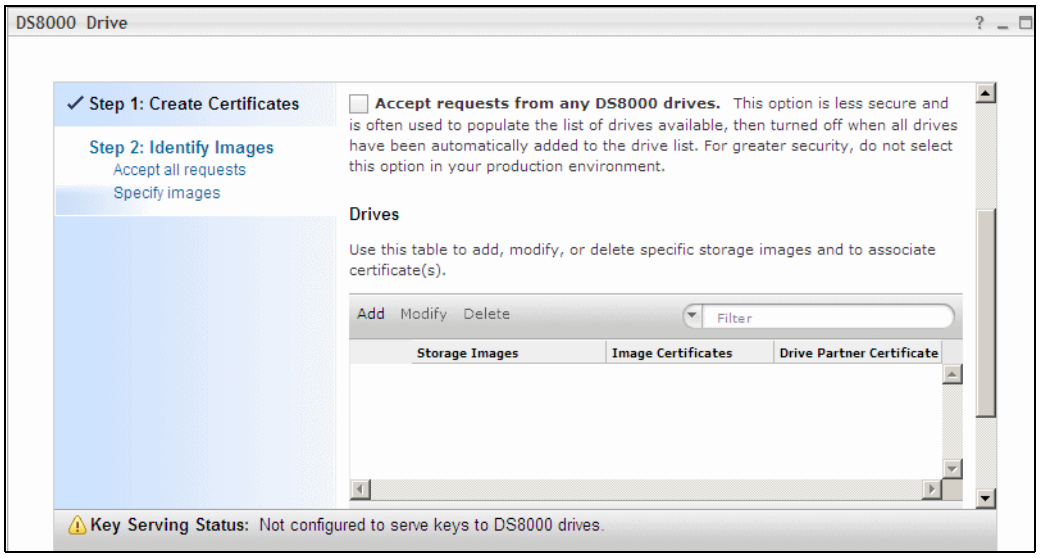


Figure 4-14 Identify storage images, Step 2

Although you can specify that *Tivoli Key Lifecycle Manager* accept requests from any DS8700 systems (drives), do not use it in a production environment; it is less secure.

5. In the Drives table, click **Add**.

   The Add Storage Image dialog, shown in Figure 4-15, is displayed.

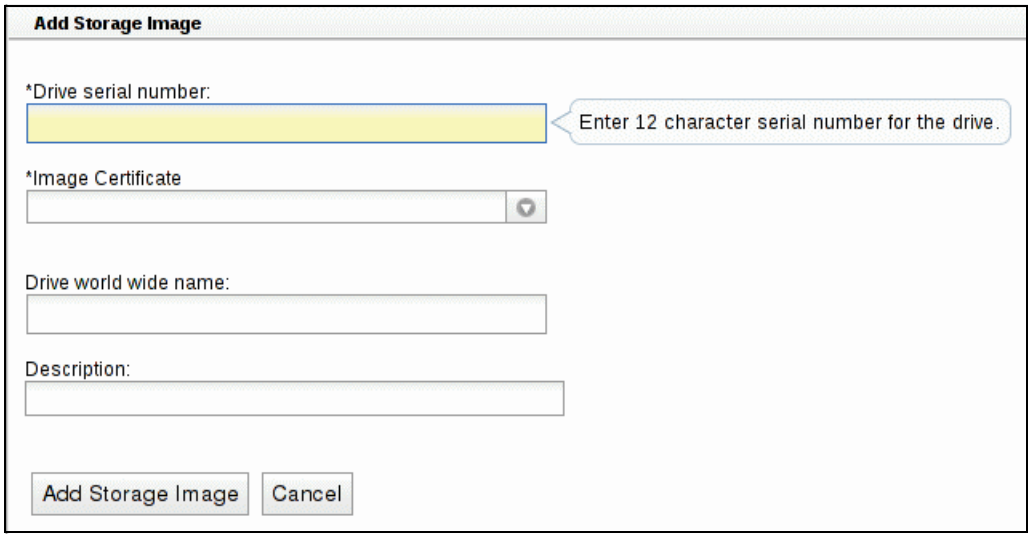


Figure 4-15 Add Storage Image

6. In the Drive serial number field, enter the DS8700 machine type and the Storage Facility Image (SFI) ID. An example of a SFI ID is 2107-75XXXXX.
7. From the Image Certificate drop-down menu, select the certificate that was created earlier (refer to Figure 4-8 on page 55, in the field named Certificate label in keystore).

   We used the example shown in Figure 4-16 on page 61.

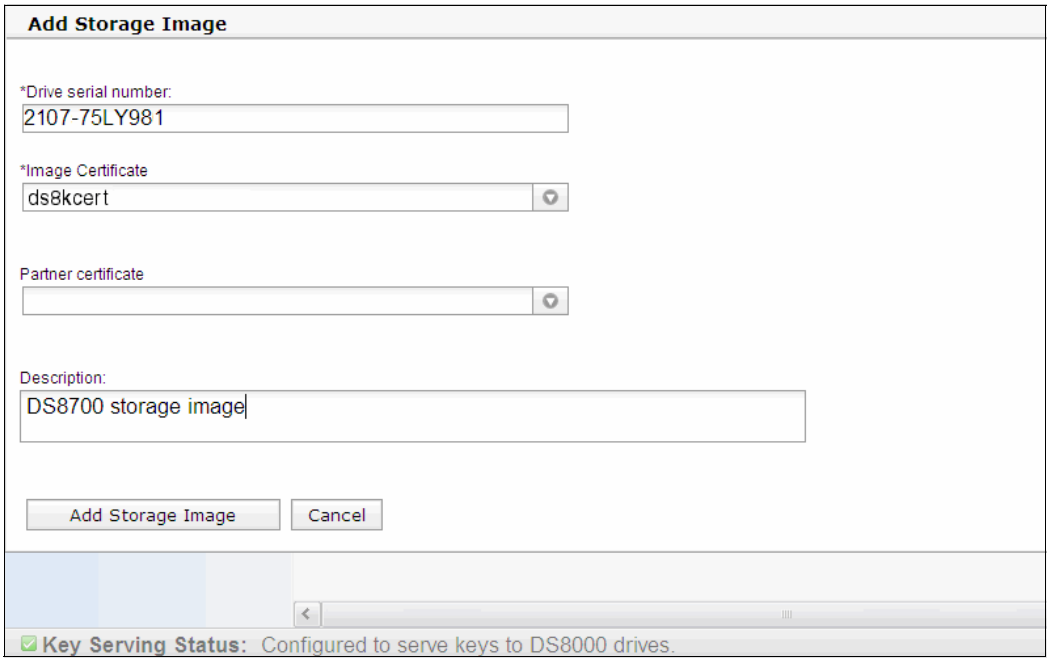


Figure 4-16 Add DS8700 and its Image Certificate

Partner Certificate field is applicable only for dual-platform support configurations. Refer to 5.4, “Tivoli Key Lifecycle Manager dual platform support implementation” on page 107, for more information.

8. Click **Add Storage Image** to continue. The window shown in the Figure 4-17 opens. The storage image is added to the Drives table.

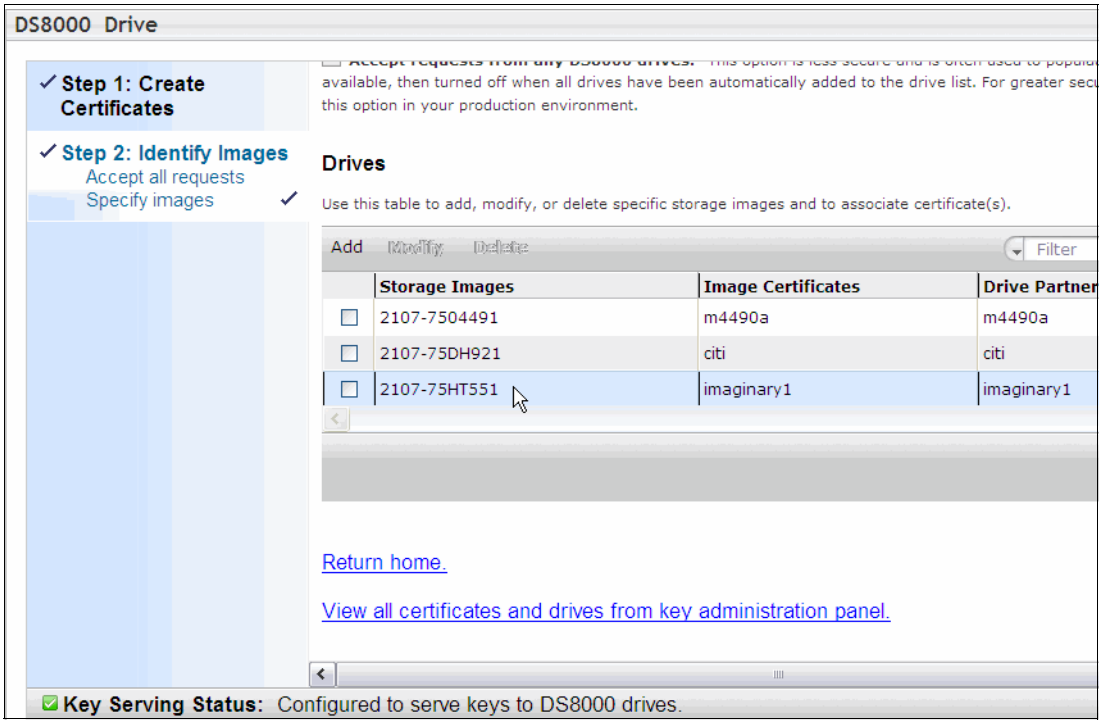


Figure 4-17 Storage image is added

Now the Key Serving Status is configured. If you click the **Return home** link at the bottom of the window, the main Welcome window opens again (Figure 4-18), which shows the configured state also.

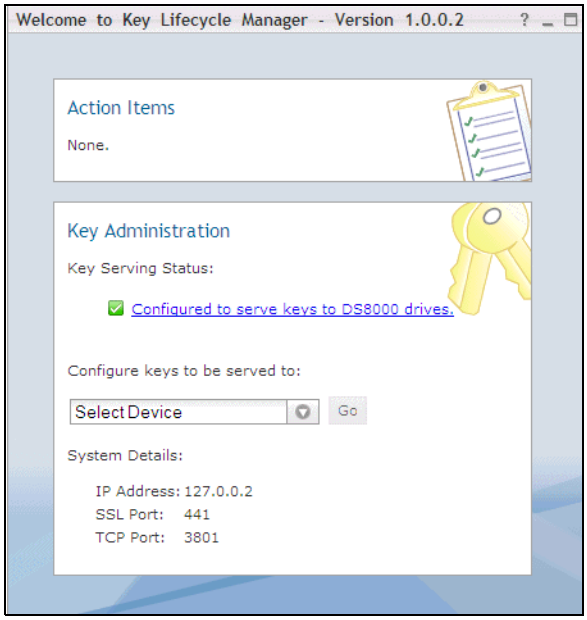


Figure 4-18 DS8700 Storage Image added

4.2 DS8700 GUI configuration for encryption

This section explains how to configure disk encryption on the DS8700 Storage Server, using the Storage Manager GUI. The high-level configuration sequence is as follows:

1. Configure the Tivoli Key Lifecycle Manager server connection to the DS8700.
2. Assign storage and security administrators.
3. Manage recovery key.
4. Configure the Encryption Group.
5. Apply activation key.
6. Configure and administer encrypted arrays.
7. Configure and administer encrypted ranks.
8. Configure and administer encrypted extent pools.

4.2.1 Configuring Tivoli Key Lifecycle Manager server connection to DS8700

The DS8700 supports up to four Tivoli Key Lifecycle Manager key server ports.

The following considerations apply to configurations:

- In multiple site configurations, at least two of the key server ports should be assigned to isolated key servers at separate physical sites. The remaining ports can be connected to general key servers.
- In single site configurations, at least two of the key server ports should be assigned to isolated key servers at the same site.

The DS8700 configuration for encryption also requires that at least two active key servers be connected and defined at the DS8700.

The DS8700 monitors all configured key servers. Customer notification is provided for loss of access to key servers and other key-sever-related errors through DS8700 customer notification mechanisms (SNMP traps and email if configured), as follows:

- Loss of access to key servers is reported at five-minute intervals.
- Loss of the ability for at least two key servers to provide key services that would prevent access to the data on the DS8700 is reported at eight-hour intervals.
- The inability of any one key server to provide key services that would prevent access to data on the DS8700 is also reported at eight-hour intervals.

To configure a Tivoli Key Lifecycle Manager server connection, perform the following steps from the DS8700 SM GUI:

1. Select **Manage Hardware**.
2. Select **Encryption**
3. Select **Key Servers.** Figure 4-19 shows the Encryption Key Servers window.

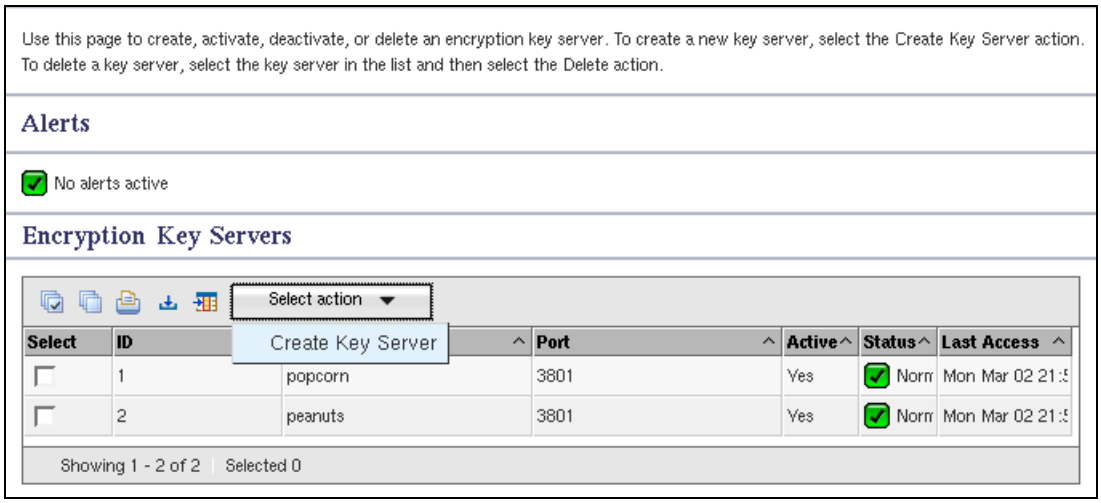


Figure 4-19 Select Create Key Server

4. Click **Select action** → **Create Key Server**. The Create Encryption Key Server dialog box opens, as shown in Figure 4-20 on page 64.

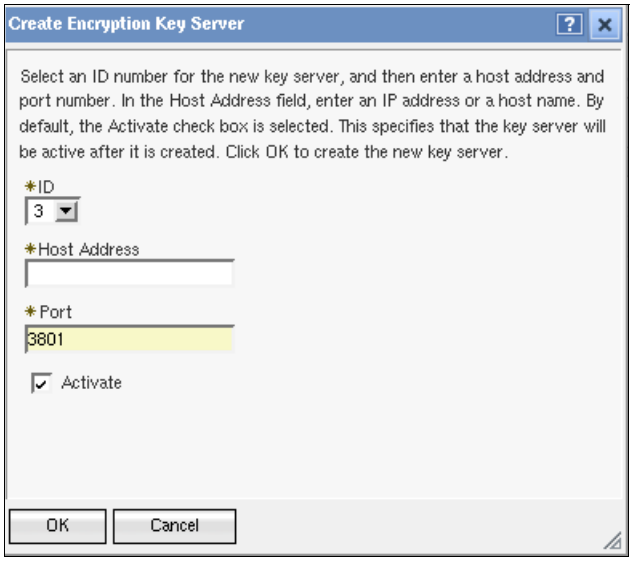


Figure 4-20 Create Encryption Key Server

5. Select the ID, specify the Host Address (IP address or name of the Tivoli Key Lifecycle Manager key server), and enter, in the Port field, the TCP Port, which is displayed in the Welcome to Key Lifecycle Manager window (in Figure 4-18 on page 62). Click **OK**. Now, the encryption key server is configured.

   Note: The TCP port can also be changed.

6. Repeat the steps to register the additional key servers (remember that at least two must be configured).

4.2.2 Assigning storage and security administrators

The DS8700 includes an internal authentication and authorization service, called the Basic authentication service. This service also provides user management and is identical to the service that is provided in other DS8700 models. The DS8700 allows you to use an external authentication service, such as a LDAP server, but still uses the internal authorization service to grant access to resources as defined by the DS8700 user group roles of admin, op_storage, op_volume, op_copy_services, service, and monitor.

An authentication policy defines the authentication service to be used, any parameters that are required to connect to and utilize that authentication service, and any mappings from that service's user groups to the DS8700 user group roles. An assumption is that any external authentication service will provide tools for user account management, therefore the DS8700 only manages user accounts defined in the Basic authentication service. Currently, the DS8700 defines two policy types, *Basic* and *SAS* (Storage Authentication Service). The Basic policy type uses the DS8700 Basic authentication service, and the SAS policy type uses the authentication service provided by an IBM System Storage Productivity Center (IBM SSPC).

With the introduction of the encryption recovery key on DS8700, a *dual control* security process is required to protect the authorized usage of the recovery key. This dual control process requires two separate user accounts to process most recovery commands. If these accounts are owned by two separate people, the recovery key cannot be used by any one person to gain access to encrypted data.

The first user role is the same *admin* user group role that exists in DS8100 and DS8300 models, but in the DS8700 it is now referred to as a *storage administrator*. The second user role, *secadmin*, is referred to as a *security administrator*.

Both users are created on DS8700 by default and you must assign these roles to two separate individuals. To work with user administration and modify default user names, sign on to the DS8700 GUI. From the selection menu on the left, click **Monitor System** → **User Administration.** A window opens in which you can select a storage complex to display all defined user IDs (Figure 4-21).

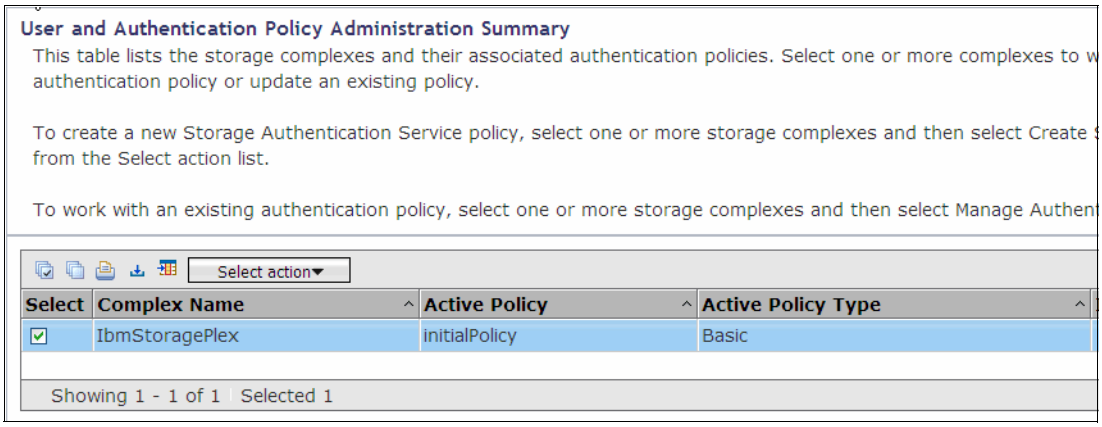


Figure 4-21 User and Authentication Policy Administration Summary

Select the storage complex, and then click **Select action** → **Manage Authentication Policy**, as shown in Figure 4-22.

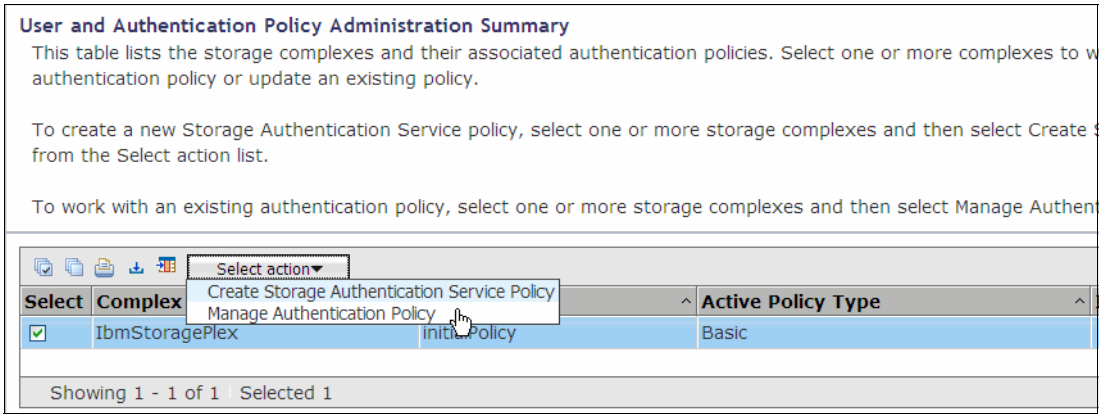


Figure 4-22 Select Manage Authentication Policy

Select your active policy, and then click **Select action** → **Properties** as shown in Figure 4-23.

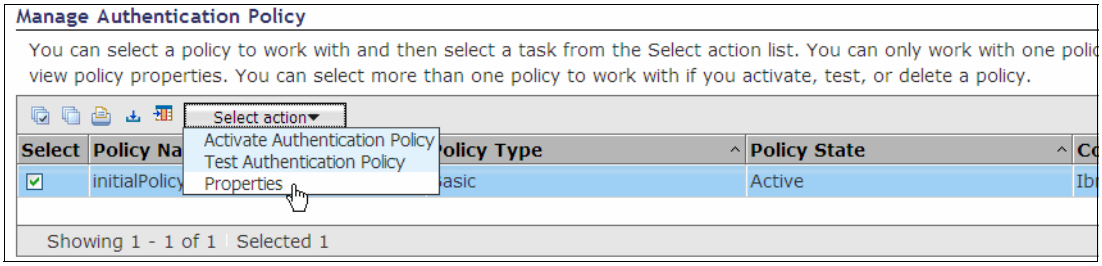


Figure 4-23 Select Authentication Policy Properties

The new window (Figure 4-24) lists all defined users.

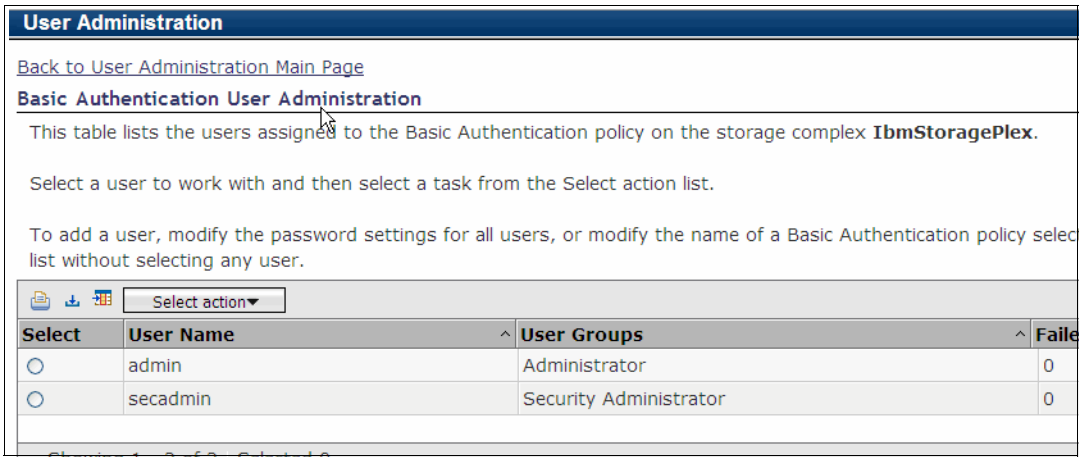


Figure 4-24 User Administration window: Default user roles are admin and secadmin

You can now select a user ID to work with, and then click **Select action** and select an action.

> **Note:** The initial admin password is *admin* and the secadmin password is *secadmin*. The first time you log in, you have to change the password to a new one. Because these users are owned by separate people, the admin and secadmin passwords must not be stored in one place.

Any user that has Security Administrator authority cannot have the authority of any other user role, and a user with any other user role cannot have the Security Administrator authority at the same time.

The secadmin user can create only new users with Security Administrator authority. As shown in Figure 4-25, all other authorities are disabled if you are logged in as the secadmin user.

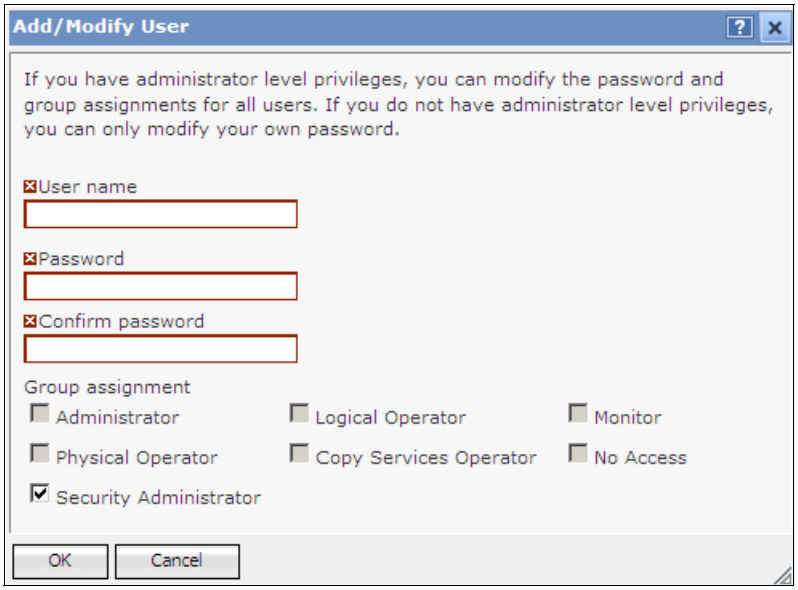


Figure 4-25 Add new user as secadmin user

Unlike secadmin user, the admin user can create users with all authorities except the Security Administrator role (see Figure 4-26).

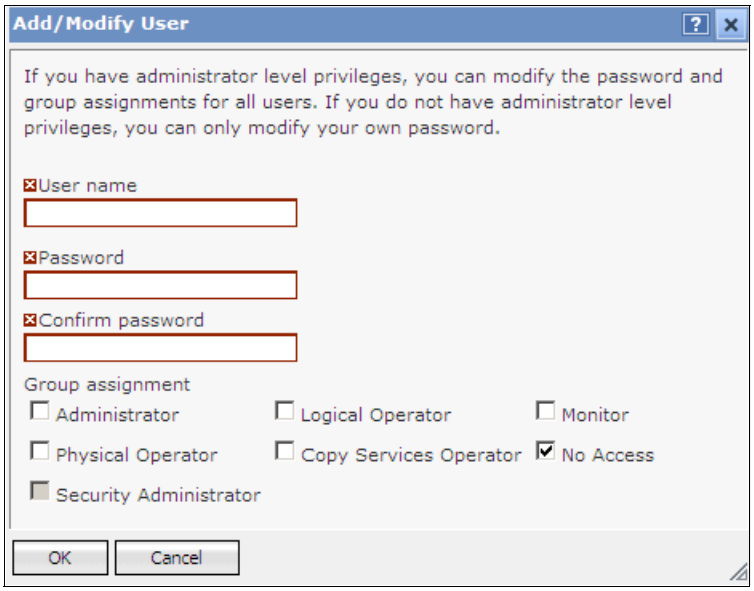


Figure 4-26 Add new user as admin user

4.2.3 Managing recovery key

Whenever an encryption technology is applied, a new kind of risk appears: the *deadlock* situation. This situation happens for example when the DS8700 cannot obtain a required data key from the Tivoli Key Lifecycle Manager server because of Tivoli Key Lifecycle Manager server or communication failure. As a consequence, all data on the DS8700 becomes inaccessible because without the keys, the data cannot be decrypted anymore.

The risk of a deadlock situation can be highly minimized by maintaining redundant (dual platform) Tivoli Key Lifecycle Manager servers, but cannot be eliminated. The *recovery key* (RK) feature provides a way to get out of a deadlock state.

Starting with the Licensed Internal Code level 65.10.xx.xx, the DS8700 offers the option to enable or disable the recovery key. Disabling the recovery key management means that you understand that DS8700 data become unrecoverable if you (permanently) lose access to the Tivoli Key Lifecycle Manager severs. However, an organization might want to disable a recovery key to further enhance data security and help satisfy Payment Card Industry (PCI) security standards, for instance.

The choice to enable or disable the recovery key management has to be taken at this stage.

Recovery key enabling

To create a recovery key, perform the following steps:

1. Log in to the DS8700 Storage Manager GUI as a user with security administrator privileges.
2. Navigate to the **Manage Hardware** → **Encryption** → **Groups** window, and then click **Select action** → **Create Recovery Key**, as shown in Figure 4-27 on page 68.

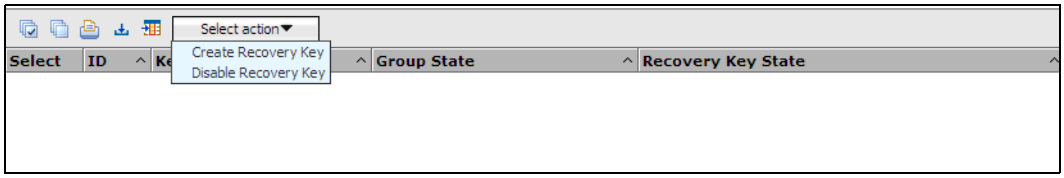


Figure 4-27 Create Recovery Key

The Create Recovery Key window opens (Figure 4-28).

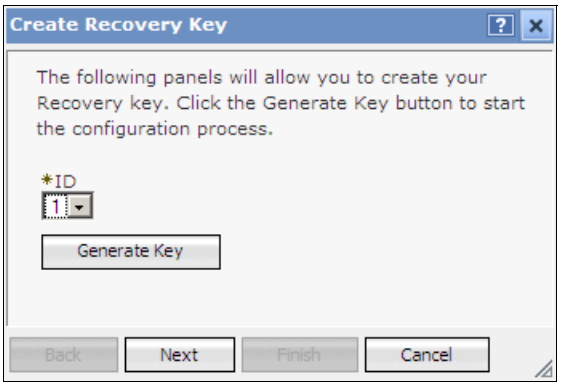


Figure 4-28 Create Recovery Key window

Currently only one Encryption Group is supported, which is already selected in the **ID** list.

3. Click **Generate Key** to continue.

   After a few seconds, the recently generated recovery key is displayed as shown in Figure 4-29. It is displayed as 64 hexadecimal characters with dashes between every four characters.

Figure 4-29 New recovery key is generated

Note: If you use the `mkreckey` DS CLI command, you can copy the new recovery key text from the terminal and save it in a file, which can be used for printing. However, we do not recommend this way, because the key can be intercepted in the network; a better approach is to write down the key onto a piece of paper directly from the screen.

The security administrator is responsible for writing down the recovery key and storing it in a safe place.

4. To ensure that the written text is correct, type it into the field of the next window (Figure 4-30) for verification.

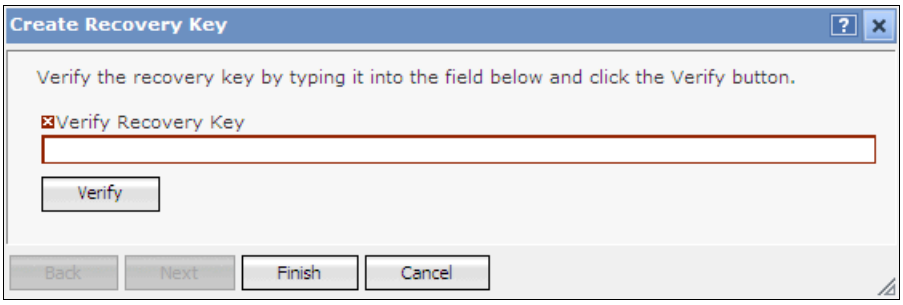


Figure 4-30 Recovery key verification

If the verification process passed, a confirmation text appears in the same window; see Figure 4-31.

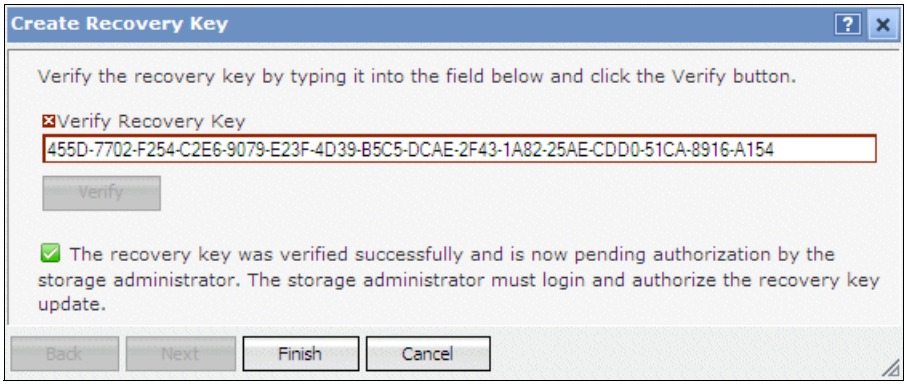


Figure 4-31 Recovery key verification passed

5. Click **Finish** to close the window. Now the recovery key has been created, but it is waiting for the storage administrator approval; see Figure 4-32.

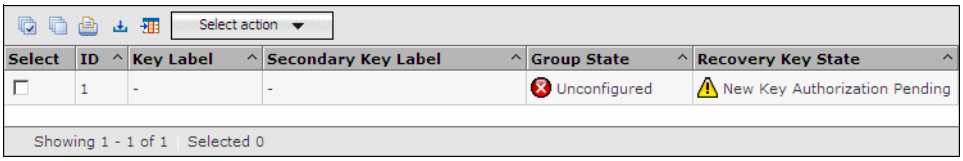


Figure 4-32 Authorization pending status

6. Log in as storage administrator and navigate to the **Manage Hardware** → **Encryption** → **Groups** window. Click **Select action** → **Authorize Recovery Key Update**, as shown in Figure 4-33.

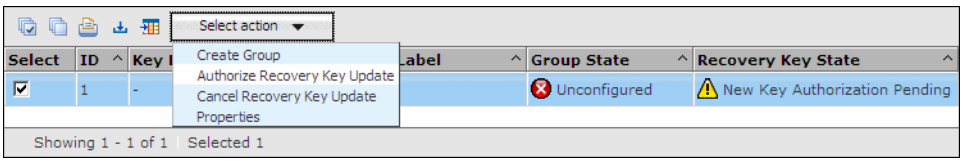


Figure 4-33 Authorize Recovery Key Update

The Authorize Recovery Key Update window opens (Figure 4-34).

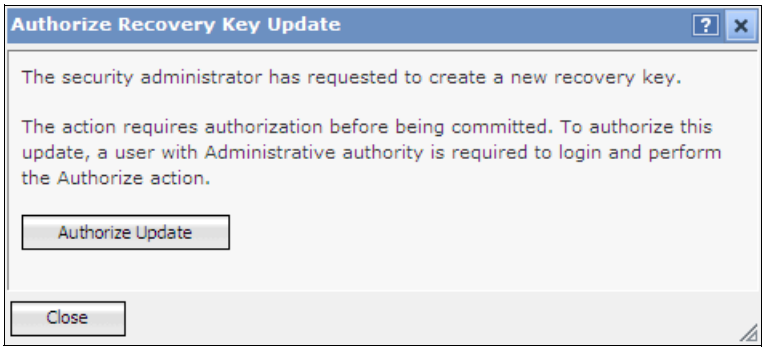


Figure 4-34 Authorize Recovery Key Update window

7. If you agree with it, click **Authorize Update** to complete the process (Figure 4-35).

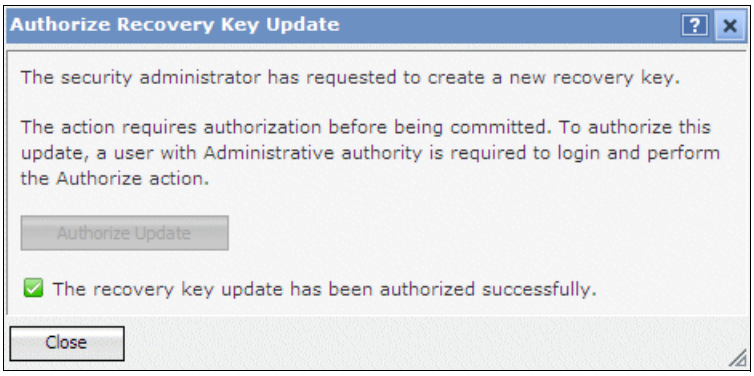


Figure 4-35 The recovery key update is authorized successfully

Now, the new recovery key has been activated and can be used in the future. In the Encryption Group panel, we can see it is in **Configured** status as shown in Figure 4-36.

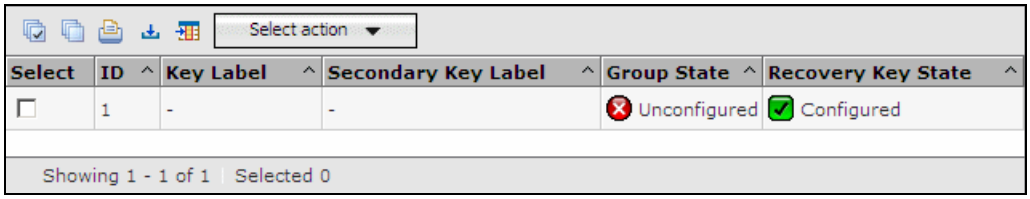


Figure 4-36 Configured recovery key

At this stage, the recovery key is configured and you can proceed to configure (or create) encryption groups, described in 4.2.4, “Creating the encryption group” on page 73. (However, if you want to disable the recovery key, see “Recovery key disabling” on page 71.) After you create the Encryption group, the recovery key Group State changes from the **Unconfigured** to **Accessible** state.

Recovery key disabling

To disable the recovery key, perform the following steps:

1. Log in to the DS8700 Storage Manager GUI as a user with security administrator privileges.
2. Navigate to the **Manage Hardware** → **Encryption** → **Groups** window, and then click **Select action** → **Disable Recovery Key**, as shown in Figure 4-37.

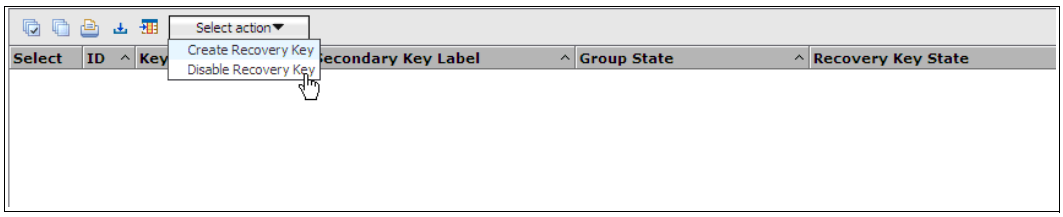


Figure 4-37 Disable Recovery Key

The Disable Recovery Key window opens (Figure 4-38).

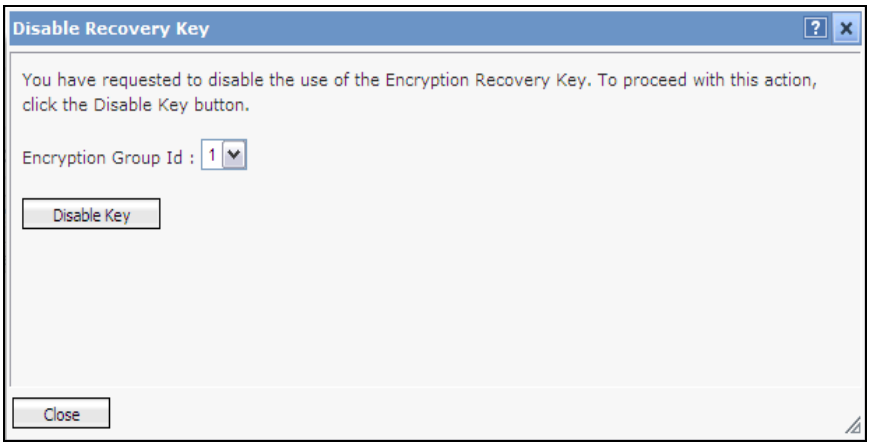


Figure 4-38 Disable Recovery Key window

3. Click **Disable Key** to continue. Now the recovery key disabling has been requested, but it is waiting for the storage administrator approval; see Figure 4-39.

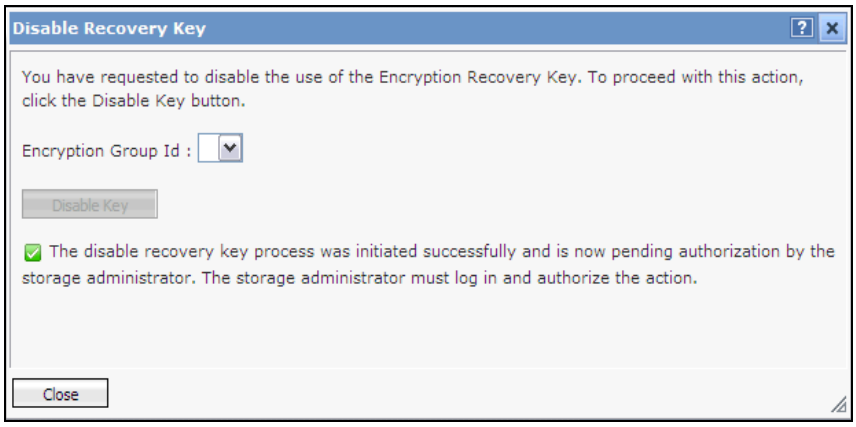


Figure 4-39 Authorization pending status

4. Log in as storage administrator and navigate to the **Manage Hardware** → **Encryption** → **Groups** window. Click **Select action** → **Authorize Recovery Key Update**, as shown in Figure 4-40.

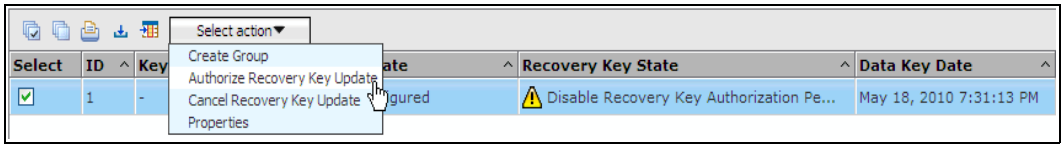


Figure 4-40 Authorize Recovery Key Update

The Authorize Recovery Key Update window opens (Figure 4-41).

Figure 4-41 Authorize Recovery Key Update window

5. Click **Authorize Update** to complete the process (Figure 4-42).

Figure 4-42 The recovery key update is authorized successfully

Now, the recovery key has been disabled. In the Encryption Group panel we can see it is in **Disabled** status as shown in Figure 4-43.

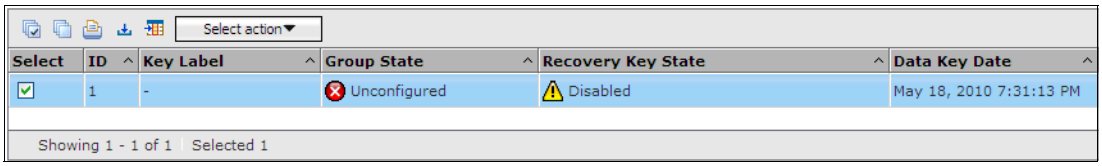


Figure 4-43 Disabled recovery key

At this stage, the recovery key is disabled and you may proceed with the next step, 4.2.4, “Creating the encryption group” on page 73. After you create the Encryption group, the recovery key Group State changes from the **Unconfigured** to **Accessible** state.

4.2.4 Creating the encryption group

The customer data within the Storage Facility Image that is encrypted is partitioned in one encryption group.The encryption group that contains encrypted data is enabled to access data through one data key obtained from a key server.

Note: Currently, the DS8700 supports only one encryption group.

An encryption group contains a set of extent pools, each of which has a set of associated ranks and volumes. IBM FlashCopy® and Remote Mirror and Copy functions can migrate data within or between encryption groups.

To configure the encryption group on the DS8700, perform the following steps:

1. Navigate to the **Manage Hardware** → **Encryption** → **Groups** window, and then click **Select action** → **Create Group**, (Figure 4-44).

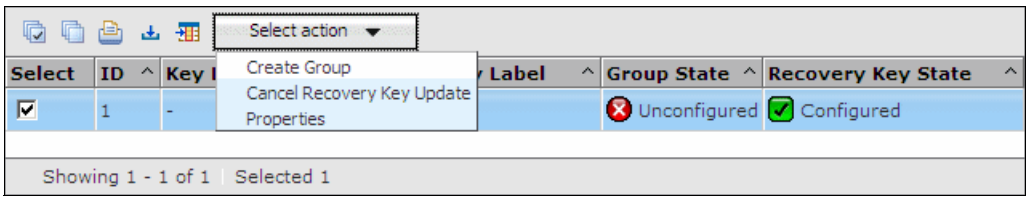


Figure 4-44 Create Encryption Groups

The Create Encryption Group window opens (Figure 4-45).

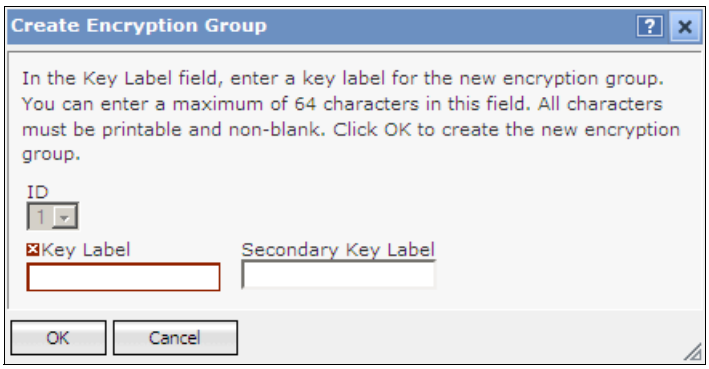


Figure 4-45 Create Encryption Groups: Add Key Label

2. Depending on how many labels you have, fill the Key Label fields accordingly. Enter the key label, which was configured during the Tivoli Key Lifecycle Manager server configuration (see Figure 4-8 on page 55, **Certificate label in keystore**), then click **OK**. In a single key label mode, leave the Secondary Key Label field empty. Dual key label is applicable when you have two Tivoli Key Lifecycle Manager servers installed on separate platforms with a different keystore type and security mode. More information about Tivoli

Key Lifecycle Manager dual platform support can be found in 5.4, “Tivoli Key Lifecycle Manager dual platform support implementation” on page 107.

Encryption Group creation takes approximately one minute. The status indicates “In progress” (Figure 4-46).

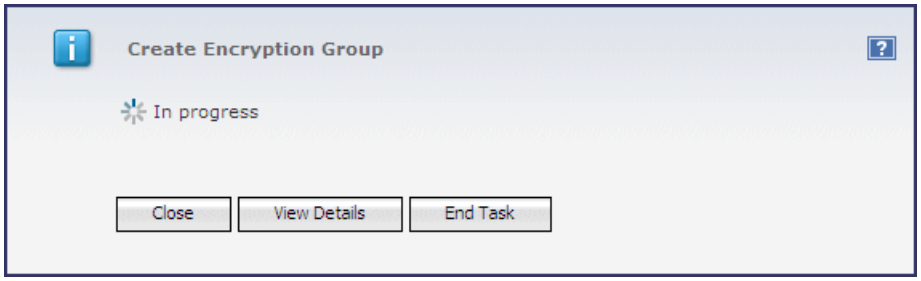


Figure 4-46 Create Encryption Group is in progress

3. When the Finished Completed message is displayed (Figure 4-47), click **Close**.

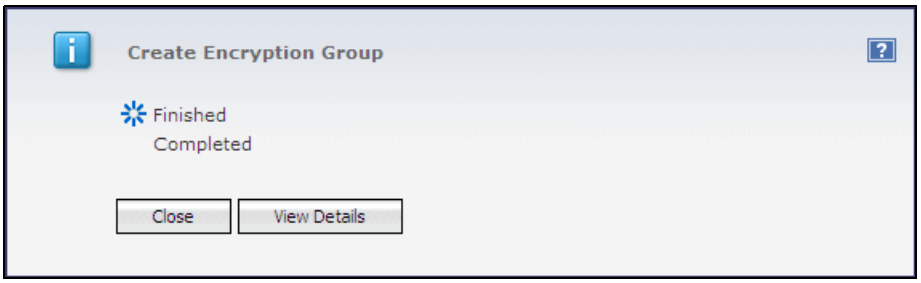


Figure 4-47 Create Encryption Group is completed

4. Verify in the Encryption Group panel that the Group State indicates Accessible, as shown in Figure 4-48. In our example, we have defined a Secondary Key Label because we have two Tivoli Key Lifecycle Manager servers installed on separate platforms. One Tivoli Key Lifecycle Manager server is on z/OS and the other on a Linux server.

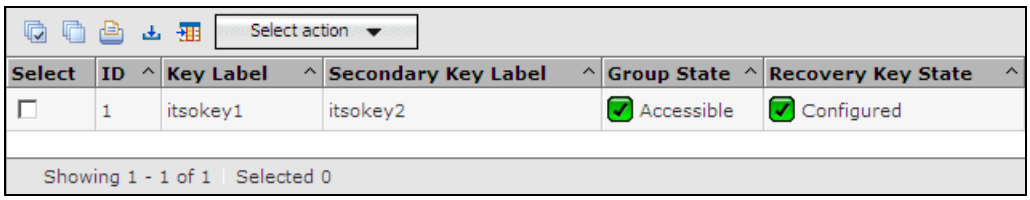


Figure 4-48 Configured Encryption Group in dual key mode

With the Licensed Internal Code level 65.10.xx or later, the rekey data key option is available on DS8700. This option allows a user with the storage administrator role to rekey the data key labels for a encryption group. See 5.5, “Rekey data key” on page 118 for further details.

4.2.5 Applying the encryption activation key

Now that the Tivoli Key Lifecycle Manager key servers have been set up and are registered with the DS8700, the encryption feature activation key must be applied, before the logical configuration (defining ranks and extent pools) can take place.

Important: Remember that the Encryption Authorization Licensed Internal Code feature key *cannot* be obtained from the Disk Storage Feature Activation (DSFA) website.

To apply the activation codes, perform the following steps:

1. Log in to the DS8700 Storage Manager GUI as a user with administrator privileges.
2. In the My Work navigation window on the DS8700 Storage Manager Welcome window, select **Manage hardware** → **Storage Complexes**, and then click **Select action** → **Apply Activation Codes**, as shown in Figure 4-49.

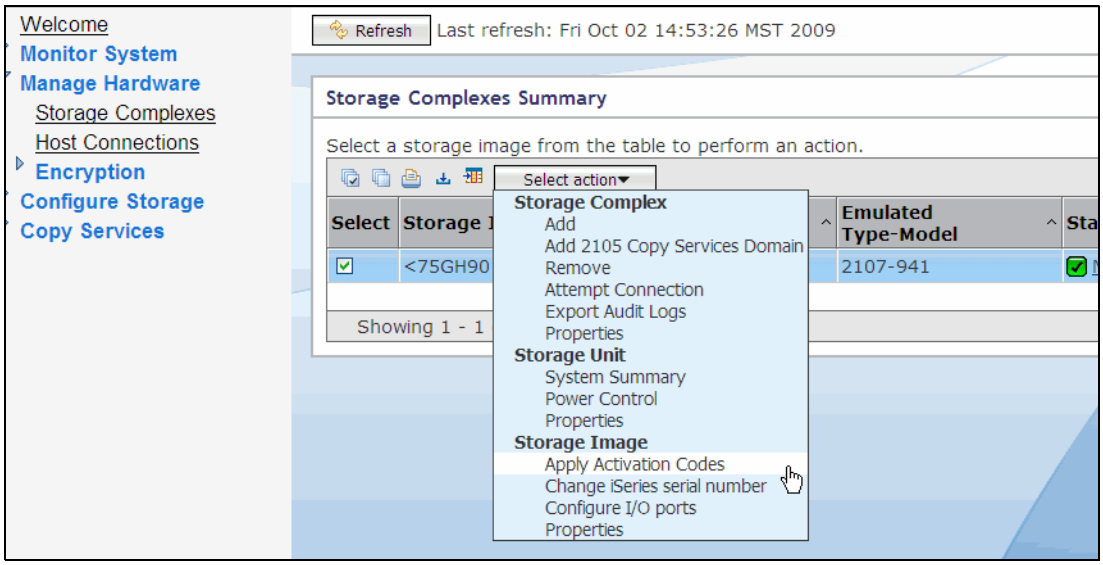


Figure 4-49 DS8700 Storage Complexes selection window

3. The Apply Activation Codes window opens (Figure 4-50 on page 76). You have an option to manually add an activation key by clicking **Select action** → **Add Activation Key**. The other option is to select **Import Key File** in case you have it in a file.

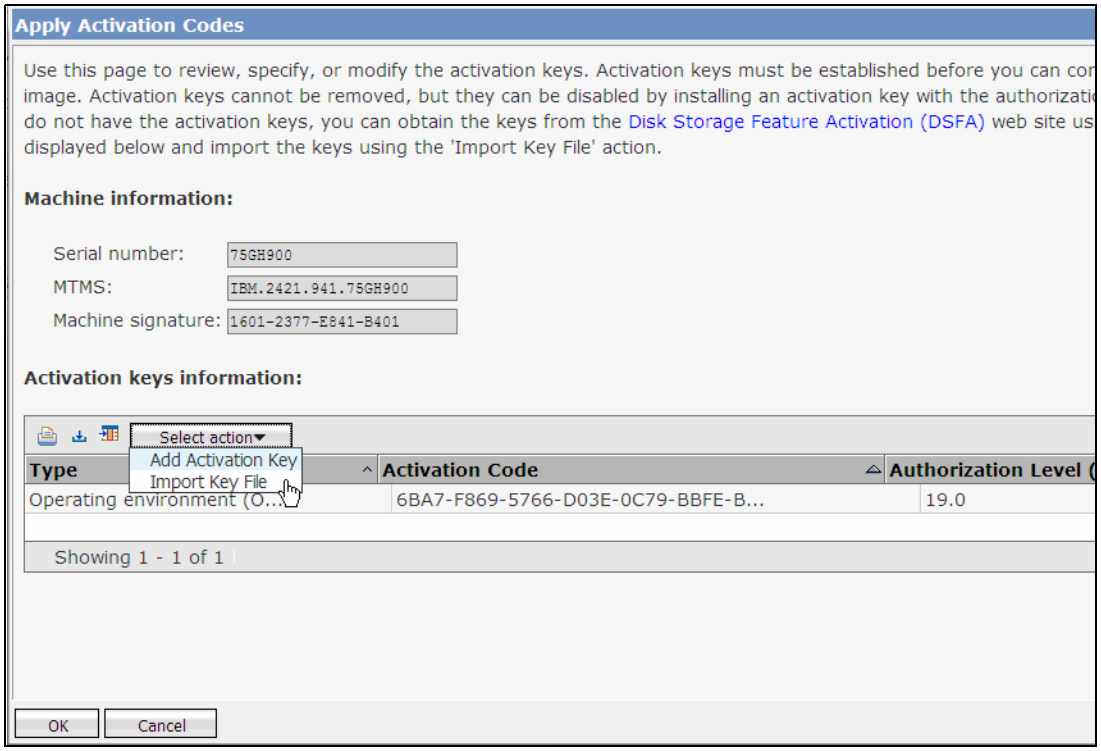


Figure 4-50 DS8700 Storage Manager: Apply Activation Codes window

4. The Import window opens; see Figure 4-51. Enter the name of the key file, then click **OK** to complete the import process.

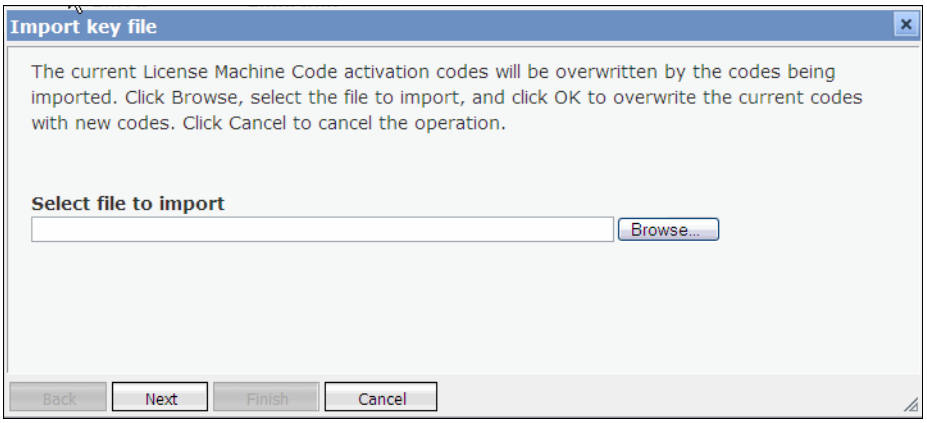


Figure 4-51 DS8700 import key file window

5. After selecting the file, click **Next** to continue. The Confirmation window displays the key name. Click **Finish** to complete the new key activation procedure. See Figure 4-52 on page 77.

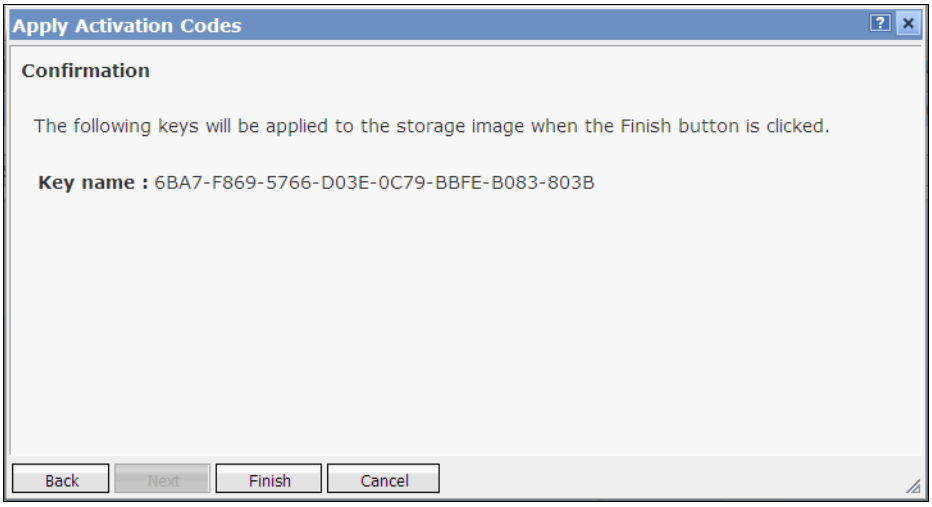


Figure 4-52 DS8700 Storage Manager: Apply Activation Codes, Confirmation window

Your encryption activation license should be listed in the Apply Activation Code window.

4.2.6 Configuring and administering encrypted arrays

> **Tip:** Creating arrays first and then ranks is not necessary. Proceeding directly with the creation of Extent Pools is possible and is explained in 4.2.8, “Configuring and administering encrypted extent pools” on page 82.

The creation of arrays is based on the array sites that are associated with the storage unit.

To create an array, perform the following steps in the DS8700 GUI:

1. Log in as a user with administrator privileges.
2. In the My Work navigation window on the DS8700 Storage Manager Welcome window, select **Configure Storage** → **Disk Configuration**.
3. Select the **Arrays** tab in the Manage Disk Configuration frame.
4. Click **Select action** → **Create Arrays**, as highlighted in Figure 4-53.

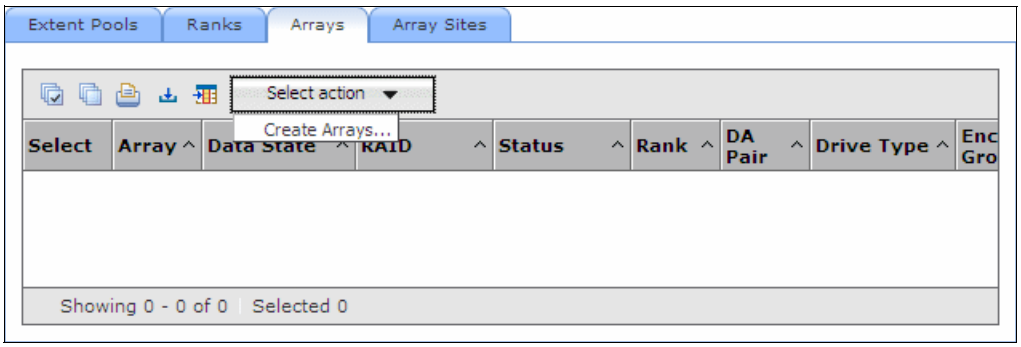


Figure 4-53 Starting the Create Arrays function

The Create New Arrays window opens (Figure 4-54 on page 78).

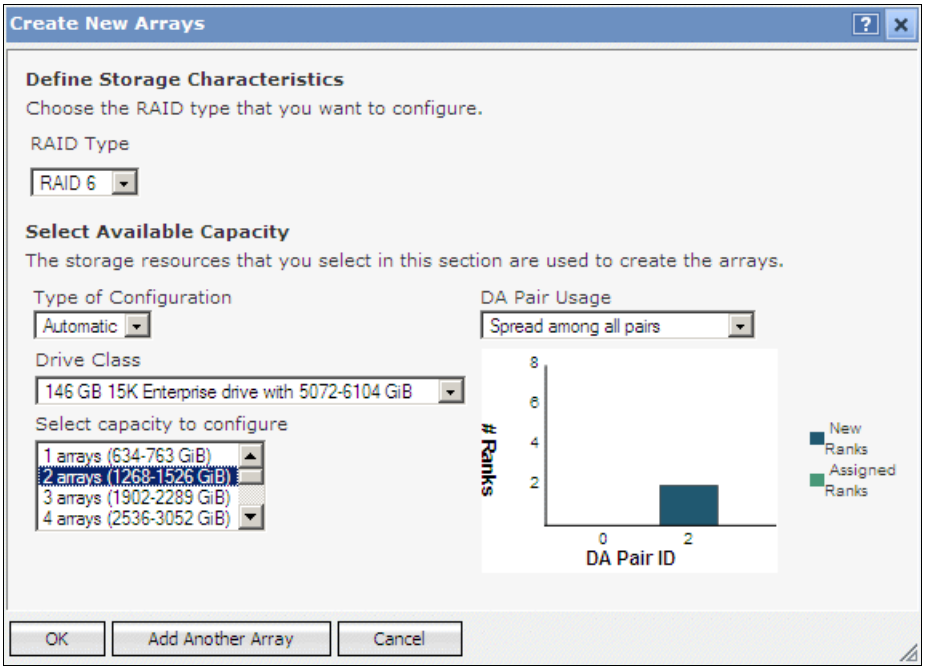


Figure 4-54 Create New Arrays window: Automatic Configuration mode

5. Specify the RAID Type you want to use. The supported RAID types are: RAID5, RAID6, and RAID10.
6. Specify the way you create arrays by selecting one of the options from the Type of Configuration list:
   – **Automatic**: In this case you must select the **Drive Class** and the number of required arrays. The system then determines an optimal setup depending on the DA Pair Usage preference.
   – **Manual**: You get a list of available array sites, and you can select them on your own.
7. When you are ready, click **OK**.

   The next window gives an overview of the new arrays, as shown in Figure 4-55. Here you can **Cancel** the operation, if something is wrong. Otherwise, select all of the rows and click **Create All** to continue. In this example, we create two RAID6 arrays.

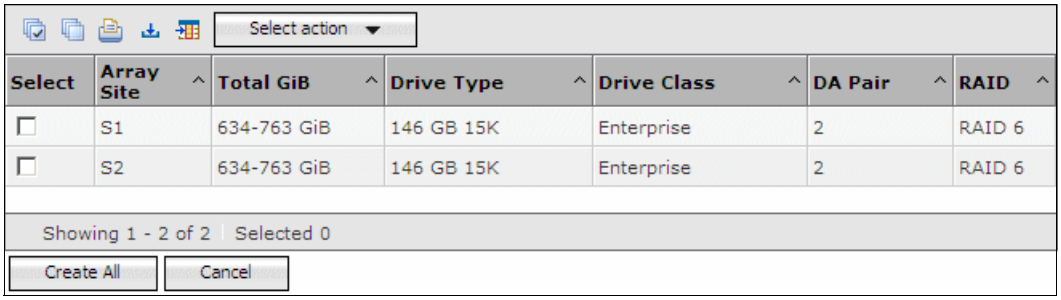


Figure 4-55 Overview of the arrays planned to create

8. Wait until the process is finished and then click **Close**, as shown in Figure 4-56 on page 79.

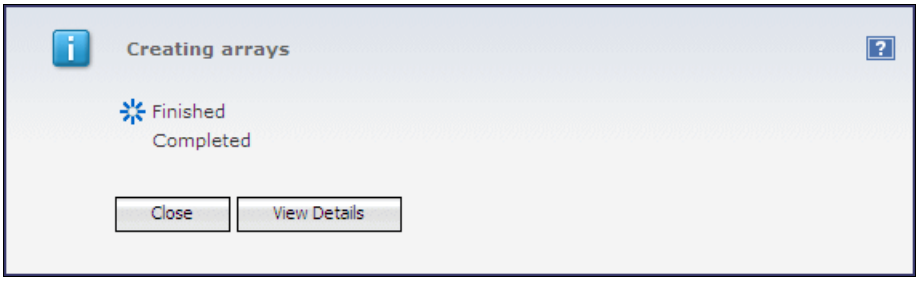
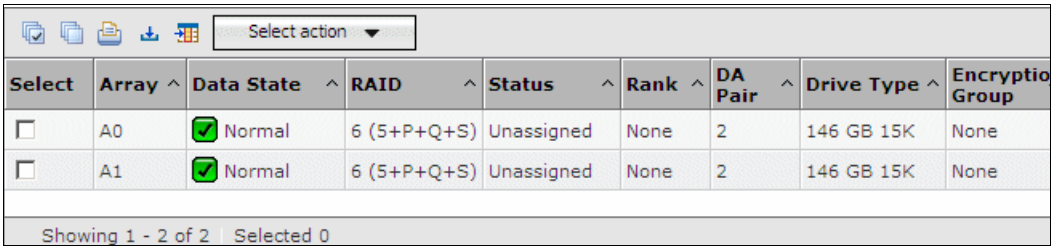


Figure 4-56 Creating arrays have been finished

9. Now that the arrays are ready, you can verify them. Open the Arrays tab again to see the results. In this example, the two arrays have been created successfully as shown in Figure 4-57.



Select	Array	Data State	RAID	Status	Rank	DA Pair	Drive Type	Encryptio Group
☐	A0	Normal	6 (5+P+Q+S)	Unassigned	None	2	146 GB 15K	None
☐	A1	Normal	6 (5+P+Q+S)	Unassigned	None	2	146 GB 15K	None

Showing 1 - 2 of 2 Selected 0

Figure 4-57 Verify the new arrays

In the Encryption Group column the state indicates None at this stage, because the encryption is defined at the rank level. This status changes after the rank is defined, as described in 4.2.7, “Configuring and administering encrypted ranks” on page 79.

4.2.7 Configuring and administering encrypted ranks

A rank is a logically contiguous storage space that is made up of one array. You can assign a rank to every unassigned array. A rank inherits the characteristics, including the RAID type, of its parent array and is given a storage type attribute of either FB (fixed block) or CKD (count key data).

The rank configuration state is unassigned until it is assigned to an extent pool. An *unassigned* rank is not associated with either rank group 0 or 1. Any unassigned rank can be assigned to an extent pool that is associated with either rank group 0 or 1.

You can assign a rank to an unassigned array and also assign the rank to an extent pool at the same time if you have already created the extent pools and the arrays. Creating extent pools first saves a step in the configuration.

To create a new encrypted rank, perform the following steps:

1. Log in as a user with administrator privileges.
2. In the My Work navigation window in the DS8700 Storage Manager Welcome window, select **Configure Storage** → **Disk Configuration**.
3. Select the **Ranks** tab in the Manage Disk Configuration frame.
4. Click **Select action** → **Create Ranks**, as highlighted in Figure 4-58 on page 80.

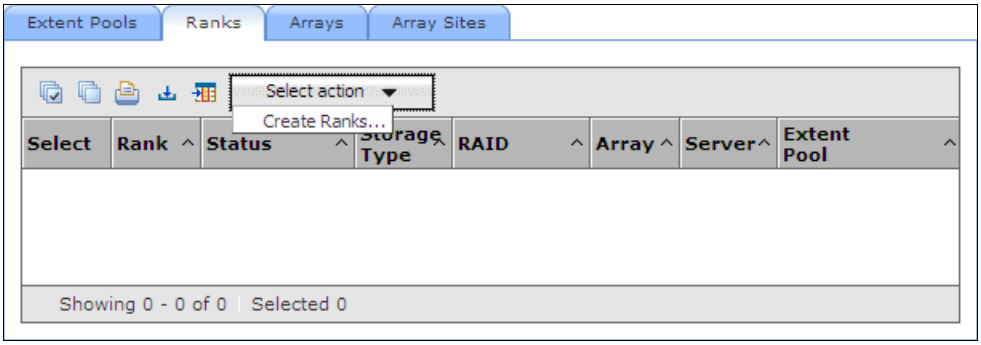


Figure 4-58 Starting the Create Ranks function

The Create New Ranks window opens, shown in Figure 4-59.

5. Specify the Storage Type (**FB** or **CKD**) and the RAID Type (**RAID5**, **RAID6**, or **RAID10**). If arrays already have been created in advance, use the same RAID type to utilize them, otherwise new arrays will be created.

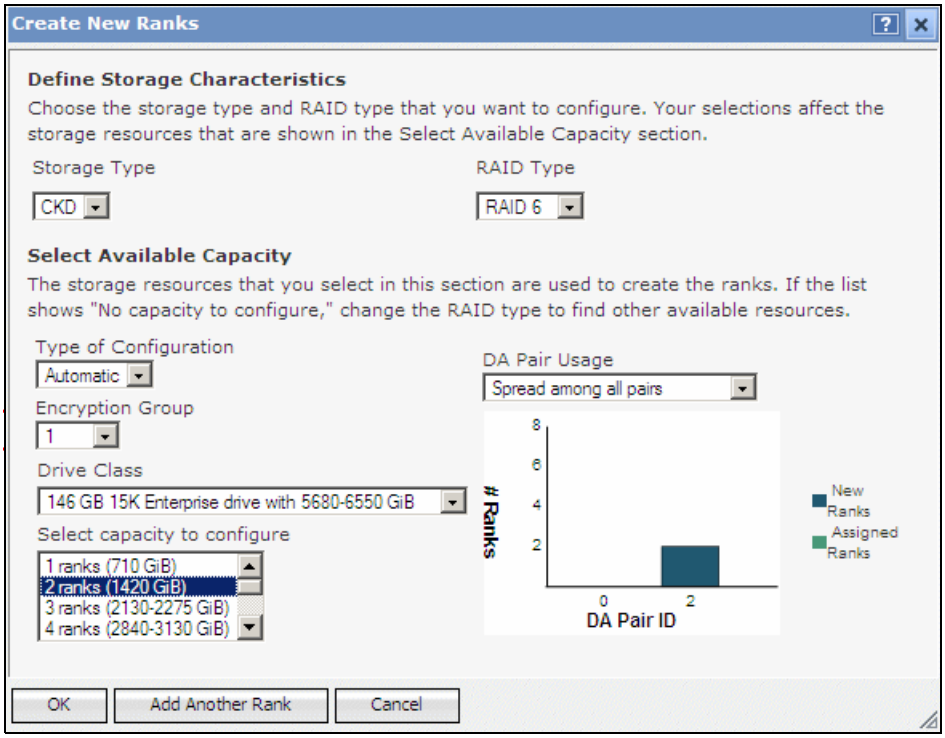


Figure 4-59 Create New Ranks window

6. Specify the way you create ranks by choosing one of the options from the Type of Configuration list:
   - **Automatic**: In this case, select the Drive Class and the number of required ranks, then the system determines an optimal setup depending on the DA Pair Usage preference.
   - **Manual**: You get a list of available ranks, and you can select them on your own.
7. Specify the **Encryption Group**. Currently the encryption intermix is not allowed, so you have to decide whether all the storage will be encrypted or not. When you create the first rank, you can enable or disable it, but further ranks must use the same setting as the first one, therefore the Encryption Group selection cannot be changed later on.

8. When you are ready, click **OK**.

   The next window gives an overview about the new ranks, as shown in Figure 4-60. Here we can **Cancel** the operation, if something is wrong. Otherwise, select all of the rows and click **Create All** to continue. In this example, we create two encrypted CKD ranks on our RAID6 arrays.

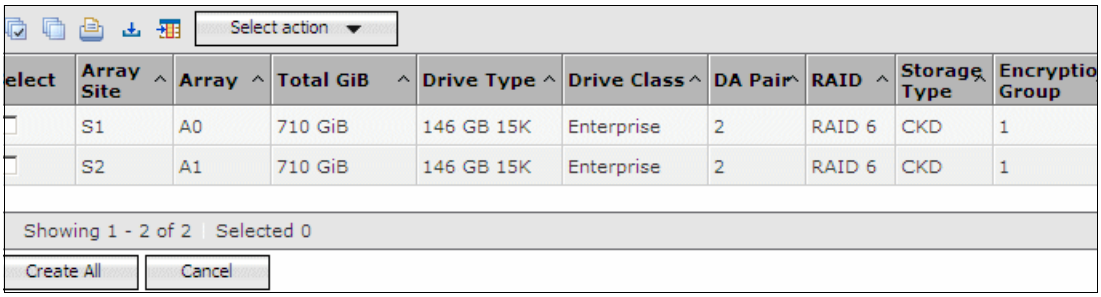


Figure 4-60 Overview of the ranks planned to create

9. Wait until it is finished. To follow the progress in real-time, go to **Monitor System** → **Long Running Task Summary** and select your running task to see the details, as shown in Figure 4-61.

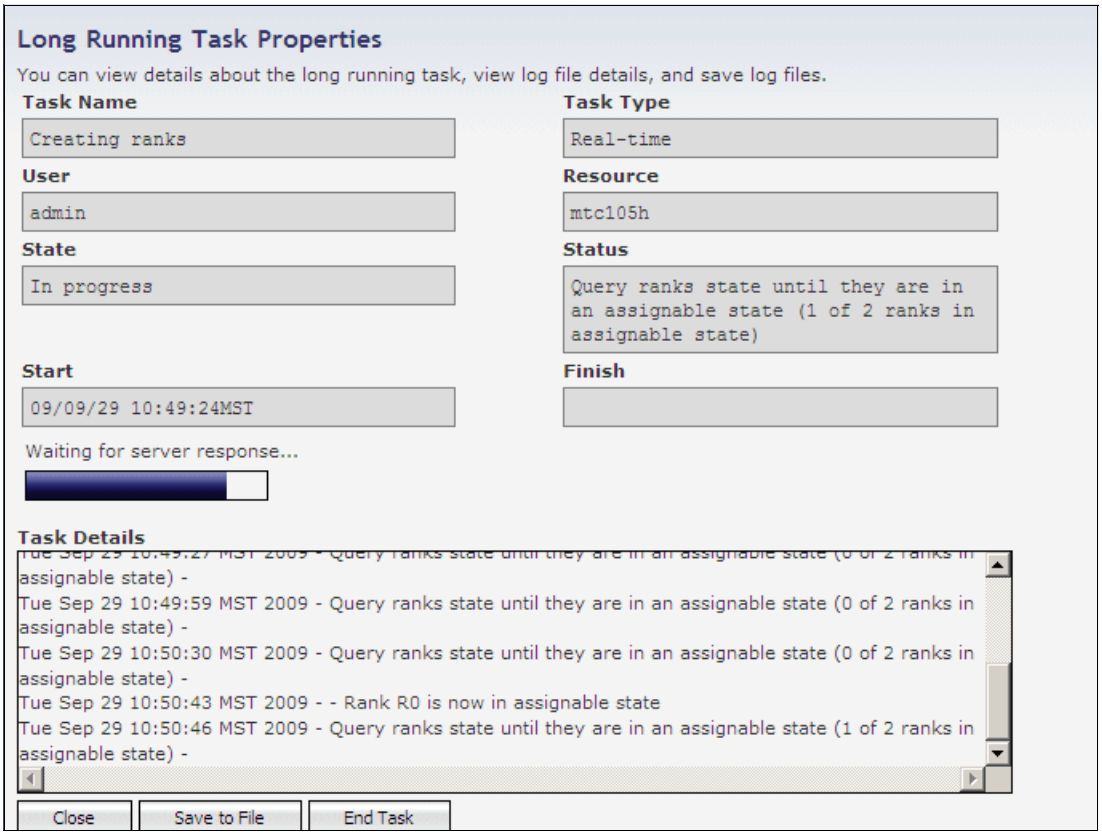


Figure 4-61 Creating ranks in progress

10. When the process is ready, you verify the new ranks under the Ranks tab. The Encryption Group status is 1, indicating that the encryption is enabled for listed ranks, as shown in Figure 4-62 on page 82.

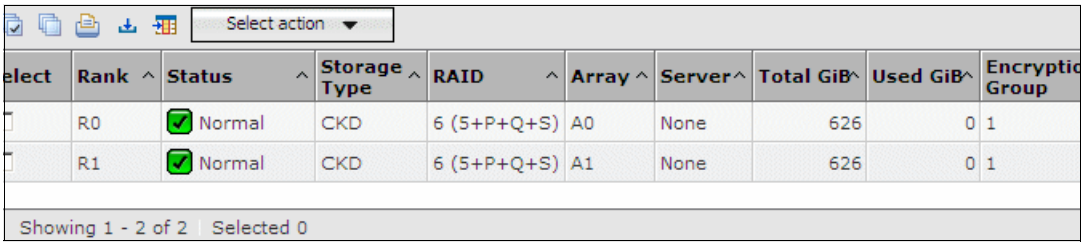


Figure 4-62 Verify the new ranks

The next step is to define encryption extent pool over previously created ranks.

4.2.8 Configuring and administering encrypted extent pools

The process of creating encrypted extent pools is similar to creating ranks. The dialog and screen sequence is the same. The only difference is in selecting ranks instead of arrays.

A good habit is to create the separate storage layers (arrays, ranks, extent pools) step by step. However, creating arrays and ranks before you start the extent pool configuration is not mandatory. If you select more ranks than already exist, the system will create them automatically based on the defined storage (FB or CKD) and RAID type. Use this shorter process only if the automatic rank and array assignment are suitable for you.

Each encrypted extent pool is defined with the rank group of 0 or 1. Extent pools that are defined for rank group 0 or 1 are assigned an even- or odd-numbered extent pool ID, respectively. Even-numbered extent pools are managed by storage server ID 0. Odd-numbered extent pools are managed by storage server ID 1.

Each rank is assigned to one extent pool. Therefore, storage server workload is affected by the rank assignments to even- and odd-numbered extent pool IDs. The best practice is to evenly distribute rank and extent pool allocations to keep the storage server workloads balanced. The *Automatic* configuration mode helps you set it in an optimal and balanced way.

To configure an extent pool, perform the following steps:

1. Log in as a user with administrator privileges.
2. In the My Work navigation window on the DS8700 Storage Manager Welcome window, select **Configure Storage** → **Disk Configuration**.
3. Select the **Extent pools** tab in the Manage Disk Configuration frame.
4. Click **Select action** → **Create Extent Pools (**Figure 4-63).

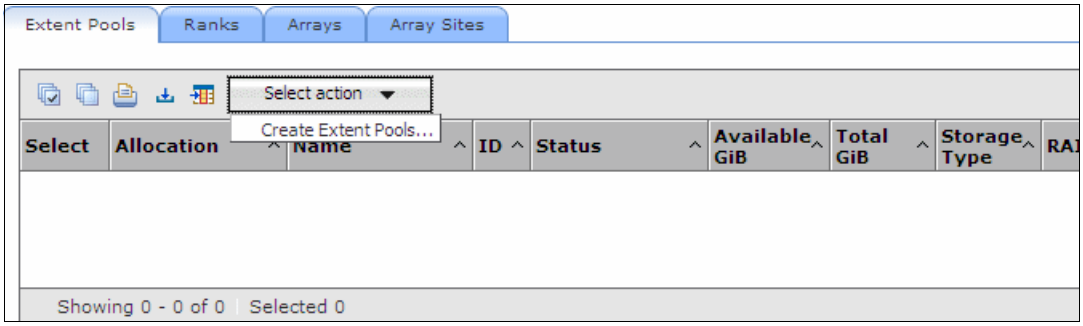


Figure 4-63 Select the Create Extent Pools function

The Create New Extent Pools window opens (Figure 4-64 on page 83).

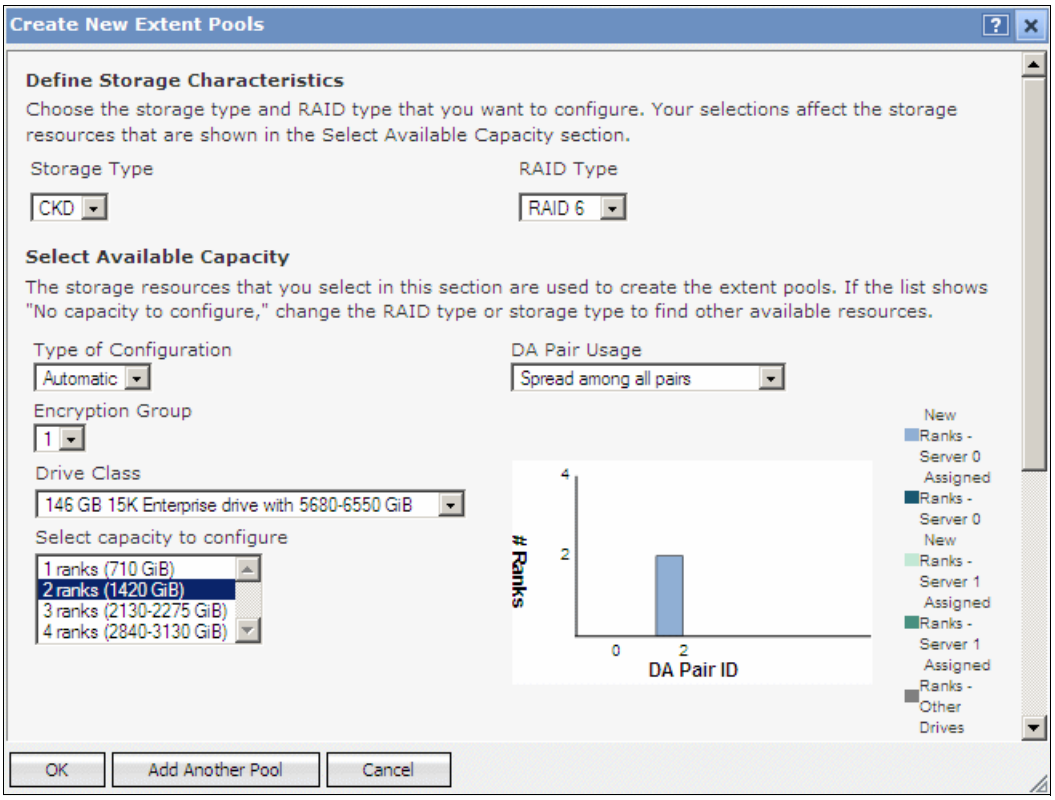


Figure 4-64 Create New Extent Pools window

5. Specify the Storage Type (**FB** or **CKD**) and the RAID Type (**RAID5**, **RAID6**, or **RAID10**). If ranks have been already created in advance, use the same RAID type to utilize them, otherwise new ranks will be created.
6. Specify the way you create extent pools by choosing one of the options from the Type of Configuration list:
   - **Automatic**: In this case you need to select the **Drive Class** and the number of required ranks, then the system determines an optimal setup depending on the **DA Pair Usage** preference. A small diagram shows the current and the new rank distributions.
   - **Manual**: You get a list of available ranks, and you can select them on your own.
7. Specify the **Encryption Group**. Currently the encryption intermix is not allowed, so if you already have some encrypted ranks, the extent pool must be encrypted also.
8. When you are ready, click **OK**.

   The next window gives an overview about the new extent pools, as shown in Figure 4-65 on page 84. Here you can **Cancel** the operation, if something is wrong. Otherwise select all of the rows and click **Create All** to continue. In this example, we create an encrypted extent pool for our existing RAID6 CKD ranks.

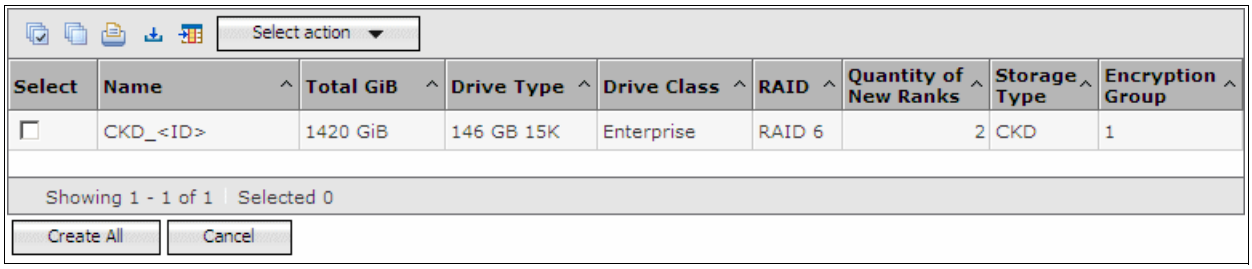


Figure 4-65 Overview of the extent pools planned to create

9. When the process is ready, you verify the new extent pools under the Extent Pools tab as shown in Figure 4-66.

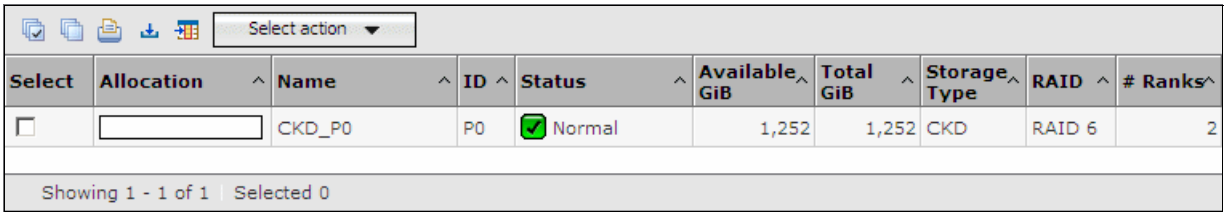


Figure 4-66 Verify the new extent pools

10. To verify that it is really encrypted, select the extent pool, and then click **Select action** → **Properties** to display more information about the extent pool. Under Drives and Volumes tab you can verify that the Encryption Group is 1, thus indicating that the encryption is enabled (Figure 4-67).

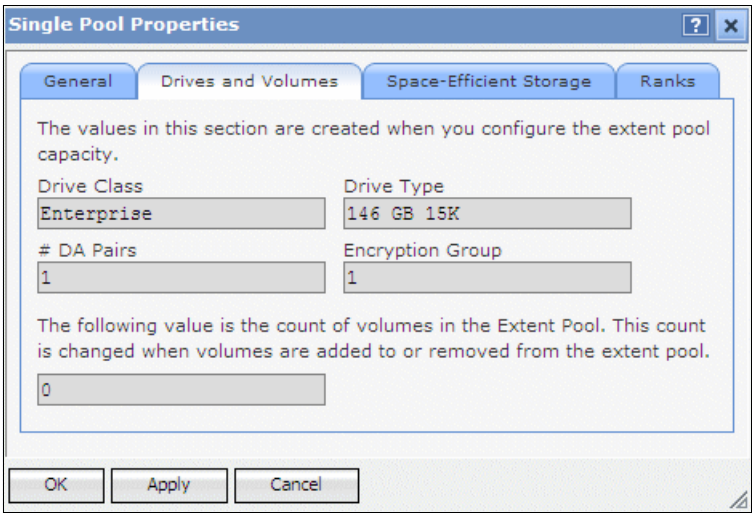


Figure 4-67 Single Pool Properties window

Now that the encrypted extent pools are ready, we can start creating encrypted volumes. Each volume has the same encryption properties as the parent extent pool and individual volume encryption property cannot be modified. After you create volumes, you can check the volume encryption status in the volume properties window. For open system volumes, go to Configure Storage Open Systems Volumes, then click the **Manage existing volumes** link in the Alert section. Select one volume, and then click **Select action** → **Properties**. The Single Volume Properties window opens and the Encryption Group status is displayed as shown in Figure 4-68 on page 85.

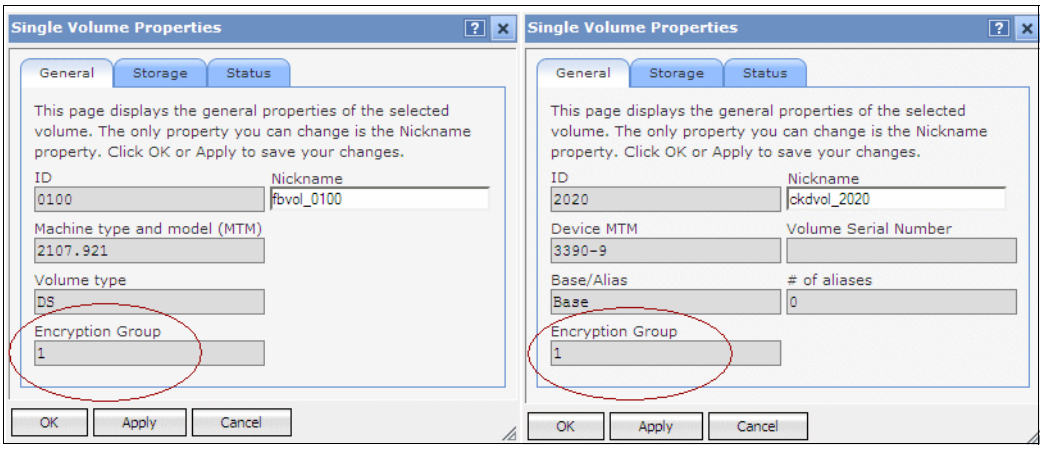


Figure 4-68 FB and CKD Volume Properties windows

4.3 DS8700 DS CLI configuration for encryption

This section explains how to configure disk encryption on the DS8700 Storage Server using the DS command line interface (DS CLI). The high-level configuration sequence is as follows:

1. Configure the Tivoli Key Lifecycle Manager server connection to the DS8700.
2. Configure recovery key.
3. Configure the encryption group.
4. Apply activation key.
5. Configure and administer encrypted arrays.
6. Configure and administer encrypted ranks.
7. Configure and administer encrypted extent pools.

Note: For detailed information about the command line interface and commands, refer to the *DS Command Line Interface User's Guide for the DS8000* at:

`http://publib.boulder.ibm.com/infocenter/dsichelp/ds8000ic/index.jsp`

4.3.1 Configuring the Tivoli Key Lifecycle Manager server connection

The DS8700 supports up to four Tivoli Key Lifecycle Manager key server connections (ports).

The following suggestions apply to configurations:

- In multiple site configurations, at least two of the key server ports should be assigned to isolated key servers at separate physical sites. The remaining ports can be connected to general key servers.
- In single site configurations, at least two of the key server ports should be assigned to isolated key servers at the same site.

The DS8700 configuration for encryption also requires that at least two active key servers be connected and defined at the DS8700.

To configure the Tivoli Key Lifecycle Manager server connection, use the **lskeymgr** and **mkkeymgr** commands as follows:

1. Issue the **lskeymgr** command to view the list of already registered Tivoli Key Lifecycle Manager servers, if any.

   Enter the **lskeymgr** command at the `dscli` command prompt with the parameters and variables as illustrated in Example 4-1. In this example, we already had one Tivoli Key Lifecycle Manager server, named `popcorn`, registered with the DS8700.

   Example 4-1 List existing Tivoli Key Lifecycle Manager servers

```
dscli> lskeymgr -l
Date/Time: 10 October 2009 9:15:57 IBM DSCLI Version: 6.5.0.220 DS
ID  state    status addr                                port
=========================================================
1   active  normal popcorn                              3801
```

2. Issue the **mkkeymgr** command to add a new Tivoli Key Lifecycle Manager server.

   Enter the **mkkeymgr** command at the `dscli` command prompt with the parameters and variables as illustrated in Example 4-2. In this example, we added a second Tivoli Key Lifecycle Manager server, named `peanuts`, to the DS8700 configuration.

   Example 4-2 Add new Tivoli Key Lifecycle Manager server

```
dscli> mkkeymgr -addr peanuts 2
Date/Time: 10 October 2009 9:15:57 IBM DSCLI Version: 6.5.0.220 DS
CMUC00354I mkkeymgr: The key server 2 has been created.
```

3. Verify that the new Tivoli Key Lifecycle Manager server was added successfully, the state is active, and the status is normal. Again, issue the **lskeymgr** command with the **-l** parameter as shown in Example 4-3.

   Example 4-3 Verifying the Tivoli Key Lifecycle Manager servers

```
dscli> lskeymgr -l
Date/Time: 10 October 2009 9:15:57 IBM DSCLI Version: 6.5.0.220 DS
ID  state    status addr                                port
=========================================================
1   active  normal        popcorn                       3801
2   active  normal        peanuts                       3801
```

4. To delete a Tivoli Key Lifecycle Manager server, you can issue the **rmkeymgr** command with the parameters and variables in Example 4-4.

   Example 4-4 Removing Tivoli Key Lifecycle Manager server from configuration

```
dscli> rmkeymgr 2
Date/Time: 10 October 2009 9:15:57 IBM DSCLI Version: 6.5.0.220 DS
CMUC00356I rmkeymgr: Are you sure you want to delete the key server 3? [y/n]:y
CMUC00357I rmkeymgr: The key server 2 has been deleted.
```

4.3.2 Managing recovery key

As already explained in 4.2.3, “Managing recovery key” on page 67, the risk of the deadlock situation can be highly minimized by maintaining redundant (dual platform) Tivoli Key Lifecycle Manager servers, but cannot be eliminated. The *recovery key* (RK) feature provides a way to get out of a deadlock state. Starting with Licensed Machine Code (LMC) level 6.5.1.xx, users can enable or disable the recovery key management.

This choice has to be made before you configure the encryption group.

Recovery key enabling

To configure recovery key using DS CLI commands, perform the following steps:

1. The DS CLI command that is used to configure recovery key must be entered interactively by a user with security administrator authority. Use the **mkreckey** command as shown in the Example 4-5.

   Example 4-5 Configure recovery key: **mkreckey** *command*

   ```
   dscli> mkreckey –dev IBM.2107-1300861 1
   Sun Aug 11 02:23:49 PST 2004 IBM DS CLI Version: 5.0.0.0 DS: IBM.2107-1300861
   CMUC00392I mkreckey: The access Recovery Key
   0123-4569-4443-3334-3334-0123-4569-3334-4443-3334-3334-0123-4569-4443-3334-3334 for
   encryption group 1 has been created, pending verification.
   ```

   You can copy the new recovery key text from the terminal and save it in a file, which can be used for printing. However, we do not recommend doing it this way, because the key can be sniffered in the network, so it is better to write down the key on a piece of paper directly from the screen.

   The security administrator is responsible for writing down the recovery key and storing the paper in a safe place.

2. The security administrator issues the **managereckey -action verify** command to ensure that the previously written key is correct. See Example 4-6.

   Example 4-6 Verify recovery key

   ```
   dscli> managereckey –dev IBM.2107-1300861 –action verify - key
   0123-4569-4443-3334-3334-0123-4569-3334-4443-3334-3334-0123-4569-4443-3334-3334 1
   Sun Aug 11 02:23:49 PST 2004 IBM DS CLI Version: 5.0.0.0 DS: IBM.2107-1300861
   CMUC00393I managereckey: The access Recovery Key for encryption group 1 has been
   verified, pending authorization.
   ```

   The recovery key is now in the Authorization Pending status.

3. After the recovery key is verified, the storage administrator authorizes the recovery key previously created. You have to log on as a user with administrator authority to issue **managereckey -action authorize** command as shown in Example 4-7.

   Example 4-7 Authorize recovery key

   ```
   dscli> managereckey –dev IBM.2107-1300861 –action authorize 1
   CMUC00406W managereckey: Are you sure that you want to authorize the creation of the
   access Recovery Key for encryption group 1? [Y/N]:Y
   Sun Aug 11 02:23:49 PST 2004 IBM DS CLI Version: 5.0.0.0 DS: IBM.2107-1300861
   CMUC00395I managereckey: The pending Recovery Key management operation for encryption
   group 1 has been authorized.
   ```

4. Check the recovery key state by issuing the **lskeygrp** command, as in Example 4-8 on page 88

Example 4-8 List the recovery key and its status

```
dscli> lskeygrp -l
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
ID  state        reckeystate reckeycreated label label2
=====================================================
1   unconfigured configured  10/10/2009    -     -
```

The `reckeystate` is configured, indicating that a new recovery key has been requested, verified, and authorized. *State* is still unconfigured because the encryption group has not been configured yet.

Now you can proceed with the configuration of the encryption group. See 4.3.3, “Configuring and administering the encryption group” on page 89. However, you may also disable the recovery key; see “Recovery key disabling” on page 88.

Recovery key disabling

To disable recovery key management using DS CLI commands, perform the following steps:

1. The DS CLI command that is used to disable the recovery key must be entered interactively by a user with security administrator authority. Use the **`managereckey`** command as shown in the Example 4-9.

   Example 4-9 Disable recovery key: **`managereckey`** *command*

   ```
   dscli> managereckey -action disable 1
   Date/Time: May 18, 2010 11:34:09 AM MST IBM DSCLI Version: 6.5.1.193 DS:
   IBM.2107-75TR731
   CMUC00416I managereckey: The access recovery key for encryption group 1 has been
   disabled, pending authorization.
   ```

   The disabling action is now in the Authorization Pending status.

2. The storage administrator authorizes the recovery key disabling. Log on as a user with administrator authority to issue **`managereckey -action authorize`** command, as shown in Example 4-10.

   Example 4-10 Disable recovery key: **`managereckey`** *command*

   ```
   dscli> managereckey -action authorize 1
   Date/Time: May 18, 2010 11:36:49 AM MST IBM DSCLI Version: 6.5.1.193 DS:
   IBM.2107-75TR731
   CMUC00418W managereckey: Are you sure that you want to authorize the disable of the
   access recovery key for encryption group 1? [Y/N]: y
   CMUC00395I managereckey: The pending recovery key management operation for encryption
   group 1 has been authorized.
   ```

3. Check the recovery key state by issuing the **`lskeygrp`** command, as in Example 4-11

   Example 4-11 List the recovery key and its status

   ```
   dscli> lskeygrp
   Date/Time: May 18, 2010 11:37:37 AM MST IBM DSCLI Version: 6.5.1.193 DS:
   IBM.2107-75TR731
   ID  state        reckeystate reckeydate datakeydate
   ==================================================
   1   unconfigured disabled    -          05/18/2010
   ```

   The `reckeystate` is disabled, indicating that the recovery key management has been disabled. *State* is still unconfigured because the encryption group has not been configured yet.

Now, you can proceed with the next step and configure the encryption group.

4.3.3 Configuring and administering the encryption group

The customer data within the Storage Facility Image that is encrypted is partitioned in one encryption group.The encryption group that contains encrypted data is enabled to access data through one data key obtained from a key server.

Note: Currently the DS8700 supports only one encryption group.

An encryption group contains a set of extent pools, each of which has a set of associated ranks and volumes. The FlashCopy and Remote mirror and copy functions can migrate data within or between encryption groups.

To configure the encryption key group using the DS CLI commands, perform the following steps:

1. Issue the **lskeygrp** command to view the state of encryption group and recovery key. Enter the **lskeygrp** command at the `dscli` command prompt with the parameters and variables shown in Example 4-12. In our example, the recovery key is configured and encryption group is unconfigured.

   Example 4-12 Listing key groups

   ```
   dscli> lskeygrp -l
   Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
   ID  state        reckeystate reckeycreated label label2
   ======================================================
   1   unconfigured configured  10/10/2009    -     -
   ```

2. Issue the **mkkeygrp** command to add the new encryption key group. Enter the **mkkeygrp** command at the `dscli` command prompt with the parameters and variables in Example 4-13. The label should match the one you defined on the Tivoli Key Lifecycle Manager server (see Figure 4-8 on page 55). Number `1` is the encryption group ID.

   Example 4-13 Creating key group

   ```
   dscli> mkkeygrp -label itsokey 1
   Date/Time: 10 October 2009 10:21:07 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
   The key server encryption key group 1 has been created.
   ```

3. Verify that the encryption group was added successfully, and that its state is accessible by issuing the **lskeygrp** command with the **-l** parameter, as shown in Example 4-14.

   Example 4-14 Listing defined key groups

   ```
   dscli> lskeygrp -l
   Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
   ID  state        reckeystate reckeycreated label label2
   ======================================================
   1   accessible configured  10/10/2009    -     -
   ```

4.3.4 Applying encryption activation key

Now that the Tivoli Key Lifecycle Manager key servers have been set up and are registered with the DS8700, the encryption feature activation key must be applied, before the logical configuration (defining ranks and extent pools) can take place.

The DS CLI **applykey** command activates the licenses for your storage unit. The DS CLI **lskey** command verifies which type of licensed features are activated for your storage unit.

To apply the encryption activation key, issue the following command:

```
dscli> applykey -file c:\keys.xml -dev IBM.2107-75LY981
```

This example presumes that your XML file is named `keys.xml` and it resides in the root directory of your `C:` drive.

Issue the **lskey** command to verify that the license codes have ben applied, as shown in Example 4-15.

Example 4-15 List all activated licenses

```
dscli> lskey IBM.2107-75LY981
Date/Time: 10 October 2009 10:30:01 IBM DSCLI Version: 6.5.0.220 DS:
IBM.2107-75LY981
Activation Key                              Authorization Level (TB) Scope
=====================================================================
Encryption Authorization                    on                       All
Global mirror (GM)                          80.3                     CKD
High Performance FICON for System z (zHPF) on                        CKD
IBM FlashCopy SE                            80.3                     CKD
IBM HyperPAV                                on                       CKD
IBM database protection                     on                       FB
Metro mirror (MM)                           80.3                     CKD
Metro/Global mirror (MGM)                   80.3                     CKD
Operating environment (OEL)                 80.3                     All
Parallel access volumes (PAV)               80.3                     CKD
Point in time copy (PTC)                    80.3                     CKD
RMZ Resync                                  80.3                     CKD
Remote mirror for z/OS (RMZ)                80.3                     CKD
```

4.3.5 Creating encrypted arrays

Complete this task to create encrypted arrays using the DS CLI commands.

Use the **lsarraysite** and **mkarray** commands to create the arrays.

An array inherits the characteristics of its parent array sites and is given a RAID type attribute (5, 6, or 10). A DS8700 array of RAID type 5, 6, or 10 is made from one (8 DDM) array site. The status of the array is *unassigned* until the array is assigned to a rank.

To create an encrypted array from unassigned array sites, perform the following steps:

1. Issue the **lsarraysite** command to view a list of array site IDs for all installed array sites. Review those arrays that are designated with the state of unassigned.

   Enter the **lsarraysite** command at the `dscli` command prompt with the following parameters and variables:

   ```
   dscli> lsarraysite -dev IBM.2107-75LY981 -state unassigned -l
   ```

2. Press Enter.

   A report of unassigned array sites is displayed. Use the list to identify unassigned array site capacity, rpm, and device adapter (DA) pair attributes. Record the RAID type for each array site. An illustration is shown in Example 4-16 on page 91.

Example 4-16 Listing arraysites

```
dscli> lsarraysite -dev IBM.2107-75LY981 -state unassigned -l
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
arsite DA Pair dkcap (10^9B) diskrpm State      Array diskclass encrypt
=======================================================================
S8     0               450.0   15000 Unassigned -     ENT       supported
```

Note: Make sure that the array site supports encryption. If this is your first time creating fixed block volumes, all the arrays are displayed with a state of unassigned.

3. Issue the **mkarray** command to create an array from one array site with the status *unassigned*. Enter the **mkarray** command at the dscli command prompt with the following parameters and variables:

   ```
   dscli>mkarray -dev storage_image_ID -raidtype 6 -arsite array_site
   ```

   Consider the following information when you create the arrays:

   – Specify one array site with identical capacity, rpm, interface, and DA pair attributes.
   – The new array inherits the capacity, rpm, interface, and DA pair characteristics of its parent array site.
   – The state of the array remains unassigned until it is assigned to a rank.

4. Verify that the array-to-array site assignment is recognized and complete by issuing either the **lsarray** or **lsarraysite** command with the **-l** parameter. Example 4-17 gives an illustration.

Example 4-17 Verifying the arrays

```
dscli> lsarray -dev IBM.2107-75LY981 -l
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
Array State      Data   RAIDtype    arsite Rank DA Pair DDMcap (10^9B) diskclass encrypt
==============================================================================
A0    Assigned   Normal 5 ( 6+P+S )   S8 R1   0 450.0 ENT supported
```

4.3.6 Creating encrypted ranks

Use the **lsarray**, **mkrank**, and **lsrank** commands to assign a rank to each unassigned array.

To create ranks, perform the following steps:

1. Ensure you have a list of the unassigned arrays for which ranks must be assigned. Issue the **lsarray** command to obtain this list if you do not already have it. Enter the **lsarray** command at the `dscli` command prompt with the following parameters and variables:

   ```
   dscli>lsarray -dev IBM.2107-75LY981 -state unassigned
   ```

2. Issue the mkrank command to assign a rank to rank group 0 or 1 according to the rank group number of the assigned extent pool ID.

   To create an encrypted rank, use the `-encryptgrp` variable.

   Enter the **mkrank** command at the `dscli` command prompt with the parameters and variables as shown in Example 4-18 on page 92 for fixed block storage type.

Example 4-18 Assigning an array to a rank

```
dscli>mkrank -dev IBM.2107-75LY981 -array A1 -stgtype fb -wait -encryptgrp 1
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
Rank IBM.2107-75HT551/R0 successfully created.
```

Notes:

- The `-encryptgrp encryption_group_ID` specifies the encryption group that this rank should use. The default is 0 (zero), which means that no encryption group is assigned to the rank.
- You can specify either the **-wait** or the **-extpool** parameter when you use the **mkrank** command. Either of these parameters allows you to be notified if the rank configuration has failed for any reason.
- If you use the **-wait** parameter, you cannot issue other commands until the entire transaction has processed.

3. Press Enter to display a report of rank assignments for your entire storage unit. Because the process of creating the rank involves formatting drives, the process can take some time before it finishes. If you want to check on the process, issue the **lsrank** command from a different DS CLI session.
4. Issue the **lsrank** command to verify that ranks and extent pools have been assigned. Enter the **lsrank** command at the `dscli` command prompt with the parameters and variables, shown in Example 4-19.

Example 4-19 Verifying the ranks

```
dscli> lsrank -dev IBM.2107-75LY981 -l
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
ID Group State  datastate Array RAIDtype extpoolID extpoolnam stgtype exts usede
xts encryptgrp
===========================================================================================
R0     0 Normal Normal    A1         10 P0       raid10P0   fb      1186     1 152
1
```

5. Press Enter to display a report of the rank assignments for your entire storage unit.

4.3.7 Creating encrypted extent pools

Complete this task to create encrypted volume extent pools. This is the first step in configuring new encrypted fixed block storage.

Issue the **mkextpool** command to create the encrypted extent pool for rank group 0 (zero).

Enter the **mkextpool** command at the `dscli` command prompt with the parameters and variables, shown in Example 4-20, for fixed block extent pools, where `-encryptgrp 1` represents the encryption group ID and the `P0` represents the extent pool name that you assign. This name can be 16 double-byte characters.

Example 4-20 Creating encrypted FB extent pool

```
dscli>mkextpool -dev IBM.2107-75LY981 -rankgrp 0 -stgtype -encryptgrp 1 fb P0
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS: IBM.2107-75LY981
Extent pool P0 successfully created.
```

Verify the extent pool assignments by issuing the **lsextpool** command when you are done creating the extent pools. Use the **-l** parameter to display a full report for the extent pools that are assigned to the storage unit.

Enter the **lsextpool** command at the `dscli` command prompt with the parameters and variables, as shown in Example 4-21.

Example 4-21 Verifying the extent pool assignments

```
dscli> lsextpool -dev IBM.2107-75LY981 -l
Date/Time: 10 October 2009 9:30:52 IBM DSCLI Version: 6.5.0.220 DS:
IBM.2107-75LY981
Name       ID stgtype rankgrp status availstor (2^30B) %allocated available rese
rved numvols numranks encryptgrp
================================================================================
raid10P0   P0 fb            0  below                34         97        34 0
24        1          1
```

All ranks and extent pools on a given encryption-capable DS8700 SFI must be configured with the same encryption group attribute. The first rank or encryption group that is configured determines what the remaining objects must be configured with. A value of 0 indicates encryption-disabled. A value of 1 indicates encryption enabled.

To change between encryption-enabled and encryption-disabled, all ranks and extent pools must be unconfigured. Unconfiguring an encryption-enabled rank causes any data that is stored on the rank to be cryptographically erased (the disk is instructed to reset its own encryption key) and subsequently overwritten to re-initialize the rank.

4.4 Encryption and Copy Services functions

Copy Services operations are not affected by encrypting drives. The encryption applies only to data at rest, which is the data that is physically written to the disk drives. If you are doing remote replication of the encrypted data, then when the data is *read* from the source disk, it is decrypted, sent across the network link, and if the target storage system is also set up for encryption, when the data is written to disk at the target site, it is encrypted again. There is no relationship between the encryption that is done at the source and the encryption at the target, they are completely independent operations with their own set of keys and potentially even their own key managers depending on how the environment is configured.

This encryption strategy would also hold true, by the way, for FlashCopy. Although this is a “T0” copy of data that only resides with the DS8700, when the source data is read and re-written to the FlashCopy target volume, it will be decrypted at *read* and re-encrypted at *write*. The encryption itself is not intrusive in terms of performance because it is all done by the drives themselves.

5

Maintaining the DS8700 encryption environment

This chapter provides information about maintenance and exploitation of your DS8700 encryption environment.

This chapter contains the following topics:

- Backup and restore
- Starting and stopping Tivoli Key Lifecycle Manager server
- Key exporting and importing tasks
- Tivoli Key Lifecycle Manager dual platform support implementation
- Rekey data key
- Recovery key usage and maintenance

© Copyright IBM Corp. 2010. All rights reserved.

5.1 Backup and restore

Tivoli Key Lifecycle Manager does not automatically synchronize between servers but it does provide a convenient backup and restore operation that can be performed using the command line or web user interface. Synchronization involves backing up Tivoli Key Lifecycle Manager and then restoring to a separate server with the same configuration parameters. Several considerations include the following items:

- Select one server to be the *main* Tivoli Key Lifecycle Manager key server and originate all backups from there. Make all changes on this main key server and then deploy it through a backup and restore to the other Tivoli Key Lifecycle Manager server.
- Both Tivoli Key Lifecycle Manager servers must be running the same OS with the same user accounts for Tivoli Key Lifecycle Manager, Tivoli Integrated Portal, and DB2.
- The restore task is a disruptive operation, therefore ensure that the other Tivoli Key Lifecycle Manager key server is active and serving keys before performing the restore.

Backup and restore tasks provide protection for critical data, and require consideration of your site practices to ensure server availability and runtime capabilities. Tivoli Key Lifecycle Manager creates backup files that contain critical data for the current state of the Tivoli Key Lifecycle Manager server.

Important: Failure to back up your keystore and other critical data properly might result in unrecoverable loss of all access to your encrypted data. Do not encrypt your backup file, or store a backup file on an encrypting device. Failure to back up data might also result in subsequent inconsistency of the key manager and potential data loss on the storage device.

5.1.1 Categories of data in a backup file

A backup file of Tivoli Key Lifecycle Manager critical data includes keystore, configuration file, metadata, and other information.

Data that requires backup protection include the following categories:

- Keystore files

  Certificates and keys (or pointers to the certificates and keys) that Tivoli Key Lifecycle Manager uses to perform key serving and other operations
- Tivoli Key Lifecycle Manager configuration files

  Properties that define selected Tivoli Key Lifecycle Manager activities such as audit settings and other values that you customize for your system configuration
- Tivoli Key Lifecycle Manager database

  Data about Tivoli Key Lifecycle Manager objects such as devices, key groups, certificates, keys, and drives

  Backup file security ensures that you do not accidentally corrupt a backup file or misplace its encryption password. Ensure that you retain the password that is used to encrypt a given backup file. The same password is required to restore and decrypt the file.

5.1.2 Backup file security

Observe the following guidelines to secure your backup files:

- Ensure that you do not accidentally corrupt a backup file or misplace its encryption password.
- Do not edit the files contained in a backup.jar file. The files will become unreadable.
- Ensure that you retain the password used to encrypt a given backup file. The same password is required to restore and decrypt the file.

5.1.3 Tivoli Storage Manager as a backup repository

Tivoli Storage Manager might be your backup repository of choice. Tivoli Storage Manager provides progressive, incremental backup, hierarchical storage management, and tape optimization.

For example, you might use the policy-based backup capabilities that Tivoli Storage Manager provides to perform scheduled backups on an hourly, daily, or weekly basis.

For critically important data such as the Tivoli Key Lifecycle Manager database, keystore, and configuration file, you might use Tivoli Storage Manager to back up that data when it changes. The Tivoli Storage Manager Server can be located on a separate computer than the one on which the DB2 server runs.

5.1.4 Backup and restore runtime requirements

Backing up and restoring data from backup files for Tivoli Key Lifecycle Manager have several runtime requirements.

Before you begin, you might prevent time out failure by increasing the time interval allowed for backup and restore transactions for large key populations. Specify a larger value for the `totalTranLifetimeTimeout` setting in the following file:

`TIP_HOME/profiles/TIPProfile/config/cells/TIPCell/nodes/TIPNode/servers/server1/server.xml`

In the example, TIP is Tivoli Integrated Portal.

In addition, the following conditions must be true:

- Ensure that the task occurs during a time interval that allows a halt to key serving activity.
- For a backup task, the Tivoli Key Lifecycle Manager server must be running in a normal operational state. The Tivoli Key Lifecycle Manager database instance must be available.
- For a restore task, the Tivoli Key Lifecycle Manager database instance must be accessible through the Tivoli Key Lifecycle Manager data source.

  Before you start a restore task, ensure that you have the password that was used when the backup file was created. Restored files must be written to the same Tivoli Key Lifecycle Manager server from which the data was previously backed up, or to an identical, replica computer.
- Ensure that directories exist and are associated with the `tklm.backup.dir` and `tklm.db2.backup.dir` properties. System and Tivoli Key Lifecycle Manager administrator accounts under which the Tivoli Key Lifecycle Manager server and the DB2 server run must have read and write access to these directories respectively.

5.1.5 Backing up critical files

The Tivoli Key Lifecycle Manager backup saves a password-protected copy of the Tivoli Key Lifecycle Manager server settings including the keystore and DB2 tables. However, when performing a restore operation, Tivoli Key Lifecycle Manager assumes that the environment is similar. The restore operations must be on the same platform with the same system user account information and Tivoli Key Lifecycle Manager and middleware file layout.

Note: Backup and restore tasks are disruptive to the Tivoli Key Lifecycle Manager server. During these tasks, Tivoli Key Lifecycle Manager temporarily stops other operations.

To back up, perform the following steps:

1. Log on to the graphical user interface. From the navigation tree, select **Tivoli Key Lifecycle Manager** → **Settings** → **Backup and Restore**.
2. In the backup and restore table, click **Create Backup**, as shown in Figure 5-1.

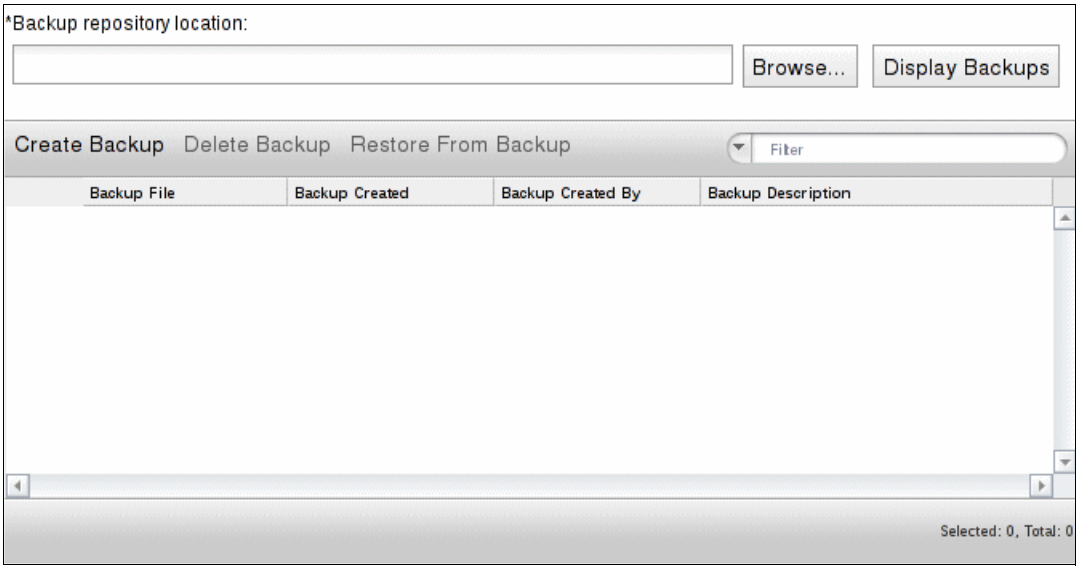


Figure 5-1 Backup and restore table

3. On the Create Backup panel, shown in Figure 5-2 on page 99, specify the required information such as the path and a value for the encryption password. Then, click **Create Backup**.

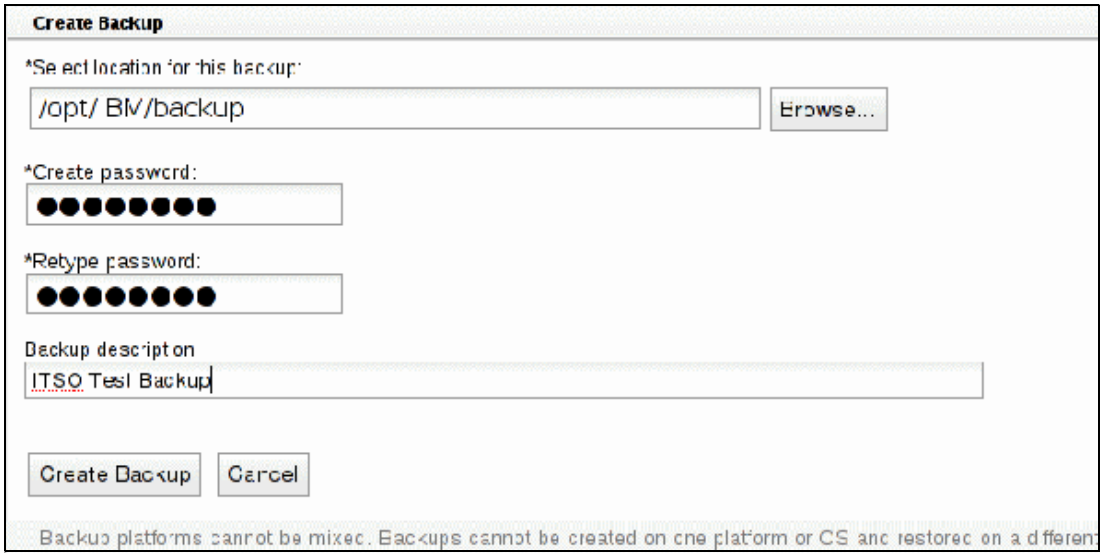


Figure 5-2 Create Backup

4. Review the directory that contains the backup files to ensure that the backup file exists. Do not edit a file in the `backup.jar` file. The file that you attempt to edit will become unreadable. The file location is as shown in the Backup File column (Figure 5-3).

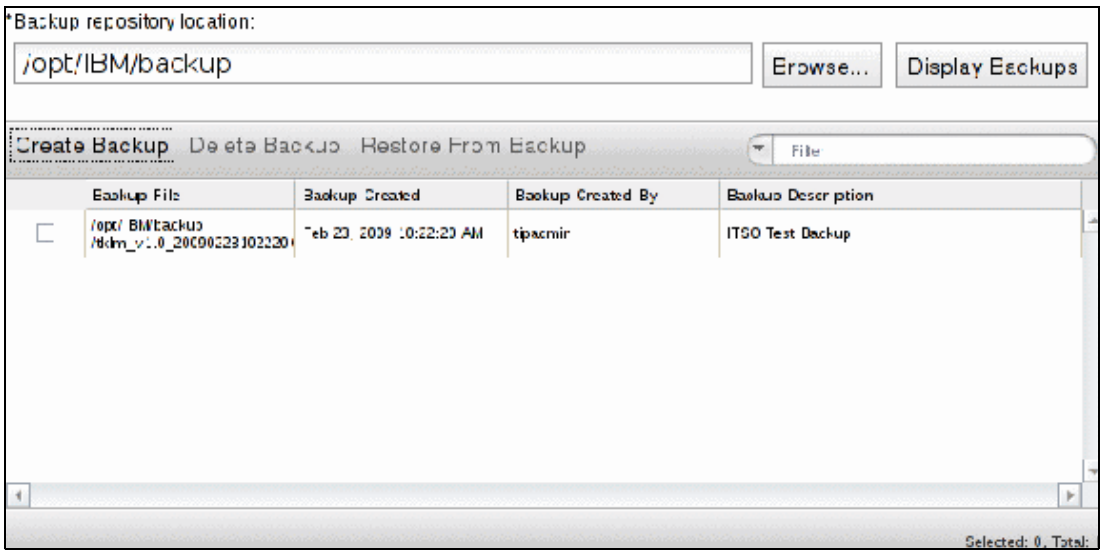


Figure 5-3 Backup file created

5.1.6 Restoring a backup file

You can use the Backup and Restore table to restore a backup file. Only one backup or restore task can run at a given time. If you restore a file to a replica computer, copy the file to that computer using media such as a disk, or electronic transmission.

Perform the following steps:

1. Log on to the graphical user interface. From the navigation tree, select **Tivoli Key Lifecycle Manager** → **Settings** → **Backup and Restore**.
2. On the Backup and Restore table, select a backup file that is listed in the table. Then, click **Restore from Backup**. The panel shown in Figure 5-4 opens.

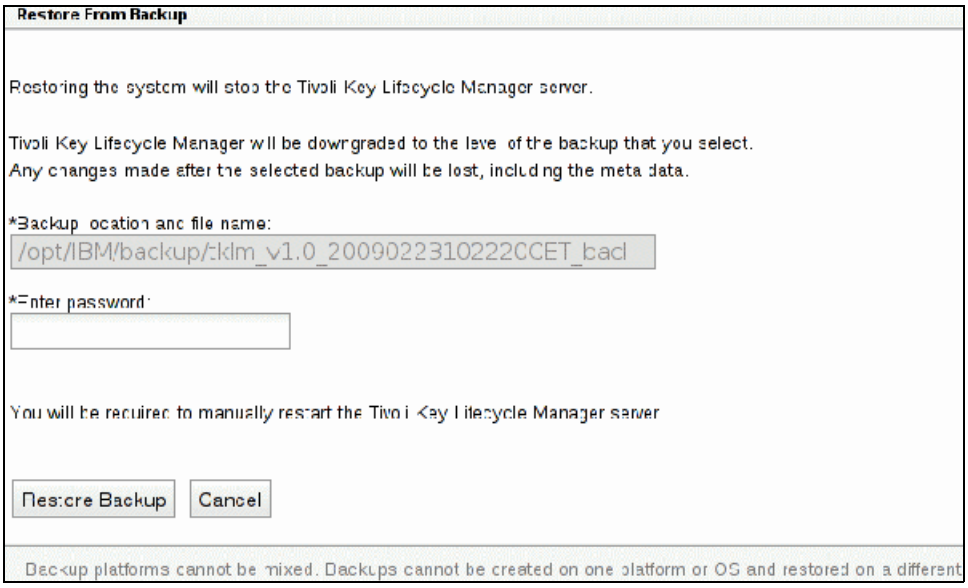


Figure 5-4 Restore from backup

3. Before the restore procedure starts, click **OK** to confirm the message indicating that the server will be stopped, as shown in Figure 5-5.

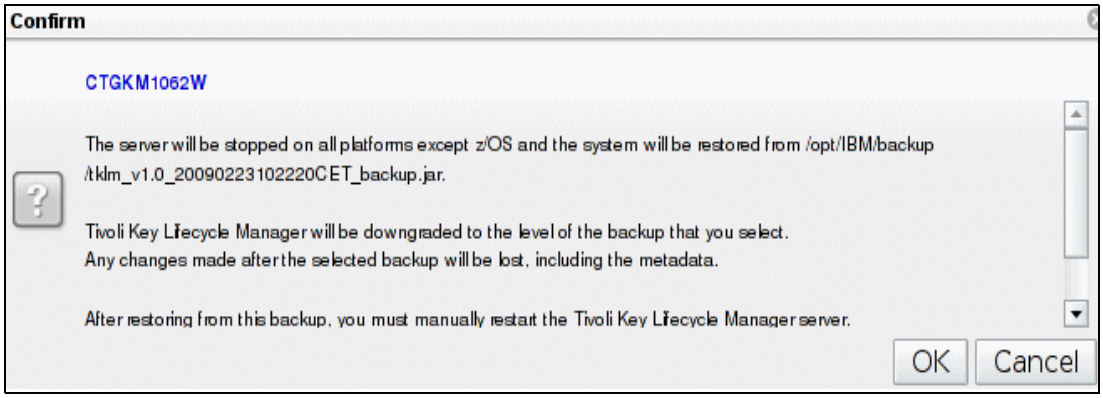


Figure 5-5 Restore warning message

4. A message indicates that the restore operation succeeded. Click **OK** to finish the restore procedure. See Figure 5-6 on page 101.

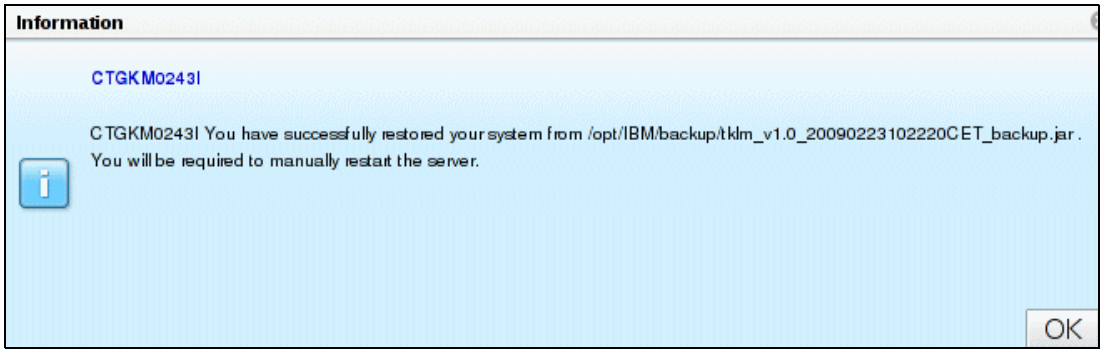


Figure 5-6 Restore successful message

5. Manually restart the Tivoli Key Lifecycle Manager server. Then, determine whether the server is in the expected state. For example, you might examine the keystore to see whether a certificate that had problems prior to restoring the backup file is now available for use. Refer to 5.2, “Starting and stopping Tivoli Key Lifecycle Manager server” on page 101.

5.1.7 Deleting a backup file

Use the graphical user interface (GUI) or command line interface (CLI) to delete a backup file for Tivoli Key Lifecycle Manager. For example, you might delete a backup file for which a business use no longer exists.

You can use the Backup and Restore table to restore a backup file, as follows:

1. Log on to the GUI. From the navigation tree, select **Tivoli Key Lifecycle Manager** → **Settings** → **Backup and Restore**.
2. Delete a selected backup file.
3. On the Backup and Restore table, select a backup file that is listed in the table. Click **Delete Backup** and confirm that you want to delete the file.
4. Examine the directory in which the backup files are stored, to determine whether the specified file was deleted.

5.2 Starting and stopping Tivoli Key Lifecycle Manager server

You might have to use the **`startServer`** or **`stopServer`** scripts to start or stop the Tivoli Key Lifecycle Manager server. Starting and stopping the Tivoli Key Lifecycle Manager application is possible by using the SSRE Integrated Solutions Console GUI if the Tivoli Key Lifecycle Manager server is installed on System z. Restarting the Tivoli Key Lifecycle Manager server, for instance, is required after completion of a restore task. You can also check Tivoli Key Lifecycle Manager status.

Starting and stopping the server by using scripts

Scripts to start and stop the Tivoli Key Lifecycle Manager server are located in the `/TIP_HOME/bin` directory for Linux and AIX platforms and `/SSRE_HOME/bin` for z/OS (OMVS). In the commands, the `server1` parameter is the default name of the configured Tivoli Key Lifecycle Manager server instance.

Start the Tivoli Key Lifecycle Manager server

To start the server, use the command for your platform, as follows:

- Windows systems:

```
StartServer.bat server1
```

- Linux, AIX and z/OS (OMVS) systems:

```
./startServer.sh server1
```

Stop the Tivoli Key Lifecycle Manager server

To stop the server, use the command for your platform, as follows:

- Windows systems:

```
StopServer.bat server1 -username TipAdminId -password Password
```

- Linux, AIX and z/OS (OMVS) systems:

```
./stopServer.sh server1  -username TipAdminId -password Password
```

When global security is enabled (which is recommended), enter the user ID and password of the Tivoli Integrated Portal administrator as parameters to the stopServer script. The script will prompt for these parameters if they are omitted, but you can specify them on the command line.

Starting and stopping from SSRE Integrated Solutions Console

Log on to the SSRE Integrated Solutions Console and from the Welcome window expand the **Applications** section (on the left side of the window) and select **Enterprise Applications**, as shown in Figure 5-7.

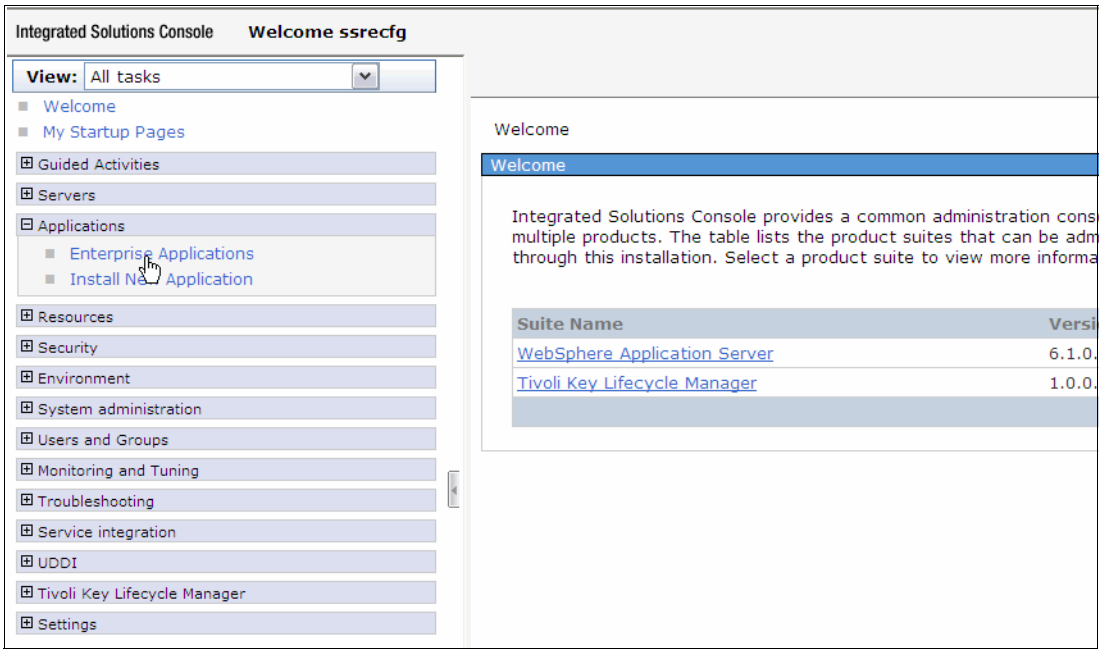


Figure 5-7 SSRE Integrated Solutions Console: select Enterprise Applications

The Enterprise Applications window opens and it displays all available applications that are running under SSRE and the status of each. A green arrow indicates that the application is started and operational. Select Tivoli Key Lifecycle Manager and click the **Stop** button; see Figure 5-8 on page 103.

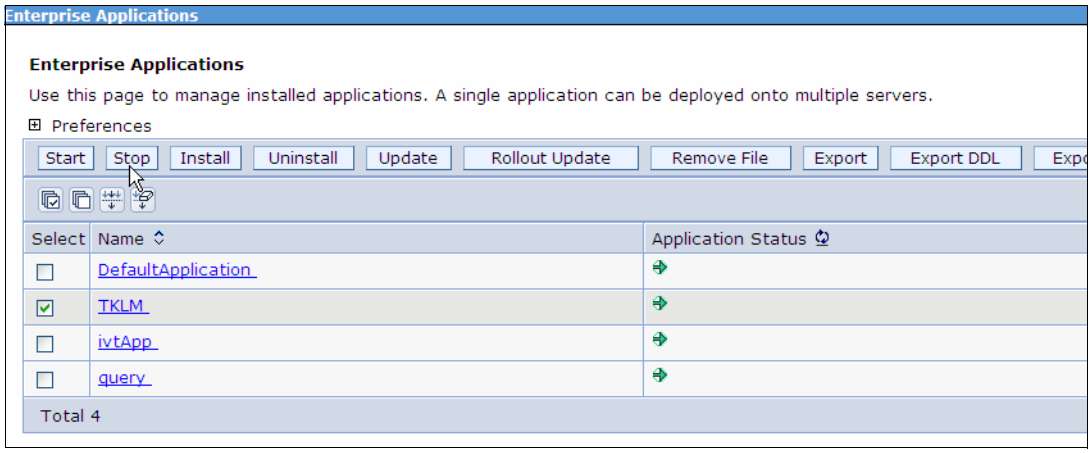


Figure 5-8 SSRE Integrated Solutions Console: Stop Tivoli Key Lifecycle Manager

Wait for the task to complete and the acknowledgment that Tivoli Key Lifecycle Manager server is stopped successfully as illustrated in the Figure 5-9.

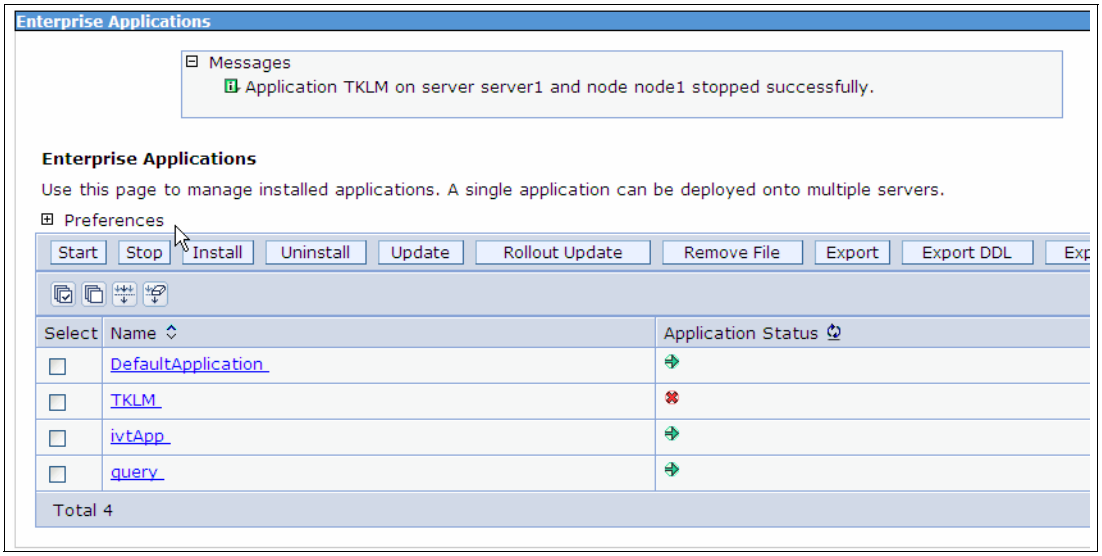


Figure 5-9 SSRE Integrated Solutions Console: Tivoli Key Lifecycle Manager is Stopped

The application status changes to red, indicating that Tivoli Key Lifecycle Manager is stopped. To restart the Tivoli Key Lifecycle Manager, select the check box next to the Tivoli Key Lifecycle Manager and click **Start**, as shown in Figure 5-10 on page 104.

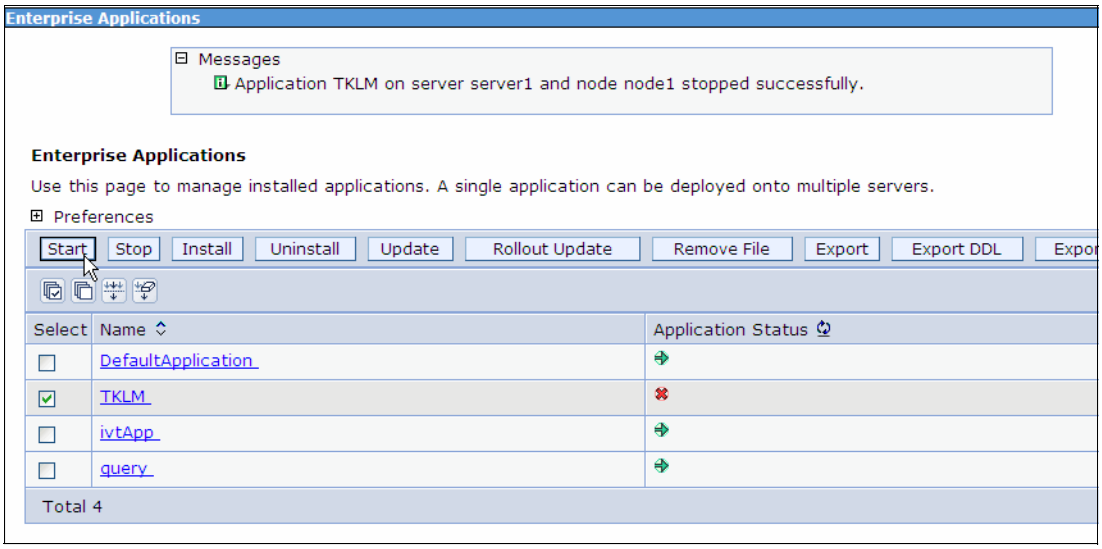


Figure 5-10 SSRE Integrated Solutions Console: Start Tivoli Key Lifecycle Manager

Wait for the message that the server is started successfully, as show in Figure 5-11.

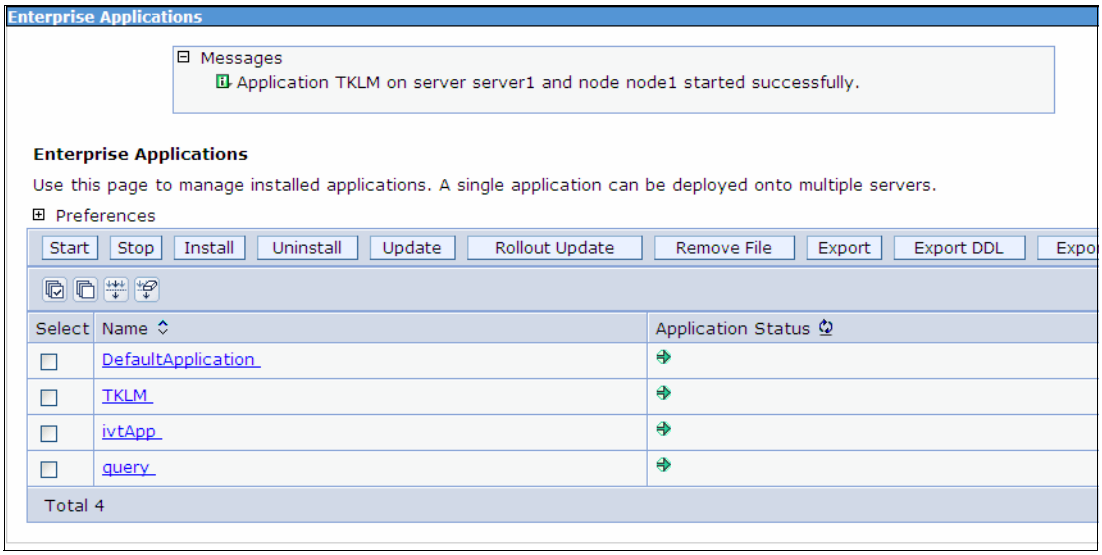


Figure 5-11 SSRE Integrated Solutions Console: Tivoli Key Lifecycle Manager is Started

Application status for Tivoli Key Lifecycle Manager changes back to started and operational.

Determining status

If you want to determine whether the Tivoli Key Lifecycle Manager server is running, simply try to log in to the Tivoli Integrated Portal. If the login is successful, the Tivoli Key Lifecycle Manager service is running. If Tivoli Key Lifecycle Manager is installed on z/OS, log in to the SSRE Integrated Solutions Console (ISC). With the SSRE ISC, you can check the status of Tivoli Key Lifecycle Manager application as shown in Figure 5-11.

Otherwise, you can issue the **`serverStatus`** command (in the `/TIP_HOME/bin` directory for Linux and AIX, or in `/SSRE_HOME/bin` for z/OS OMVS) with the server instance, username and password parameter, as illustrated in Example 5-1 on page 105.

Example 5-1 Check server status

```
cmd> ./serverStatus server1 -username TipAdminId -password Password
ADMU0116I: Tool information is being logged in file
           /opt/IBM/tivoli/tip/profiles/TIPProfile/logs/server1/serverStatus.log
ADMU0128I: Starting tool with the TIPProfile profile
ADMU0500I: Retrieving server status for server1
ADMU0508I: The Application Server "server1" is STARTED
```

In Windows systems, you can also check in the Services window to verify that the service is running, as shown in Figure 5-12.

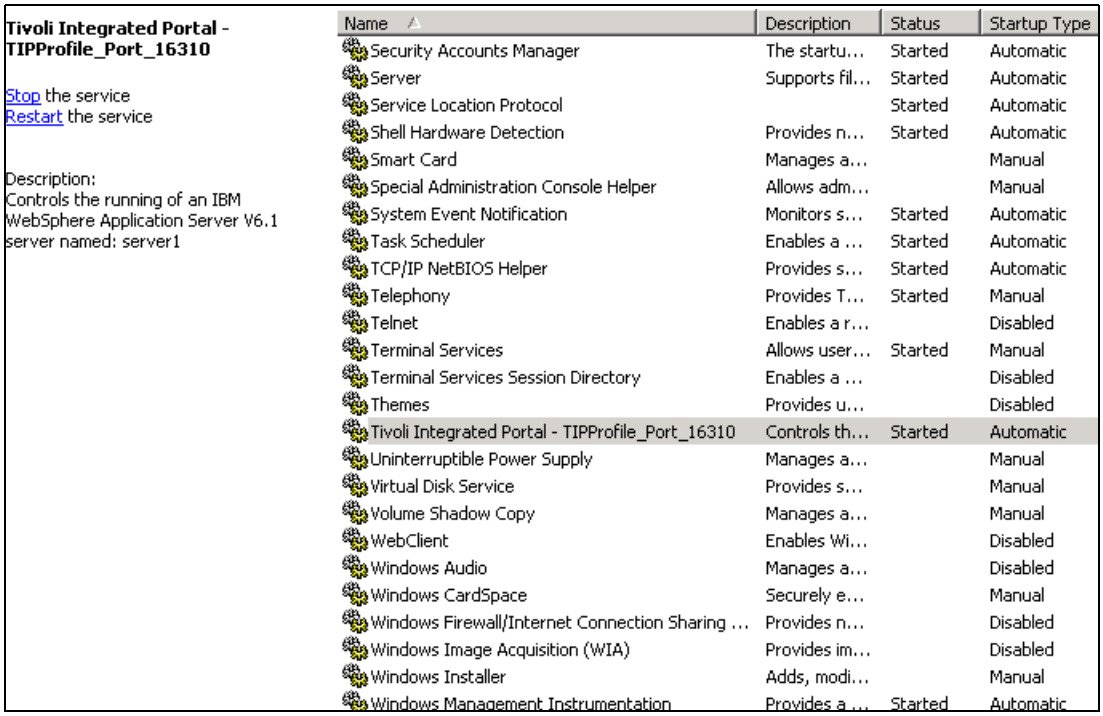


Figure 5-12 WIndows server state check

In Linux, issue the **ps** command, as shown in Example 5-2.

Example 5-2 Linux server state check

```
ps -ef | grep tip | grep server1
root      3747     1 67 16:20 pts/7    00:00:55 /opt/IBM/tivoli/tip/java/bin/java
-Declipse.security -Dwas.status.socket=59520
...../opt/IBM/tivoli/tip/profiles/TIPProfile/config TIPCell TIPNode server1
```

5.3 Key exporting and importing tasks

If you have two Tivoli Key Lifecycle Manager servers running on separate OS platforms and if both key server platforms operate in *clear* key mode (exporting of both private and public keys is allowed), backup and restore functions are not supported. The only way to keep them synchronized is to export the certificate from one server and restore it on the other server by using the export and import functions.

Note: Only the Tivoli Key Lifecycle Manager CLI mode can be used for this process because the GUI does not support these functions.

5.3.1 Exporting keys

To export keys, perform the following steps:

1. Open a command window, go to `<tip installation directory>/bin` folder, and execute the **`wsadmin`** script to export keys from the primary Tivoli Key Lifecycle Manager server (server1) as illustrated in Example 5-3.

   Example 5-3 Issue the wsadmin command

   ```
   23a4088:/opt/IBM/tivoli/tip/bin/wsadmin.sh -username tipadmin -password
   tipadmin -lang jython
   WASX7209I: Connected to process "server1" on node TIPNode using SOAP connector;
   The type of process is: UnManagedProcess
   WASX7029I: For help, enter: "$Help help"
   ```

2. Use the **`tklmKeyExport`** command with the `-alias`, `-fileName`, `-keyStoreName`, and `-type` parameters to export secret or private keys. The `-alias` parameter is the given name from the Tivoli Key Lifecycle Manager server DS8700, the `-keyStoreName` is the master keystore name for the Tivoli Key Lifecycle Manager server. See Example 5-4.

   Example 5-4 Exporting the keystore

   ```
   wsadmin>print AdminTask.tklmKeyExport('[-alias itsokey -fileName TKLM_DS8K
   -keyStoreName "ITSO Sample Keystore" -type privatekey]')
   CTGKM0001I Command succeeded.
   ```

   The file `TKLM_DS8K` was created in `/opt/IBM/tivoli/tip/products/tklm` with the filename `TKLM_DS8K`.

3. Copy and archive the exported key to the new Tivoli Key Lifecycle Manager server. The exported keys are regular files on the file system. The way that they are transferred depends on the operating systems.

For more Tivoli Key Lifecycle Manager CLI information, refer to the CLI online reference at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tklm.doc/ref/ref_ic_cli.html

5.3.2 Importing keys

Perform the following steps on the second Tivoli Key Lifecycle Manager server:

1. Open a command window, go to the `<tip installation directory>/bin` folder, and use the **`wsadmin`** command to import keys from Tivoli Key Lifecycle Manager server1. See Example 5-5.

   Example 5-5 Issue the wsadmin command

   ```
   23a4089:/opt/IBM/tivoli/tip/bin/wsadmin.sh -username tipadmin -password tipadmin -lang
   jython
   WASX7209I: Connected to process "server1" on node TIPNode using SOAP connector;  The
   type of process is: UnManagedProcess
   WASX7029I: For help, enter: "$Help help"
   ```

2. Use the **tklmKeyImport** command with the `-alias`, `-fileName`, `-password`, `-keyStoreName`, `-usage`, and `-type` parameter to export secret or private keys. The `-alias` parameter is the given name from the Tivoli Key Lifecycle Manager server DS8700, the `-password` is the key password from the DS8700, the `-keyStoreName` is the master keystore name for the Tivoli Key Lifecycle Manager server, and the `-usage` defines the storage type. See Example 5-6.

Example 5-6 Import the keystore

```
wsadmin>print AdminTask.tklmKeyImport('[-alias itsokey -fileName
/root/fromTKLM_server1/TKLM_DS8K -password passw0rd -keyStoreName "ITSO Sample
Keystore" -usage DS8K -type privatekey]')
CTGKM0001I Command succeeded.
```

For more Tivoli Key Lifecycle Manager CLI information refer to the online CLI reference at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tklm.doc/ref/ref_ic_cli.html

5.4 Tivoli Key Lifecycle Manager dual platform support implementation

When there are two Tivoli Key Lifecycle Manager key server platforms and at least one of the key server platforms is operating in secure key mode, a second key label is typically required. A key server operating in secure-key mode typically does not support the export of any private keys outside of the key server platform. The Tivoli Key Lifecycle Manager key server platform, which operates in secure mode, is available on z/OS platform when using JCECCAKS or JCECCARACFKS keystore type and private keys are stored and protected by Integrated Cryptographic Service Facility (ICSF).

For more information about Tivoli Key Lifecycle Manager dual platform support concept and design, refer to 2.5, “Dual key server support” on page 34.

In our example, we have the following key servers:

- Isolated Key Server (IKS): Based on IBM System x hardware, which stores the keys in the master keystore file that is written on its internal hard drive.
- Tivoli Key Lifecycle Manager for z/OS: Runs on IBM System z (z/OS), with JCECCARACFKS keystore type and private keys are stored and protected by Integrated Cryptographic Service Facility (ICSF). Therefore, only the public key can be exported.

In this section, we describe the setup tasks for two Tivoli Key Lifecycle Manager servers: `itso-tklm1` and `itso-tklm2`. In our example, IKS is defined as `itso-tklm1` server and Tivoli Key Lifecycle Manager for z/OS as `itso-tklm2`.

When both Tivoli Key Lifecycle Manager servers are installed and ready, make sure they are updated to the latest version. Currently, the dual platform key server support requires Tivoli Key Lifecycle Manager V1.0 fix pack 2 or later. If you need to upgrade your servers, use the IBM Fix Central portal to download the latest fix pack, which also contains a manual that describes the upgrade process.

First, we create the master keystores on every Tivoli Key Lifecycle Manager instances. On IKS, only the JCEKS keystore type is currently supported; for Tivoli Key Lifecycle Manager for z/OS, more options are available, depending on the ICSF existence. The process is the same

as the process that is described in 4.1.2, “Creating SSL certificate for secure communication” on page 52. Use the same keystore name on each Tivoli Key Lifecycle Manager server.

To complete Tivoli Key Lifecycle Manager dual platform support implementation, the following steps are required:

1. Creating the certificates
2. Exchanging the public keys
3. Adding DS8700 storage image

5.4.1 Creating the certificates

Now, we can create the new certificates on each Tivoli Key Lifecycle Manager platform. To do this, using the Tivoli Key Lifecycle Manager servers in CLI mode is faster and easier because the commands are almost the same for each platform. Only a few characters are different for each platform. Therefore, a better approach is to use terminals, where we can copy the preformatted commands.

Tip: When using Tivoli Key Lifecycle Manager CLI on z/OS (OMVS) you may need to convert to BPXFX111 character conversion table before you start your OMVS session. Use the following command:

```
omvs convert((BPXFX111))
```

This command allows you to enter the square brackets, [ ], which are used in Tivoli Key Lifecycle Manager CLI commands. Otherwise, you might get the following syntax error message because of code page differences:

```
SyntaxError: Lexical error at line 1, column 28. Encountered: "\u00dd" (221),
after : ""
```

Log on to each Tivoli Key Lifecycle Manager server. From the command window, go to the `<TIP HOME>/bin` folder for open systems platform, and `<SSRE HOME>/bin` for z/OS, then execute the **`wsadmin.sh`** script to invoke Tivoli Key Lifecycle Manager CLI, as shown in Example 5-7.

Example 5-7 Invoke Tivoli Key Lifecycle Manager CLI

```
wsadmin.sh -username ssrecfg -password ssrecfg -lang jython
```

Use the **`tklmCertCreate`** command to create a new certificate. In Example 5-8, `itsokey1` is created on the `itso-tklm1` server. In Example 5-9, `itsokey2` is created on `itso-tklm2` server.

Example 5-8 Create certificate on IKS

```
wsadmin>print AdminTask.tklmCertCreate ('[-type selfsigned -cn itsokey1 -alias
itsokey1 -usage DS8K -validity 1095 -keyStoreName "ITSO Sample Keystore"]')
CTGKM0503I Created a key pair and self-signed certificate: itsokey1
```

Example 5-9 Create certificate on Tivoli Key Lifecycle Manager for z/OS

```
wsadmin>print AdminTask.tklmCertCreate ('[-type selfsigned -cn itsokey2 -alias
itsokey2 -usage DS8K -validity 1095 -keyStoreName "ITSO Sample Keystore"]')
CTGKM0503I Created a key pair and self-signed certificate: itsokey2
```

Note: Do not give the same name for the certificates. A preferred way is to use the same expiration date in the `validity` parameter.

You can also specify more parameters for customizing your organization details. For more Tivoli Key Lifecycle Manager CLI information refer to the online CLI reference at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tklm.doc/ref/ref_ic_cli.html

If you already have certificates from a third-party provider, import them instead of creating new self-signed keys

The new certification can be listed using the **`tklmCertList`** command, as shown in Example 5-10 for `itso-tklm1` key server. The verbose mode (`-v y`) provides more details about it. Issue the same command on the other Tivoli Key Lifecycle Manager key server too.

Example 5-10 List certifications

```
wsadmin>print AdminTask.tklmCertList('[-alias itsokey1 -keyStoreName "ITSO Sample
Keystore" -v y]')
CTGKM0001I Command succeeded.
CTGKM0661I Found 1 certificates.

uuid                    CERTIFICATE-252c1687-6f03-4048-97e0-fc85003aadee
alias(es)               itsokey1
certifications          null
information             null
key store name(s)       ITSO Sample Keystore
key store uuid(s)       KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
owner                   null
key state               active
issuer name             CN=itsokey1
subject name            CN=itsokey1
target public key uuid  null
activation date         Sep 23, 2009
archive date            Sep 16, 2037
compromise date         null
creation date           Sep 23, 2009
expiration date         Sep 22, 2012
destroy date            null
retirement date         Sep 17, 2032
serial number           1253745487459693000
```

Each certification contains both public and private keys. You can view them by using the **`tklmKeyList`** command as shown in Example 5-11.

Example 5-11 List keys

```
wsadmin>print AdminTask.tklmKeyList ('[-alias itsokey1 -keyStoreName "ITSO Sample
Keystore" -v y]')
CTGKM0001I Command succeeded.
CTGKM0710I Found 2 keys.

uuid                  KEY-1ffcc23b-6ecd-4111-be66-e5e6d1ecbb12
information           null
alias(es)             itsokey1
```

```
key algorithm         AlgorithmName
key length (in bits)  2048
key type              RSAPrivateKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
key state             active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: 40 8e 4f eb e7 44 78 50  e7 9c 2f bd 4c 65 8d 60
..O..DxP....Le..


uuid                  KEY-c0abd46e-32c1-4524-b8a1-3ac70ddce66b
information           null
alias(es)             itsokey1
key algorithm         AlgorithmName
key length (in bits)  2048
key type              RSAPublicKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
key state             active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: b0 f5 7d ab eb 85 f1 85  32 e3 82 bc 75 b1 05 b3
........2...u...
```

The `key type` line shows whether the type is a public or private key.

5.4.2 Exchanging the public keys

The main concept of the dual platform key support feature is that the isolated key server (IKS) and the Tivoli Key Lifecycle Manager for z/OS have their own asymmetric key pair and also know the public key from the partner Tivoli Key Lifecycle Manager server. If a Tivoli Key Lifecycle Manager server breaks down, the other server has all the keys required for unwrapping the DS8700 data key, so the redundancy is provided on the Tivoli Key Lifecycle Manager side as well.

Public keys must be exchanged between the Tivoli Key Lifecycle Manager servers in the following order:

1. Export the public keys on both Tivoli Key Lifecycle Manager servers.
2. Copy the exported key files to the other server.
3. Import the public keys on Tivoli Key Lifecycle Manager servers accordingly.

Note: Only the Tivoli Key Lifecycle Manager CLI mode can be used for this process because the GUI does not support these functions.

Exporting the public key

Execute the **tklmCertExport** command to export the public key to a regular file in an existing directory on the local filesystem. Use a filename that contains the name of the key to avoid confusion. You have to specify the `uuid` to select the proper certification to export. This `uuid` value can be obtained from **tklmCertList** output as shown in Example 5-10 on page 109.

Example 5-12 and Example 5-13 show how to export a public key.

Example 5-12 Export IKS public key into a file

```
wsadmin>print AdminTask.tklmCertExport('[-uuid
CERTIFICATE-252c1687-6f03-4048-97e0-fc85003aadee -fileName /root/itsokey1.cert]')
CTGKM0001I Command succeeded.
/root/itsokey1.cert
```

Example 5-13 Export Tivoli Key Lifecycle Manager for z/OS public key into a file

```
wsadmin>print AdminTask.tklmCertExport('[-uuid
CERTIFICATE-0eba963e-873d-4cf2-9666-bd8cce196974 -fileName /root/itsokey2.cert]')
CTGKM0001I Command succeeded.
/root/itsokey2.cert
```

Transferring the key files to the other server

The exported keys are regular files on the filesystem. How those can be transferred depends on the operating systems. IKS uses SLES v10, and wide range of options are available. If possible, copy the files over the network by using a secure transfer method like SCP or SFTP.

In this example, the `itsokey1.cert` file from IKS is transferred to Tivoli Key Lifecycle Manager for z/OS server, and `itsokey2.cert` file from Tivoli Key Lifecycle Manager for z/OS server is transferred to IKS.

Important: If any kind of media (floppy or compact disk, or USB pen drive) was involved during the file transfer, make sure that it is permanently erased. Do not leave any temporary files outside of the Tivoli Key Lifecycle Manager servers.

Importing the public key

Use the **tklmCertImport** command to import the public key file into your keystore. The import command includes an **alias** parameter. It specifies a unique name for the certificate. You have to give the same certification name as used on the partner server to keep the Tivoli Key Lifecycle Manager infrastructure consistent. In Example 5-14 on page 112, we import the Tivoli Key Lifecycle Manager for z/OS public key on the IKS server and we use for **alias** (certificate key label) the same name we used to define certificate key label on Tivoli Key Lifecycle Manager for z/OS, which is `itsokey2`.

Similarly, in Example 5-15, we import the IKS public key on the Tivoli Key Lifecycle Manager for z/OS server and we use for **alias** (certificate key label) the same name we used to define certificate key label on IKS, that is `itsokey1`.

Example 5-14 Import the Tivoli Key Lifecycle Manager for z/OS public key on the IKS

```
wsadmin>print AdminTask.tklmCertImport('[-fileName /root/itsokey2.cert -alias
itsokey2 -keyStoreName "ITSO Sample Keystore" -usage DS8K]')
CTGKM0001I Command succeeded.
```

Example 5-15 Import the IKS public key on the Tivoli Key Lifecycle Manager for z/OS server

```
wsadmin>print AdminTask.tklmCertImport('[-fileName /root/itsokey1.cert -alias
itsokey1 -keyStoreName "ITSO Sample Keystore" -usage DS8K]')
CTGKM0001I Command succeeded.
```

Key exchange verification

At this time, a a good practice is to check the content of the keystores before moving forward.

Both Tivoli Key Lifecycle Manager servers must have a complete key pair (public and private) and a single public key from the partner server. So, altogether two certificates with three keys inside are stored on each server. Example 5-16 and Example 5-17 on page 113 demonstrate the content of the IKS server.

Example 5-16 List of certifications on the IKS server after key exchange

```
wsadmin>print AdminTask.tklmCertList('[-v y]')
CTGKM0001I Command succeeded.
CTGKM0661I Found 2 certificates.

uuid                    CERTIFICATE-252c1687-6f03-4048-97e0-fc85003aadee
alias(es)               itsokey1
certifications          null
information             null
key store name(s)       ITSO Sample Keystore
key store uuid(s)       KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
owner                   null
key state               active
issuer name             CN=itsokey1
subject name            CN=itsokey1
target public key uuid  null
activation date         Sep 23, 2009
archive date            Sep 16, 2037
compromise date         null
creation date           Sep 23, 2009
expiration date         Sep 22, 2012
destroy date            null
retirement date         Sep 17, 2032
serial number           1253745487459693000

uuid                    CERTIFICATE-7e18fcd9-402d-4d26-856e-912e7917ee7c
alias(es)               itsokey2
certifications          null
information             null
key store name(s)       ITSO Sample Keystore
key store uuid(s)       KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
```

```
owner                   null
key state               active
issuer name             CN=itsokey2,OU=,O=,L=,ST=,C=
subject name            CN=itsokey2,OU=,O=,L=,ST=,C=
target public key uuid  null
activation date         Sep 23, 2009
archive date            Sep 16, 2037
compromise date         null
creation date           Sep 23, 2009
expiration date         Sep 22, 2012
destroy date            null
retirement date         Sep 17, 2032
serial number           1253760544252699000
```

Example 5-17 List of keys on the primary server after key exchange

```
wsadmin>print AdminTask.tklmKeyList ('[-v y]')
CTGKM0001I Command succeeded.
CTGKM0710I Found 3 keys.

uuid                   KEY-1ffcc23b-6ecd-4111-be66-e5e6d1ecbb12
information            null
alias(es)              itsokey1
key algorithm          AlgorithmName
key length (in bits)   2048
key type               RSAPrivateKey
owner                  null
certifications         null
key store name(s)      ITSO Sample Keystore
key store uuid(s)      KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
key state              active
activation date        Sep 23, 2009
archive date           Sep 16, 2037
compromise date        null
creation date          Sep 23, 2009
expiration date        Sep 22, 2012
destroy date           null
retirement date        Sep 17, 2032
hash value             0000: 40 8e 4f eb e7 44 78 50  e7 9c 2f bd 4c 65 8d 60
..O..DxP....Le..


uuid                   KEY-c0abd46e-32c1-4524-b8a1-3ac70ddce66b
information            null
alias(es)              itsokey1
key algorithm          AlgorithmName
key length (in bits)   2048
key type               RSAPublicKey
owner                  null
certifications         null
key store name(s)      ITSO Sample Keystore
key store uuid(s)      KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
key state              active
activation date        Sep 23, 2009
archive date           Sep 16, 2037
```

```
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: b0 f5 7d ab eb 85 f1 85  32 e3 82 bc 75 b1 05 b3
........2...u...


uuid                  KEY-da187ac1-95b1-4c94-8836-a3fa45015794
information           null
alias(es)             itsokey2
key algorithm         RSA
key length (in bits)  2048
key type              RSAPublicKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-83bd0463-92e9-4e3d-8380-345107d60300
key state             pre-active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: e3 29 8a 0d 5c f9 64 e8  9b 17 80 09 bc 5a 18 7e
......d......Z..
```

On the IKS server, both the public and private key of `itsokey1` and only the public key of `itsokey2` exist, which complies with the requirements.

Example 5-18 and Example 5-19 on page 115 show the state of the Tivoli Key Lifecycle Manager for z/OS server.

Example 5-18 List of certifications on the Tivoli Key Lifecycle Manager for z/OS server after key exchange

```
wsadmin>print AdminTask.tklmCertList('[-v y]')
CTGKM0001I Command succeeded.
CTGKM0661I Found 2 certificates.

uuid                    CERTIFICATE-0eba963e-873d-4cf2-9666-bd8cce196974
alias(es)               itsokey2
certifications          null
information             null
key store name(s)       ITSO Sample Keystore
key store uuid(s)       KEYSTORE-0c8fd4ca-c21f-4cc9-8487-c1d734e275af
owner                   null
key state               active
issuer name             CN=itsokey2, OU=, O=, L=, ST=, C=
subject name            CN=itsokey2, OU=, O=, L=, ST=, C=
target public key uuid  null
activation date         Sep 23, 2009
archive date            Sep 16, 2037
```

```
compromise date         null
creation date           Sep 23, 2009
expiration date         Sep 22, 2012
destroy date            null
retirement date         Sep 17, 2032
serial number           1253760544252699000

uuid                    CERTIFICATE-5fd17bda-6131-4386-beca-856f616485b2
alias(es)               itsokey1
certifications          null
information             null
key store name(s)       ITSO Sample Keystore
key store uuid(s)       KEYSTORE-0c8fd4ca-c21f-4cc9-8487-c1d734e275af
owner                   null
key state               active
issuer name             CN=itsokey1
subject name            CN=itsokey1
target public key uuid  null
activation date         Sep 23, 2009
archive date            Sep 16, 2037
compromise date         null
creation date           Sep 23, 2009
expiration date         Sep 22, 2012
destroy date            null
retirement date         Sep 17, 2032
serial number           1253745487459693000
```

Example 5-19 List of keys on the Tivoli Key Lifecycle Manager for z/OS server after key exchange

```
wsadmin>print AdminTask.tklmKeyList ('[-attributes "{state active}" -v y]')
CTGKM0001I Command succeeded.
CTGKM0710I Found 3 keys.

uuid                  KEY-4ed8b668-5d61-4311-85be-c639a02a75cd
information           null
alias(es)             itsokey2
key algorithm         AlgorithmName
key length (in bits)  2048
key type              RSAPrivateKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-0c8fd4ca-c21f-4cc9-8487-c1d734e275af
key state             active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: dc 10 49 33 b2 48 9d e5  81 28 9f a6 f5 59 50 c0
..I3.H.......YP.
```

```
uuid                  KEY-c7c3a79b-d1b7-44c2-99e3-dd04ef0d2b42
information           null
alias(es)             itsokey2
key algorithm         AlgorithmName
key length (in bits)  2048
key type              RSAPublicKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-0c8fd4ca-c21f-4cc9-8487-c1d734e275af
key state             active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: e3 29 8a 0d 5c f9 64 e8  9b 17 80 09 bc 5a 18 7e
......d......Z..


uuid                  KEY-79fd184f-888e-43ad-8b15-990a3edaa8da
information           null
alias(es)             itsokey1
key algorithm         RSA
key length (in bits)  2048
key type              RSAPublicKey
owner                 null
certifications        null
key store name(s)     ITSO Sample Keystore
key store uuid(s)     KEYSTORE-0c8fd4ca-c21f-4cc9-8487-c1d734e275af
key state             active
activation date       Sep 23, 2009
archive date          Sep 16, 2037
compromise date       null
creation date         Sep 23, 2009
expiration date       Sep 22, 2012
destroy date          null
retirement date       Sep 17, 2032
hash value            0000: b0 f5 7d ab eb 85 f1 85  32 e3 82 bc 75 b1 05 b3
........2...u...
```

The number of the certifications and keys are correct. If we compare the outputs of **tklmCertList** and **tklmKeyList** commands, we see that the adequate certification serial numbers and key hash values match with each pair, which proves that the keys are exactly the same replicas.

5.4.3 Adding DS8700 storage image

After completing the key exchange process, the Tivoli Key Lifecycle Manager servers are ready to identify new DS8700 Storage Facility Images (SFI) with dual keys. In this step, we can use the GUI again, however the Tivoli Key Lifecycle Manager **tklmDeviceAdd** CLI command is also available.

Figure 5-13 shows the sample GUI windows on both servers.

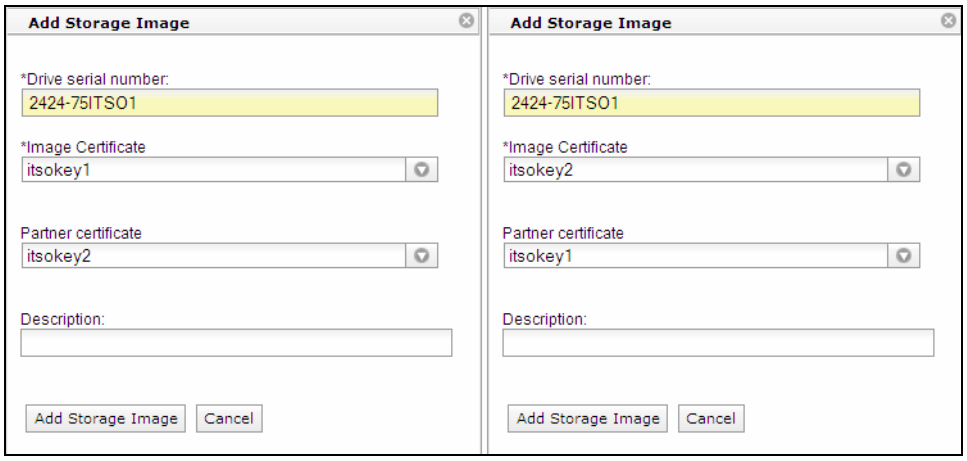


Figure 5-13 Adding DS8700 Storage Image on IKS (left) and Tivoli Key Lifecycle Manager for z/OS (right) servers

Notice that on the IKS server (left), the IKS certificate key label (`itsokey1`) is selected for Image Certificate and the Tivoli Key Lifecycle Manager for z/OS certificate key label (`itsokey2`) is assigned as Partner Certificate. On the Tivoli Key Lifecycle Manager for z/OS server side (right), the certificate key label relationship is reversed. The `itsokey2` is defined as Image Certificate and the `itsokey1` is the Partner certificate.

Click the **Add Storage Image** button on both servers.

On the IKS server, we can verify the result, as shown in Figure 5-14.

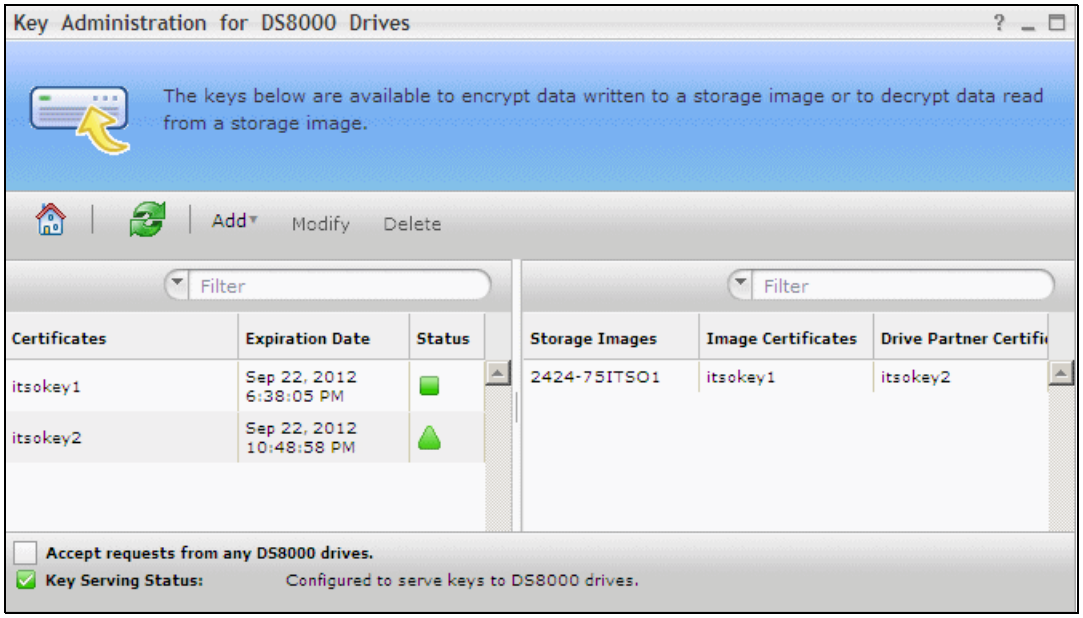


Figure 5-14 Configured DS8700 Storage Image on IKS server

Figure 5-15 on page 118 shows the Tivoli Key Lifecycle Manager for z/OS server state.

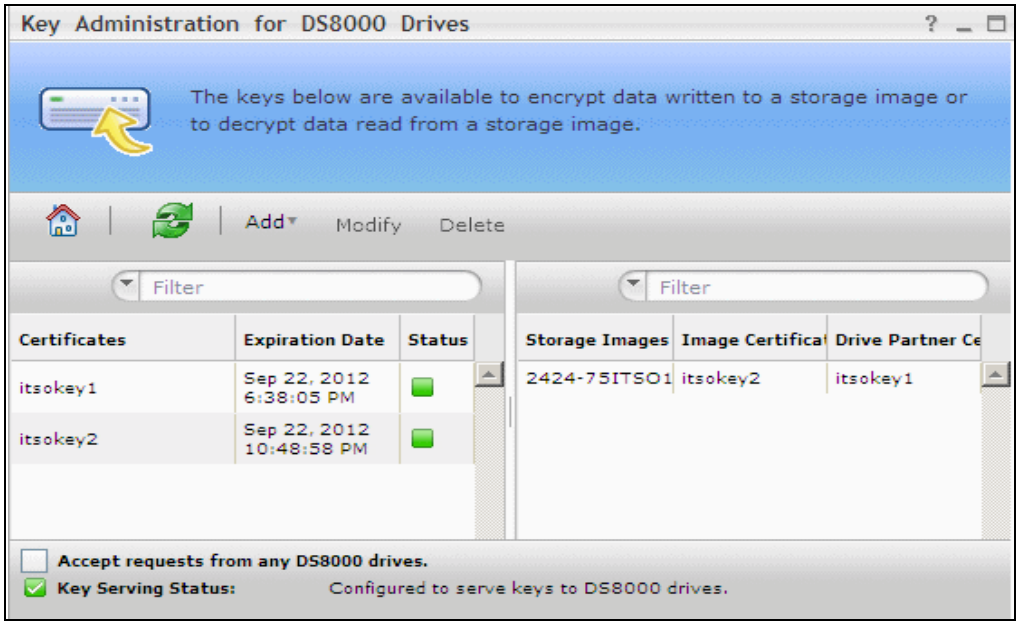


Figure 5-15 Configured DS8700 Storage Image on Tivoli Key Lifecycle Manager for z/OS server

Now the Tivoli Key Lifecycle Manager infrastructure is ready for serving keys to authorized storage images. The same certificates names `itsokey1` and `itsokey2` are used when configuring the encryption group on the DS8700. They are referred to as *Key Label* and *Secondary Key Label* as described in 4.2.4, “Creating the encryption group” on page 73.

5.5 Rekey data key

With Licensed Internal Code level 65.10.xx or later, the Rekey Data Key option is available on DS8700. This option allows a user with the storage administrator role to rekey the data key labels for a encryption group. A customer might want to use this function to periodically change the data key labels.

The following procedure describes how to rekey the data key labels. Let assume that the new data key label `itsokey2` has already been defined in the Tivoli Key Lifecycle Manager servers. Navigate to the **Manage Hardware** → **Encryption** → **Groups** panel and select the **Rekey Data Key** menu entry, as shown in Figure 5-16.

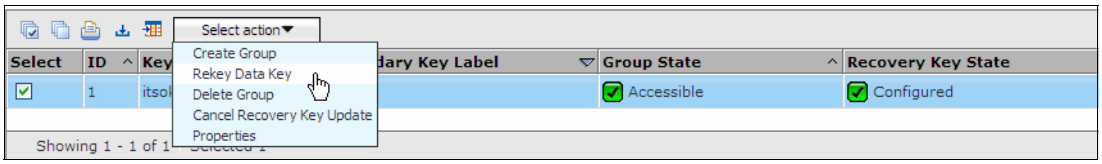


Figure 5-16 Starting the Rekey Data Key function

The Rekey Data Key window opens, shown in Figure 5-17 on page 119. Type the new data key label into the field **Key Label** and then click **Rekey**. Note that the **Secondary Key Label** field must be used only when the dual key server support is enabled (see 5.4, “Tivoli Key Lifecycle Manager dual platform support implementation” on page 107)

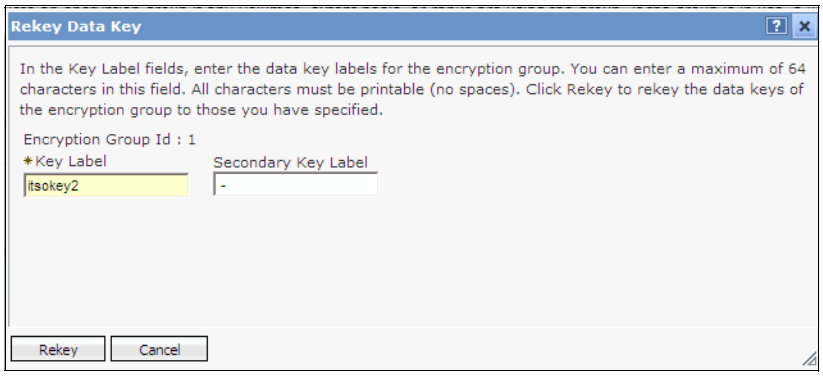


Figure 5-17 Rekey Data Key window

If the key is valid, a successful confirmation message is displayed; see Figure 5-18.

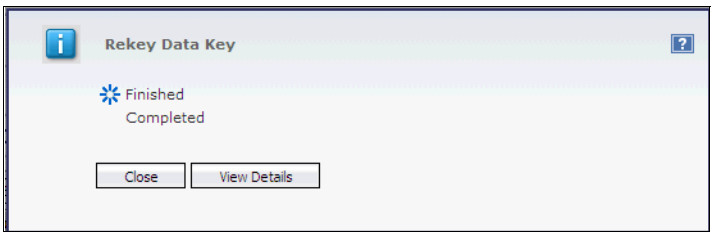


Figure 5-18 Rekey Data Key action completed

The encryption group main panel shows the new data key labels; see Figure 5-19.

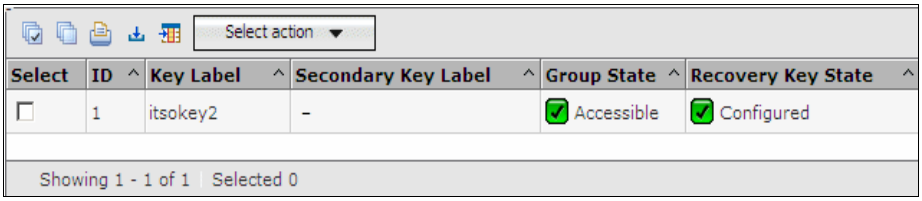


Figure 5-19 Encryption group main panel

5.6 Recovery key usage and maintenance

Starting with the Licensed Internal Code level 65.10.xx.xx, two recovery-key-related options are available:

- Recovery key enabling
- Recovery key disabling

When the recovery key is enabled several functions are available to manage and exploit the recovery key after its creation:

- Validate recovery key
- Initiate recovery key
- Re-key recovery key
- Delete recovery key

When the recovery key is disabled, the recovery key enabling is still possible.

We describe the details of each in this session.

5.6.1 Validate recovery key

Part of the recovery key creation process is the verification of a newly created recovery key. Verification ensures that the recovery key was written correctly. However, after the encryption environment is operational, the recovery key will be used only in the deadlock situation. Therefore, verify that the stored recovery-key key is still valid. The validation process can be performed occasionally and it can be included in your task list for the maintenance of the encryption environment. The security administrator is able to validate the recovery key, at any time. Only users with security administrator authority can validate the recovery key. Verifying the recovery key does not change anything in the system and the storage administrator approval is not needed.

Navigate to the **Manage Hardware** → **Encryption** → **Groups** panel to select the **Validate Recovery Key** menu entry, as shown in Figure 5-20.

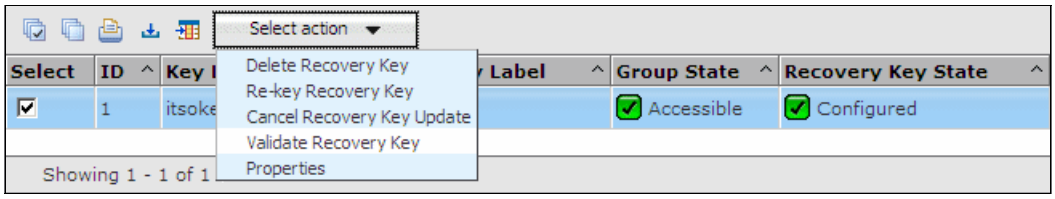


Figure 5-20 Starting the Validate Recovery Key function

The Validate Recovery Key window opens, shown in Figure 5-21. Type the key into the field and then click **Validate**.

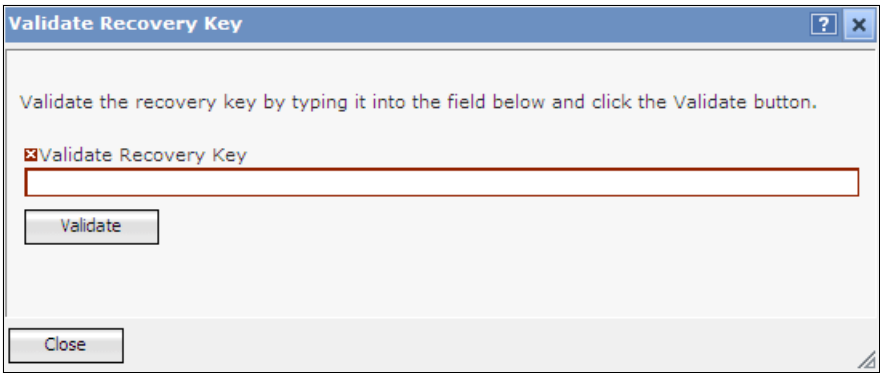


Figure 5-21 Validate Recovery Key window

If the key is valid, a successful confirmation message is displayed below the **Validate** button, as shown in Figure 5-22 on page 121.

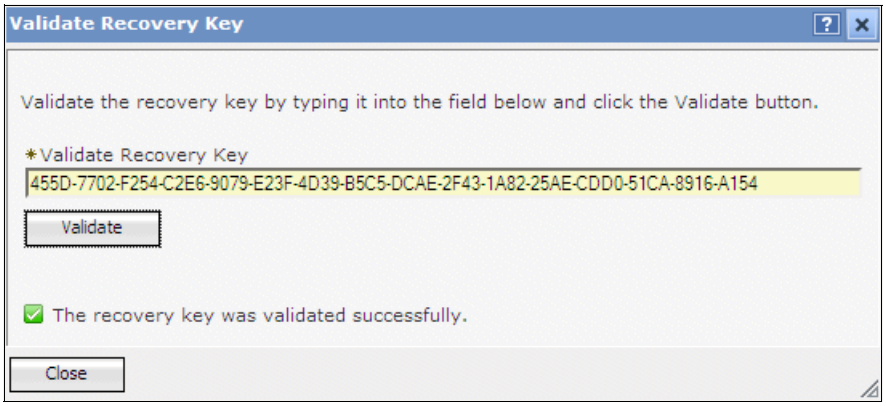


Figure 5-22 The recovery key was validated successfully

The flowchart of this process is simple, as shown in Figure 5-23.

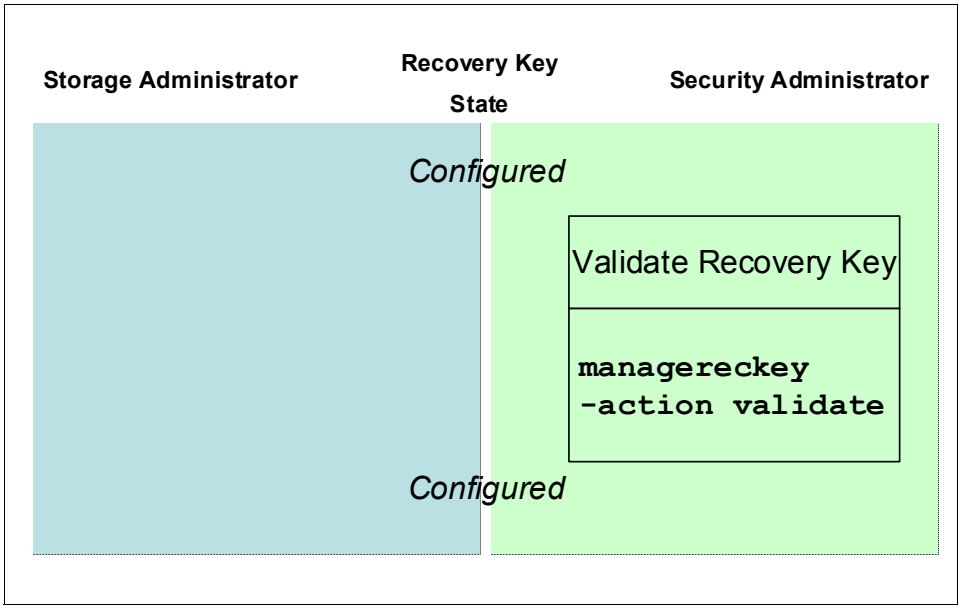


Figure 5-23 Validate recovery key flowchart

5.6.2 Initiate recovery key

If the Tivoli Key Lifecycle Manager servers are down or inaccessible for any reason, the DS8700 Storage Facility Image (SFI) cannot be started, because without the keys the storage remains in locked mode. In this situation, two choices are available:

- Repair at least one of the Tivoli Key Lifecycle Manager servers or its network connectivity to be able to serve the necessary key for the DS8700.
- Initiate the recovery process to unlock the DS8700 and to let the volumes be accessible again.

Consider the first option and check the Tivoli Key Lifecycle Manager servers' status first. Try to fix the problem with Tivoli Key Lifecycle Manager servers if the problem is not complex, and obviously if you have time to meet your service level agreement. Otherwise, use the recovery key to initiate the process to unlock DS8700 volumes.

Simulating Tivoli Key Lifecycle Manager failure

In this section, we walk through a real-life example to demonstrate how the deadlock recovery works. You can use this scenario soon after implementing encryption in your environment to test and document system recovery in a deadlock situation.

As a first step in our example, we shut down all Tivoli Key Lifecycle Manager servers connected to the DS8700. You can check the Tivoli Key Lifecycle Manager status from the DS8700 Storage Manager GUI. Navigate to **Manage Hardware** → **Encryption** → **Key Servers**. See Figure 5-24.

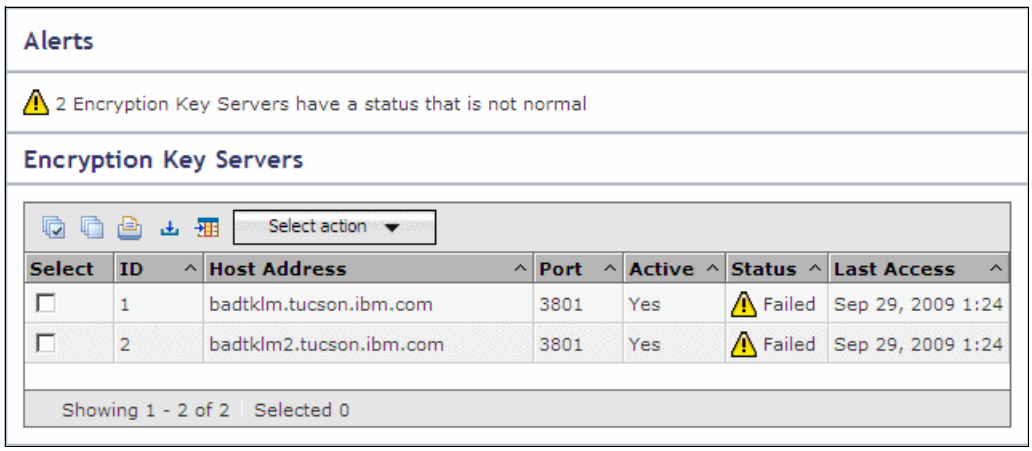


Figure 5-24 Tivoli Key Lifecycle Manager servers status is Failed

The DS8700 Hardware Management Console (HMC) monitors the availability of the Tivoli Key Lifecycle Manager servers. The connection is verified every five minutes; the Last Access column in Figure 5-24 shows the last time stamp of when the Tivoli Key Lifecycle Manager server answered to the heartbeat request. If the HMC detects an outage of a Tivoli Key Lifecycle Manager server, the `BE14EAF1` SRC is reported (see Figure 5-25), which is only an early notification. When the outage exceeds the four-hour limit, the `BE14EAF2` error is reported as a call-home event, so both the customer and IBM are alerted of this incident.

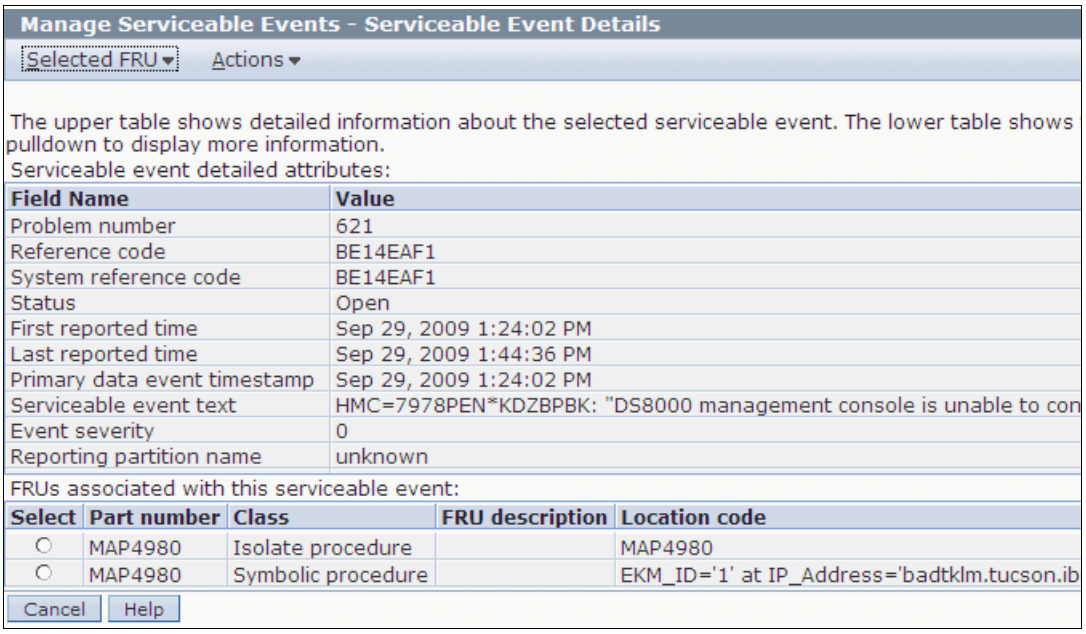


Figure 5-25 BE14EAF1 SRC on DS8700 HMC

The Tivoli Key Lifecycle Manager servers are accessed only when the key is required for the DS8700 (excluding the heartbeat verification). In most cases, this happens when the DS8700 is booting up, but several of the reliability, availability, and serviceability (RAS) functions, such as Concurrent Code Load (CCL), also trigger a Tivoli Key Lifecycle Manager key retrieval. The easiest way to reach this point is to turn off the DS8700 by using the DS8700 HMC GUI.

If we start the DS8700 again while the Tivoli Key Lifecycle Manager servers are still down, we notice that the initialization progress is much slower. The reason is that some warm starts are initiated to configure the storage devices. However, without getting the necessary keys from the Tivoli Key Lifecycle Manager servers, the configuration phase fails, and the DS8700 has to give up and report the failure.

Possible SRC codes are as follows:

- `BE14E008`: Unable to retrieve keys from any configured encryption key management servers because of communication errors between the HMC and encryption key management server (or servers). Figure 5-26 shows a sample SRC.

Manage Serviceable Events - Serviceable Event Details

Selected FRU ▾ Actions ▾

The upper table shows detailed information about the selected serviceable event. The lower table shows t

Serviceable event detailed attributes:

Field Name	Value
Problem number	624
Reference code	BE14E008
System reference code	BE14E008
Status	Open
First reported time	Sep 29, 2009 2:12:53 PM
Last reported time	Sep 29, 2009 2:12:53 PM
Primary data event timestamp	Sep 29, 2009 2:10:54 PM
Serviceable event text	EncryptionKeyManagerServer error: Unable to retrieve keys from any conf
Event severity	0
Reporting partition ID	001

FRUs associated with this serviceable event:

Select	Part number	Class	FRU description	Location code
○	MAP4980	Isolate procedure		MAP4980
○	MAP4980	Symbolic procedure		EKM_ID='1' at IP_Address='badtklm.tucson.ib
○	MAP4980	Symbolic procedure		EKM_ID='2' at IP_Address='badtklm2.tucson.i
○	MAP4980	Symbolic procedure		EKM_ID='0' at IP_Address='null'

Figure 5-26 BE14E008 SRC on DS8700 HMC

- `BE1E2058`: Config Error - Global Data Inaccessible is shown in Figure 5-27 on page 124.

Manage Serviceable Events - Serviceable Event Details

Selected FRU Actions

The upper table shows detailed information about the selected serviceab
associated with this event. Select a FRU and use the "Selected FRU" pulld
Serviceable event detailed attributes:

Field Name	Value
Problem number	623
Reference code	BE1E2058
System reference code	BE1E2058
Status	Open
First reported time	Sep 29, 2009 2:12:55 PM
Last reported time	Sep 29, 2009 2:12:55 PM
Primary data event timestamp	Sep 29, 2009 2:11:01 PM
Serviceable event text	Config Error - Global Data Inaccessible
Event severity	0
Reporting partition ID	001

FRUs associated with this serviceable event:

Select	Part number	Class	FRU description	Location code
○	MAP4970	Isolate procedure		n/a

Figure 5-27 BE1E2058 SRC on DS8700 HMC

You can check the encryption group status from the DS8700 GUI. Navigate to **Manage Hardware** → **Encryption** → **Groups**. The Encryption Group also reflects the Inaccessible state, as shown in Figure 5-28.

Alerts

1 Encryption Groups are Inaccessible

Manage Encryption Groups

Select action

Select	ID	Key Label	Secondary Key Label	Group State	Recovery Key State
☐	1	itsokey1	itsokey2	Inaccessible	Configured

Showing 1 - 1 of 1 Selected 0

Figure 5-28 Encryption Group is Inaccessible

Initiate recovery process

To start the recovery process, log on to DS8700 SM GUI as a user with security administrator authority and go to **Manage Hardware** → **Encryption** → **Groups**. Select your Key Label, and then click **Select action** → **Initiate Recovery**, as highlighted in Figure 5-29.

Select action

- Delete Recovery Key
- Re-key Recovery Key
- Cancel Recovery Key Update
- Validate Recovery Key
- Initiate Recovery
- Properties

Select	ID	Key	Label	Group State	Recovery Key State
☑	1	itsoke		Inaccessible	Configured

Showing 1 - 1 of 1

Figure 5-29 Starting the recovery process

The **Initiate Recovery** window opens (Figure 5-30).

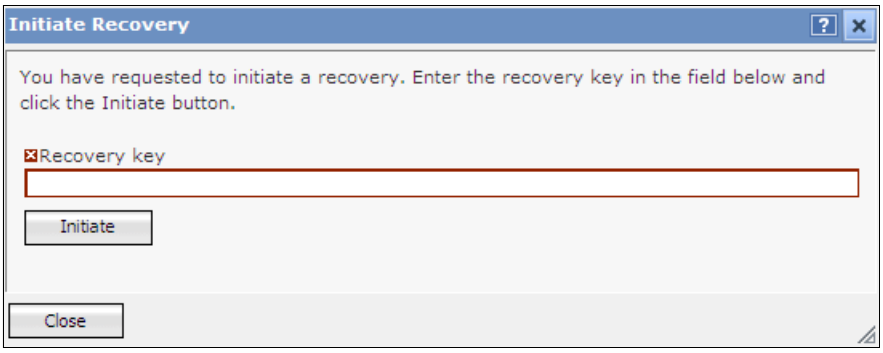


Figure 5-30 Initiate recovery window

The recovery process requires the valid recovery key. Enter the key into the input field in the same way as it was printed on the screen (uppercase characters with dash separation). Then, click **Initiate**. A confirmation message is displayed (Figure 5-31).

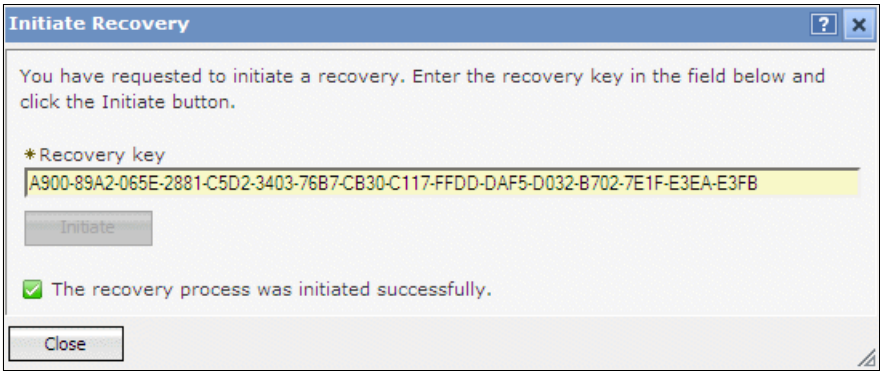


Figure 5-31 The recovery process was initiated successfully

Click **Close** to close the window. Figure 5-32 shows that the Recovery Key State has changed to Request Recovery Authorization state. Group State is still Inaccessible.

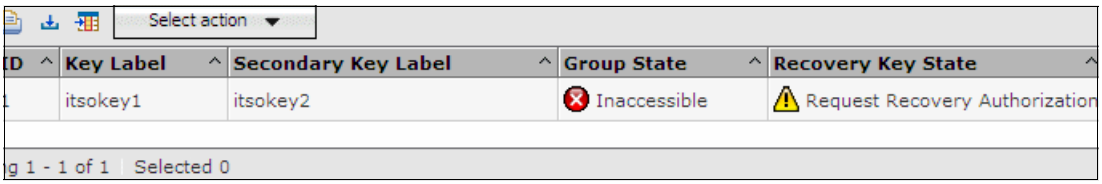


Figure 5-32 Request Recovery Authorization Pending state

At this stage, the user with administrator authority has to approve this recovery request. Therefore, log on as security administrator and go to **Manage Hardware** → **Encryption** → **Key Servers**. Select your key label, and then click **Select action** → **Authorize Recovery Key Update** (Figure 5-33 on page 126) to finalize the recovery.

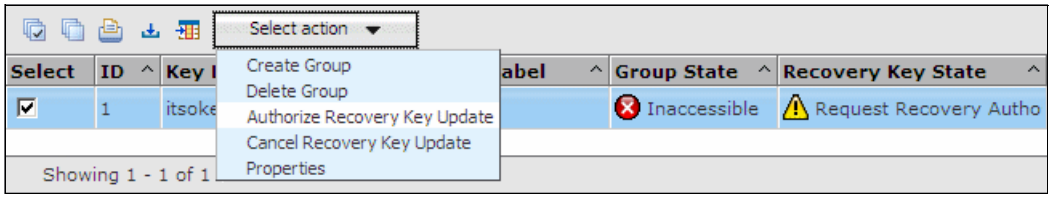


Figure 5-33 Start Authorize Recovery Key Update

The Authorize Input Recovery Key window opens (Figure 5-34).

Figure 5-34 Authorize Input Recovery Key window

If you click the **Authorize Update** button, you get a confirmation message. To complete the Recovery process, the DS8700 must be restarted as indicated in the message shown in Figure 5-35.

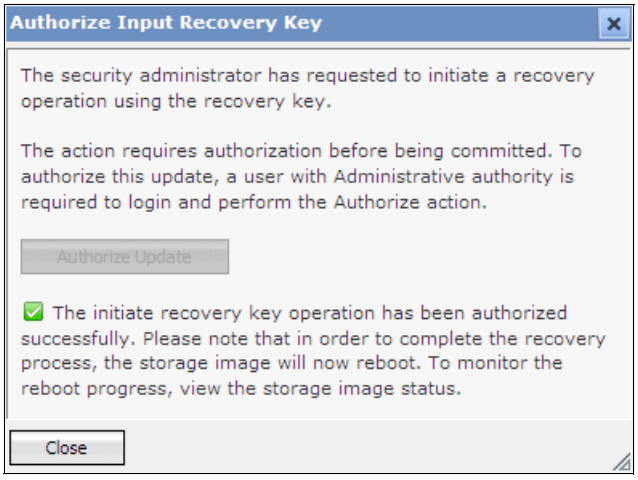


Figure 5-35 Recovery process has been authorized successfully

You can follow the reboot process by viewing the DS8700 status messages at the HMC (Figure 5-36 on page 127).

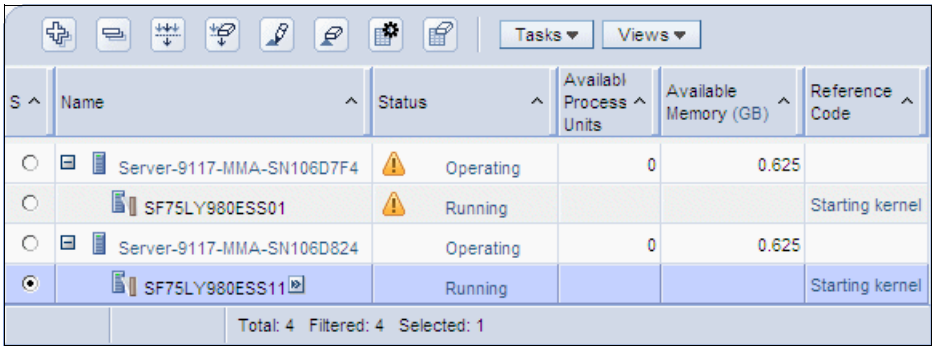


Figure 5-36 SFI reboot is in progress

Depending on the DS8700 configuration, you have to wait several minutes to finish the initialization. When it is ready, the Group State in the Manage Encryption Groups window changes to **Accessible**, as shown in Figure 5-37. This state means that the DS8700 volumes are online and accessible from hosts.

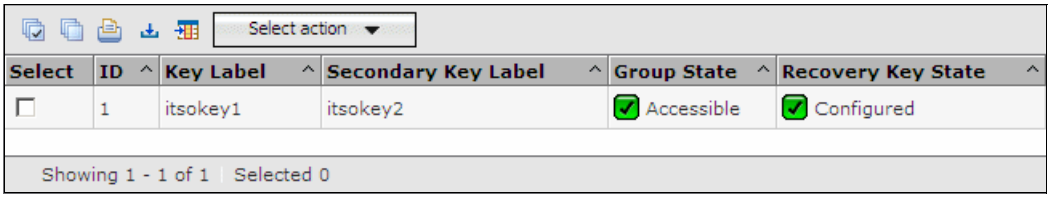


Figure 5-37 Group State is Accessible again

Important: This is not a permanent operation; the recovered DS8700 has been unlocked until the next power cycle only. However, while the system is up and running, the customer has time to repair the Tivoli Key Lifecycle Manager infrastructure. If all the Tivoli Key Lifecycle Manager servers are lost forever (and there is no backup available), the Encryption Group must be re-created in the future, which is a destructive process, and all the customer data must be offloaded first.

Figure 5-38 on page 128 shows the flowchart and corresponding DS CLI commands of the recovery process.

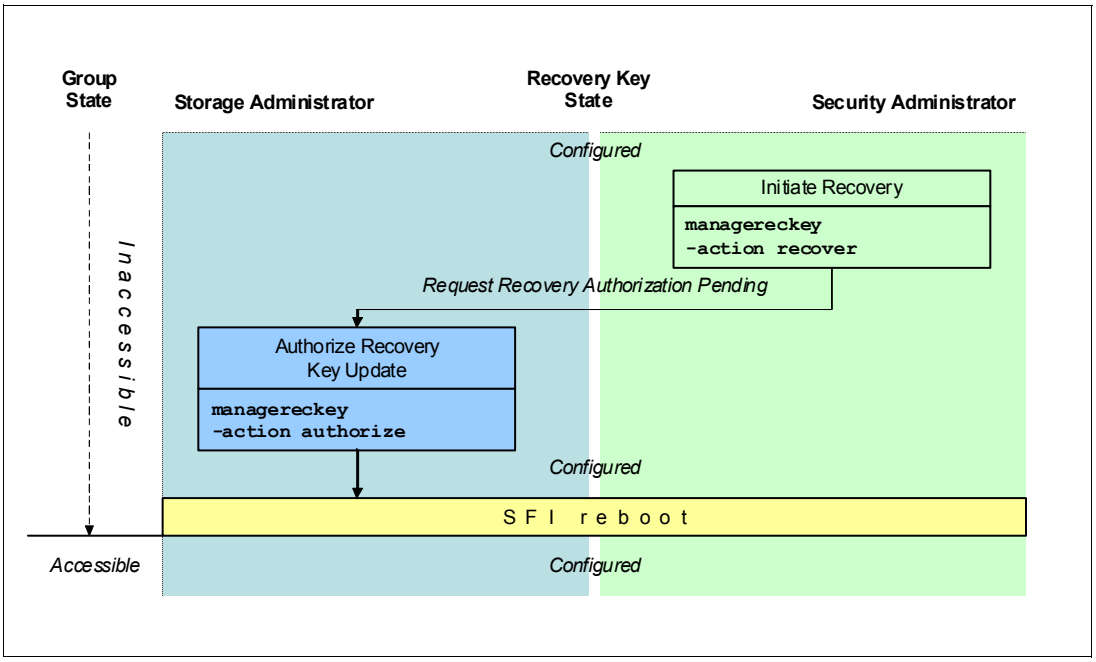


Figure 5-38 Initiate Recovery flowchart

5.6.3 Re-key recovery key

In the case of a lost recovery key, or an unauthorized person is suspected to access the key, the security administrator can decide to regenerate the new recovery key. This means the old key is revoked and it cannot be used anymore. The name of the process is *re-key recovery key*.

Log on to DS8700 SM GUI as a user with security administrator authority and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your Key Label, and then click **Select action** → **Re-key Recovery Key**, as shown in Figure 5-39.

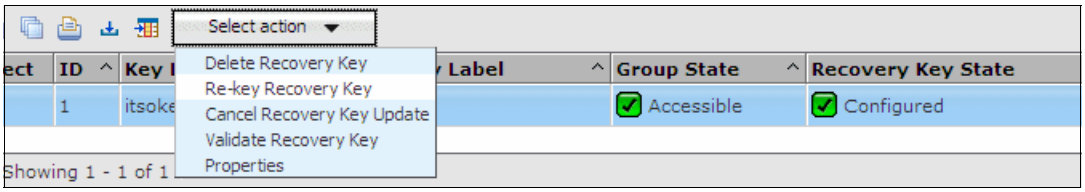


Figure 5-39 Starting the Re-key Recovery Key function

The Re-key Recovery Key window opens, as shown in Figure 5-40.

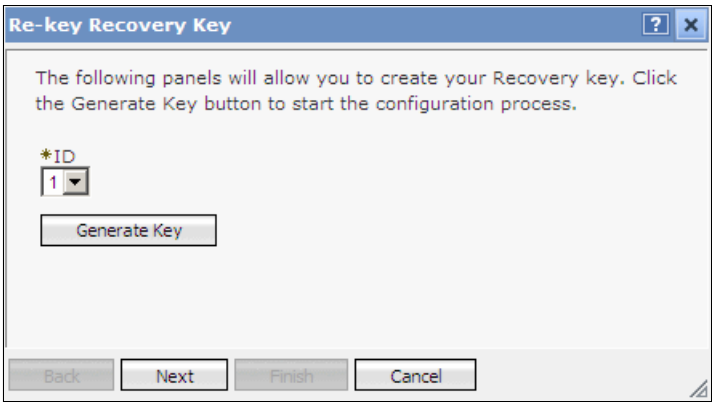


Figure 5-40 Re-key Recovery Key window

Click **Generate Key** to establish a new key, which is then displayed in the window (Figure 5-41).

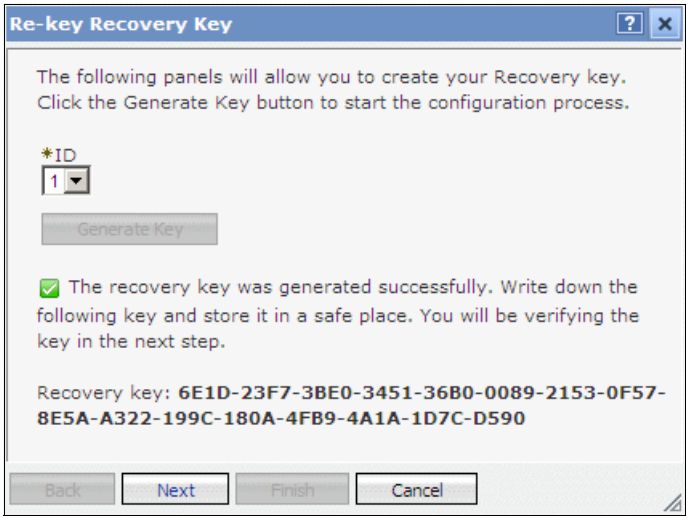


Figure 5-41 Re-key recovery key has been generated

Although the new key replaces the old one, do not shred the old key yet, because the old key is still active until the storage administrator approves the key update. The process is similar to the process of creating a new key (refer to 4.2.3, “Managing recovery key” on page 67). We must verify that the new key is written down correctly. Type the key text into the input field, and click **Verify**; see Figure 5-42 on page 130.

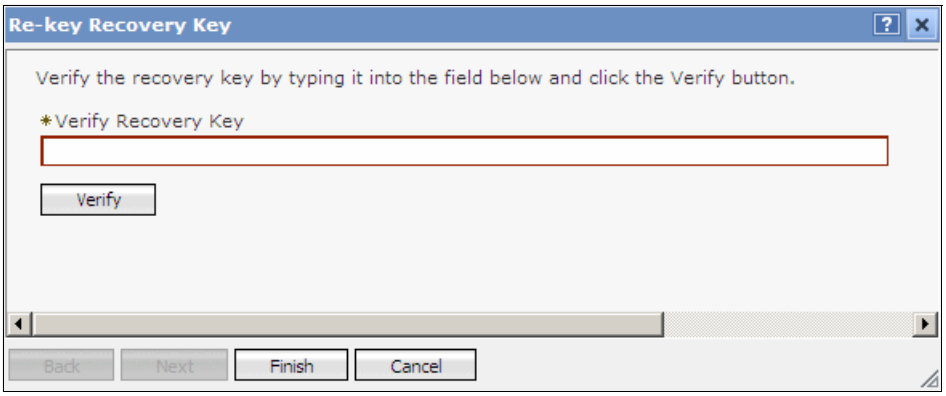


Figure 5-42 Re-key recovery key verification

If the new key is correct, a confirmation message is displayed shortly (Figure 5-43).

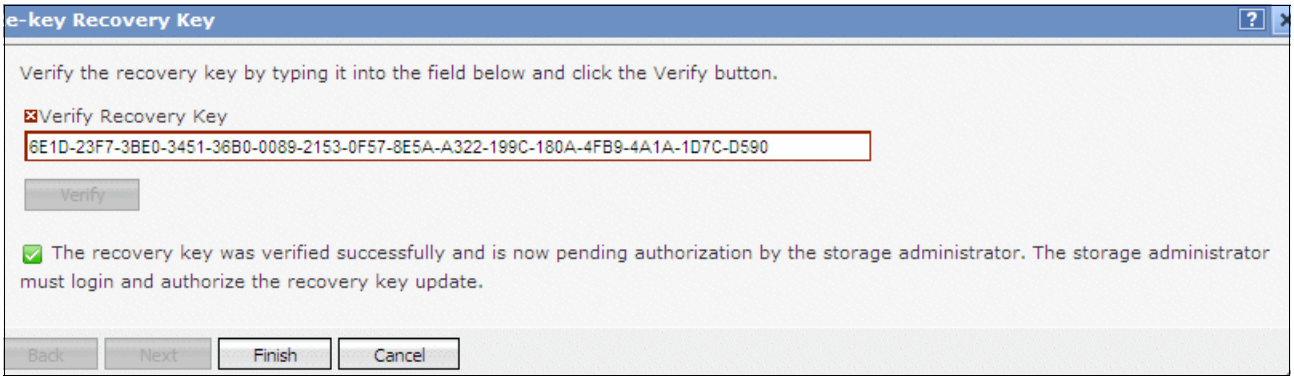


Figure 5-43 Re-key recovery key was verified successfully

Click **Finish** to close the window.

In the Manage Encryption Group window, the Recovery Key State changed to Re-key Authorization Pending state indicating that any user with administrator authority has to approve this re-key request. Log on as a user with administrator authority and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your Key Label, and then click **Select action** → **Authorize Recovery Key Update**, as in Figure 5-44.

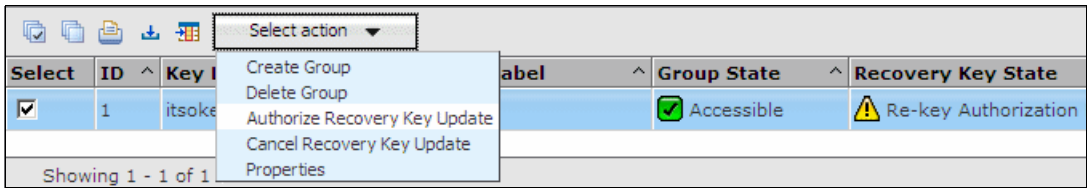


Figure 5-44 Starting the Authorize Recovery Key Update function

The Authorize Recovery Key Update window opens, as shown in Figure 5-45. If you agree with the authorization, click **Authorize Update**.

Figure 5-45 Authorize Recovery Key Update window

Figure 5-46 shows that the new key has been activated.

Figure 5-46 The recovery key has been authorized successfully

Now, only the new recovery key is valid; the old key has been revoked. The state of the key changes back to a Configured state.

The flowchart of the re-key recovery key process is depicted in Figure 5-47 on page 132. The corresponding DS CLI commands are also provided.

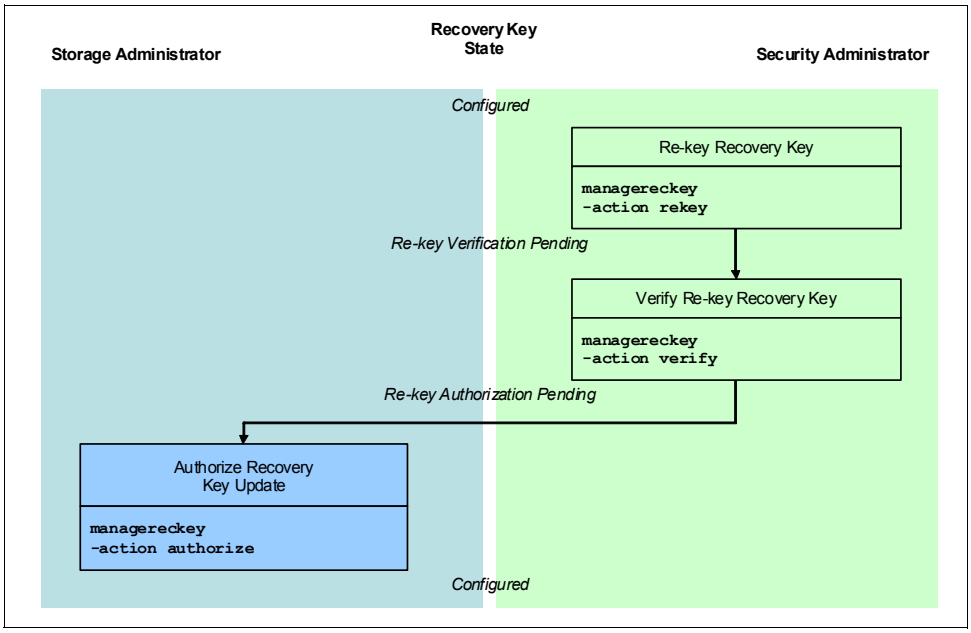


Figure 5-47 Re-key recovery key flowchart

5.6.4 Delete recovery key

Deleting the actual recovery key might be needed only if the customer wants to convert an encryption-enabled DS8700 to an encryption-disabled mode. As a prerequisite, the encryption group must be in **Unconfigured** state, which means that the customer data has been already erased.

Log on to DS8700 GUI as a user with security administrator authority and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your Key Label, and then click **Select action** → **Delete Recovery Key**, as shown in Figure 5-48.

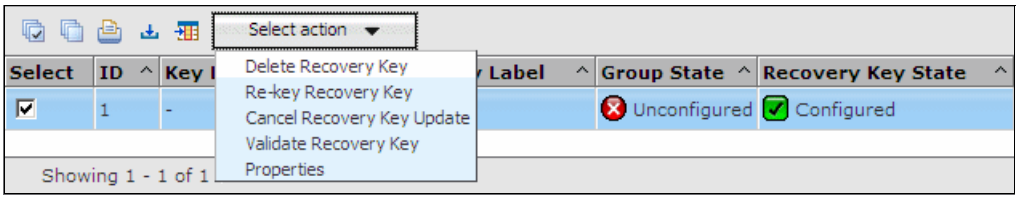


Figure 5-48 Starting the Delete Recovery Key process

The Delete Recovery Key window opens (Figure 5-49).

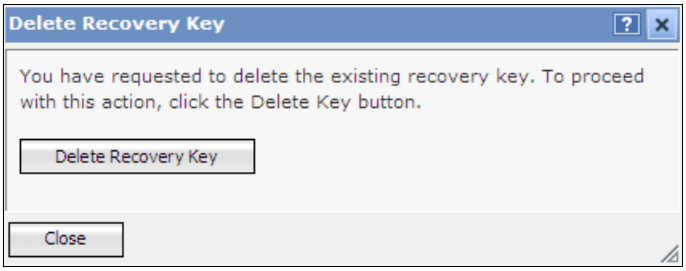


Figure 5-49 Delete Recovery Key window

Click **Delete Recovery Key**. A confirmation message appears below the button, as seen in Figure 5-50. Click **Close** to close the window.

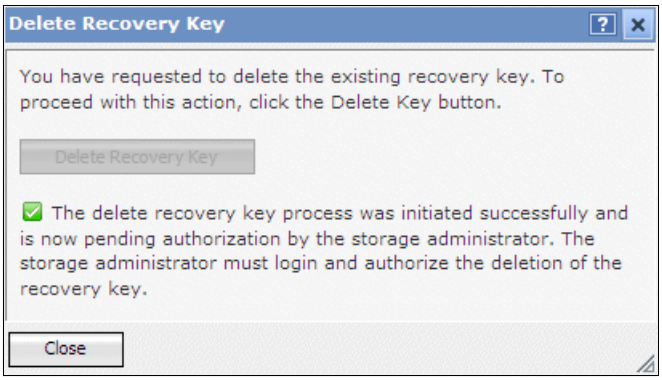


Figure 5-50 Delete recovery key process was initiated successfully

Figure 5-51 shows that the recovery key State changed to a Deconfigure Key Authorization Pending state.

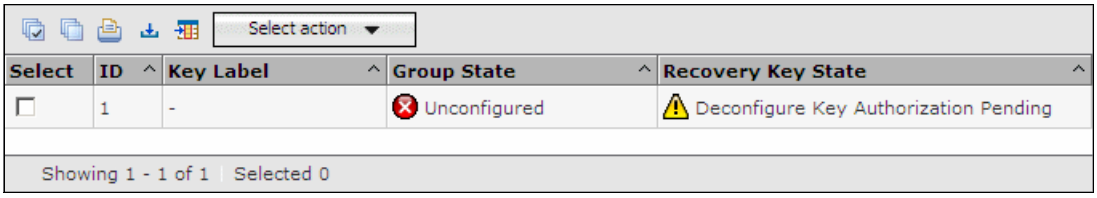


Figure 5-51 Deconfigure Key Authorization Pending

At this stage, the system is waiting for a storage administrator to authorize this request. Log on as a user with administrator authority and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your Key Label, and then click **Select action** → **Authorize Recovery Key Update**, as shown in Figure 5-52.

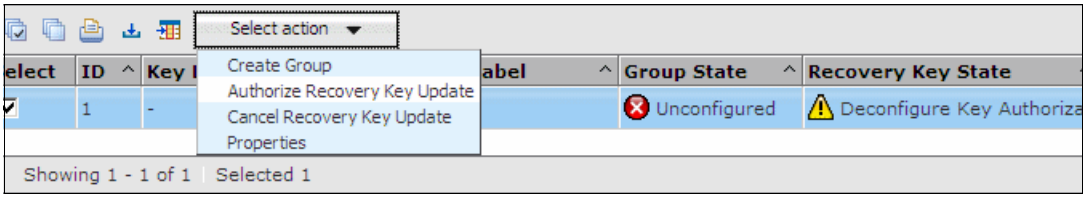


Figure 5-52 Starting the Delete Recovery Key process authorization

The Authorize Recovery Key Update window opens, as shown in Figure 5-53 on page 134.

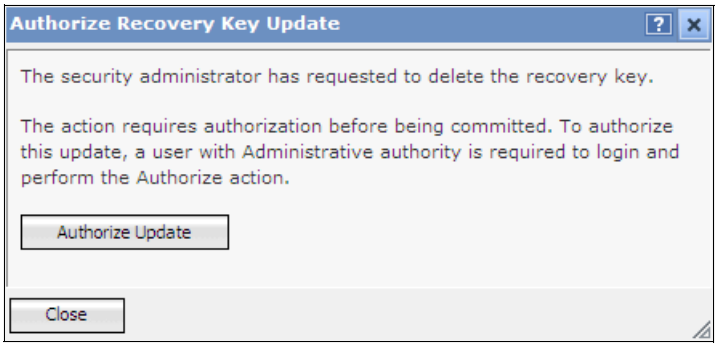


Figure 5-53 Authorize Recovery Key Update window

If you agree, click **Authorize Update** to complete the request. A confirmation message appears, as shown in Figure 5-54.

Figure 5-54 Delete Recovery Key process has been authorized successfully

Finally, the Encryption Group panel is empty as shown in Figure 5-55. Starting from this state, non-encrypted arrays and ranks can be created on this storage system.

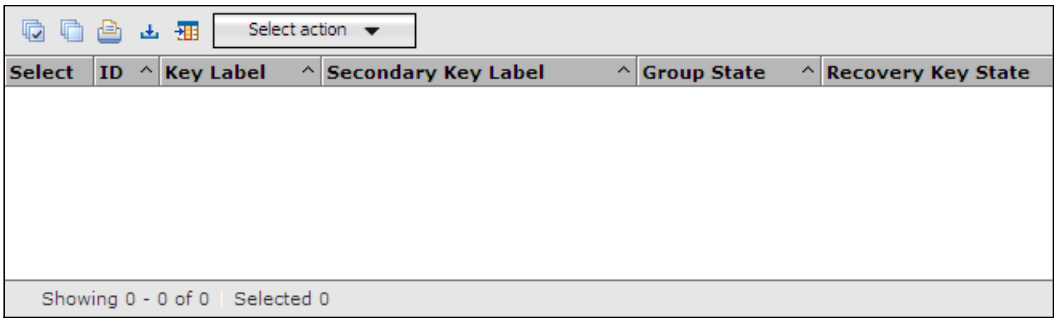


Figure 5-55 Empty Encryption Group panel

The Figure 5-56 on page 135 shows the flowchart and the corresponding DS CLI commands of the delete recovery key process.

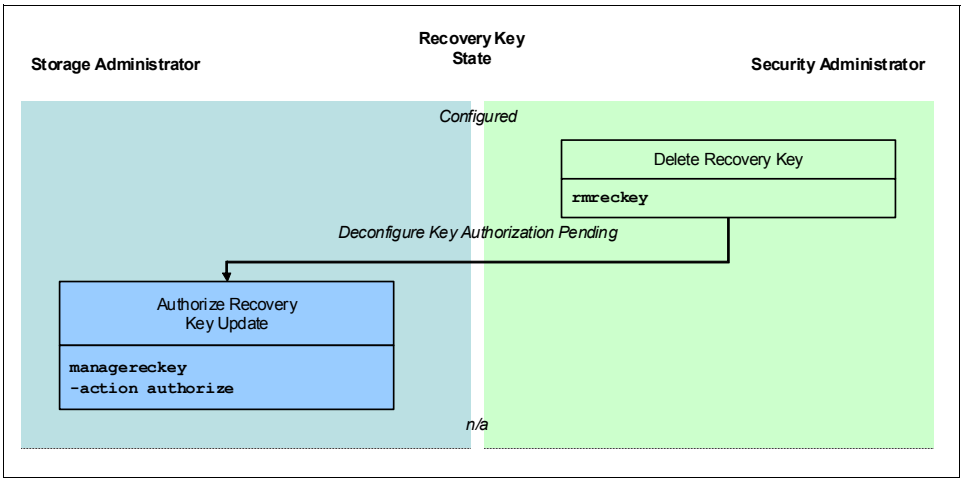


Figure 5-56 Delete recovery key flowchart

5.6.5 Recovery key enabling

The recovery key enabling process allows a user to enable the recovery key management in a scenario where the recovery key management was initially disabled. Because the recovery key must be created before configuring an encryption group, to enable the recovery key management, the encryption group has to be in *unconfigured* state. This means that the customer data has to be erased *before* the recovery key enabling. In other words, the transition between a recovery-key-disabled management to a recovery-key-enabled management implies a full data erasing operation.

To enable the recovery key involves both security administrator and storage administrator roles. Log on to DS8700 GUI as a user with security administrator authority and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your encryption group, and then click **Select action** → **Enable Recovery Key**, as illustrated in Figure 5-57.

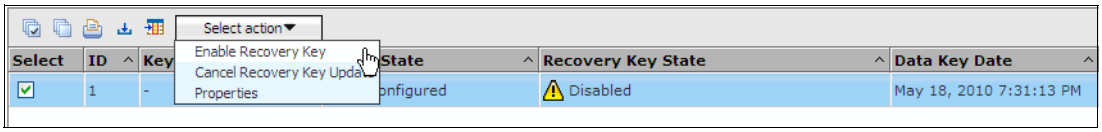


Figure 5-57 enable Recovery Key

The Enable Recovery Key window opens (Figure 5-58).

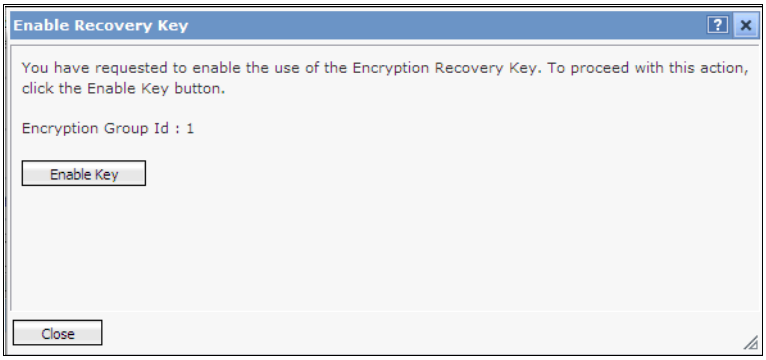


Figure 5-58 Enable Recovery Key window

Click **Enable Key** to request the recovery key enabling, as shown in Figure 5-59.

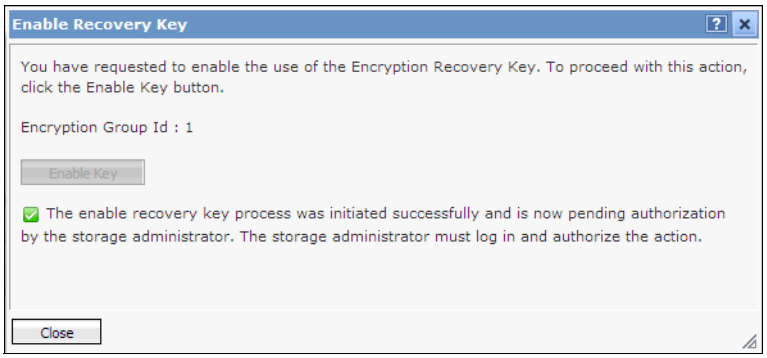


Figure 5-59 Enable Recovery Key window with the request pending message

Log on as storage administrator and navigate to the **Manage Hardware** → **Encryption** → **Groups** panel. Select your encryption group, and then click **Select action** → **Authorize Recovery Key Update**, as illustrated in Figure 5-60.

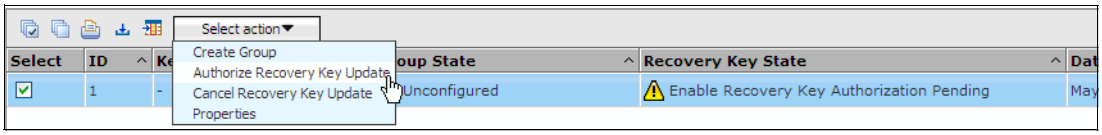


Figure 5-60 Authorize Recovery Key Update

The Authorize recovery key update window opens (Figure 5-61).

Figure 5-61 Authorize Recovery Key Update window

Click **Authorize Update** to authorize the key enabling. A confirmation message is displayed (Figure 5-62).

Figure 5-62 Authorize Recovery Key Update with the confirmation message

The encryption group is removed, as shown in Figure 5-63. Now a recovery key can be requested by using the standard procedure reported in Chapter 4, “DS8700 encryption implementation” on page 49.

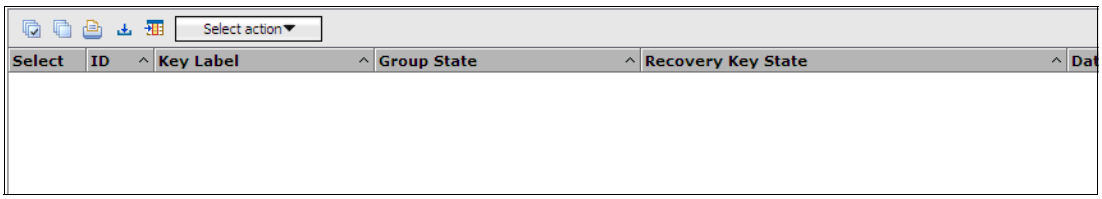


Figure 5-63 Empty encryption group panel

5.6.6 Recovery key state summary

This chapter has introduced recovery key functions. In most cases, the recovery key has multiple temporary states. Figure 5-64 summarizes all possible recovery key states and their relationship.

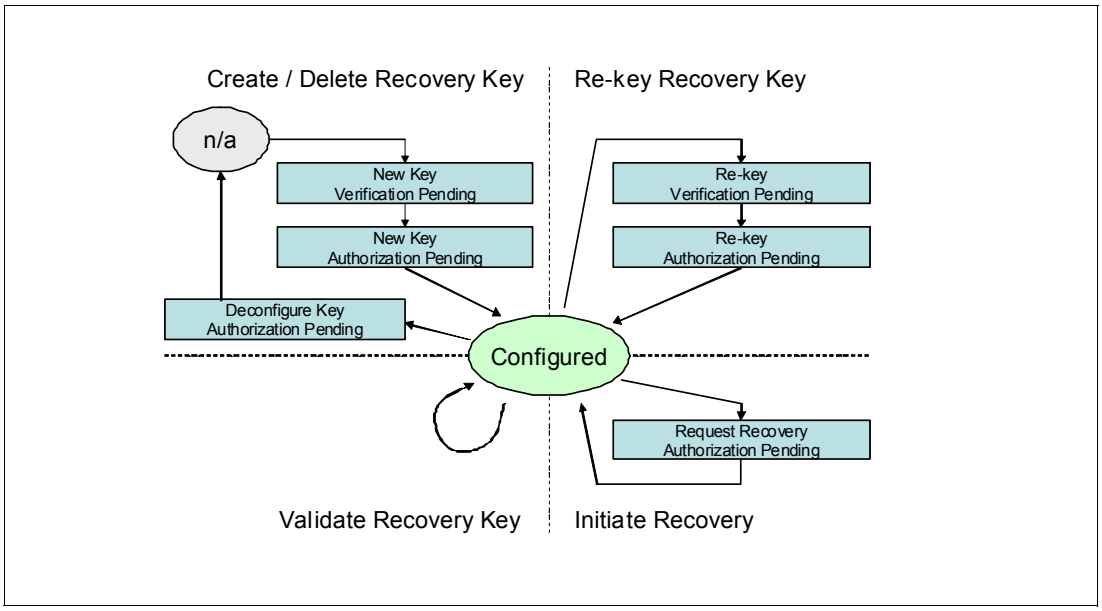


Figure 5-64 Overview of the recovery key states

6

Integrating encryption devices

This chapter describes the implementation of the DS8000 Full Disk Encryption with existing installations of encrypted storage (tape or disk).

This chapter contains the following topics:

- Integration with existing encryption installations
- Migration from EKM to Tivoli Key Lifecycle Manager
- Managing multiple devices

© Copyright IBM Corp. 2010. All rights reserved.

6.1 Integrate with existing tape encryption installations

If you were already using encryption for tape devices with the IBM Encryption Key Manager (EKM) as your key management system, you have to first migrate your EKM key server (or servers) to Tivoli Key Lifecycle Manager before you can integrate the DS8000 encryption feature into your existing environment. Also, remember, that at least two Tivoli Key Lifecycle Manager servers are required for supporting DS8000 encryption.

6.1.1 Migrating from EKM to Tivoli Key Lifecycle Manager

Tivoli Key Lifecycle Manager supports the migration from a EKM-based installation to a Tivoli Key Lifecycle Manager based installation. The migration requires that you stop the EKM server. You can either schedule a maintenance window to shut down the EKM and migrate, or if you have redundant EKMs, you can stage this migration. Refer to the *Tivoli Key Lifecycle Manager Installation and Configuration Guide,* SC23-9977 for detailed information.

The migration procedure can also be found online at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.tklm.doc/install/cpt/cpt_ic_plan_migration.html

The following list contains a quick overview of considerations to be aware of when migrating from EKM to Tivoli Key Lifecycle Manager:

- Back up your EKM configuration prior to migration.
- Write down the path to the EKM. This path must not contain any spaces.
- Schedule an outage for your EKM server. Note that if you have redundant EKMs, you do not have to stop all EKMs; you may stop only the one being migrated to Tivoli Key Lifecycle Manager. After you have migrated the first EKM to Tivoli Key Lifecycle Manager, a best practice is to use the Tivoli Key Lifecycle Manager backup and restore function to configure the remaining Tivoli Key Lifecycle Managers.
- Migration types that are not supported include:
  - Administrator SSL keystores or truststores
  - PKCS11Impl keystores and truststores
- EKM configuration files will be converted to Tivoli Key Lifecycle Manager configuration files.
- Keystore will be migrated.
- Drive table will be migrated into DB2.
- The `Keygroups.xml` file (if it exists) will be migrated into DB2.

The first task after the migration must be a backup of the new Tivoli Key Lifecycle Manager server. Do this before you start any administration tasks or even installing a second TKM server.

After the second Tivoli Key Lifecycle Manager server is up and running (migrated or installed), you start with adding the new DS8000 Systems as described in 4.1.3, “Configuring the DS8700 Storage Facility Image” on page 58.

6.2 Multiple encrypted disk or tape devices

Tivoli Key Lifecycle Manager is able to manage multiple encrypted devices of various types (disk, tape, or others). Tivoli Key Lifecycle Manager manages the encryption key and the nature and number of the encrypted device is irrelevant to Tivoli Key Lifecycle Manager.

There is however a dependency for licensing driven by the number of storage devices or the volume of encrypted storage.

IBM Tivoli product licensing documents are available at:

http://www.ibm.com/software/tivoli/products/licensing.html

Figure 6-1 illustrates a possible configuration for Tivoli Key Lifecycle Manager installation with tape and disk encryption devices.

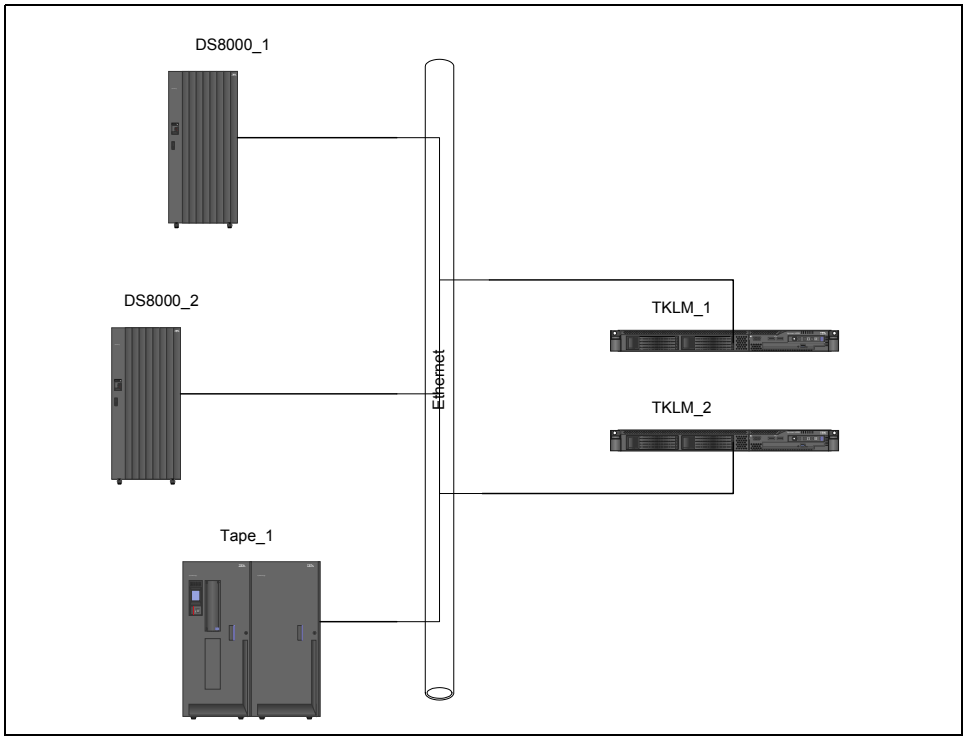


Figure 6-1 Tivoli Key Lifecycle Manager with several DS8000 and tape encryption devices

From a connectivity standpoint, the managed devices and the Tivoli Key Lifecycle Manager servers simply require an Ethernet connection. A maximum of four Tivoli Key Lifecycle Manager IP connections can be setup in the DS8000 HMC. In other words, each DS8000 can connect to a maximum of four independent Tivoli Key Lifecycle Manager servers (with synchronized Keystores) for redundant key access.

The other consideration is in the number of separate devices that can be handled from each Tivoli Key Lifecycle Manager, as mentioned previously, depends on the license you have for the Tivoli Lifecycle Key Manager software.

A

Tivoli Key Lifecycle Manager Installation

This Appendix provides an outline for the installation of the Tivoli Key Lifecycle Manager.

Note: For detailed information, refer to the I*BM Tivoli Key Lifecycle Manager Installation and Configuration Guide*, SC23-9977.

In addition, refer to the Tivoli Integrated Portal information center:

http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.tip.doc/welcome_tip_ic.htm

© Copyright IBM Corp. 2010. All rights reserved.

Hardware requirements

Ensure that the computer has the required memory, speed, and available disk space.

Remember that for DS8000 encryption support, you must install a Tivoli Key Lifecycle Manager server (isolated server) that exactly matches the following hardware and software requirements:

- IBM System x3650 with L5420 processor
  - Quad-core Intel Xeon® processor L5420 (2.5 GHz, 12 MB L2, 1.0 GHz FSB, 50 W)
  - 6 GB memory
  - 146 GB SAS RAID 1 storage
  - SUSE Linux 10 with Service Pack 3)
  - Dual Gigabit Ethernet ports (standard)
  - Power supply
- Tivoli Key Lifecycle Manager v1
  - Includes DB2 9.1 FB4

Note: No other hardware or software is allowed on this server. An isolated server must only use internal disk for all files necessary to boot and have the Tivoli Key Lifecycle Manager key server become operational.

Table A-1 lists the general hardware requirements.

Table A-1 Tivoli Key Lifecycle Manager hardware requirements

System components	Minimum values	Suggested values
System memory (RAM)	2 GB	4 GB
Processor speed	► For Linux and Windows systems: 2.66 GHz single processor ► For AIX and Sun Solaris systems: 1.5 GHz (2–way)	► For Linux and Windows systems: 3.0 GHz dual processors ► For AIX and Sun Solaris systems: 1.5 GHz (4–way)
Disk space free for product and prerequisite products such as DB2 Database and keystore files	15 GB	30 GB

Operating system requirement and software prerequisites

Table A-2 identifies the operating systems requirements for installation.

Table A-2 Tivoli Key Lifecycle Manager software requirements

Operating system	Patch, maintenance level at time of initial publication
AIX Version 5.3 64–bit, and Version 6.1	For Version 5.3, use Technology Level 5300-04 and Service Pack 5300-04-02
Sun Server Solaris 10 (SPARC 64–bit) Note: Tivoli Key Lifecycle Manager runs in a 32–bit JVM.	None
Windows Server 2003 R2 (32–bit Intel)	None
Red Hat Enterprise Linux AS Version 4.0 on x86 32–bit	None
SUSE Linux Enterprise Server Version 9 on x86 (32–bit) and Version 10 on x86 (32–bit)	None

On Linux platforms, Tivoli Key Lifecycle Manager requires the following package:

```
compat-libstdc++-33-3.2.3-47.3
```

On Red Hat systems, to determine if you have the package, run the following command:

```
rpm -qa | grep -i "libstdc"
```

Java runtime environment requirements

The Tivoli Key Lifecycle Manager requirement for a *version* of Java runtime environment depends on which operating system is used:

- On distributed systems, use IBM Java runtime environment that is included with embedded WebSphere Application Server.
- On all systems, use of an independently installed development kit for Java, from IBM or other vendors, is not supported.

Database authority and requirements

The Tivoli Key Lifecycle Manager requirement for a *database* depends on which operating system is used.

On distributed systems, use DB2 Enterprise Edition Version 9.1 with Fix Pack 4 on the same computer on which the Tivoli Key Lifecycle Manager server runs.

Before installing the application, see the DB2 documentation on the following websites, if applicable, for the additional kernel settings:

- AIX systems

  No documentation required.

- Linux systems

  http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.uprun.doc/doc/t0008238.htm

- Solaris systems

  http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.uprun.doc/doc/t0006476.htm

- Window systems

  No documentation is required.

Installing Tivoli Key Lifecycle Manager on a Windows platform

On distributed systems, use one of the following modes to install Tivoli Key Lifecycle Manager on Windows platform:

- A GUI-based installation driven by a wizard.
- A console mode installation that runs in a console window. This mode scrolls information onto the screen, then prompts for your entries one line at a time.
- A silent installation that runs unattended, using response files for the configuration options.

GUI mode installation

To install Tivoli Key Lifecycle Manager in GUI mode, run the wizard to enter configuration options:

1. Start the installation wizard by navigating to the directory where you stored the installation files, and run this command:

   ```
   install
   ```

   On Windows, **`install`** is the `install.exe` file.

2. Select a setup language, and then click **OK**, as shown in Figure A-1 on page 147.

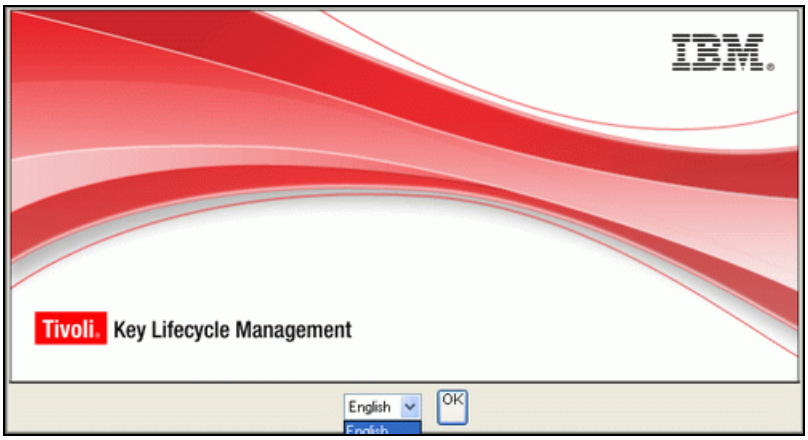


Figure A-1 Tivoli Key Lifecycle Manager installation wizard

3. After the installation wizard opens the introduction window, click **Next** to proceed to the next window.

   The Software License Agreement window opens (Figure A-2).

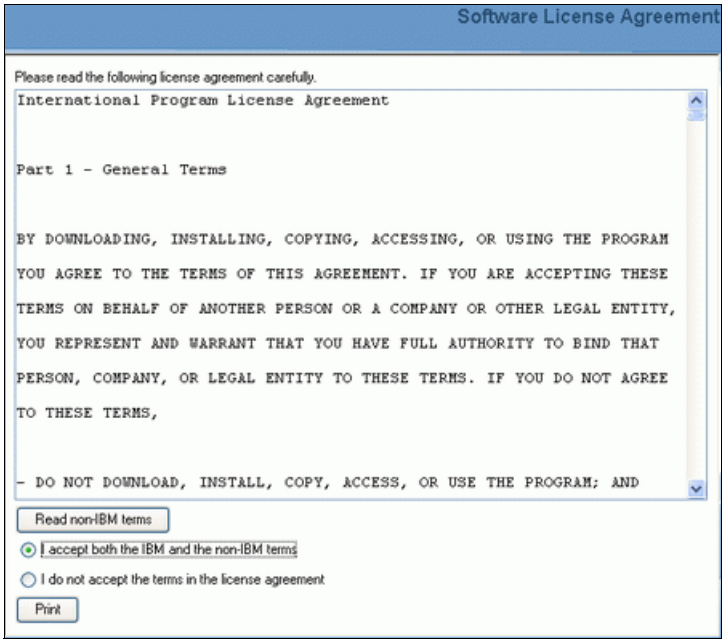


Figure A-2 License Agreement

4. Read the License Agreement, and if acceptable, click **Yes**.

   The installation wizard presents several panels where several panels gather configuration information for DB2.

5. Select a DB2 destination, and then click **Choose**, as shown in Figure A-3 on page 148.

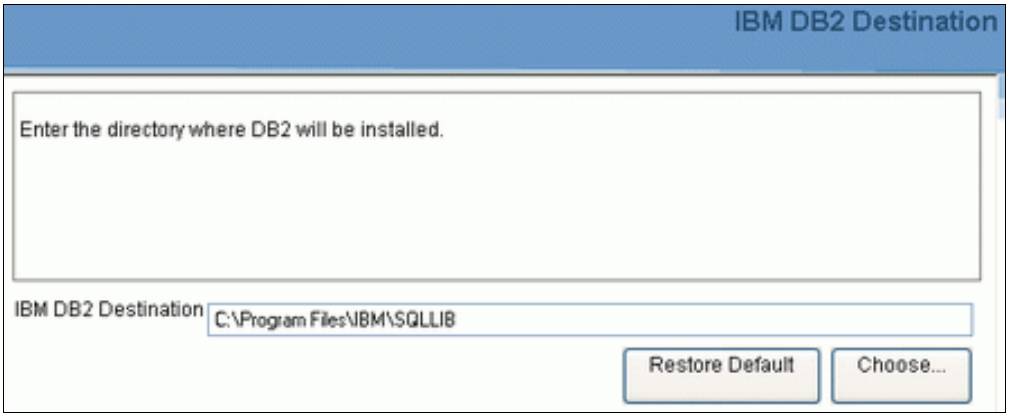


Figure A-3 Specify DB2 destination

6. The default DB2 values are displayed, as shown in Figure A-4. Enter a password where required and click **Next**.

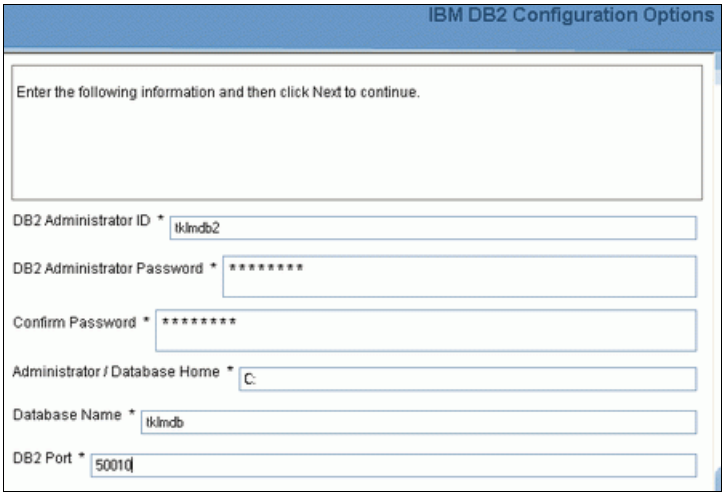


Figure A-4 DB2 defaults

7. The Summary of prerequisites is displayed, as shown in Figure A-5. Click **Next** to accept the options.

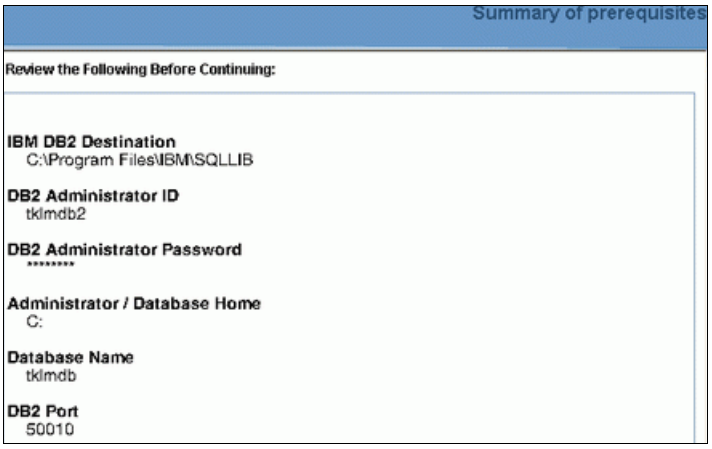


Figure A-5 DB2 installation summary

DB2 is now being installed.

8. As shown in Figure A-6, the DB2 portion of the installation is now completed. Click **Next**, to proceed with the remainder of the Tivoli Key Lifecycle Manager installation.

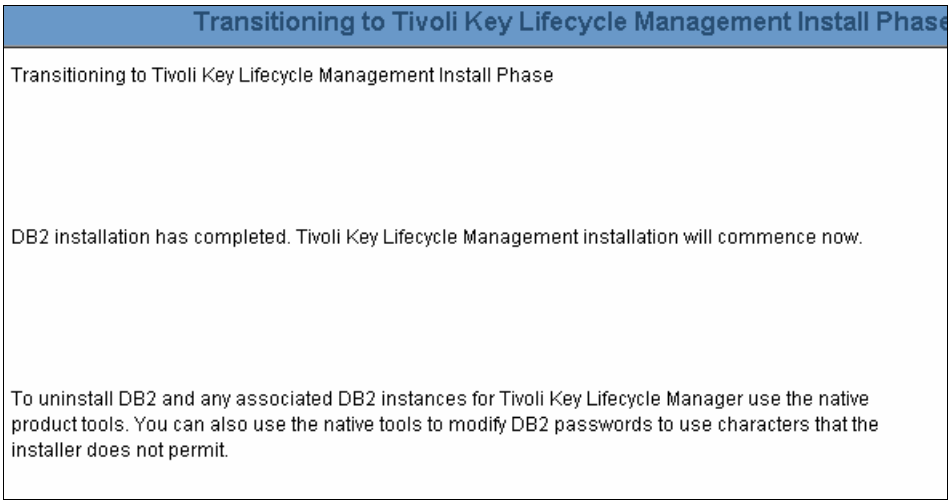


Figure A-6 DB2 installation completed

The installation wizard configures the autonomic deployment engine used in the process. This is shown in Figure A-7. The Tivoli Key Lifecycle Manager and Tivoli Integrated Portal installation begins.

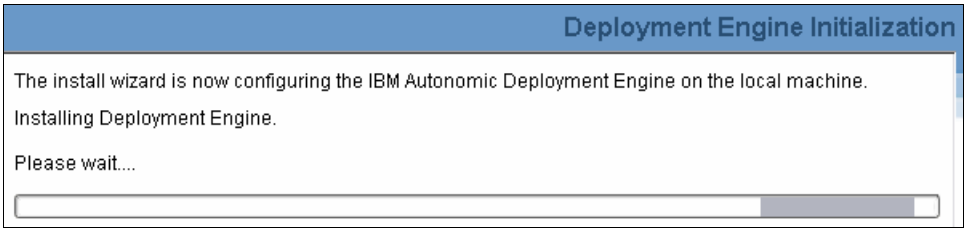


Figure A-7 Deployment engine initialization

9. Select a directory where you want to install the Tivoli Integrated Portal (TIP), as shown in Figure A-8, and click **Next**.

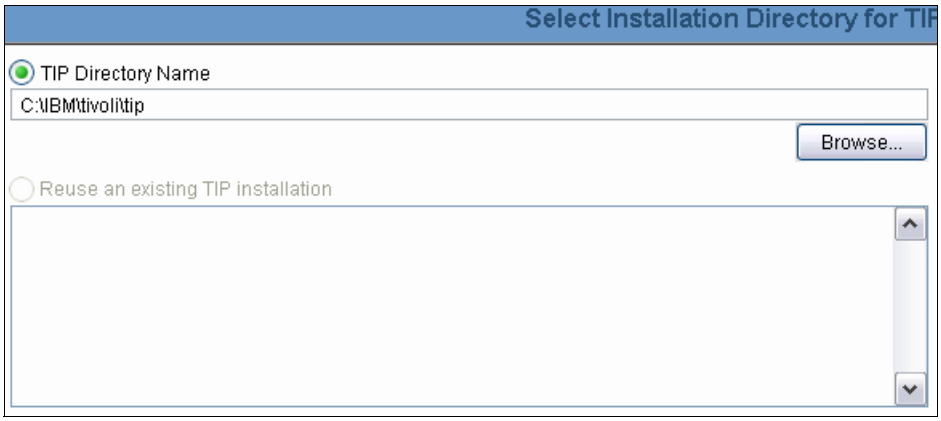


Figure A-8 TIP installation directory

10. Enter a password for TIP as shown in Figure A-9, and click **Next** (not shown in Figure A-9).

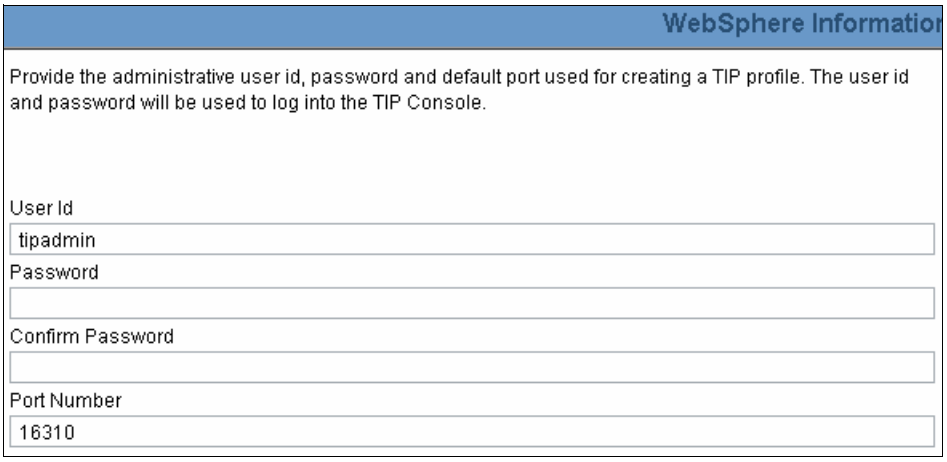


Figure A-9 Enter TIP user ID and password

11. The next window, shown in Figure A-10, allows you to migrate an existing EKM implementation. We do not have an EKM in our environment, so we simply click **Next** (not shown in Figure A-10).

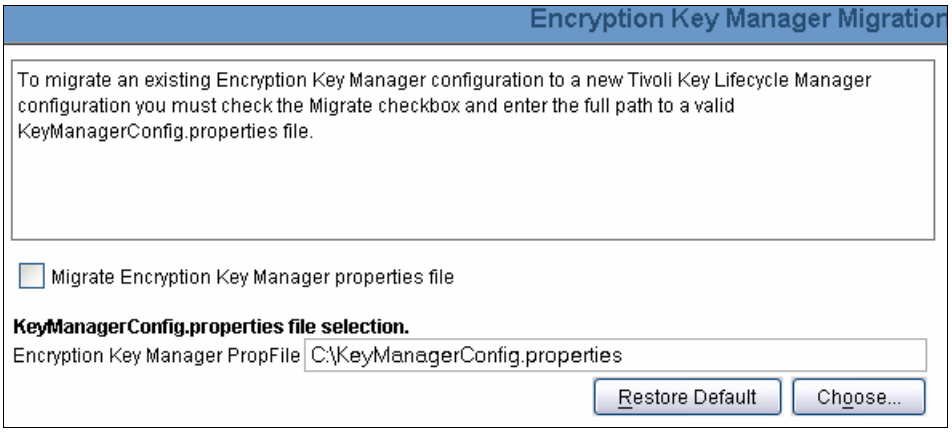


Figure A-10 Migrate EKM

12. A summary of the installation options is displayed as illustrated in Figure A-11. If you have to make changes, go back now and make them now; otherwise, click **Install**.

Disk space information for installation target.
Required disk space: 446 MB
Available disk space: 37612 MB

Please review your selection and click the Install button when you are ready.

- Embedded WebSphere Application Server
- Integrated Solutions Console
- CGI Server
- Tivoli Time Scheduling Services
- Tivoli Common Reporting
- TIP Transformation
- TIP Server
- Tivoli Key Lifecycle Management
- Security Service
- TIP Dependency Management
- Start WebSphere Application Server
- Tivoli Common Reporting Configuration
- Tivoli Key Lifecycle Manager Post Configuration for Tivoli Integrated Portal
- Stop WebSphere Application Server
- Start WebSphere Application Server

Figure A-11 Tivoli Key Lifecycle Manager installation options summary

As shown in the Figure A-12, the Tivoli Key Lifecycle Manager is now installed.

Install Complete

Congratulations IBM Tivoli Key Lifecycle Manager has been successfully installed to:

C:\IBM\tivoli\tip

Click "Done" to quit the installer.

The Tivoli Integrated Portal can be accessed with this URL:

http://yourhostname:16310

Note that during install a new user; TKLMAdmin was created and an administrative role was assi gned to this user. See the product documentation for the usage and password for this new user a nd role.

Figure A-12 Tivoli Key Lifecycle Manager install completed

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

For information about ordering these publications, see “How to get Redbooks” on page 154. Note that some of the documents referenced here may be available in softcopy only.

- *IBM DS8700 Series: Architecture and Implementation,* SG24-8776
- *IBM System Storage Tape Encryption Solutions*, SG24-7320

Other publications

These publications are also relevant as further information sources:

- *IBM Tivoli Key Lifecycle Manager Installation and Configuration Guide,* SC23-9977
- *IBM Tivoli Key Lifecycle Manager Program Directory (for z/OS)*, GI11-4300
- *IBM Tivoli Key Lifecycle Manager Quick Start Guide*, GI11-8738

Online resources

These web sites are also relevant as further information sources:

- Tivoli Key Lifecycle Manager

  http://www.ibm.com/software/tivoli/products/key-lifecycle-mgr/
- IBM System Storage DS8000

  http://www.ibm.com/systems/storage/disk/ds8000/index.html
- IBM Security page

  http://www.ibm.com/security/index.html
- IBMJCEFIPS provider

  http://www.ibm.com/developerworks/java/jdk/security/50/FIPShowto.html
- Tivoli Integrated Portal information center

  http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.tip.doc/ctip_install_overview.html
- Migration from EKM to Tivoli Key Lifecycle Manager

  http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.tklm.doc/install/cpt/cpt_ic_plan_migration.html

© Copyright IBM Corp. 2010. All rights reserved.

- IBM Tivoli product licensing documents are available at:

  http://www.ibm.com/software/tivoli/products/licensing.html

- Modifying kernel parameters (Linux)

  http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.uprun.doc/doc/t0008238.htm

- Modifying kernel parameters (Solaris Operating Environment)

  http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.uprun.doc/doc/t0006476.htm

How to get Redbooks

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

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

A
activation key 74, 89
admin, definition of 65
Advanced Encryption Standard (AES) 4, 9
algorithms
  encryption 6
  RSA 5
alias 111
applykey 90
array encryption 77, 90
asymmetric 4–6, 8, 10
asymmetric encryption 5
audit 43
availability 41

B
backup 43, 96, 99
Backup File column 99
best practices 37, 95
BPXFX111 108

C
CA 8
certificate 8
certificate authority (CA) 8
certificate replacement interval 47
ciphertext 4
Concurrent Code Load (CCL) 123
Copy Services 93
cryptographically erased 14
cryptography 3

D
Data Encryption Standard (DES) 4
data key (DK) 18
DB2 tables 11
deadlock 1, 10, 27, 120
deadlock prevention 42
device session key 17
DFSA 39, 75
digital certificates 8
digital signature 6, 15
Disk Storage Feature Activation (DSFA) 39, 75
DoD 5220.20-M 3
DS command line interface (DS CLI) 85
dual Hardware Management Console (HMC) 44
Dual key server 34
dual platform key support 110

E
EKM 11–12, 140
email 63
encrypted arrays 77, 90
encrypted extent pools 92
encrypted group recovery key (EGRK) 30
encrypted primary recovery key (EPRK) 31
encrypted ranks 79, 91
encryptgrp 92
encryption
  asymmetric 5
  symmetric 4
Encryption Authorization Licensed Internal Code feature key 39
encryption deadlock 1, 10, 27
  permanent 27
  temporary 27
encryption group 17, 40, 73, 89, 92
encryption hardware 14
encryption key
  availability 9
  security 9
Encryption Key Manager
  See EKM
encryption mechanism 13
encryption-capable 14, 39
export or import 47
exporting keys 106
extent pool 79
  encryption 82, 92
externally encrypted data key (EEDK) 18

F
failover 47
FDE 1, 5, 14
feature key 39
FIPS 140-2 certification 12
FlashCopy 73, 93
Full Disk Encryption
  See FDE

G
general key server 43
group key (GK) 25

H
hash 6
HMC 14, 44
  dual 44
  redundant configuration 45
hybrid encryption 5, 9

I
IBMJCEFIPS 12
ICSF 107

© Copyright IBM Corp. 2010. All rights reserved.

Image Certificate 117
import or export 47
importing keys 106
Integrated Cryptographic Service Facility (ICSF) 107
isolated key server 11, 42–43, 62

J
Java Cryptography Extension (JCE) 12
Java security keystore 12
JCE 12
JCECCAKS 107
JCECCARACFKS 107

K
key encrypting key (KEK) 16
Key Label 73
key management 16
key server
  backup 43
  general 43
  isolated 43
keygroups.xml 140
keys
  exporting 106
  importing 106
keystore 12, 30, 41, 140
keystore file 96

L
locked 22–23
lsarraysite 90
lsextpool 93
lskeygrp 88–89
lskeymgr 86
lsrank 92

M
managereckey 87–88
Master Key 53
mkarray 90
mkextpool 92
mkkeygrp 89
mkkeymgr 86
mkrank 91
mkreckey 68, 87–88
monitoring 42

O
OMVS 108

P
Partner certificate 117
PCI (Payment Card Industry) 67
primary recovery key (PRK) 29
private key 4–7
public key 5–6, 8
public/private key 9

R
rank, encryption 79, 91
recovery key (RK) 1, 29, 31, 67, 87
  delete 132
  initiate 121
  re-key 128
  validate 120
recovery signature (RS) 29
Redbooks website 154
  Contact us xi
redundant HMC configuration 45
Remote Mirror and Copy 73
restore 96, 99
role 28
RSA 9
  algorithm 5

S
SAS 64
SAS policy 64
secadmin 66
  definition of 65
secondary recovery key (SRK) 29
secret key encryption 4
Secure Sockets Layer (SSL) 5
security administrator 28, 33, 65
serverStatus 104
SNMP 42, 44, 63
SSRE 104
StartServer.bat 102
StopServer.bat 102
storage administrator 28, 33, 65
Storage Authentication Service
  See SAS
symmetric 4, 8, 10
symmetric encryption 3–4
synchronize keys 34
system assurance call 39

T
Tape Encryption 5
Tivoli Integrated Portal 11, 50–51
Tivoli Key Lifecycle Manager 10
  configuration file 96
  database 96–97
  failover 47
  Master Key 53
Tivoli Key Lifecycle Manager server
  start 101
  stop 101
Tivoli Storage Manager 97
tklmCertExport 111
tklmCertImport 111
tklmCertList 109
tklmDeviceAdd 116
truststore 140

U

unlocked 22

W

WebSphere Application Server 11
wrapped key method 16
wsadmin.sh 108

IBM System Storage DS8700 Disk Encryption

Benefit from a robust and sophisticated key management

Learn about recovery key and dual platform key servers

Encrypt data at rest with no performance degradation

IBM recognizes the requirement for data protection, both from hardware or software failures, and also from physical relocation of hardware, theft, and re-tasking of existing hardware.

The IBM DS8000 Series offers encrypted Fibre Channel drives. These Full Disk Encryption drive sets are used with key management services provided by IBM Tivoli Key Lifecycle Manager software to allow encryption for data at rest on a DS8700 Storage System. Use of encryption technology has a number of considerations that are critical for you to understand to maintain the security and accessibility of encrypted data.

This IBM Redpaper publication contains information that can help customers and storage administrators plan for disk encryption. It also explains how to install and manage the encrypted storage.

More important, this paper documents how to comply with IBM requirements for using the IBM DS8000 encrypted storage.

This edition applies specifically to the IBM System Storage DS8700 with DS8000 Licensed Machine Code (LMC) level 6.5.1.xx (bundle version 75.1.xx.xx).

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

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

For more information:
ibm.com/redbooks

REDP-4500-02