Redbooks
ibm.com/redbooks

Data-at-rest Encryption for the IBM Spectrum Accelerate Family

Bert Dufrasne

Roman Fridli

Andrew Greenfield

Security

Storage

IBM

Redpaper

International Technical Support Organization

Data-at-rest Encryption for the IBM Spectrum Accelerate Family

April 2019

REDP-5402-02

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

Third Edition (April 2019)

This edition applies to IBM FlashSystem A9000 and A9000R Software V12.3

This document was created or updated on April 5, 2019.

Contents

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

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 jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 actual people or business enterprises 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

AIX®
DB2®
IBM®
IBM FlashSystem®
IBM Spectrum™
IBM Spectrum Accelerate™
IBM Spectrum Virtualize™
MicroLatency®
Power Systems™
Redbooks®
Redpaper™
Redbooks (logo)  ®
Storwize®
System Storage®
Tivoli®
XIV®

The following terms are trademarks of other companies:

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.

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

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

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

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

Preface

With the ever-growing landscape of national, state, and local regulations, industry requirements, and increased security threats, ensuring the protection of an organization's information is a key part of operating a successful business.

Encrypting data-at-rest is a key element when addressing these concerns. Most storage products offer encryption at an additional cost. The IBM® Spectrum Accelerate family, which includes IBM XIV® Storage System, IBM FlashSystem® A9000, IBM FlashSystem A9000R system(s), and IBM Spectrum™ Accelerate Software provides data-at-rest encryption at no charge. Clients can take advantage of encryption and still benefit from the lower total cost of ownership (TCO) that the IBM Spectrum Accelerate™ family offers.

For IBM FlashSystem A9000 and A9000R, clients now have a choice between an external key manager-based implementation or a local key based encryption implementation. The local key solution offers a simplified deployment of data-at-rest encryption.

This IBM Redpaper™ publication explains the architecture and design of the XIV and IBM FlashSystem A9000 and A9000R encryption solutions. Details are provided for configuring and implementing both solutions.

Note: For the external key manager-based solution, IBM Security Key Lifecycle Manager Version 2.5 and 2.6 were used in preparation of this paper.

Authors

This paper was produced by a team of IBM specialists from around the world.

Bert Dufrasne is an IBM Certified Consulting IT Specialist and Project Leader for IBM System Storage® disk products at the International Technical Support Organization (ITSO), San Jose Center. He has worked at IBM in various IT areas. He has authored 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. He holds a master's degree in Electrical Engineering.

Roman Fridli is a certified IBM XIV Product Field Engineer based in Switzerland. He joined IBM in 1998 as a Customer Engineer for IBM Power Systems™ and Intel Servers, including point-of-sale devices. Since 2012, he has worked for the XIV PFE EMEA Team based in Mainz, Germany. He holds a degree in Electrical Engineering and multiple certifications in the storage solution and networking areas.

Andrew Greenfield is an IBM Global XIV and Flash Solution Engineer who is based in Phoenix, Arizona. He holds numerous technical certifications from Cisco, Microsoft, and IBM. He is also responsible for many of the photos and videos that are featured in this book and at http://www.IBM.com. Andrew brings over 24 years of data center experience with Fortune 100 companies to the team. He graduated Magna cum Laude, Honors, from the University of Michigan, Ann Arbor. Andrew has also written other IBM FlashSystem and XIV Gen3 IBM Redbooks publications.

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

Markus Oscheka
Bob Liu
Harry McGregor
Vladimir Shalikashvili
IBM

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from 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 might also include minor corrections and editorial changes that are not identified.

Summary of Changes for Data-at-rest Encryption for the IBM Spectrum Accelerate Family as created or updated on April 5, 2019.

April 2019, Third Edition

This revision includes the following new and changed information.

New information

- Updated IBM XIV information
- Updated to include procedures for local key encryption for IBM FlashSystem A9000 and IBM FlashSystem A9000R
- Support for Gemalto Safenet KeySecure external key server

Changed information

- Updated changes in XIV 3rd Party Key Managers
- Updated to reference the longer keys used by modern SSL standards

1

Encryption overview

With the ever-growing landscape of national, state, and local regulations, industry requirements, and increased security threats, ensuring the protection of an organization's information is a key part of operating a successful business. Businesses today need tools to protect against the known threats and to guard against as yet unknown threats. Effective threat and vulnerability management must be proactive rather than reactive, preventing problems rather than responding to them.

Encrypting "data-at-rest" is a key element when addressing these concerns. Most storage products offer encryption at an additional cost. IBM XIV Storage System, IBM Spectrum Accelerate Software, and IBM FlashSystem A9000 and A9000R include data-at-rest encryption at no charge.

For IBM FlashSystem A9000 and A9000R, clients now have a choice between the external key manager-based implementation and a local key-based encryption implementation. The local key solution offers a simplified deployment of data-at-rest encryption, but clients must ensure that it is adequate for their data security requirements.

Note: XIV and IBM FlashSystem A9000 and IBM FlashSystem A9000R systems provide data-at-rest encryption at no charge. However, when using the external key server, such as the IBM Security Key Lifecycle Manager, appropriate licensing is required.

This chapter gives an overview of encryption, and covers the following topics:

- Introduction to data-at-rest encryption
- Threats and security challenges
- Need for encryption
- Encryption concepts
- Encryption challenges

1.1 Introduction to data-at-rest encryption

The XIV and IBM FlashSystem A9000 and IBM FlashSystem A9000R systems offer *hot encryption*. When encryption is enabled on the systems, all data that is contained on that array is encrypted within minutes, with no performance impact. This process is also transparent to any host activities and applications. After existing data has been encrypted, there is no performance penalty on reading or writing new data to disk.

Such support for data-at-rest encryption provides advantages and has certain characteristics:

- Future non-destructive hot encryption is applied to the data already stored on the system without data rewrite.
- It offers flexibility in the business decision-making process with the option to deploy the storage system today and decide to apply encryption later, when the need for encryption arises.

Members of the IBM Spectrum Accelerate family support an encryption process that requires at least one external key server, such as the IBM Security Key Lifecycle Manager (SKLM) server. Starting with software version 12.1, IBM FlashSystem A9000 and IBM FlashSystem A9000R offer a local encryption key solution.

1.1.1 External key management

External key management as provided by SKLM facilitates the generation and lifecycle management of encryption keys. External key management augments security by storing keys separately from the data, and by presenting a secured and well-defined interface for key services.

When implementing the external key server solution, it is critical for availability and disaster protection to have more than one key server installed, preferably in different locations.

The IBM Security Key Lifecycle Manager server does not need to be dedicated to a particular storage system and can be shared across multiple products in the data center.

Important: Encryption must be deployed with careful planning and a full understanding of the interaction among the required products to avoid deadlock situations. The compatibility between different operating system types and versions, and key server types and versions, must be considered as well.

1.1.2 Local key management

Local key management enables system support for key management services without requiring a dedicated, independent key management server. By avoiding a dedicated key server, the cost and complexity of managing keys can be reduced, potentially translating into a lower system total cost of ownership.

Some businesses do not consider the deployment of a dedicated key management system a necessity. They prefer to avoid the expenses and administration typically warranted by the deployment of external key management as long as certain requirements are met.

Tip: If you adopt the external key server implementation, you then switch *non-disruptively* to the local key encryption. However, the reverse operation of changing from local key to external key server first erases any data already on disk.

1.2 Threats and security challenges

Companies face many threats and security challenges:

- Increasing number and sophistication of threats. You must be able to defend against all threats rather than respond only to intrusions.
- Prevention of data breaches and inappropriate data disclosure and 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 employee who inadvertently 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 must prove that your physical assets are secure.

1.3 Need for encryption

Organizations experience a continual 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 the subsequent demand to simplify the complexity of encryption keys management throughout the lifecycle.

In particular, security exposures occur when data storage devices such as disk drives and IBM MicroLatency® Modules leave the company's premises, which usually happens when a data storage device fails and the IBM Service Support Representative (IBM SSR) replaces it with a new part. Sometimes, data storage devices are replaced proactively and the data can still be accessed. IBM has a procedure to delete all data on those parts; however, this task is no longer under the control of the client. Some clients buy back the drives and destroy them themselves, but this procedure can be expensive. A similar concern is when clients return the whole storage system to IBM. IBM erases all data, but this step is not sufficient for some clients. IBM offers a service that is called IBM Certified Secure Data Overwrite Service to erase all data, with several passes, in compliance with United States Department of Defense regulations (DoD 5220.22-M).

When data on the storage system is encrypted, these concerns become resolved. Without the proper decryption key, the data on the storage device or even on the entire storage system is available only as ciphertext, and is therefore unreadable.

The question of what to encrypt and what to leave in clear text often arises. With overall system performance that is not affected by encryption on IBM FlashSystem A9000, IBM FlashSystem A9000R, and XIV systems, it makes sense to encrypt everything. This is easier than choosing which data falls under 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 important.

Important: IBM FlashSystem A9000, IBM FlashSystem A9000R, and XIV systems provide storage device-based encryption for data at rest. If encryption over the network is required, additional encryption services must be investigated and deployed.

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

1.4 Encryption concepts

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

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).

1.4.1 Symmetric key encryption

XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R encryption use symmetric key encryption for data-at-rest solutions.

Symmetric key encryption uses the same key to encrypt plain text to ciphertext and to decrypt the ciphertext to regenerate the plain text. This method is called *symmetric encryption* because same key is used for both encryption and decryption.

Anyone who obtains the key can transform the ciphertext back to plain text. If you want to preserve confidentiality, you must keep your key a secret. 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.

Figure 1-1 shows an encryption and decryption data flow path. The symmetric key is used to encrypt a secret file. The decryption of the text uses the same symmetric key to decrypt the data back to readable text.

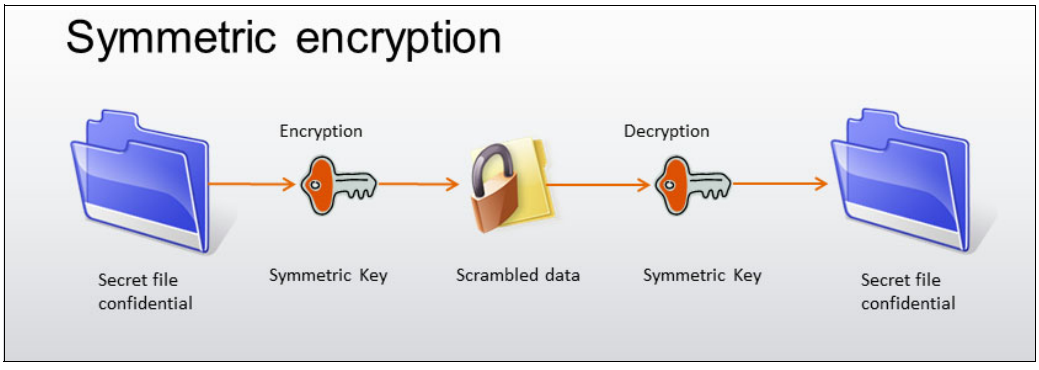

Figure 1-1 Symmetric encryption

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

Each key defines a one to one mapping to the ciphertext. Without the key, as shown in Figure 1-2, deriving the mapping is not feasible.

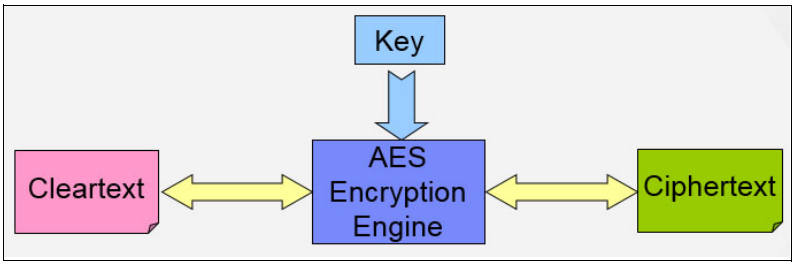

Figure 1-2 Key mapping of ciphertext

1.5 Encryption challenges

Encryption, as described previously, depends on encryption keys. Those keys must be, at the same time, kept secure and available, and responsibilities must be split:

- Key security

  To preserve the security of encryption keys, the implementation must be such that no one individual (person or system) has access to all of the information that is 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 and decrypt the data keys). This *wrapped key* method is used with XIV, IBM FlashSystem A9000, and IBM FlashSystem A9000R.

  See “XIV disk encryption” on page 22 for important differences between external and local key security and their benefits.

- Key availability

  More than one individual (person or system) has access to any single piece of information that is necessary to determine the encryption key.

  In a key server based solution, redundancy is provided by having multiple isolated key servers. In addition, backups of the key server’s data are maintained.

  For the local key solution, the key is stored on each local boot device and each controller of IBM FlashSystem A9000 / A9000R.

- Recovery key and separation of responsibilities

  For XIV, IBM FlashSystem A9000, and IBM FlashSystem A9000R, a manual *recovery key* can also be optionally configured, providing the possibility to recover the master key if the key server or local key becomes unavailable.

  To prevent one person from gaining access to the data, the handling of a recovery key requires at least two people with the role of Security Administrator, which ensures that one person cannot access the data, and it also ensures separation between the Security Administrator and Storage Administrator roles. XIV, IBM FlashSystem A9000, and IBM FlashSystem A9000R also enable operation without a recovery key, but this is not preferable because it puts data at risk if the key servers are no longer accessible.

Key management is the most complex part of an encryption system:

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

  Managing keys with IBM Security Key Lifecycle Manager (abbreviated as SKLM in paths and file names) offers production-ready key management. It offers three advantages:

  - Separated, centralized, and simplified key management
  - Separation of key storage from data storage
  - XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R provide KMIP Version 1.2 support

  One critical consideration with an external key server implementation is that the key server must not depend on any storage system to which it provides encryption keys.

  Furthermore, this external key server must not store any of its code or any information about those keys that it manages for storage systems on those storage systems themselves.

  If this consideration is not accounted for, it becomes possible to experience *encryption deadlock,* where a key server cannot function because it depends on storage that cannot release data because it needs to communicate with that key server. It is analogous to having a bank vault that can be unlocked with a combination, but the only copy of the combination is locked inside the vault.

- For less complex environments, managing the IBM A9000 / R keys with the local key option has the benefit of simple key management through XCLI, without the need for any external host.

2

Planning

Implementing data-at-rest encryption requires careful planning, in particular when opting for the external key server solution. The local key option available for the IBM FlashSystem A9000 and IBM FlashSystem A9000R requires some planning too. To ensure accurate procedures and point out differences, planning is covered in these sections:

- Planning and implementation process flow
- Best practices for an external key server
- IBM Security Key Lifecycle Manager licensing for IBM FlashSystem A9000 and IBM FlashSystem A9000R
- IBM FlashSystem A9000 / A9000R with local encryption

2.1 Planning and implementation process flow

The data-at-rest encryption-capable XIV has only one encryption option as shown in Figure 2-1. It outlines the planning and implementation process. Figure 2-1 shows the additional local key option for IBM FlashSystem A9000 or IBM FlashSystem A9000R.

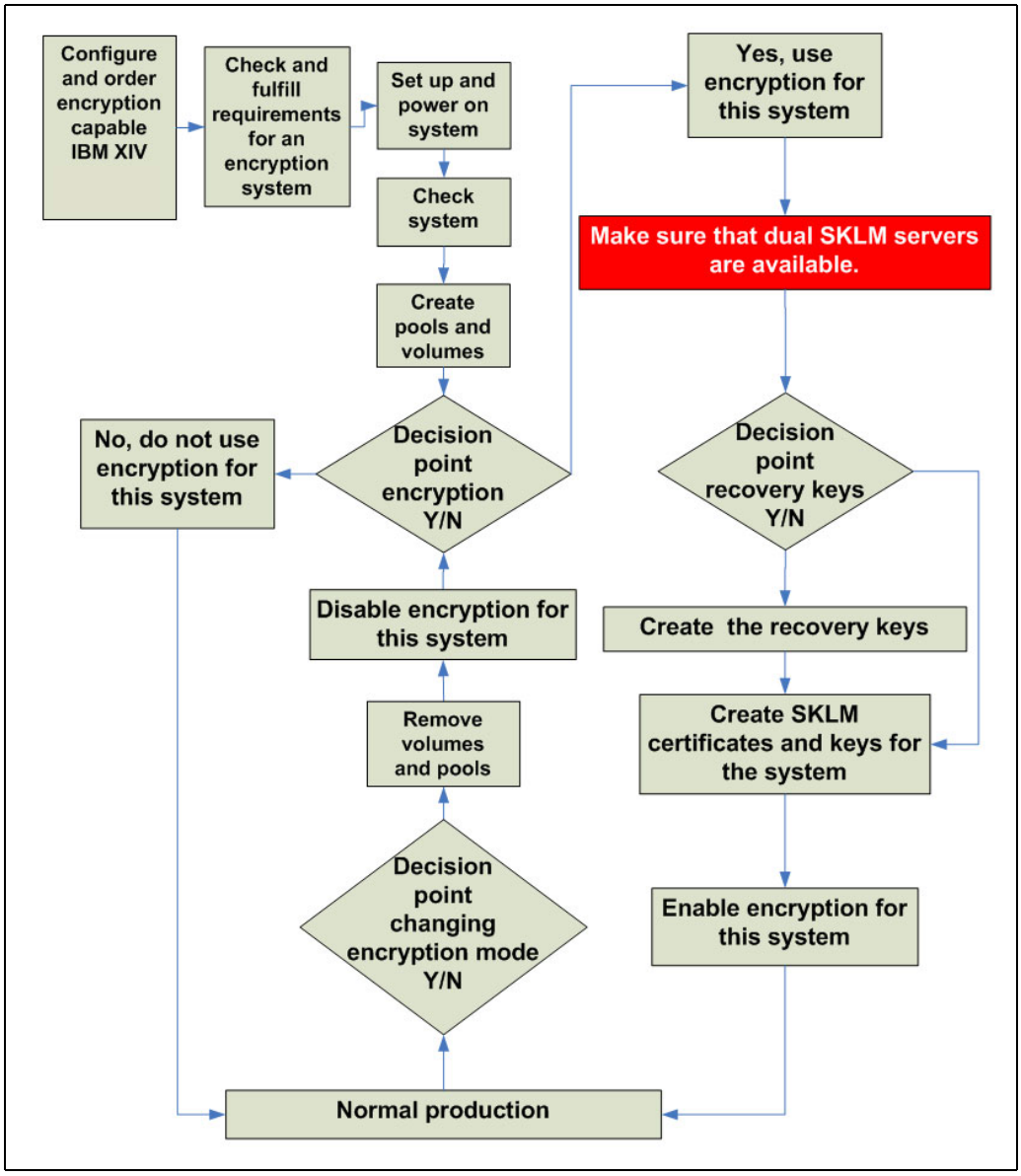

Figure 2-1 XIV Implementation process flow

Important: It is impossible to enable encryption on an XIV unless all of its current disks and modules are in a fully enabled and operational status. This condition is to ensure a full encrypted and secured status upon key generation. If there are any disks or modules that are in a failed state, they must be remedied before the encryption enrollment will proceed.

The IBM FlashSystem A9000 / A9000R can use either an internal key or external key server, but not both at the same time. All of the modules must be in a ready state. The system will not allow encryption enrollment if any of the key components are in a failed state.

Note: IBM FlashSystem A9000 / A9000R systems allow online conversion from an accessible, external key server to the local key option without any disruption to production or data. This process is documented in 4.7, “Converting from external to local key encryption” on page 115.

Converting from a local key encryption to an external key server implies a complete erase of all pools and volumes first.

Figure 2-2 shows the process flow for local key option for IBM FlashSystem A9000 or IBM FlashSystem A9000R. For more information, see 4.6, “Local encryption mechanism and process” on page 109.

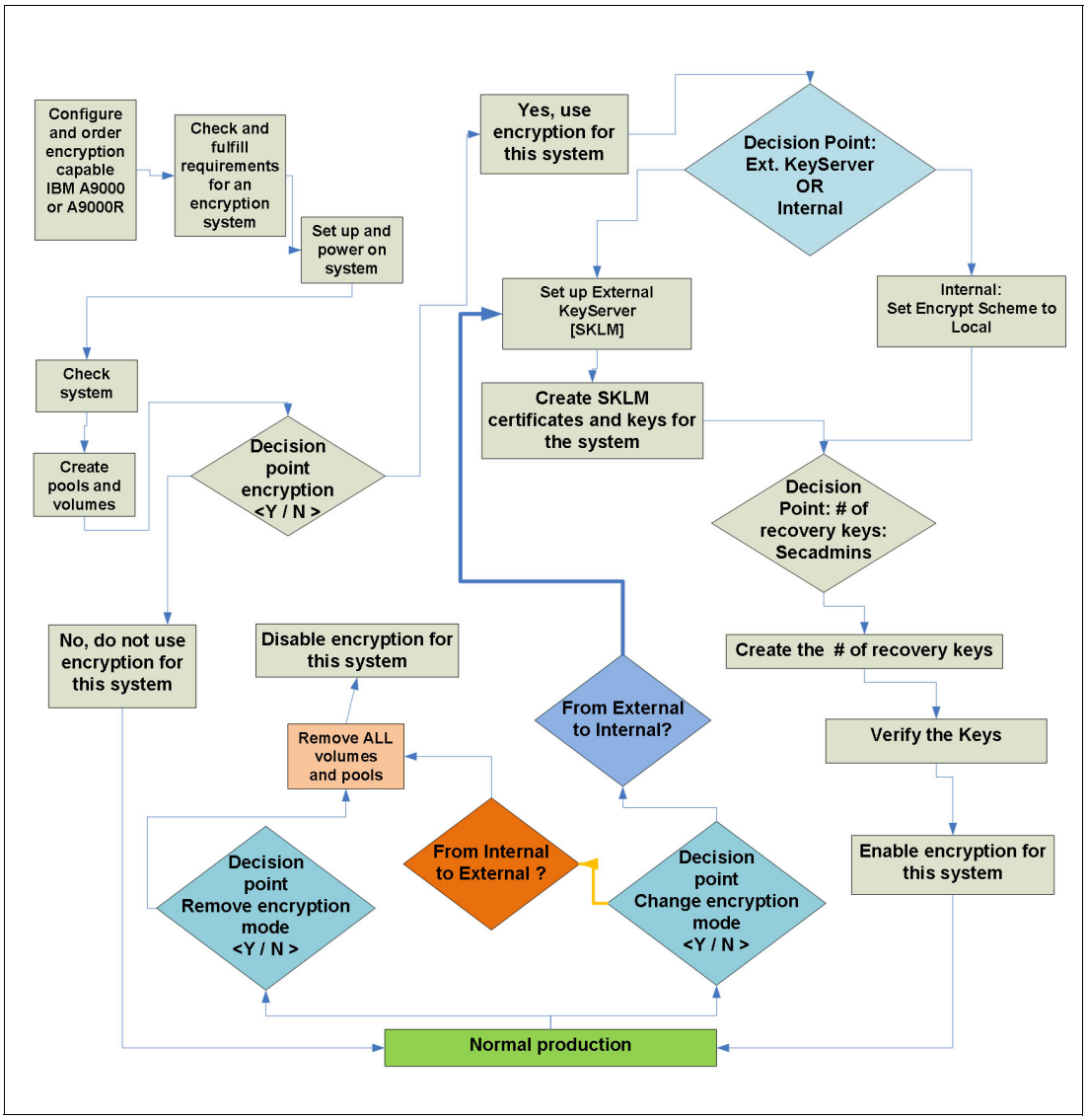

Figure 2-2 A9000 Encryption Implementation process flow

Security Administrator role

User roles for XIV, IBM FlashSystem A9000, and IBM FlashSystem A9000R systems include a *Security Administrator* role. The goal is to split security-related tasks from the Storage Administrator user role. The Security Administrator is the only role with authority to configure and enable encryption. The Security Administrator cannot reach any other menu items, such as **Volume view** or **Create**.

Recovery Keys

Optionally, a manual recovery key can also be configured, providing the possibility to recover the master key if the key server becomes unavailable.

Important: A recovery key can be created only if data-at-rest encryption is not yet enabled. You cannot create a recovery key after encryption is activated. However you do have the option of rekeying, which is discussed in 3.5.6, “Recovery key rekey” on page 51 for XIV and in 4.3.3, “Recovery key rekey” on page 101 for the A9000 or A9000R.

Activating encryption

In an XIV system that is equipped with self-encrypting drives (SEDs) and encryption enabled, all data that is stored on the XIV is encrypted. The same applies to the MicroLatency modules and vaulting devices in IBM FlashSystem A9000 and IBM FlashSystem A9000R.

You can activate encryption concurrently with data that is already stored in the storage systems. This capability is referred to as *hot encryption*.

When hot encryption is started in an XIV system, data in the flash cache is erased. Therefore, after encryption is finished, the flash cache must start “learning” again.

Important: Deactivating encryption cryptographically erases all data on the drives and MicroLatency modules. Therefore, you must back up any data that must be kept, or migrate it to another system, before deactivating encryption on XIV systems.

Copy Services function considerations

If volumes on an encrypted XIV or IBM FlashSystem A9000 and IBM FlashSystem A9000R system are mirrored to a non-encrypted system, the data is not encrypted on the target storage. Therefore, it is not secured. Also, if the target XIV or IBM FlashSystem A9000 and IBM FlashSystem A9000R encryption is activated, the data is not encrypted when it is transferred between the two systems unless you take suitable measures to protect data in transit, such as a WAN Virtual Private Network (VPN) tunnel.

2.1.1 Additional planning when using an external key server

After the IBM Security Key Lifecycle Manager installation, a few tasks are required to implement and activate encryption on the storage system.

Configuring the recovery key is strongly advised after the external key server is successfully added on the IBM XIV. The recovery key is generated from either the GUI or the command line (XCLI).

The use of an external key server is a separately purchased product that can support the KMIP standard required by the IBM FlashSystem A9000 or IBM FlashSystem A9000R system.

Important: The IBM Security Key Lifecycle Manager is an external key server that can be installed as a virtual machine (VM) as well as a physical machine. In that case, make sure that it does not use the encrypted XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R systems as a storage device.

Such configuration can lead to an encryption deadlock situation, where an IBM Security Key Lifecycle Manager server cannot function because it depends on the very storage that cannot release data because it needs to communicate with that server.

An encryption-enabled XIV, IBM FlashSystem A9000 and A9000R using the external key server option, requires at least one key server to be configured. However, the preferred practice is to have at least two key servers configured (a primary server and a backup server). If the IBM Security Key Lifecycle Manager server is not reachable when a previously encrypted XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R is powering on (or restarted), the storage system is not accessible to read or write data for the hosts.

After the XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system starts, they must be able to communicate with at least one key server to obtain the encryption keys. Communication between the XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system and the IBM Security Key Lifecycle Manager server is through a Key Management Interoperability Protocol (KMIP) over Secure Sockets Layer (SSL) protocol. The physical connection between the XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system and the key server is through a Internet Protocol network, as shown in Figure 2-3.

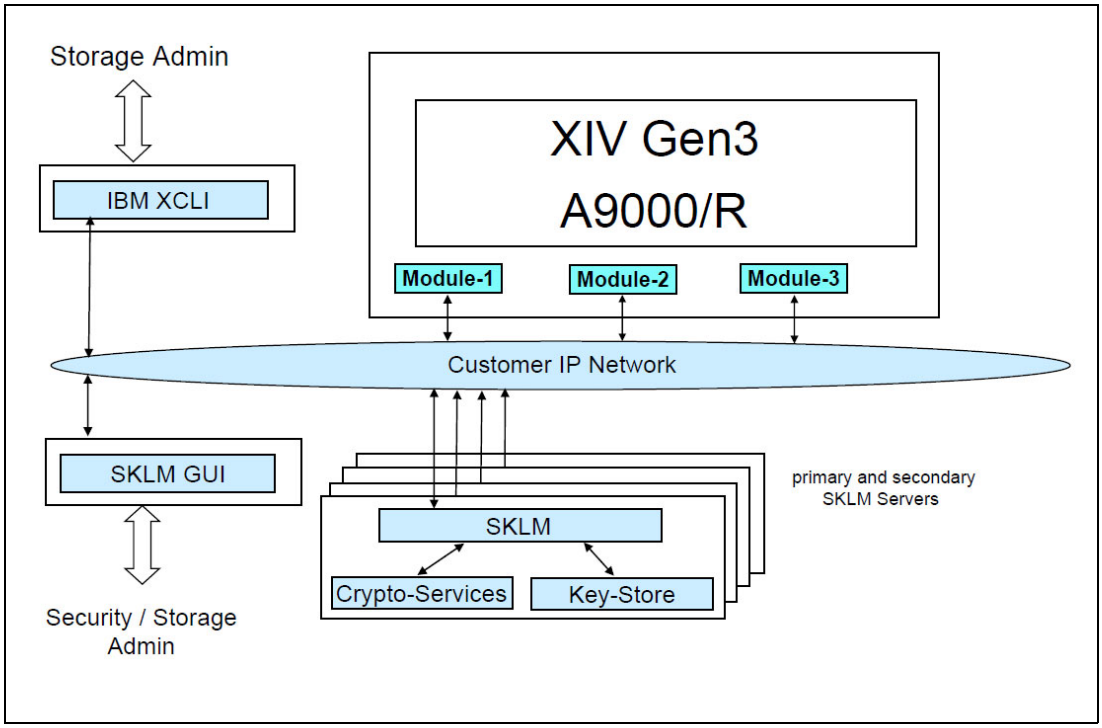

Figure 2-3 XIV and A9000 / R connection with IBM Security Key Lifecycle Manager (SKLM)

A key server can be configured to serve keys to any device that IBM Security Key Lifecycle Manager supports, including other encryption-enabled XIV systems or supported IBM tape drives.

Important: If the IBM Security Key Lifecycle Manager server is not reachable when a previously encrypted XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R is powering on (or restarted), the storage system is not accessible to read or write data for the hosts. This restriction is why it is critically important to have at least two IBM Security Key Lifecycle Manager servers in your IP network.

2.1.2 Digital certificates

Digital certificates are a way to bind information with an identity. Digital certificates are exchanged between the IBM Security Key Lifecycle Manager and XIV systems so that each one can verify the other's identity before sending the sensitive keying information. This process ensures that the sender can trust the receiver. This technique is critical to prevent any "man in the middle" hacking attempts, by not only having a digital fingerprint, but also encrypting all communications to ensure that any listening devices are unable to decipher the contents.

Certificates can be signed by a *certificate authority* (CA). If users trust the CA and can verify the CA's signature, they can also verify that certain information belongs to a person or an entity that is identified in the certificate.

These items are part of the information that is stored in a digital certificate:

- Name of the issuer
- Subject distinguished name (DN)
- Public key that belongs to the owner
- Validity date for the public key
- Serial number of the digital certificate
- Digital signature of the issuer

Digital certificates: Each XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system has a unique digital default device certificate that is installed at the time of manufacture. In addition, users can install their own digital certificates if they choose. MD5 signature algorithm-based certificates should not be used because of their vulnerability.

2.1.3 IBM Security Key Lifecycle Manager licensing for IBM FlashSystem A9000 and IBM FlashSystem A9000R

IBM Security Key Lifecycle Manager uses resource value units (RVUs) in its licensing for the IBM FlashSystem A9000 and IBM FlashSystem A9000R. However, if the storage system is placed behind an IBM Spectrum Virtualize™ system, such as IBM SAN Volume Controller or IBM Storwize®, such as the V9000, the licensing uses standard Effective Capacity. These options are explored in the following sections.

IBM Security Key Lifecycle Manager licensing outside of virtualization

Because the IBM FlashSystem A9000 or IBM FlashSystem A9000R offer compression and data deduplication features, the licensing model is based on RVUs. These units are not linear, so the following tables are useful to order the correct RVU based on the size of the array.

For the A9000 model, the table is shown in Figure 2-4.

FlashSystem A9000	
Modules	**RVU**
12 x 1.2 TB	15
12 x 2.7 TB	31
12 x 5.7 TB	51

Figure 2-4 A9000 RVU values

IBM FlashSystem A9000R has many other configuration options. Figure 2-5 shows the proper values to be ordered for IBM Security Key Lifecycle Manager for each IBM FlashSystem A9000R configuration.

FlashSystem A9000R With 2.9 TB MicroLatency® modules			FlashSystem A9000R: With 5.7 TB MicroLatency® modules		
Grid elements	**Effective (TB)**	**RVU**	**Grid elements**	**Effective (TB)**	**RVU**
2	300	52	2	600	80
3	450	67	3	900	106
4	600	81	4	1200	131
5	750	94	5	1500	152
6	900	108	6	1800	173

Figure 2-5 IBM FlashSystem A9000R RVU values

Additionally, there are other options that an IBM Sales Executive can use to license based on physical capacity when ordering.

2.1.4 IBM Security Key Lifecycle Manager licensing under virtualization

With the explosion of storage virtualization, and the modern ability for a single encrypted volume to span multiple storage systems, a different licensing model is used when IBM Spectrum Virtualize is in front of an XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system.

In this virtualized environment, storage functions are typically handled by the highest layer, in this case by IBM Spectrum Virtualize, as shown in Figure 2-6.

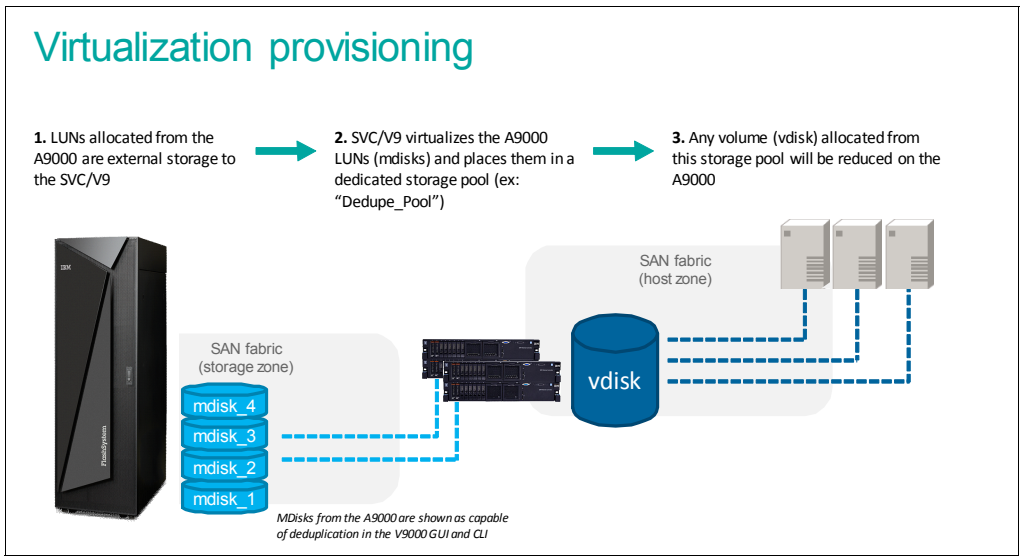

Figure 2-6 IBM FlashSystem A9000R using SVC to provision single encrypted volume to hosts

As shown in Figure 2-6 on page 13, IBM Spectrum Virtualize (or SAN Volume Controller) has no awareness of data reduction or over-provisioning on the back-end storage that it virtualized.

With IBM Spectrum Virtualize, you can use standard external virtualization licensing per TB (SVC 5641-VC7 or VSC 5608-AE1). However, in special conditions, it might make more sense to license based on Physical Capacity instead of Effective Capacity. Have your IBM Sales Executive contact their Storage Virtualization champion.

2.2 Best practices for an external key server

The following information helps ensure that you employ best practices for encrypting storage environments. It includes preferred practices for mitigating the risk of an encryption deadlock.

2.2.1 Security

User roles for XIV, IBM FlashSystem A9000, and IBM FlashSystem A9000R systems include a *Security Administrator* role. The goal is to split security-related tasks from the Storage Administrator user role. The Security Administrator is the only role with authority to configure and enable encryption. The Security Administrator cannot reach any other menu items, such as **Volume view** or **Create**.

2.2.2 Availability

Here are the considerations and preferred practices:

- IBM XIV, IBM FlashSystem A9000 or A9000R system

  The system must be configured with all management modules or grid controllers IP addresses to provide redundant access to the network. An exception is the A9000 version where only two out of three grid controllers contain management IP addresses.

- IBM Security Key Lifecycle Manager key server
  - Configure redundant key servers to each encrypting storage device. Have independent and redundant key servers on each site.
  - To start the IBM Security Key Lifecycle Manager key server operation after power-on without human intervention, the key server must be set up to automatically power on when power is available and to automatically start the key server application. The application must be configured to automatically start.

2.2.3 Encryption administration

Here are the considerations and preferred practices:

- General:
  - The change management processes must cover any procedures that are necessary to ensure adherence to guidelines that are required to ensure the correct configuration of key servers and encrypted storage.

- All personnel who have any of the following assignments or capabilities are required, at least annually, to review a client document that describes these risks and the processes that are adopted to mitigate them:
  - Responsibility for the implementation of IBM Security Key Lifecycle Manager key servers or encrypted storage products.
  - Responsibility to manage the placement or relocation of data that is related to or required by any IBM Security Key Lifecycle Manager key server.
  - Access authority to configure IBM Security Key Lifecycle Manager key servers or encrypted storage products.
  - Responsibility to rekey the recovery key of the XIV systems.
- Implement automated monitoring of the availability of any equipment that is associated with the management of key services and take steps to keep them operational. This equipment can include, but is not limited to, key servers, SNMP managers, LDAP servers, and domain name servers.
- Pay particular attention to disaster recovery plans and scenarios, and consider the availability of key servers, key server backups, and key server synchronization. A preferred practice is to establish the independence of each recovery site from the other recovery site.
- If recovery key management is enabled, the client must have a documented process to handle and maintain the recovery keys of each XIV system, IBM FlashSystem A9000, or IBM FlashSystem A9000R system. This key is the last resort to unlock the storage system if the IBM Security Key Lifecycle Manager environment is destroyed or totally inaccessible. The recovery key is *not* used while IBM Security Key Lifecycle Manager remains available.

► IBM Security Key Lifecycle Manager key server:

- Configuration of redundant key servers (at least two) is required. Redundancy implies independent servers and independent storage devices.
- Configuration of one key server with dedicated hardware and non-encrypted storage resources at each recovery site is required.

> **Two key servers:** XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R systems require at least one key server to be configured, but a preferred practice is to use two for redundancy.

The following tasks must be accomplished:

- Implementing a key server environment that is independent from non-key server applications so that management of the key server can be restricted to those personnel that are specifically authorized to manage key servers.
- Implementing a key server that is physically and logically isolated from other applications that might 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 of storing (initially or through data migration) code and data objects that are 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 secondary key server that is not dependent on the availability of the primary key server.

- Configuration of additional key servers on generalized server hardware and generalized storage is allowed. 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 preferred practice and reduces the probability that all key servers experience a simultaneous power loss.
- Clients must ensure that all key servers that a particular storage device is configured to and communicate with have a consistent keystore content relative to any wrapping keys that are 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.
- Back up key server data after it is updated. Do not store the backups on encrypted storage media that depend on a key server. For more information, see 5.1, “Automated replication” on page 120.
- Periodically audit to ensure that all online and backup data that is required to make each key server operational is stored on media that is not dependent on the key server to access the data.
- Under normal circumstances, clients must not delete keys on the key server. Deletion of all copies of a key is a cryptographic erase operation. It affects all data that is encrypted under this key.

► IBM XIV Storage System and IBM FlashSystem A9000 / A9000R:

- The XIV and IBM FlashSystem A9000 / A9000R monitor all configured IBM Security Key Lifecycle Manager key servers. Customer notification is provided through the XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R client notification mechanism (SNMP traps, email, Short Message Service (SMS)), or combinations of them, when configured, when a key validation issue with 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. Such a condition reflects a degraded key server environment.

  The following conditions are monitored and reported:

  - If the storage system cannot receive a required data key during power-on from the key servers, it reports the error condition to the client and to IBM. In this case, the associated XIV system that has encryption activated is inaccessible to attached hosts because the storage devices cannot be unlocked during a power-on. If the storage system can obtain the required data key from a key server, after reporting the error, it reports the condition to the client and to IBM.
  - IBM FlashSystem A9000 or IBM FlashSystem A9000R system reboot is required to make the data accessible.
  - The ability of each key server to serve data keys that are configured on the XIV, IBM FlashSystem A9000, or IBM FlashSystem A9000R system is verified at hourly intervals. Loss of the ability to unwrap a configured data key is reported to the client and to IBM.

2.2.4 Multiple key servers for redundancy

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

On Microsoft Windows systems and other systems, such as Linux or IBM AIX®, both computers must have the required memory, speed, and available disk space to meet the workload.

This is not a failover or clustered server from an IBM Security Key Lifecycle Manager point of view. The redundancy is managed by setting up multiple key manager destinations at the XIV system, IBM FlashSystem A9000, or IBM FlashSystem A9000R.

Synchronization is achieved through operating system independent UI-based replication. Starting with Version 2.6, IBM Security Key Lifecycle Manager provides a GUI for automated replication configuration. You can configure the replication program to replicate IBM Security Key Lifecycle Manager critical data across clone servers when new keys are added to the master server.

The automated replication process enables cloning of IBM Security Key Lifecycle Manager environments to multiple servers in a manner that is independent of operating systems and directory structure of the server. For example, you can replicate data from a master server on a Windows system to a clone server on a Linux system. You can clone a master IBM Security Key Lifecycle Manager server with up to 20 copies.

For more information, see IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/en/SSWPVP_2.6.0/com.ibm.sklm.doc/admin/cpt/cpt_ic_admin_replication_clone_master.html

On older versions of the IBM Security Key Lifecycle Manager server, back up one server and restore the backup configuration on the other server, assuming that both servers have the same operating system. If you have servers with different operating systems, you must use the export/import function. Plan to perform this backup/restore or export/import operations when the following events take place:

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

2.2.5 Setting up IBM Security Key Lifecycle Manager servers

For efficient planning, complete the pre-installation worksheets that are available in IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/SSWPVP_2.6.0/com.ibm.sklm.doc/install_guide/cpt/cpt_insguide_worksheets.html

For more information, see IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/en/SSWPVP_2.6.0/com.ibm.sklm.doc/welcome.htm

IBM FlashSystem A9000 and IBM FlashSystem A9000R systems require a reboot to make the data accessible.

The ability of each key server to serve data keys that are configured on the IBM FlashSystem A9000, or IBM FlashSystem A9000R system is verified at hourly intervals. Loss of the ability to unwrap a configured data key is reported to the client and to IBM.

2.3 IBM FlashSystem A9000 / A9000R with local encryption

This section is specific for the IBM FlashSystem A9000 / A9000R using the local key encryption method. It offers an easy and simple method for enabling data at rest encryption. However, remember that the local encryption, by definition, can only be used for local volumes and not any other (external) A9000 arrays and their corresponding volumes.

The following sections highlight specific consideration for the local key encryption.

2.3.1 Security

Because the keystore will be internal to the IBM FlashSystem A9000 / R, it is critical that all of the keys are protected. Be sure to safeguard both the master and the recovery keys.

Although it is possible to enable encryption without recovery keys, do not do so because this technique provides no way to recover data access if the master key is lost.

Because the key is stored internally, if the system is to be moved to a new location, to prevent theft of data, consider erasure of the system. This process can be done with various encryption commands or is also available as a service from IBM.

Additionally, best practices say that anytime a grid server or boot SSD is replaced, issue a rekey to ensure complete security.

2.3.2 Availability

Because the key server is local and internal to the IBM A9000 / R, there are the special considerations and preferred practices:

- The system must be configured with all management modules or grid controllers IP addresses to provide redundant access to the network. An exception is the A9000 version where only two out of three grid controllers contain management IP addresses.
- All of the modules must be in a ready state. The system will not allow encryption enrollment if any of the key components are in a failed state.

2.3.3 Encryption administration

Because the encryption users are contained inside the IBM A9000 / R storage array and not in an external key server, best practices for securing user credentials should be observed:

- General:
  - All personnel who have any of the following assignments or capabilities are required, at least annually, to review a client document that describes these risks and the processes that are adopted to mitigate them:
    - Responsibility for the implementation or maintenance of the local key server of encrypted storage products.
    - Access authority to configure the IBM A9000 / R local key server or encrypted storage products.
    - Responsibility to rekey the recovery key of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system.
  - Implement automated monitoring of the availability of any equipment that is associated with the management of key services and take action to keep them operational. This equipment can include, but is not limited to, the SNMP managers, LDAP servers, and domain name servers.
  - Pay particular attention to disaster recovery plans and scenarios, and consider the availability of key server backups. A preferred practice is to establish the independence of each recovery site from the other recovery site.
  - Have a documented process to handle and maintain the recovery keys of each IBM FlashSystem A9000 or IBM FlashSystem A9000R system. This key is the last resort to unlock the IBM FlashSystem A9000, or IBM FlashSystem A9000R system if something happens to the master key.
  - Back up key server data after it is updated. Do not store the backups on encrypted storage media that depend on a key server.
  - Under normal circumstances, clients must not delete keys on the key server. Deletion of all copies of a key is a cryptographic erase operation. It affects all data that is encrypted.
- IBM FlashSystem A9000 family:
  - The IBM FlashSystem A9000 / A9000R monitors all configured IBM Security Key Lifecycle Manager key servers. Customer notification is provided through the IBM FlashSystem A9000 or IBM FlashSystem A9000R client notification mechanism (SNMP traps, email, SMS), or combinations of them, when configured, when a key validation issue with 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. Such a condition reflects a degraded key server environment.

    The following conditions are monitored and reported:
    - If the storage system cannot receive a required data key during power-on from the local key server, it reports the error condition to the client and to IBM. If the storage system can obtain the required data key from its local key server, after reporting the error, it reports the condition to the client and to IBM.

- The ability of the local key server to serve keys that are configured on the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is verified at hourly intervals. Loss of the ability to unwrap a configured data key is reported to the client and to IBM.

2.3.4 Safeguard keys for redundancy

Unlike the prior options, if using the local key server option, the only safeguard is to ensure the master key and the recovery keys are well documented and protected.

3

Implementing encryption on XIV

This chapter describes how to configure and implement data-at-rest encryption for the XIV system.

It covers these topics:

- Encryption process overview
- IBM Security Key Lifecycle Manager installation
- XIV data-at-rest encryption configuration
- Recovery key use and maintenance
- Activating or deactivating encryption
- Verifying encryption state

Note: Several illustrations in this chapter are based on Version 2.6 of the IBM Security Key Lifecycle Manager GUI. Older versions, including IBM Tivoli® Key Lifecycle Manager, were supported when this paper was written.

Information presented in this chapter also applies to IBM Spectrum Accelerate deployed on hardware equipped with appropriate Self Encrypting Drives.

For information specific to IBM FlashSystem A9000 and IBM FlashSystem A9000R, see Chapter 4, “Implementing encryption on IBM FlashSystem A9000 and A9000R” on page 57.

3.1 XIV disk encryption

The IBM XIV Storage System Gen3 data-at-rest encryption relies on the system's self-encrypting drives (SEDs) and external key manager software. It helps secure data with industry-standard encryption for data at rest, without affecting performance. When encryption is enabled, the optional SSDs that are used as flash cache are also encrypted by using software-based AES 256-bit keys encryption. The XIV system secures data at rest and offers a simple, cost-effective solution (cryptographic erasure) for securely erasing any disk drive that is being retired or repurposed.

Encryption support is offered starting with XIV System Software Version 11.4 on XIV Gen 3 Model 214 systems, with 1 TB (stripped-down 2 TB), 2 TB, 3 TB, 4 TB, and 6 TB drive capacities, if those drives are SED. All 4 TB and 6 TB drives available in the XIV system are SED, but on older models, not all 1 TB, 2 TB, and 3 TB drives are SED. Any system that was ordered after 8 October 2013, regardless of capacity, has SED. The flash cache encryption is software-based. On an XIV system, upon hot encryption, the flash cache is emptied, and the XIV system must relearn the workload.

Encryption is also supported through a system software upgrade to Version 11.4 or later on all XIV 214 systems that include SED. No additional hardware changes are required to apply data-at-rest encryption functions on those systems. All SED drives can operate transparently in non-encrypting systems.

3.1.1 Self-encrypting drives

The XIV system supports data encryption with the SEDs. All disks in the XIV system must be the SED type and of the same capacity. No intermixture is allowed. These disks have encryption hardware, and they can encrypt and decrypt data at full disk speed without affecting the performance.

Encryption-capable XIV systems with SEDs can be used without activated encryption. By nature, SEDs encrypt data by using a default access key. However, because the drive is not enrolled, the individual key is not protected, and the data remains readable. In this context, *enrolling* means configuring the drive to lock its encryption key with an externally provided key, as described in "Enrolling" on page 23. After a drive becomes enrolled, the access key is locked, and data on the drive is no longer readable without the external key.

Safe Drive Retirement

With SEDs in the XIV system, Safe Drive Retirement (Figure 3-1 on page 23) is another secure feature. When systems are retired, moved, or sold, the keys can be discarded. No data can be read when this feature is used. The IBM XIV is cryptographically erased, as described in 3.2, "Encryption process overview" on page 24.

Cryptographically erased: If all copies of the encryption key are lost (whether intentionally or accidentally), it is no longer possible to decrypt the associated ciphertext. The data that is contained in the ciphertext is said to be *cryptographically erased*. The data is lost because it is not feasible to be decrypted without the key.

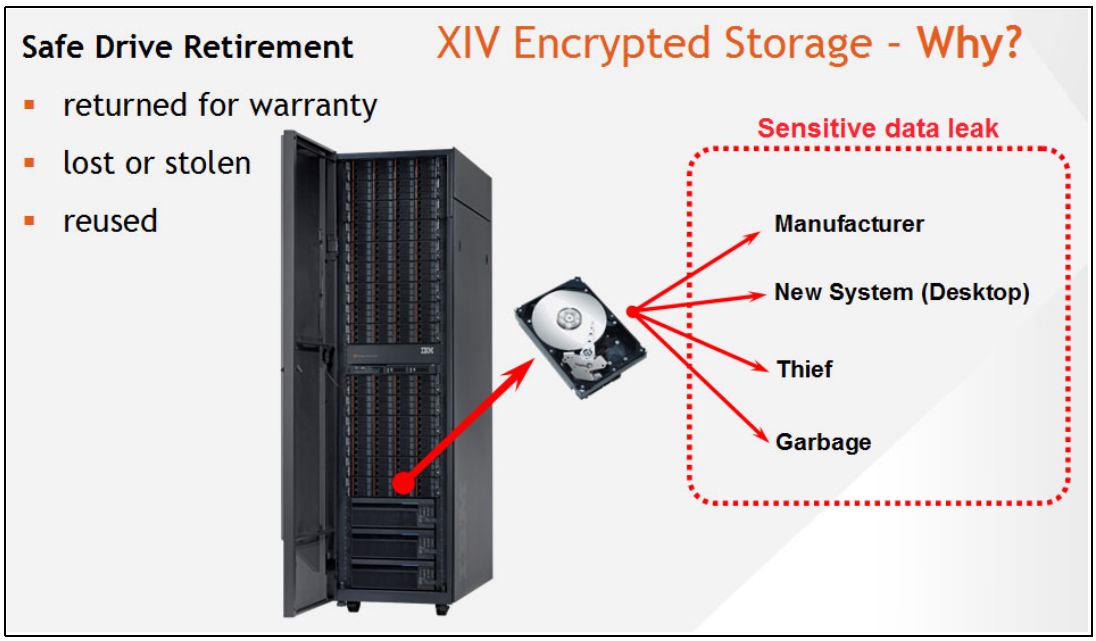

Figure 3-1 Safe Drive Retirement

Banding

A *band* is a contiguous region on the disk. *Banding* is the process of defining one or more bands on the drive. Only SED drives can be banded.

In XIV systems, each drive is configured with two bands: One for internal use by the XIV system and one for the user data, which is always encrypted with a drive-unique Data Encryption Key (DEK). That DEK is never accessible from outside the drive.

When a band is defined for user data, a new encryption key is generated and associated with this band. This process effectively and permanently “erases” all data that was previously stored in the band.

Enrolling

Enrolling is a process of instructing the key server to encrypt the key (DEK) that is associated with a specific band. This goal is accomplished by an externally provided key that is called the Data Access Key (DAK).

The enrolling is performed when the encryption is activated either through the XIV GUI or XIV command-line interface (XCLI).

Before enrollment, the DEK is encrypted with the Manufactured Secure ID (MSID), which is a hardcoded known value in the drive firmware. The MSID is set by the disk manufacturer. It is unchangeable and readable.

Unenrolling is instructing the disk to wrap the key with the MSID again, which makes the data readable. When a disk’s band is enrolled, the band becomes unreadable or locked during power-on or reset.

Unlocking a disk requires the same DAK that was used to enroll it. After the DAK is provided, the drive decrypts the DEK and uses it to access the data. The XIV software is responsible for providing each drive with its DAK.

Figure 3-2 shows the enrolling process, showing the authentication and encryption key relationship.

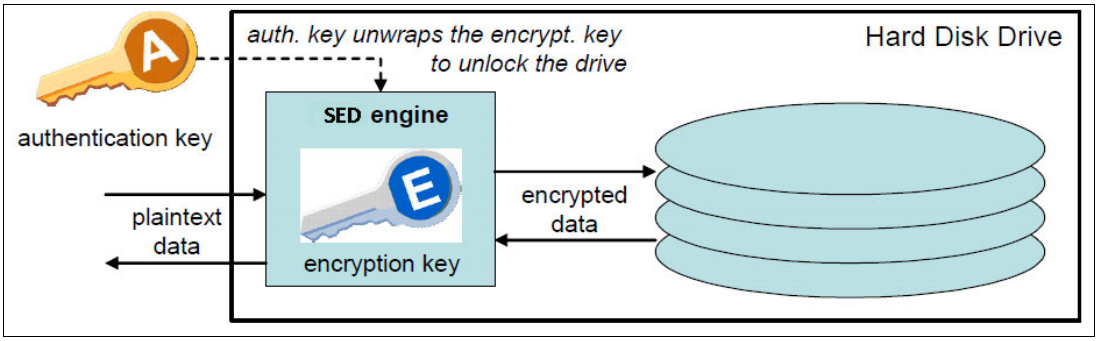

Figure 3-2 Enrolling a SED hard disk drive

3.2 Encryption process overview

The XIV data-at-rest encryption initial configuration starts with installing and configuring the external key server. In our testing, we used IBM Security Key Lifecycle Manager Version 2.6. After the key server is installed and configured, the XIV system and the key server must be able to connect to one another. They establish a trusted connection by exchanging their certificates, as explained in 3.4.1, “Overview of configuration steps” on page 26. Then, the XIV system generates a random XIV master key (XMK), which is used to create the DAKs:

- For each SED in all modules, a DAK is generated that is based on the XMK and the serial number of the XIV module.
- For the second-level read cache (SSDs), the device access key is generated based on the XMK and the module’s serial number.

Next, the XIV system requests and receives the externally stored key (ESK) from the key server. The ESK is used to wrap (encrypt) the XMK that is stored in the XIV system. Figure 3-3 shows the process.

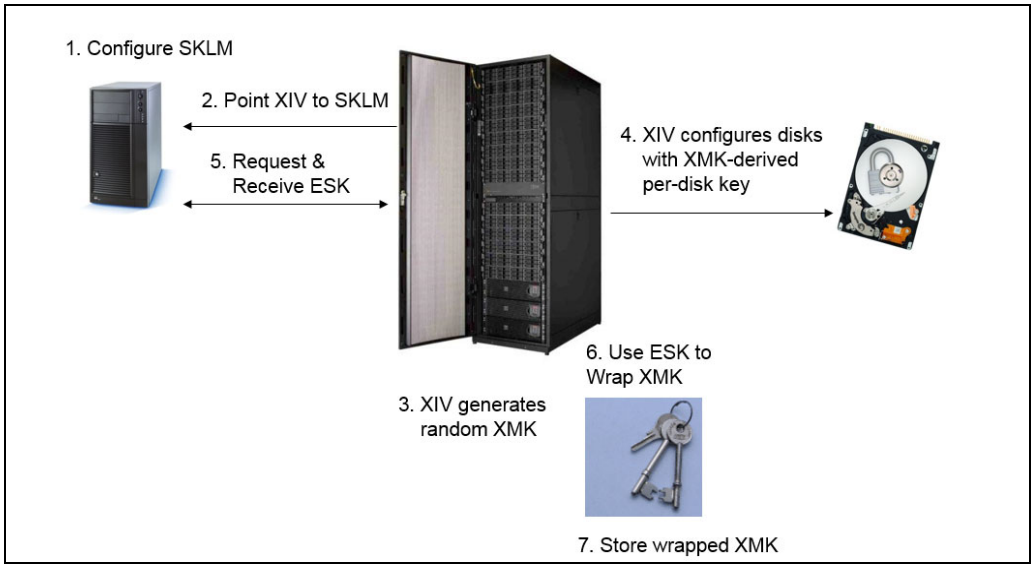

Figure 3-3 Initial configuration

Encryption enablement on the XIV system encrypts the SEDs and the cache SSDs. This non-destructive encryption process is applied to the data that is already stored on the system without data rewrite.

After the XIV data-at-rest encryption is activated, upon booting (after power maintenance, for example), the main encryption startup sequence is shown in Figure 3-4. The SEDs are locked during a reboot. Therefore, you need a valid connection to the key server.

When the XIV system is booting, it establishes a Secure Sockets Layer (SSL) tunnel that is based on the Key Management Interoperability Protocol (KMIP) if the certificates on the XIV system and on the key server match. The XIV system then requests the ESK, which the key server provides. It is used to unwrap the XMK to derive the DAKs that unlock the SEDs and the encryption-activated flash cache.

Note: If no valid key server is available, the XIV system boots into maintenance mode with no host I/O possible, and all disks are locked. However, a simple XIV system reboot does not lock the disks because they are not power-cycled.

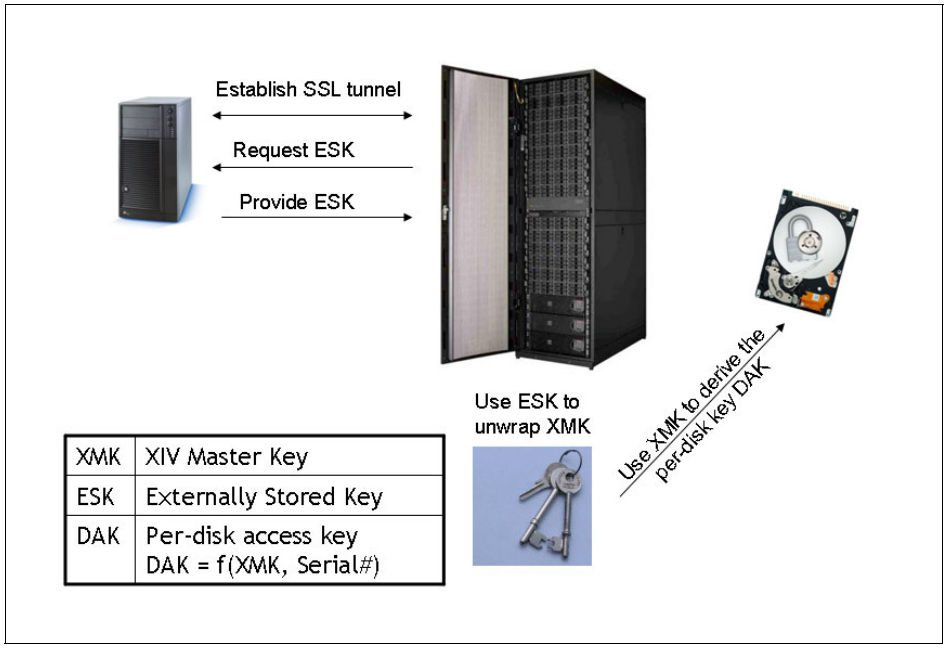

Figure 3-4 Encryption startup sequence

3.3 IBM Security Key Lifecycle Manager installation

Starting with the IBM Security Key Lifecycle Manager Version 2.6, the supported operating systems are SUSE Linux, Red Hat Linux, AIX, and Windows.

For specific supported operating system requirements, see IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/SSWPVP_2.6.0/com.ibm.sklm.doc/install_guide/cpt/cpt_ic_release_oview_sw.html?view=embed

Use the REST interfaces as an alternative to the command-line interface, which might become deprecated in future versions of IBM Security Key Lifecycle Manager.

All references to the alias property of cryptographic keys and certificates in the GUI, CLI, and REST interface will be deprecated in the later versions of IBM Security Key Lifecycle Manager.

You can find installation instructions online in IBM Knowledge Center and chose your desired IBM Security Key Lifecycle Manager version:

https://www.ibm.com/support/knowledgecenter/en/SSWPVP_2.6.0/com.ibm.sklm.doc/install_guide/top/landing-install.html/

A collection of IBM Security Key Lifecycle Manager V2.6 PDF documents can be found online at the following website:

http://www.ibm.com/support/docview.wss?uid=swg27046975

3.4 XIV data-at-rest encryption configuration

This section describes the steps that are required to prepare IBM Security Key Lifecycle Manager to serve an encryption-enabled XIV system. It is based on the assumption that an IBM Security Key Lifecycle Manager server is installed and ready to be configured.

3.4.1 Overview of configuration steps

These steps are required to configure IBM Security Key Lifecycle Manager for the XIV system:

1. Create the IBM Security Key Lifecycle Manager master keystore.
2. Verify the date and time of the XIV system and the key server.
3. Create secadmin role users.
4. Manage certificates:
   a. Copy the XIV device-specific certificate from the XIV system and add the IBM Security Key Lifecycle Manager (import certificate).
   b. Create an IBM Security Key Lifecycle Manager self-signed certificate on the IBM Security Key Lifecycle Manager GUI (add the KMIP-based SSL certificate).
   c. Export the IBM Security Key Lifecycle Manager certificate.
5. Define the IBM Security Key Lifecycle Manager key server on the XIV system (import the `cert.pem` IBM Security Key Lifecycle Manager Certificate).
6. Create the XIV device in the XIV device group (XIV group type).

3.4.2 Detailed configuration steps

This section describes the steps to configure and implement an XIV system with IBM Security Key Lifecycle Manager.

The IBM Security Key Lifecycle Manager solution provides simple-to-use installation options and a management console.

Step 1: IBM Security Key Lifecycle Manager master keystore

When successfully installed, Version 2.6 of the IBM Security Key Lifecycle Manager generates the AES 256-bit master key for data encryption automatically. To conform to the HIPAA and PCI-DSS standards and for increased data security, a 256-bit length master key is used for encrypting IBM Security Key Lifecycle Manager sensitive data, such as key material. The keystore holds all keys and certificates that are managed by IBM Security Key Lifecycle Manager.

If this is a IBM Security Key Lifecycle Manager installation earlier than Version 2.6 or the previous Tivoli Key Lifecycle Manager, you must create the master keystore manually.

Step 2: Verifying the date and time of the XIV system and the key server

When you implement encryption by using keys and certificates, you must have a matching date and time to avoid validity issues, especially when the key server and the system to be encrypted are in different time zones. A mismatch might lead to certificates becoming valid at a future time in the other time zone.

Complete the following steps:

1. Log in to the XCLI by using an administrator role-based user and run the command **time_list**, as shown in Figure 3-5.

```
XIV_ITSO>>time_list
Time       Date         Time Zone   Daylight Saving Time
16:07:38   2016-10-17   UTC         no
```

Figure 3-5 The time_list command

2. You can match times between the key server and the XIV system by adjusting the time zone or setting the time manually. If you choose to match the time zone, make sure that you use a valid time zone. Valid time zones can be shown by running the **timezone_list** command, as shown in Figure 3-6.

```
XIV_ITSO>>timezone_list
Timezone
Universal
Egypt
Africa/Nairobi
Africa/Mbabane
Africa/Niamey
Africa/Dakar
Africa/Khartoum
Africa/Casablanca
...
Europe/Ulyanovsk
NZ-CHAT
EET
GB
GMT
```

Figure 3-6 Excerpt of the timezone_list command

3. Run the **timezone_set** command to adjust the time zone of the XIV system with one of the valid time zones that are listed, as shown in Figure 3-7.

```
XIV_ITSO>>timezone_set timezone=UTC
Command executed successfully.
```

Figure 3-7 The timezone_set command

4. If the date and time do not match between the key server and the XIV, you can set it on the XIV system by running the **time_set** command, as shown in Figure 3-8.

```
XIV_ITSO>>time_set time=2016-10-18.12:02:00
Command executed successfully.
```

Figure 3-8 The time_set command

5. Check that your XIV system shows the encryption support in the output of the **state_list** command, as shown in Figure 3-9.

```
XIV_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Supported
data_reduction_state     Online
```

Figure 3-9 Show encryption support by running the state_list command

If the encryption category does not state a supported value, contact IBM Support.

Step 3: Creating the secadmin role users

Managing and configuring encryption with an XIV system requires a minimum of two secadmin role users. To create them, you can either use the XIV GUI, the IBM HSM UI, or the XCLI.

Complete the following steps:

1. When using the XCLI, add the security admin role users by running the **user_define** command, as shown in Figure 3-10.

```
user_define user=secadmin1 category=securityadmin password=passw0rd
password_verify=passw0rd

user_define user=secadmin2 category=securityadmin password=passw0rd
password_verify=passw0rd
```

Figure 3-10 Define secadmin users in XCLI

2. Run the **user_list** command to verify that the creation of the secadmin users was successful, as shown in Figure 3-11.

```
XIV_ITSO>>user_list
Name                Category     Group     Active
xiv_development     xiv_development        yes
xiv_maintenance     xiv_maintenance        yes
admin               storageadmin           yes
technician          technician             yes
xiv_hostprofiler    xiv_hostprofiler       yes
manager_server_user storageadmin           yes
secadmin1           securityadmin          yes
secadmin2           securityadmin          yes
```

Figure 3-11 The user_list command example

You can change certain user's parameters by running the **user_update** command, as shown in Figure 3-12.

```
XIV_ITSO>>user_update
user=              password=          password_verify=   email_address=
number=            area_code=         exclusive=
```

Figure 3-12 The user_update command example

Tip: After a Security Administrator role base user owns a recovery key, the user name cannot be changed anymore, as shown in Figure 3-13.

```
XIV_ITSO>>user_rename user=secadmin1 new_name=secadmin1b
Error:   USER_OWNS_RECOVERY_KEY
Details: User owns recovery key and therefore cannot be deleted or renamed
```

Figure 3-13 User owns a recovery key

Step 4: Managing certificates

To manage and configure the IBM Security Key Lifecycle Manager certificates, log in to the XCLI. You must log on to your XIV system as a Security Administrator to complete the following steps:

1. First, copy the XIV device-specific certificate from the XIV system by exporting the certificate, and add it to the IBM Security Key Lifecycle Manager. To get the certificate name that is required in step 2 on page 30, run the **pki_list** command, which shows the XIV default device-specific certificate that was installed during manufacturing, as shown in Figure 3-14.

```
XIV_ITSO>>pki_list
Name Fingerprint                       Has signed certificate Services
XIV  2c4cd4f951e48d664ebd978357e5a16b yes
                  XCLI,CIM,IPSEC,KMIP
```

Figure 3-14 The pki_list command

2. Using the name that you get in the output from the command that you run in Figure 3-14 on page 29 (where this example says <*default certificate*>), run **pki_show_certificate name=<*default_certificate*>**, as shown in Figure 3-15.

```
XIV_ITSO>>pki_show_certificate name=XIV
Certificate:
...
-----BEGIN CERTIFICATE-----
MIIDVjCCAj6gAwIBAgIEAJiaaDANBgkqhkiG9w0BAQsFADAiMQswCQYDVQQGEwJV
UzETMBEGA1UEChMKaWJtWElWRGlzazAeFw0xNjEwMTcyMzIzMDVaFw0OMTEwMTEy
MzIzMDVaMDkxCzAJBgNVBAYTAlVTMRMwEQYDVQQKEwppYm1YSVZEaXNrMRUwEwYD
VQQDEww5ODM1LTYwMDMzMTAwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQCJeFEdWePwcucUlwvAdRZleKLXhDlHg4fKDQ+BspI4NOX7Rpq3PrOoi2JpRqRW
G4z1rFecRfOqlaE4UlimKOzC4Cx5YpXsEONG2L98aMLP/qGpDQfRhMABOAXTEKum
p8FryHe2MAOrfXZ5EnnDq9YPmvueBbMHBuxboW3/XKTLFkA3PGNssA+nS2WucfSI
VcgM7kmzpXWpbqDxUOVigcx+BAOHZNha/9Lur8MR7DSmba9mFdNkVpemHg7Scru5
1Z19Xi+7upY9XZaHleSC13m2pUmv2ZvNdaDoAPF8W5TwgWP9WO9rAXOjdAFcZE3B
8bNuI3qfQWQ2wO8kuwdzgNZnAgMBAAGjfTB7MEoGA1UdIwRDMEGAFOZrcCLEx/xs
trBUvc+gKcAUMIZroSakJDAiMQswCQYDVQQGEwJVUzETMBEGA1UEChMKaWJtWElW
RGlza4IBADAOBgNVHQ8BAf8EBAMCBaAwHQYDVROOBBYEFL7sJFutxiki1eR3wSyd
SWnn/FMxMAOGCSqGSIb3DQEBCwUAA4IBAQAHClyYYzEkcmo23UWYNQEa7Z1+7WpF
steIgYx9eWHAxiHceo+ItOVT4bovmjH+JgAjWb44Hj7FVkJGl4hokDZPmD+bMy+m
y1ahZCcGzOGExaBjj33CTlUixnK2ZyEOEY8SYrDn1fifGMTAWBCWtTJzeVOOcOp5
aBcZn89DohhoURXMQvxVvkLQ+Q2LWJ7fV/zrNxgFQHIEbUdpu2PJwRFWqzbbc3Bm
Jou1rZ2FlmYSwrFbremhA54VtjogP35UC/Vb9wAPTYgNxF822vF1mWE4jVYr2xys
UzWUZM8/HdMz1sWycpcedhR6xELTZ2HMrql3XJRHSPk9W/R9XY4UHbEb
-----END CERTIFICATE-----
```

Figure 3-15 The pki_show_certificate excerpt

3. Copy the portion that includes `----BEGIN CERTIFICATE----` through `----END CERTIFICATE----`, paste it into a blank text editor, and save the file as <*filename*>`.pem`.

   As you can see, this manufacturing default certificate is readable. After the data-at-rest encryption is activated, the public key is wrapped, and it is encrypted.

4. Select **Keep pending client device communication certificates**, which can be set in the Key Serving Parameters tab of the Configuration tab, as shown in Figure 3-16.

   When the setting is disabled, client device communication certificates must be manually imported in the wsadmin CLI and do not show up in the Welcome window automatically.

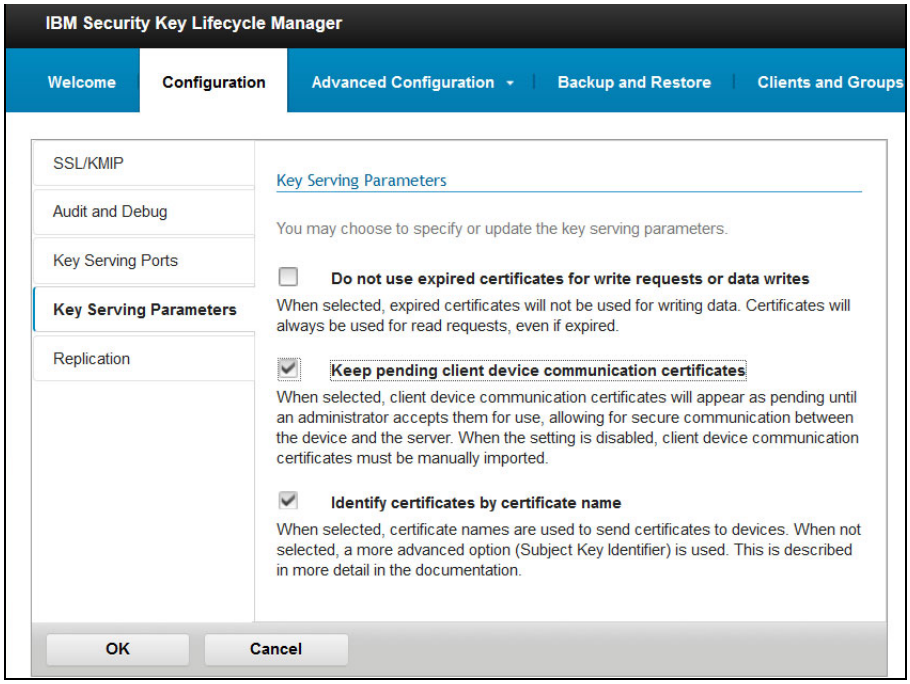

Figure 3-16 Keep pending client device communication certificates

5. In the Welcome window of the IBM Security Key Lifecycle Manager GUI, select the XIV Device Group and then click **Go to** → **Manage keys and devices**, as shown in Figure 3-17.

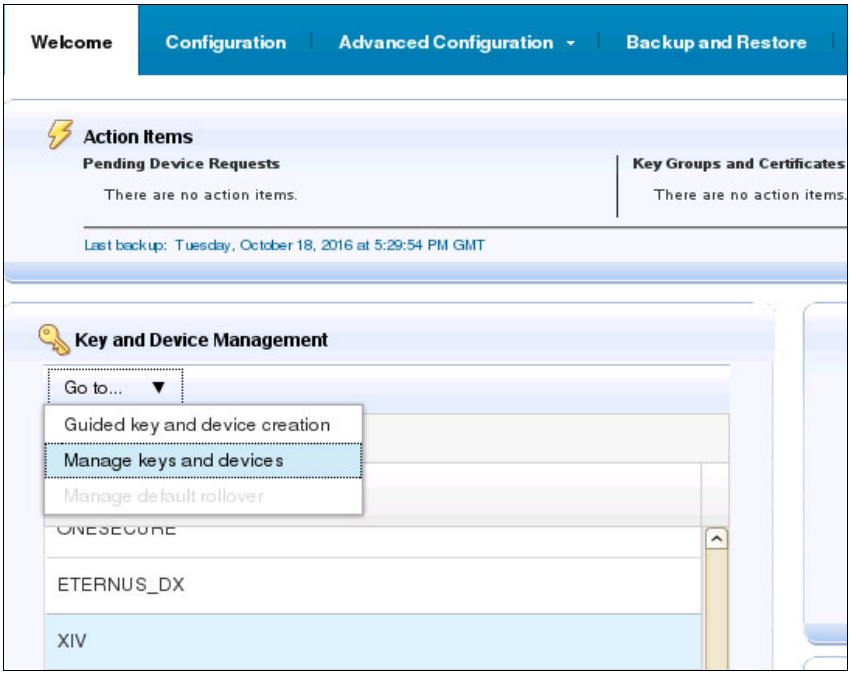

Figure 3-17 Manage keys and devices

Select **Machine affinity** and select **Hold new device requests pending my approval**, as shown in Figure 3-18. This action ensures that when you add the XIV device later that it shows as **Pending** in the Welcome window.

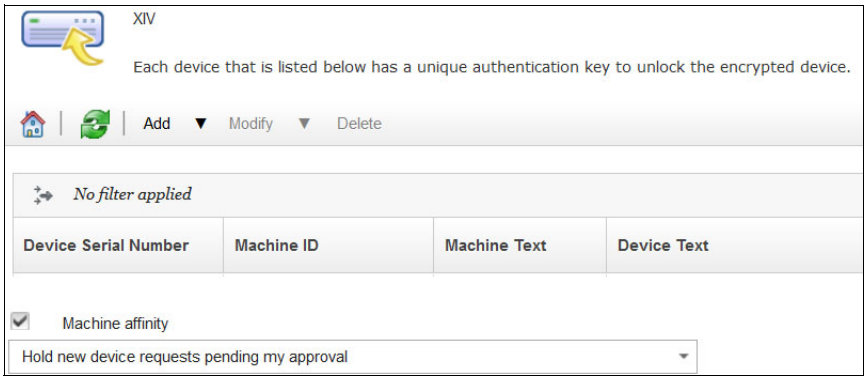

Figure 3-18 Machine affinity and Hold new device requests pending approval

6. Import the newly created SSL certificate as “trusted” in the IBM Security Key Lifecycle Manager web GUI. Click **Advanced Configuration** → **Client Certificates**, and click **Import** (under the SSL/KMIP Certificate for Clients section), as shown in Figure 3-19.

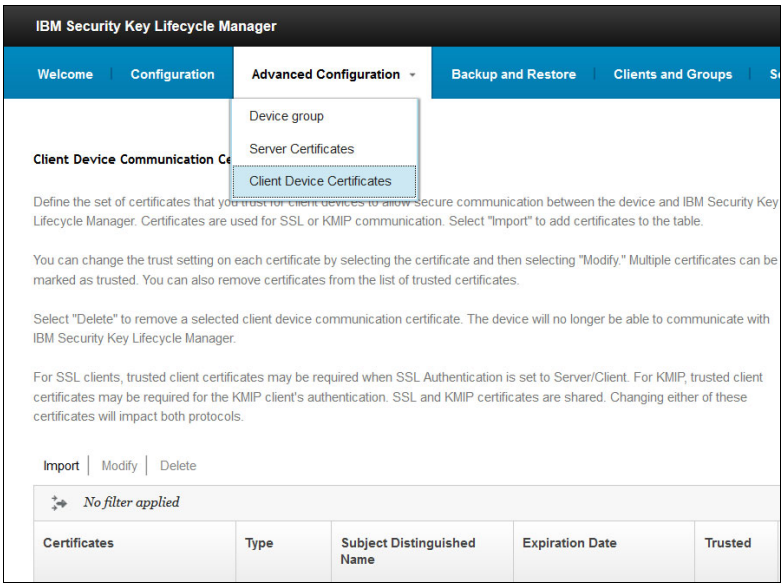

Figure 3-19 Client Device Communication Certificates display

7. Click **Import** (under the SSL/KMIP Certificate for Clients section), as shown in Figure 3-20.

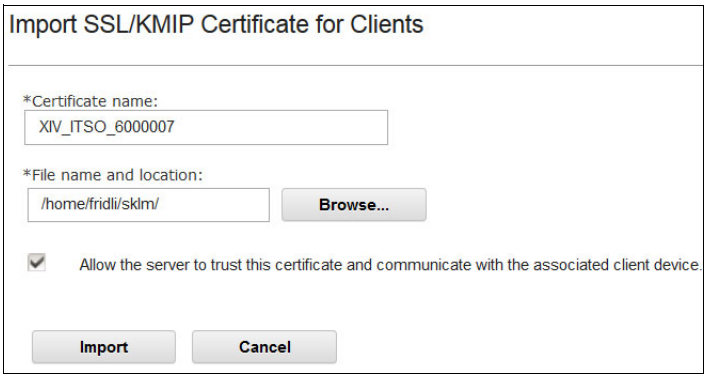

Figure 3-20 Import SSL/KMIP Certificate for Clients

The certificate shows as trusted with a type of **SSL/KMIP** and its name and expiration date are shown, as shown in Figure 3-21.

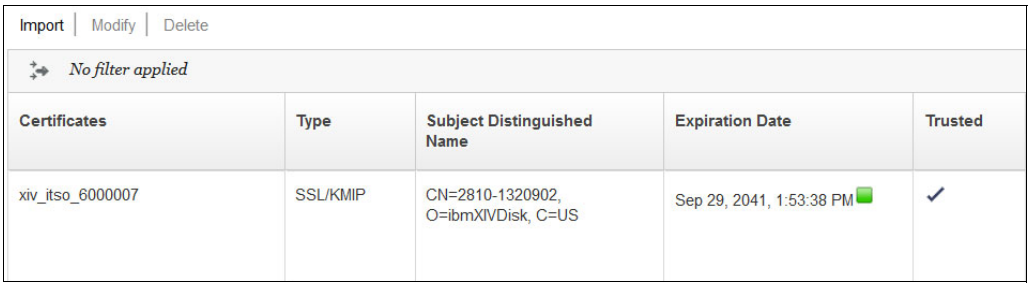

Figure 3-21 Show the trusted device-specific certificate

8. To define the key server in the XIV system, you must create and export the key server certificate on the IBM Security Key Lifecycle Manager and add it to the XIV systems afterward.

   Create an IBM Security Key Lifecycle Manager self-signed certificate in the IBM Security Key Lifecycle Manager GUI (add the SSL/KMIP certificate for key serving).

   If this is a new key server installation, you must have an Action Item in the IBM Security Key Lifecycle Manager Welcome window to create the server certificate, as shown in Figure 3-22.

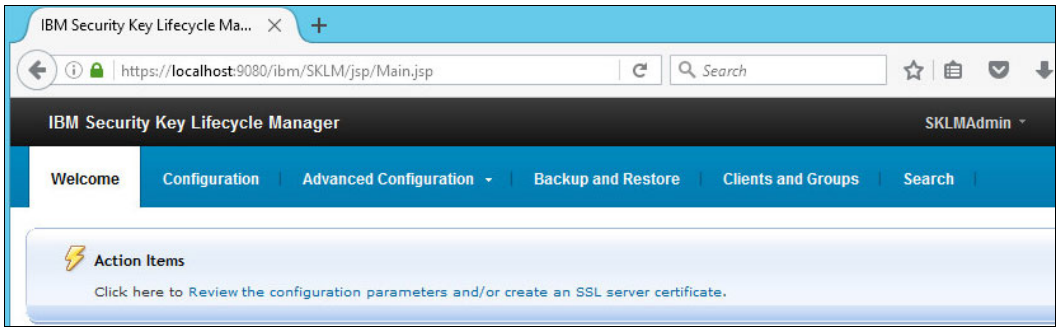

Figure 3-22 IBM Security Key Lifecycle Manager Welcome window with initial configuration Action Item

If the Action Item in the Welcome window is not available anymore, the server certificate can be created by completing the following steps:

a. In the window that is shown in “Step 2: Verifying the date and time of the XIV system and the key server” on page 27, under Advanced Configuration, click **Create self-signed certification**.

b. When the window that is shown in Figure 3-23 opens, create the certificate that is used to encrypt data for secure communication over SSL.

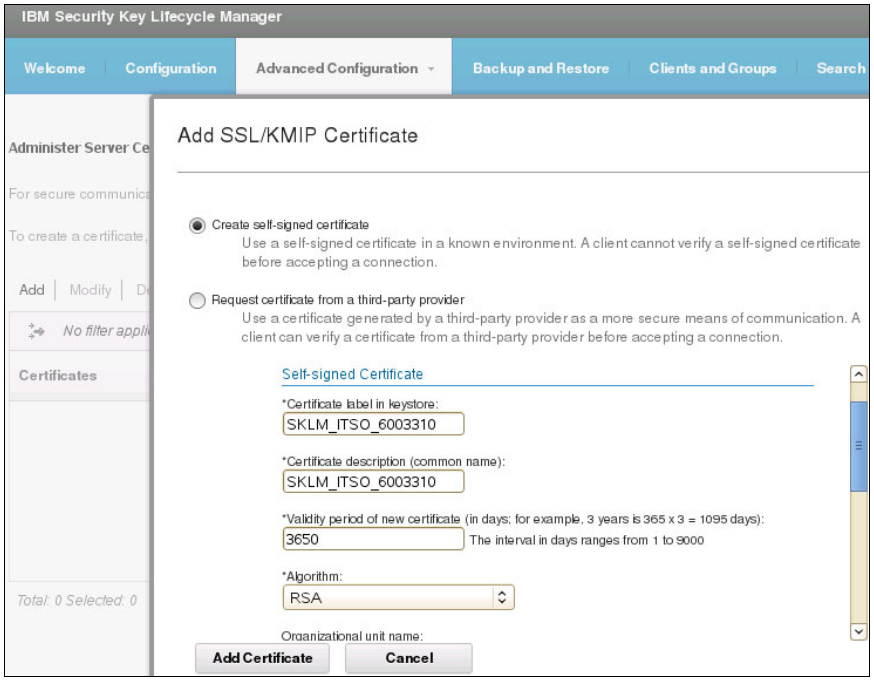

Figure 3-23 Add SSL/KMIP Server Certificate

> **Tip:** Do not confuse this certificate with the *device* certificate that is associated with the XIV system.

c. Select **Create self-signed certificate**. Third-party signed certificates are also supported.

> **Caution:** Although using an existing certificate from the keystore is possible, using the same certificate to encrypt disk data and protect the communication protocol is *not* recommended.

d. Choose a descriptive label and a certificate expiration validity in days in accordance with your security guidelines. You can also enter certificate parameters.

e. Click **Add Certificate**.

As indicated in the Warning notice that is shown in Figure 3-24, the SSL/KMIP certificate is updated. For this change to take effect, you must restart the server. Stopping and starting the key server is described in 5.2, “Starting and stopping an external IBM Security Key Lifecycle Manager server” on page 121. Also, create a backup to ensure that you can restore this data.

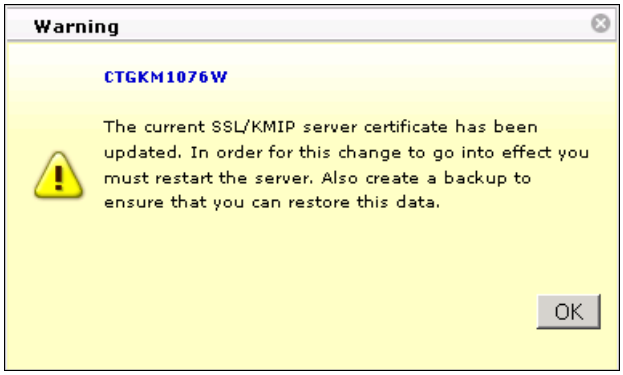

Figure 3-24 Reminder to restart the server

9. In the left pane of the IBM Security Key Lifecycle Manager GUI, click **Welcome** to return to the Welcome window, as shown in Figure 3-25.

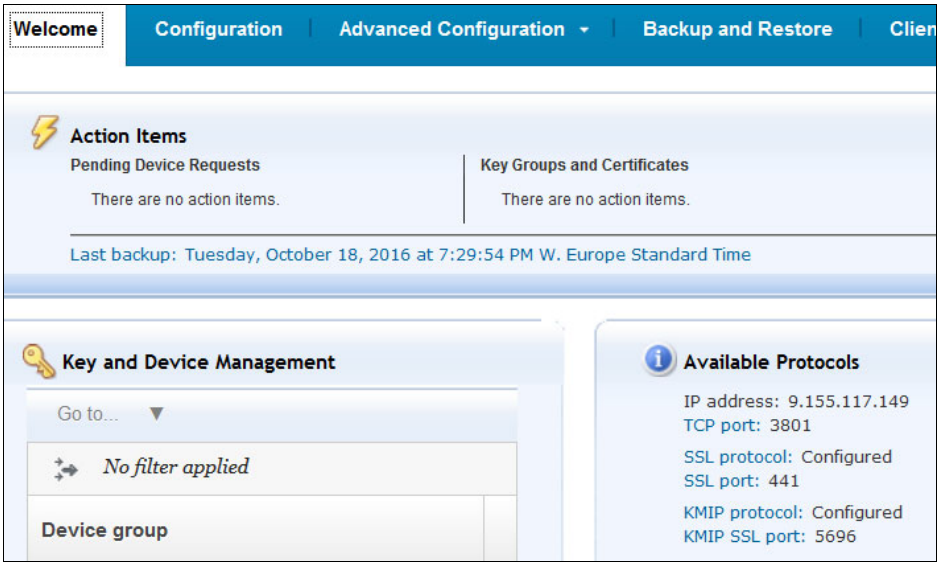

Figure 3-25 Welcome window

You have now created the IBM Security Key Lifecycle Manager master keystore and the SSL certificate. As a result, the Available protocols section now has both *SSL protocols* and *KMIP protocols* configured.

After it is created, you must export the certificate. In previous versions earlier than IBM Security Key Lifecycle Manager V2.6, you can do this only by running **`wsadmin`**.

10. Exporting IBM Security Key Lifecycle Manager server certificates can be done in the IBM Security Key Lifecycle Manager GUI starting with Version 2.6 by completing the following steps:

    a. Log in to the IBM Security Key Lifecycle Manager server after if restarts and click **Advanced Configuration** → **Server Certificates**, as shown in Figure 3-26.

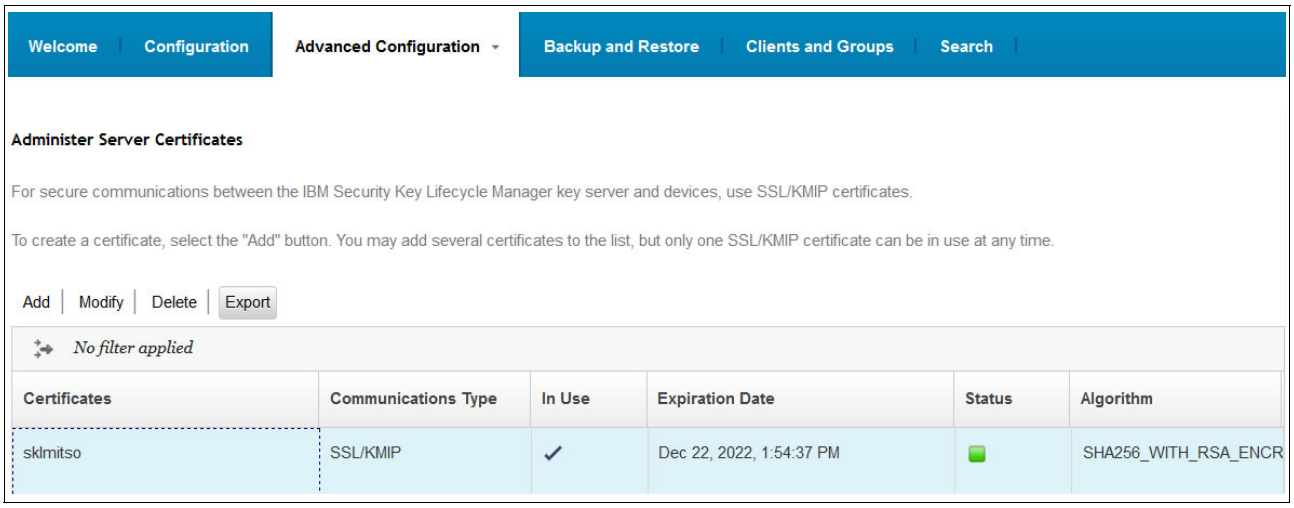

Figure 3-26 Export the server certificate

    b. Enter a server certificate file name, including the `.pem` file ending, and browse to the file location where you want it to be saved. Choose certificate type **base64** because the DER format is not supported in an XIV system, as shown in Figure 3-27.

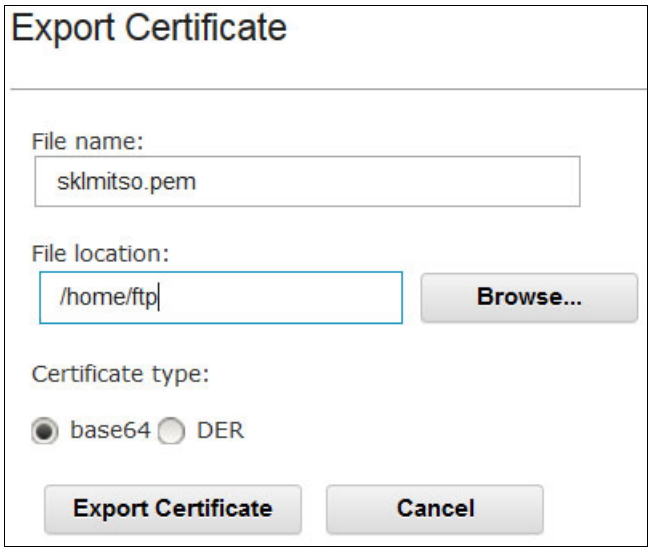

Figure 3-27 Export certificate type base64

11. For IBM Security Key Lifecycle Manager Version 2.5 and earlier, run **wsadmin** to export the server certificate:
   - For a Microsoft Windows operating system, open a DOS prompt with Administrator privileges. Run the following **wsadmin** command:

     ```
     cd <WAS_HOME> wsadmin -username sklmadmin -password <sklmadmin password>
     -lang jython
     ```

     <WAS_HOME> is, for example, `C:\Program Files (x86)\IBM\WebSphere\AppServer`.
   - For a Linux operating system, open a UNIX terminal session, and run the following command:

     ```
     cd <WAS_HOME>/bin>rm -f ./tmp/cert.der
     ./wsadmin.sh -username SKLMAdmin -password <sklmadmin password> -lang jython
     ```

     <WAS_HOME> is, for example, `/opt/IBM/WebSphere/AppServer/bin/`.
12. To view all existing certificates, run the **print AdminTask.tklmCertList()** command that is shown in Figure 3-28.

```
wsadmin>print AdminTask.tklmCertList('[-alias XIV_itso_6003310]')
CTGKM0001I Command succeeded.

uuid = CERTIFICATE-23882e94-73b3-4282-b253-185569a76a4d
alias = XIV_itso_6003310
key store name = defaultKeyStore
key state = ACTIVE
issuer name = O=ibmXIVDisk,C=US
subject name = CN=2812-6003310,O=ibmXIVDisk,C=US
creation date = 10/18/16 6:22:45 PM GMT+00:00
expiration date = 10/12/41 1:23:05 AM GMT+00:00
serial number = 10001000
```

Figure 3-28 List all certificates command in wsadmin

13. Print the certificate that is created in step 8 on page 33 by running the **print** command:

    ```
    print AdminTask.tklmCertList('[-alias <label provided in Step 2>]')
    ```

    Figure 3-29 shows an example of the result.

```
wsadmin>print AdminTask.tklmCertList('[-alias XIV_itso_6003310]')
CTGKM0001I Command succeeded.

uuid = CERTIFICATE-23882e94-73b3-4282-b253-185569a76a4d
alias = XIV_itso_6003310
key store name = defaultKeyStore
key state = ACTIVE
issuer name = O=ibmXIVDisk,C=US
subject name = CN=9835-6003310,O=ibmXIVDisk,C=US
creation date = 10/18/16 6:22:45 PM GMT+00:00
expiration date = 10/12/41 1:23:05 AM GMT+00:00
serial number = 10001000
```

Figure 3-29 List a specific certificate in wsadmin

14. Take the Universally Unique Identifier (UUID) information from the output of step 12 on page 37 and use it to export the certification file. You might want to change the **-fileName** option to something other than `/tmp/cert.der` if you want to save it in a different folder.

    The specified folder and file name are relative. Therefore, if you specify `/tmp/cert.der`, it is saved in a subdirectory of your IBM Security Key Lifecycle Manager installation directory.

15. Export the certification file by running the following command and format it as base64:

```
print AdminTask.tklmCertExport('[-uuid
CERTIFICATE-a44aba79-6bcc-47dd-94c0-23ddb5db102c -format base64 -fileName
/tmp/cert.pem ]')
```

    This is a successful output response:

```
CTGKM0001I Command succeeded /tmp/cert.pem
```

    This `.pem` file is the certificate that is passed by a parameter in the XCLI **encrypt_keyserver_define** command (as described in “Step 5: Defining the key server on the XIV system”).

    On a Windows server, you can specify the path as shown in this example:

```
print AdminTask.tklmCertExport
   ('[-uuid CERTIFICATE-a44aba79-6bcc-47dd-94c0-23ddb5db102c
      -format base64 -fileName d:\\mypath\\mycertfilename.pem]')
```

Step 5: Defining the key server on the XIV system

You can define a key server on an XIV system by adding the IBM Security Key Lifecycle Manager certificate that you generated and exported to the XIV system.

Complete the following steps:

1. If you are using the XIV GUI, log in as a Security Administrator.
2. Click **Systems** → **System Settings** → **Manage Key Servers** and click the plus sign (**+**) icon.
3. Enter the name for your key server, the key server IP address or DNS name, and select the key server certificate that you generated previously, as shown in Figure 3-30.

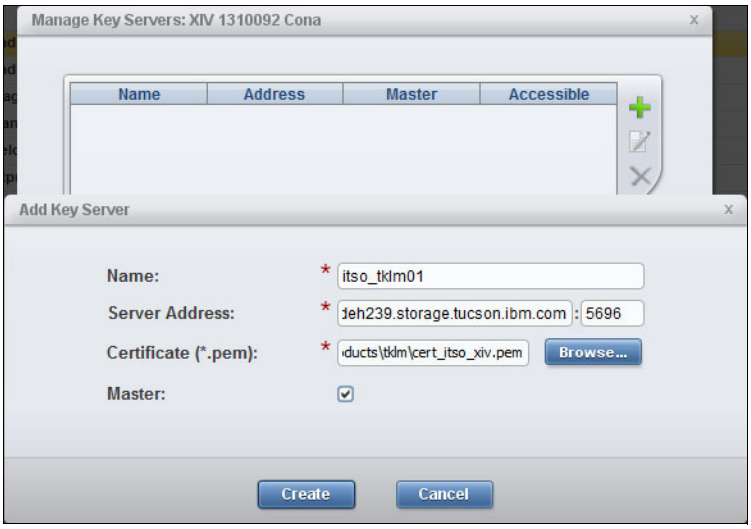

Figure 3-30 Add Key Server window

4. Click **Create**. The Manage Key Servers window shown in Figure 3-31 indicates that the key server is now trying to establish the connection with the key server.

Figure 3-31 Manage Key Servers

After the connection is established, the Accessible column displays **Yes**, as shown in Figure 3-32.

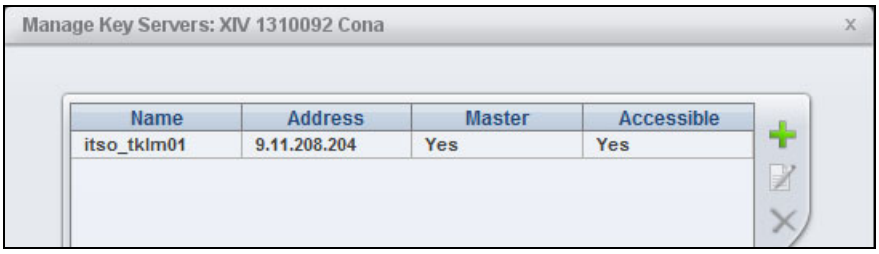

Figure 3-32 Added Key Server

You can also use the XCLI **`encrypt_keyserver_define`** command while logged in to the XIV system as a security admin user to define the key server.

Each operating system has its own way of creating line breaks in a file. Those breaks are not always visible, and if the server certificate is specified to be added as a file name, it can lead to a `bad cert` status after the key server is added.

One way to eliminate this error is to edit the server certificate in a text editor and create the whole **`encrypt_keyserver_define`** command as one line, including the certificate.

Complete the following steps:

1. At the certificate part of the command, add an asterisk (*) character at the end of each line, as shown in Figure 3-33.

```
-----BEGIN CERTIFICATE-----*
MIIDVjCCAj6gAwIBAgIEAJiaaDANBgkqhkiG9w0BAQsFADAiMQswCQYDVQQGEwJV*
UzETMBEGA1UEChMKaWJtWElWRGlzazAeFw0xNjAzMjIxNjI2NDJaFw0OMTAzMTYx*
NjI2NDJaMDkxCzAJBgNVBAYTAlVTMRMwEQYDVQQKEwppYm1YSVZEaXNrMRUwEwYD*
VQQDEwwyODEwLTYwMTM3NzUwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB*
AQCF3l+xvVIAv4Kz5yIYNP8IKXL/RlmGcTrJNmGfMzpEYcNwqYMqHzwn1TsXePN9*
vfhI6ZUJSWHH35wemZLm93SseGrZ4fXloe7k2KLPiggJiQ9p5rPN9WNYViAeCBVY*
QLCowuHBPO7FJqM8TBHJLk49BbxbSJOcfAP/TyiaCs5HWEYlcwtdqifRMnO5WD7l*
ijeMktNLVxgt8hjTF9LeWsY2oFOaNAt7EkMdOIpLwiqeeKqN19X5ovGTBCzo7K5N*
4eDuYtBCL5Ys5HZmEQKMY8egOLOZLsO2ZpSu+PMeOg6cCEBkqjmQehaKvvFkuf37*
3qGroMEANFvGeHj6e/gDKhITAgMBAAGjfTB7MEoGA1UdIwRDMEGAFOZrcCLEx/xs*
trBUvc+gKcAUMIZroSakJDAiMQswCQYDVQQGEwJVUzETMBEGA1UEChMKaWJtWElW*
RGlza4IBADAOBgNVHQ8BAf8EBAMCBaAwHQYDVROOBBYEFNOVEsjxUCSOSL+bagge*
bVHZxAJNMAOGCSqGSIb3DQEBCwUAA4IBAQB3VL+g/NICkvSmc4+N9i8TaTe6mL+w*
YQrTvG2MH8bIZEnKyGrddY1QNU8f49ECW7QV/WgbCrYiBRAcJiSibiL/7xcDf+G8*
brmUGOPELjEa1h9AKtIMNNFt3zUt28VdkuvxQiOtels93DDEtDrc1d+e+1OeQb5i*
/NnvG1ifHOZngYBeXCu4B39TDkA4FZOOsb5koj8IVxpzh9WCPZfWfqNUJEgpkRX1*
JMs1BOgy4kjsVQvajpH4IIvu897Oy5H6heIDnydsR7bsUFGw4edMx823StHlHYJ8*
ODGKj7zxHSrEc7NDHFSYQDnXX3/WmU8y1olScntv5dTwmShzzlxcdtbB*
-----END CERTIFICATE-----
```

Figure 3-33 Add asterisks at the end of line of the server certificate

2. Then, remove the line break at the end of the line behind the asterisks so that you end up having the entire command on one line, as shown in Figure 3-34.

```
XIV_ITSO>>encrypt_keyserver_define name=sklm1 ipv4=9.123.123.123 master=yes
certificate="-----BEGIN
CERTIFICATE-----*MIICyTCCAbGgAwIBAgIGKSiyd1FPMAOGCSqGSIb3DQEBCwUAMBQxEjAQBgN
VBAMTCXNrbG1pdHNv*MzAeFw0xNjEwMjEwOTIxMzlaFw0yMjEyMjQwOTIxMzlaMBQxEjAQBgNVBA
MTCXNrbG1pdHNvMzCC*ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAIGeknLONw39+wYw4
ZTvUWYjtNuOOeUJ1weO*UgG6pdfZR1N5xj5ijEYENiWMHFxkKO6Su87lQLXTEvL3QsQlxrAmOKMu
itOQxNugguEsLaCim1Nhawv1Da*Rtd1eDlAKeDlAfERReiRaZHaiVnUw8uvURzbLlofgg+/iKfnP
eFLBeBqrnkIl7FqrdVNOTObbFxPxoV6*DfM49pjG8lM5kAR93RO5UysSlz6NOw9srK6eGCfNJGWq
zpsHYK8gBmfxDHco/j/hD4+1TjLbnxna*w8gdQYTFRDkWR4oteFKDBlLM7szIHLZ3VS2CjkYW/VZ
hXGcqzzALy3ZP5uvWFFKWT5THO3I8OtJe*xpkCAwEAAaMhMB8wHQYDVROOBBYEFDUo+63Y6PQCGt
9qM8P6Oj+cZfC7MAOGCSqGSIb3DQEBCwUA*A4IBAQBwx7WrZGN9E77LcFQmjkWty9Oy44cPHuf5B
7dO64x4q3pZOqFAuY1OBpAExwOghDgW6IeY*nkaWWGfIfTie/NA+U/sP2toVVtR/xuNQSp2WKnxM
nd8ddNosHo2q9Xprx4d2CJS1H7oUj+Maf7OL*napObaHKyaa3veLHOfxyN/HWFQTfOKfOqYwqlCy
px3nLw3XF+a3PKAAZA1hSnMfdNG59RJY+xoa4*fzrmAleVx6iUqZV3jy3Eg2mf3Pam/qPlPbelmz
l4SdJBc+XMfJ2dquidQKedVYymSVy977NF4Tws*erD6HgQHSkfR3FEM+b7EfOUPFIBrys8rKtLRb
Wvovebq*-----END CERTIFICATE-----"
```

Figure 3-34 The encrypt_keyserver_define one-line command by using the server certificate

3. You can verify that the key server was added successfully by running the **encrypt_keyserver_list** command that is shown in Figure 3-35.

```
XIV_ITSO>>encrypt_keyserver_list
Module Name App/Key Status Last_time_checked Master Port Address Keyserver Type
1      sklm1 NOKEY        2016/10/26 13:48:45 yes    5696  9.123.123.123   TKLM
2      sklm1 NOKEY        2016/10/26 13:48:45 yes    5696  9.123.123.123   TKLM
3      sklm1 NOKEY        2016/10/26 13:48:45 yes    5696 9.123.123.123   TKLM
```

Figure 3-35 The encrypt_keyserver_list command

The NOKEY status shows that the XIV has not been accepted on the key server. You see a Pending Devices link in the Welcome window of the key server GUI, as shown in Figure 3-36.

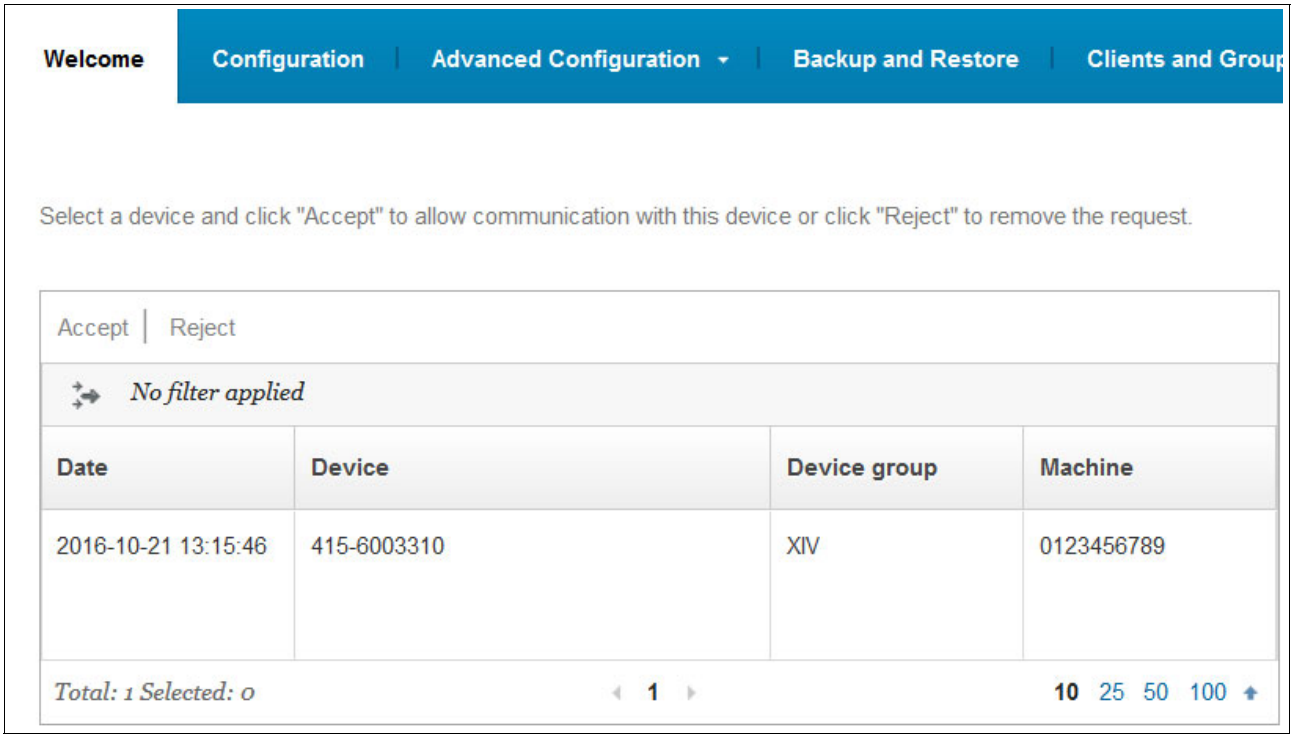

Figure 3-36 Pending Devices in the Welcome window

4. Click the **Pending Devices** action item and the XIV that you added appears, as shown in Figure 3-37.

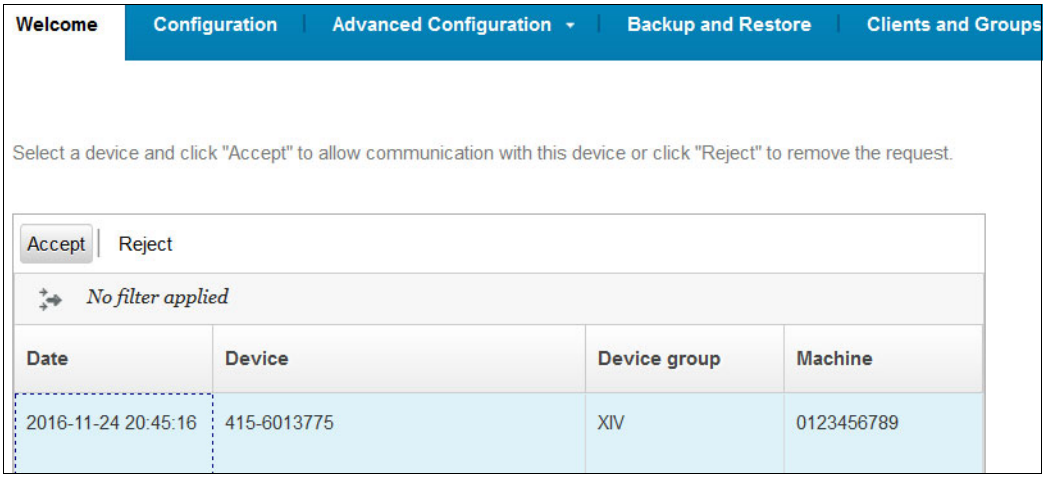

Figure 3-37 Accept the pending XIV device

5. Highlight the XIV system and click **Accept** to add it to the key server. If this device served keys in the past from that key server, a warning message, as shown in Figure 3-38, appears and advises you to take a backup before accepting the pending device. This situation can happen during migrations or if encryption was enabled and disabled before. It is a preferred practice to create a backup now to be able to revert to the current state of configuration of the key server.

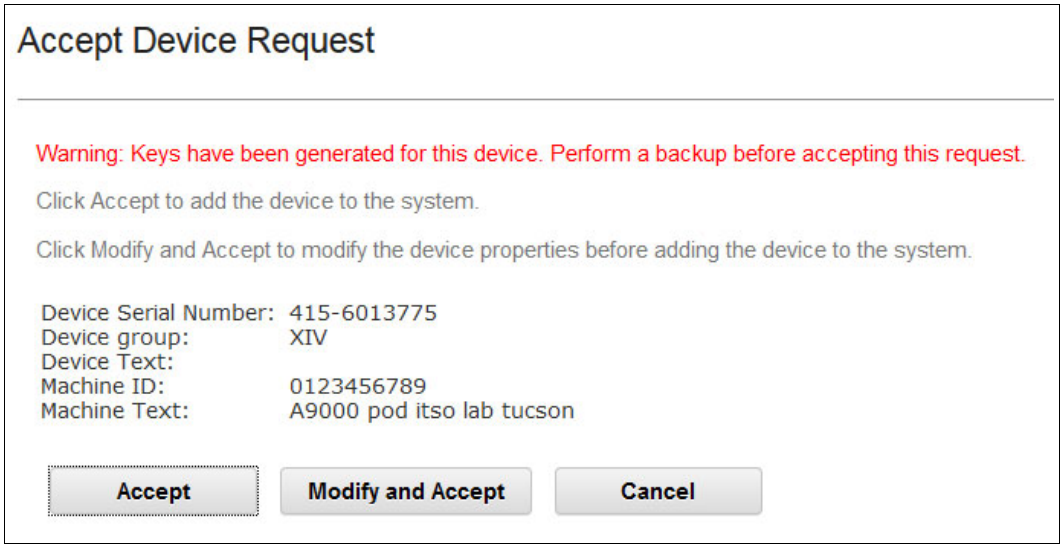

Figure 3-38 Accept Device Request

6. After the device request is accepted, it shows as a client device communication certificate in the Advanced Configuration section, as shown in Figure 3-39.

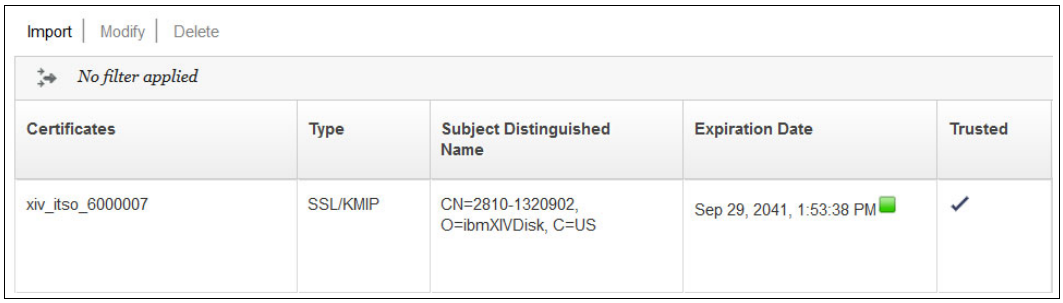

Figure 3-39 Client device communication certificates

7. Repeat the same procedure for any additional secondary key servers. If any of the secondary key server states are `NOKEY`, run **`encrypt_keyserver_check_status`** and then **`encrypt_keyserver_list`**. If that does not change the `NOKEY` status, you can replicate the complete configuration of the primary IBM Security Key Lifecycle Manager server to the secondary ones. The same can be accomplished by a manual backup/restore as well.

3.5 Recovery key use and maintenance

To protect against the possibility (following a disaster, for example) that all Security Key Lifecycle Managers become unusable and unrecoverable, the XIV system enables you to create a *recovery key*, as shown in Figure 3-40. With a recovery key, Security Administrators can unlock an XIV system without the involvement of a key server.

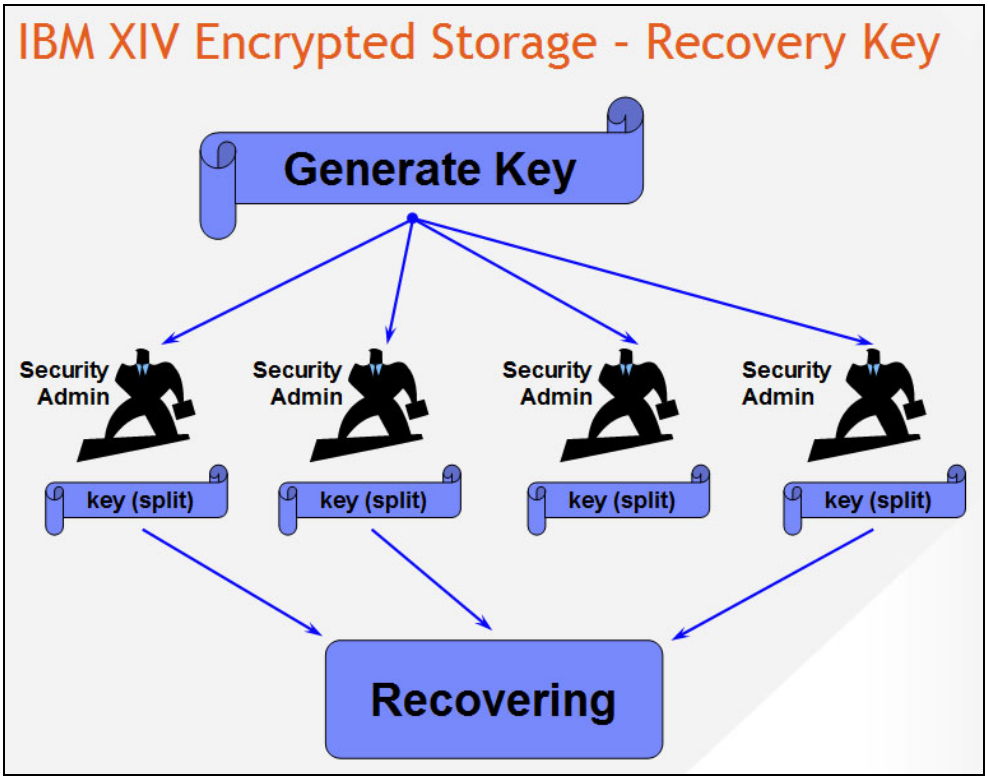

Figure 3-40 Recovery key

Encryption can be activated either in the XIV GUI or through the XCLI. If that action is through the XIV GUI, the recovery key is mandatory. The option to activate data-at-rest encryption without recovery keys is possible, but only through the XCLI by running the **`encrypt_enable`** command with the **`recovery_keys=no`** flag. The recovery keys are split according to the number of defined Security Administrators and created separately for each Security Administrator.

The recovery key is used to unwrap the XMK, which unlocks the drives.

Important: A recovery key can be created only if data-at-rest encryption is not yet enabled. You cannot create a recovery key when XIV encryption is already activated.

Managing the recovery key requires at least two Security Administrators. They maintain the recovery key and keep it safe.

Client responsibility: Although an XIV system supports two roles, Storage Administrator and Security Administrator, only a Security Administrator is allowed to use the recovery key. The client is responsible for assigning at least two *separate* individuals as Security Administrators to prevent data access by a single person.

3.5.1 Process for recovery keys

The recovery keys can be generated only if data-at-rest encryption is deactivated on the XIV system. In other words, configuring the Recovery keys is best done before initially activating encryption. Make sure that at least two XIV Security Administrators are defined on the system. Recovery key creation requires communication with the key server.

The following steps are required to configure recovery keys:

1. Generate recovery keys for each Security Administrator.
2. Get keys for each Security Administrator (make a note of them for later).
3. Verify keys for each Security Administrator to make the keys usable.

The XIV generates a random recovery key and a related wrapping key. The recovery key can also be rekeyed, which generates a new recovery key. That new key must be acquired and verified again by each defined Security Administrator.

3.5.2 Recovery key generation with the XIV GUI

Complete the following steps:

1. In the XIV GUI, log in as a Security Administrator and click **All Systems** → **List**, select your system, and select **Generate Recovery Key**, as shown in Figure 3-41.

Figure 3-41 Generate Recovery Key

2. You must choose at least two Security Administrators, add them to the Recovery Key Owners section on the right side, and then click **Start**, as shown in Figure 3-42.

   You can add more users who can unlock a locked IBM XIV. If you do that, the least number that you designate in the “Minimum recovery users” field is required to unlock the XIV system. For example, if you define three Security Administrators and add them to Recovery Key Owners, but you select only two as minimum recovery users, only two are necessary to unlock the XIV system, but three of them will be able to do so.

Figure 3-42 Generate Recovery Key pane

3. Now that the recovery key is generated, you can verify whether the process was successful by clicking **Show Results**, as shown in Figure 3-43.

Figure 3-43 Generate recovery keys result

A default text editor opens and shows the `Completed Successfully` log message that is shown in Figure 3-44.

```
Generating Recovery Keys in 1 systems
-------------------------
[+] | XIV 1310092 Cona - Completed Successfully
```

Figure 3-44 Recovery key generation log

When you close this window, the Generate Recovery Key window that is shown in Figure 3-45 informs you that each of the Security Administrators must log in to acquire the keys that are generated for them.

Figure 3-45 Recovery keys generation completed successfully" message

If the XIV data-at-rest encryption is already enabled, the process continues, but as a result, it completes with an error. When you click **Show Results**, your default text editor opens to display an error message similar to the one in Figure 3-46.

```
Generating Recovery Keys in 1 systems
-------------------------
[-] | XIV 1310092 Cona - Completed with Errors -
[-] Failed executing encrypt_recovery_key_generate
    users="itsosecadmin,itsosecadmin2" min_req="2"
[-] reason: Encryption has already been enabled.
```

Figure 3-46 Recovery key generation error log

3.5.3 Recovery key generation with XCLI

If you prefer, you can generate the recovery key through the XCLI by completing the following steps:

1. Start with the **encrypt_recovery_key_generate** command that is shown in Table 3-1.

Table 3-1 The encrypt_recovery_key_generate command

Category	Command	Description
System	**encrypt_recovery_key_generate**	Specifies which Security Administrators receive recovery key shares and the minimum number of recovery key shares that must be entered

Here is an example of this command:

```
XIV 1310092>>encrypt_recovery_key_generate min_req=2
users=itsosecadmin,itsosecadmin2,itsosecadmin3
Command executed successfully.
```

2. Each defined Security Administrator must collect and verify the keys that are generated individually by using their credentials to log in to the XCLI, as shown in Table 3-2.

Table 3-2 The encrypt_recovery_key_get command

Category	Command	Description
System	**encrypt_recovery_key_get**	Retrieves the recovery key share that is generated for the current user

Here is an example of this command:

```
XIV 1310092>>encrypt_recovery_key_get
Command executed successfully.
key=628O7CB19O2AMO74EDLA4EV8FOC574E4OA1564F5557OCDEEBED37BC3876789
```

3. All defined Security Administrators must verify their keys, as shown in Table 3-3.

Table 3-3 The encrypt_recovery_key_verify command

Category	Command	Description
System	**encrypt_recovery_key_verify**	Confirms that the current user has correctly copied the recovery key share that is presented by **encrypt_recovery_key_get**

Here is an example of this command:

```
XIV 1310092>>encrypt_recovery_key_verify
key=628O7CB19O2AMO74EDLA4EV8FOC574E4OA1564F5557OCDEEBED37BC3876789
Command executed successfully.
recovery_status=Key accepted, 1 of 3 fragments have been verified
remaining_fragments=2
```

4. The state of verification can be checked by the **encrypt_recovery_key_status** command that is shown in Table 3-4.

Table 3-4 The encrypt_recovery_key_status command

Category	Command	Description
System	**encrypt_recovery_key_status**	Shows status of recovery keys

Here is an example of this command:

```
XIV 1310092>>encrypt_recovery_key_status
Date Created          User           Status
2013-10-24 11:55:15   itsosecadmin   Verified
2013-10-24 11:55:15   itsosecadmin2  Unverified
2013-10-24 11:55:15   itsosecadmin3  Unverified
```

After all defined Security Administrators have collected and verified their keys, the XIV data-at-rest encryption can be activated. For more information, see 3.6.1, “Activating data-at-rest XIV encryption” on page 54.

You can run **encrypt_recovery_key_list** to show the number of shares that have recovery keys and how many of them are required for recovery.

3.5.4 Recovery key verification by using the XIV GUI

Complete the following steps:

1. The recovery key must be acquired by each Security Administrator. The Security Administrators must log in with their own credentials, copy and paste the key into the **Verify Key** field, and then activate it by clicking **Activate Recovery Key**, as shown in Figure 3-47.

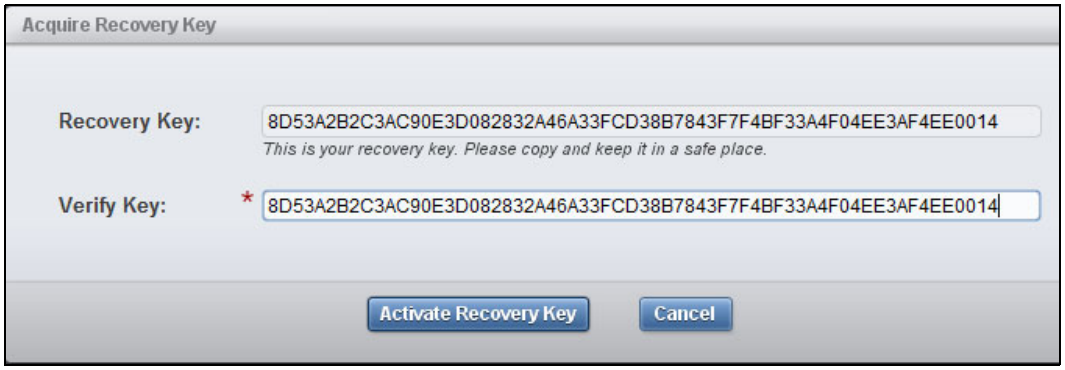

Figure 3-47 Activate Recovery Key

2. An information window is shown indicating that, after verification, you cannot acquire the key again. Save the key in a text file and keep it in a secured place that is physically separate from both the XIV system and the key servers. Click **Continue** to proceed, as shown in Figure 3-48.

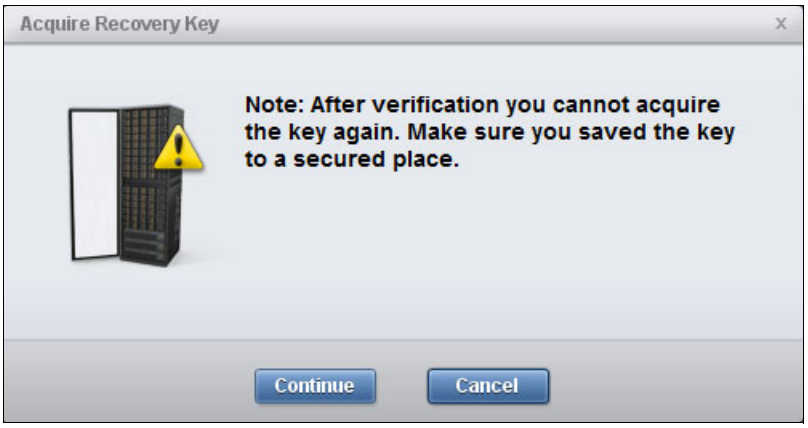

Figure 3-48 Acquire Recovery Key note

If you look at the systems lists through the XIV GUI, the Recovery Key column changes to "1 of 2 acquired."

Now, the next Security Administrator must log in to the XIV GUI with correct credentials, and then repeat the Activate Recovery Key procedure.

Tip: If multiple IBM XIV systems are defined in the XIV GUI, you can decrease the loading time after login by right-clicking the XIV system that you want to access, clicking **Modify IP addresses** without changing any entry, and clicking **Update**.

3.5.5 Recovery key verification by using the XCLI

Complete the following steps:

1. You can use the XCLI to verify the recovery keys that are defined by using the **`encrypt_recovery_key_verify`** command that is shown in Table 3-5.

Table 3-5 The encrypt_recovery_key_verify command

Category	Command	Description
System	**`encrypt_recovery_key_verify`**	Confirms that the current user has correctly copied the recovery key share that is presented by the **`encrypt_recovery_key_get`** command.

Here is an example of this command:

```
XIV_ITSO>>encrypt_recovery_key_verify
key=628O7CB19O2AMO74EDLA4EV8FOC574E4OA1564F5557OCDEEBED37BC3876789
Command executed successfully.
recovery_status=Key accepted, 1 of 2 fragments have been verified
remaining_fragments=1
```

2. The state of verification can be checked by running the **encrypt_recovery_key_status** command that is shown in Table 3-6.

Table 3-6 The encrypt_recovery_key_status command

Category	Command	Description
System	**encrypt_recovery_key_status**	Shows the status of the recovery keys

Here is an example of this command:

```
XIV_ITSO>>encrypt_recovery_key_status
Date Created          User          Status
2016-11-24 11:55:15   itsosecadmin1 Verified
2016-11-24 11:55:15   itsosecadmin2 Unverified
```

You can run **encrypt_recovery_key_list** to show the number of shares that have recovery keys and how many of them are required for recovery.

3. The next Security Administrator must log in to the XCLI with the correct credentials and repeat the Activate Recovery Key procedure.
4. Save the key in a text file and keep it in a secure place that is physically separate from both the XIV system and the IBM Security Key Lifecycle Manager servers.

Important: The rekey process is not finished until all defined Security Administrators have confirmed their keys. Until all newly generated keys are confirmed, the “old” keys remain active. The key switch from old to new is an atomic operation carried after all new keys are confirmed.

After all defined Security Administrators have collected and verified their keys, the XIV data-at-rest encryption can be activated. For more information, see 3.6.1, “Activating data-at-rest XIV encryption” on page 54.

3.5.6 Recovery key rekey

Rekeying is the process of changing cryptographic values in the chain between the key server, recovery key, and DAKs so that the previous value no longer enables access to the system.

The rekey and Verify Recovery Key functions can be performed any time while the recovery key is configured and a key server is available. A key server is required to enable the XIV system to verify that it is in the correct environment.

Only when the key server can decrypt the data key can the XIV system be sure that it is in the same environment. Only then, it generates a new recovery key. For example, on an XIV system that was stolen and put in a separate environment, rekeying the recovery key is not possible.

During a rekeying operation, the following actions are performed:

1. The XIV system sends the ESK to the key server and requests a rekey validation.
2. The key server verifies the identity of the XIV system by using its certificates.
3. The key server signals the XIV system that it can proceed to generate a recovery key.
4. The XIV system generates a recovery key.

Changing the recovery key does not erase the data.

An Unconfigure function of the recovery key is not available after data-at-rest encryption is activated, but you can use the Regenerate Recovery Key function to change your keys.

The recovery key can also be rekeyed to replace the current recovery key with a new one. All defined Security Administrators must collect and verify the new recovery key.

To regenerate the recovery key, you can use either of the following approaches:

- In the XIV GUI, log in as a Security Administrator, and select **All Systems** → **List**, select your system, and select **Re-Generate Recovery Key**, as shown in Figure 3-49.

Figure 3-49 Re-Generate Recovery Key

- The recovery key can also be rekeyed in the XCLI by using the **encrypt_recovery_key_rekey** command that is shown in Table 3-7.

Table 3-7 The encrypt_recovery_key_rekey command

Category	Command	Description
System	**encrypt_recovery_key_rekey**	Restarts the recovery key generation process that is described in **encrypt_recovery_key_generate**.

Here is an example of this command:

```
XIV 1310092 Cona>>encrypt_recovery_key_rekey
Command executed successfully.
```

3.5.7 Using a recovery key to unlock an XIV system

On an XIV system with a recovery key configured, an option exists to let a Security Administrator enter the recovery key.

After a power-off followed by a power-on action, if the XIV system cannot get the required data key from the master key server, it attempts to contact all other configured key servers to obtain the required key. If that is not successful, the Security Administrator provides the recovery key. The XIV system uses the recovery key to unwrap the XMK that unlocks the drives. After access to data is restored, the XIV system is available to serve host I/O again.

Each Security Administrator enters their individual parts of the recovery key until the number of defined minimum required Security Administrators is reached and it is again possible to unlock the disks. The XCLI command to do so is **encrypt_recovery_key_enter**, as shown in Table 3-8.

Table 3-8 encrypt_recovery_key_enter

Category	Command	Description
System	**encrypt_recovery_key_enter**	Unlocks encrypted disks when the system reboots and cannot access any of the defined key servers if the recovery keys were defined

This example shows the command:

```
encrypt_recovery_key_enter
key=62807CB1902AM074EDLA4EV8F0C574E40A1564F55570CDEEBED37BC3876789
```

As soon as the last one of the minimum number of defined Security Administrators has logged in with credentials and entered a recovery key, the XIV system unlocks and activates the data-at-rest encryption again, as shown in Figure 3-50.

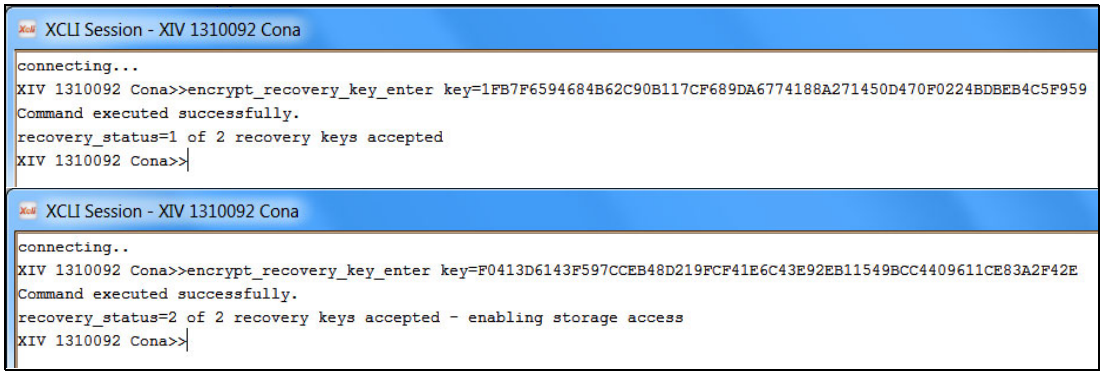

Figure 3-50 The encrypt_recovery_key_enter command

Important: If your XIV system has a microcode level that is earlier then 11.5, the **encrypt_recovery_finish** command is not available. In this case, an IBM SSR must access the system with *technician* authority and change the state of the XIV system from `maintenance` to `on` by running a **state_change target_state=on** command. A Storage Administrator does not have the authority to run that command.

Now, you can verify by running the **state_list** command again that the system_state has changed to on, as shown in Figure 3-51. The I/O operation has resumed and the XIV system is serving data to its hosts again.

```
XIV_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
```

Figure 3-51 The state_list command

3.6 Activating or deactivating encryption

Now that the implementation and configuration of the XIV system and its corresponding key server are finished, you can enable (activate) the data-at-rest encryption in the XIV system.

3.6.1 Activating data-at-rest XIV encryption

For data-at-rest encryption to complete successfully, all of these prerequisites must be fulfilled:

- The current encryption state must be DISABLED (displayed as Supported by **state_list**).
- One master key server must be configured successfully, and recovery keys must be generated and verified by at least two separate Security Administrators, unless a **recovery_keys=no** parameter was passed. This action can be handled either in the XIV GUI or through the XCLI.

In the XIV GUI, log in as Security Administrator, and click **Systems** → **System Settings** → **Activate Encryption**, as shown in Figure 3-52.

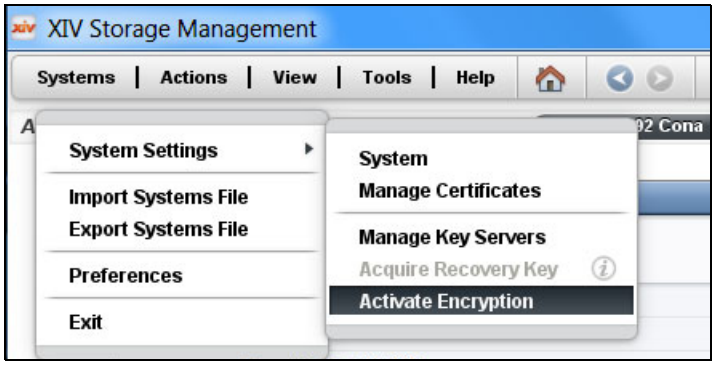

Figure 3-52 Activate data-at-rest XIV encryption

This command is entered by a Security Administrator to enable the data protection feature. Optionally, the data-at-rest encryption can be activated from the XCLI by using the command that is shown in Table 3-9.

Table 3-9 The encrypt_enable command

Category	Command	Description
System	**encrypt_enable**	Enables the data protection feature

This example shows the command:

```
XIV 1310092 Cona>>encrypt_enable
Warning:   ARE_YOU_SURE_YOU_WANT_TO_ENABLE_ENCRYPTION y/n: y
Command executed successfully.
```

3.6.2 Deactivating XIV data-at-rest encryption

You can disable the data protection feature. A prerequisite is that no volumes are defined on the system. In addition to disabling the data protection, a cryptographic erase is performed on all protected bands to ensure that all existing user data is no longer accessible. After the command completes successfully, all bands are left in an unlocked state. Disabling encryption when the encryption state is other than ACTIVE is an error (**state_list** needs to show it as “Enabled”).

In the XIV GUI, log in as Security Administrator, and click **Systems** → **System Settings** → **Deactivate Encryption**, as shown in Figure 3-53.

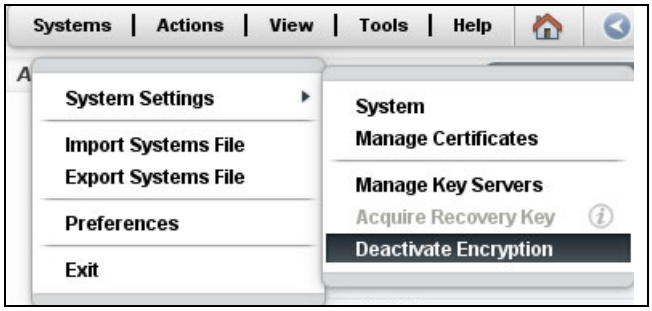

Figure 3-53 Deactivate Encryption

As Figure 3-54 shows, the system prompts you to verify that you want to deactivate encryption on the XIV system.

Figure 3-54 Deactivate Encryption verification

3.7 Verifying encryption state

These actions enable users to verify the encryption state of the system:

- The encryption state column in the output of the XCLI **state_list** command shows any of these states: Supported (encryption is disabled), Enabling/Activating, Partial, Enabled/Activated, Enabling on Boot, or Disabling. It is shown as Enabled in Figure 3-55.

```
XIV_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
```

Figure 3-55 Encryption state

- The key server state can be checked with the **encrypt_keyserver_list** command, as shown in Figure 3-56. The status is checked once every hour or if something has changed or was updated.

```
XIV_ITSO>>encrypt_keyserver_list
Module Name App/Key Status Last_time_checked Master Port Address Keyserver Type
1      sklm1 ACTIVE        2016/10/26 13:48:45 yes    5696  9.123.123.123   TKLM
2      sklm1 ACTIVE        2016/10/26 13:48:45 yes    5696  9.123.123.123   TKLM
3      sklm1 ACTIVE        2016/10/26 13:48:45 yes    5696 9.123.123.123   TKLM
```

Figure 3-56 Key server state

- Starting with XIV software Version 11.3, in the new encryption state column in the output of the XCLI **disk_list** command, states can be Banded or Enrolled.
- The **ssd_list** command shows the encryption state of the solid-state drives (SSDs) that are used as flash cache in the system. Encryption-related columns are encryption_state and secure_erase_status.

More data-at-rest encryption-related XCLI commands are listed in IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/api/redirect/ibmxiv/r2/index.jsp

The *Command-Line Interface (CLI) Reference Guide* includes an “Encryption enablement and support commands” section, and can be downloaded from IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/en/STJTAG/com.ibm.help.xivgen3.doc/xiv_pubsrelatedinfoic.html

4

Implementing encryption on IBM FlashSystem A9000 and A9000R

This chapter describes how to configure and implement data-at-rest encryption for IBM FlashSystem A9000 and IBM FlashSystem A9000R systems. It includes a specific section for the local key server, which shares many of the same processes and flow as the external key server.

This chapter covers the following topics:

- IBM FlashSystem A9000/A9000R encryption
- External Encryption mechanism and process
- Recovery key use and maintenance
- Activating and deactivating encryption
- Verifying the encryption state
- Local encryption mechanism and process
- Converting from external to local key encryption

Note: Several figures s in this chapter are based on Version 2.6 of the IBM Security Key Lifecycle Manager GUI and Gemalto SafeNet KeySecure Version 8.

4.1 IBM FlashSystem A9000/A9000R encryption

IBM FlashSystem A9000 / A9000R offers a local encryption solution in addition to the enterprise centralized external key management solution.

In both cases, data-at-rest encryption also protects the stored data if any of the SSDs or flash enclosure MicroLatency modules are stolen, improperly discarded, or accessed without authorization.

For IBM FlashSystem A9000 and A9000R, SSDs in grid controllers act as vaulting devices that hold the de-staged cache data and metadata during an IBM FlashSystem A9000 or IBM FlashSystem A9000R system shutdown. The vaulting SSDs are also securely erased and encrypted after each successful devault. The encryption of system data and metadata is not required, so system data and metadata are not encrypted.

The storage system secures all written data with industry-standard AES-256 encryption for data-at-rest on media. All encryption activities are carried out using on-board hardware to avoid any performance impact.

Each MicroLatency module has dual Field Programmable Gate Array (FPGA) control for data-at-rest encryption that goes unused if not enabled. Thus, the use and enabling of encryption has no performance impact. Data at rest is protected by an Advanced Encryption Standard (XTS-AES) algorithm by using the 256-bit symmetric option in xor-encrypt-xor (XEX)-based tweaked-codebook mode with ciphertext stealing (XTS) mode, as defined in the IEEE1619-2007 standard.

SEDs: Certain IBM products that implement encryption of data at rest, which is stored on a fixed-block storage device, implement encryption by using self-encrypting drives (SEDs). IBM FlashSystem A9000 / R flash module chips do not use SEDs. All encryption and decryption activities are performed by dedicated hardware chips embedded on the IBM MicroLatency modules, which can be thought of as the functional equivalent of Self-Encrypting Flash Controller (SEFC) cards.

4.2 External Encryption mechanism and process

When the user enables encryption, a random XIV Master Key (XMK) is generated for IBM FlashSystem A9000, or IBM FlashSystem A9000R system. This key is protected (*wrapped*) by the externally stored key (ESK), creating an Encrypted Master Key (EXMK).

Optionally, a manual recovery key (REK) can also be configured and used to wrap the XMK, creating a Recovery Encrypted Master Key (REXMK) and providing the possibility to recover the master key if the key server becomes unavailable.

Important: A recovery key can be created only if data-at-rest encryption is not yet enabled. You cannot create a recovery key after encryption is activated. However you do have the option of rekeying a recovery key.

When you enable encryption on an IBM FlashSystem A9000 or IBM FlashSystem A9000R system, the following keys are generated:

- For each MicroLatency module, a TAK (Access Key) is generated in each flash enclosure that is based on the XMK and the serial number of the flash enclosure.
- For the vaulting devices (SSDs), the device access key is generated that is based on the XMK and the grid controllers MegaRaid adapter serial number.

The encryption for the IBM FlashSystem A9000 or IBM FlashSystem A9000R can be enabled during the installation of the system or later. If encryption is not enabled, the system might not meet the customer's compliance standards or the legal compliance standards and the data might not be protected against security threats.

Encryption enablement on IBM FlashSystem A9000 or IBM FlashSystem A9000R systems encrypts the MicroLatency modules and the SSDs in a hot encryption manner.

Important: The encryption for IBM FlashSystem A9000 or IBM FlashSystem A9000R can be disabled only when no pools or volumes are defined. Therefore, you must back up any data that is to be kept (or migrate it to another system) before you deactivate encryption on any active encrypted A9000/R system.

4.2.1 External key server encryption process overview

Following the external key server methodology, the external key server must be installed and configured first. Then, connectivity between the IBM FlashSystem A9000 or A9000R and an external key server must be verified.

IBM certified Gemalto SafeNet KeySecure as another viable external key server in addition to IBM Security Key Life Cycle Manager (SKLM). The configuration steps for SKLM are described in 4.2.3, "IBM Security Key Lifecycle Manager installation" on page 62. The configuration steps for or SafeNet KeySecure are described in 4.2.5, "Setting up SafeNet KeySecure encryption" on page 80.

In either case, the key server first must establish a trusted connection by exchanging their certificates, as described in "Overview of configuration steps" on page 61.

Then, as shown in step 3 in Figure 4-1 on page 60, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system generates a random master key (XMK) that is used to create Access Keys for MicroLatency modules and vaulting SSDs devices. Although Figure 4-1 on page 60 shows SKLM, the same process applies for Gemalto SafeNet.

Next, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system requests and receives the ESK from the key server. The ESK is used to wrap (encrypt) the XMK that is stored in the IBM FlashSystem A9000 or IBM FlashSystem A9000R system.

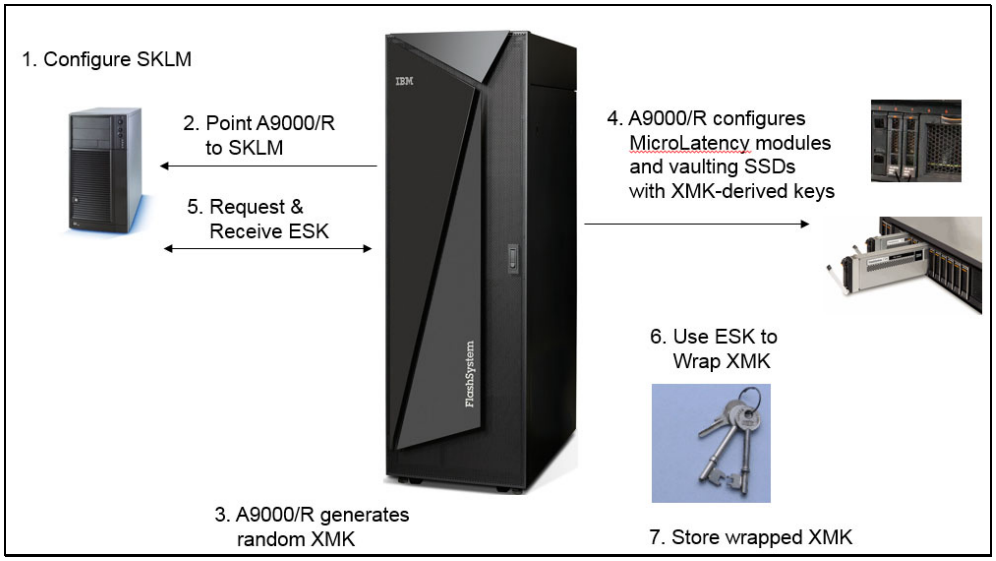

Figure 4-1 Initial configuration

After IBM FlashSystem A9000 or IBM FlashSystem A9000R data-at-rest encryption is activated and upon starting (after power maintenance, for example), the main encryption start sequence occurs, as shown in Figure 4-2.

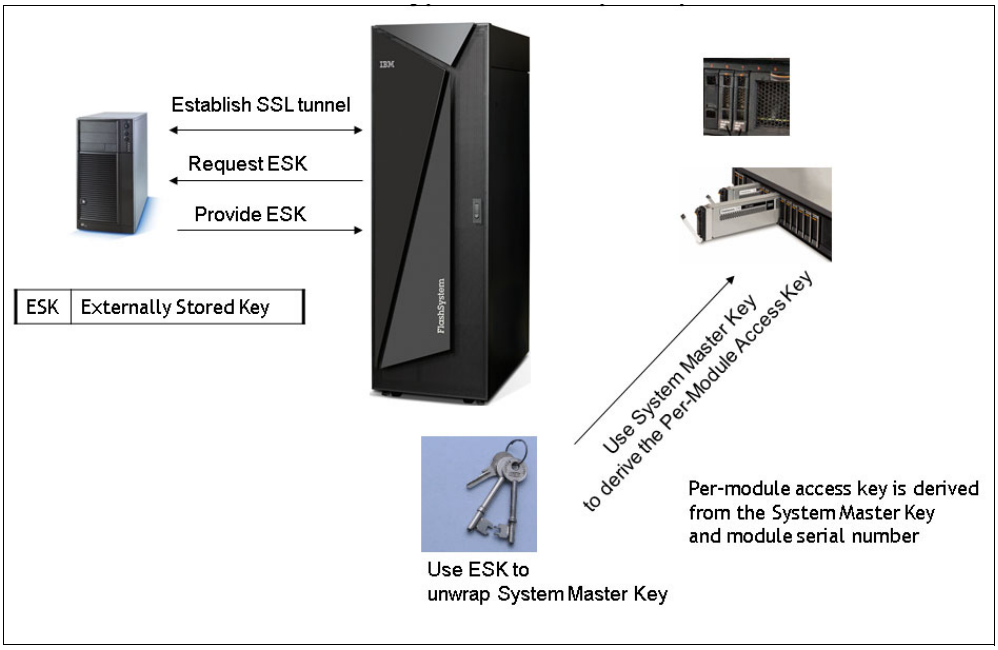

Figure 4-2 External encryption startup sequence

The MicroLatency modules and vaulting SSDs are locked during a restart. Therefore, you need a valid connection to the key server.

When the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is powering on from a reboot or cold start, it attempts to establish a Secure Sockets Layer (SSL) tunnel that is based on the Key Management Interoperability Protocol (KMIP).

The identity certificates on the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and the key server must match. The IBM FlashSystem A9000 or IBM FlashSystem A9000R system then requests the ESK, which the key server provides. It is used to unwrap the XMK to derive the Access Keys that unlock the MicroLatency modules.

Note: If no valid key server is available at power-on, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system will boot and stay in maintenance mode with no host I/O possible because all MicroLatency modules are cryptographically locked. This same key server check also occurs during a restart of the array, whereby the array will lock all modules until a valid key server is contacted and provides the correct keys.

4.2.2 External key server encryption configurations

This section describes the steps that are required to prepare IBM Security Key Lifecycle Manager to serve an encryption-enabled IBM FlashSystem A9000 or IBM FlashSystem A9000R system. It is based on the assumption that IBM Security Key Lifecycle Manager server is installed and ready to be configured.

Overview of configuration steps

Complete the following steps to configure IBM Security Key Lifecycle Manager for an IBM FlashSystem A9000 or IBM FlashSystem A9000R system:

1. Install and define either of the following components:
   - IBM Security Key Lifecycle Manager master keystore.
   - Gemalto SafeNet KeySecure infrastructure or appliance.
2. Verify the date and time of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and key server. We advise you to use an NTP server to ensure every element is locked into the same date and time stamp. Failure or “drift” of any element’s system clock can present issues or failed authentication.
3. Create secadmin role users on each A9000/ R.
4. Complete the following steps to create and manage certificates:
   a. Copy the IBM FlashSystem A9000 or IBM FlashSystem A9000R device-specific certificate from the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, and add the IBM Security Key Lifecycle Manager or SafeNet (import the certificate).
   b. On SKLM, verify the “Keep pending client device communication” option certificates and FlashSystem A9000 or A9000R affinity, and hold new device requests pending approval.
   c. Create an IBM Security Key Lifecycle Manager self-signed certificate on the IBM Security Key Lifecycle Manager GUI (add the KMIP-based SSL server certificate).
   d. Export the IBM Security Key Lifecycle Manager server certificate.
5. Define the IBM Security Key Lifecycle Manager key server on the IBM FlashSystem A9000 or IBM FlashSystem A9000R system (import the `cert.pem` IBM Security Key Lifecycle Manager Certificate).

Important: As a next step, a preferred practice is to define recovery keys *before* encryption is activated.

4.2.3 IBM Security Key Lifecycle Manager installation

Starting with IBM Security Key Lifecycle Manager Version 2.6, the supported operating systems are SUSE Linux, Red Hat Linux, AIX, and Windows.

For specific supported operating system requirements, see IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/SSWPVP_2.6.0/com.ibm.sklm.doc/install_guide/cpt/cpt_ic_release_oview_sw.html?view=embed

Use the REST interfaces as an alternative to the command-line interface (CLI), which might become deprecated in future versions of IBM Security Key Lifecycle Manager.

All references to the alias property of cryptographic keys and certificates in the graphical user interface, command-line interface, and REST interface will be deprecated in the later versions of IBM Security Key Lifecycle Manager.

You can find installation instructions online in the IBM Knowledge Center and chose your desired IBM Security Key Lifecycle Manager version:

https://www.ibm.com/support/knowledgecenter/en/SSWPVP_2.6.0/com.ibm.sklm.doc/install_guide/top/landing-install.html/

A collection of IBM Security Key Lifecycle Manager Version 2.6 pdf documents can be found at the following website:

http://www.ibm.com/support/docview.wss?uid=swg27046975

4.2.4 SKLM External key server configuration steps

This section describes the steps to configure and implement the IBM FlashSystem A9000 or IBM FlashSystem A9000R system with an external key server, such as IBM Security Key Lifecycle Manager. This section also applies when converting from an internal key server to an external one.

The IBM Security Key Lifecycle Manager solution provides simple-to-use installation options and a management console.

Step 1: IBM Security Key Lifecycle Manager master keystore

Version 2.6 of the IBM Security Key Lifecycle Manager generates the AES 256-bit XMK automatically for data encryption after a successful installation of IBM Security Key Lifecycle Manager. To conform with the HIPAA and PCI-DSS standards and for increased data security, a 256-bit length SMK is used for encrypting IBM Security Key Lifecycle Manager sensitive data, such as key material. This keystore holds all keys and certificates that are managed by IBM Security Key Lifecycle Manager. On older key server versions, you might have to manually create the master keystore.

Step 2: Verifying the date and time of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and the key server

When you implement external key server encryption, you must have a matching date and time to avoid validity issues, especially when the key server and the system to be encrypted are in different time zones. A mismatch might lead to certificates becoming valid at a future time in the other time zone.

Complete the following steps:

1. Log in to the IBM FlashSystem A9000 or IBM FlashSystem A9000R XCLI by using an administrator role-based user and run **time_list**, as shown in Figure 4-3.

```
A9000_ITSO>>time_list
Time       Date         Time Zone   Daylight Saving Time
16:07:38   2016-10-17   UTC         no
```

Figure 4-3 The time_list command

2. You can match times between the external key server and the IBM FlashSystem A9000 or IBM FlashSystem A9000R system by adjusting the time zone or setting the time manually. If you choose to adjust the time zone, make sure that you use a valid time zone. Otherwise, your IBM FlashSystem A9000 or IBM FlashSystem A9000R system might disappear from the HSM UI. If that happens, deleting the IBM FlashSystem A9000 or IBM FlashSystem A9000R system entry in the HSM UI and adding it again brings it back. Valid time zones can be shown by running the **timezone_list** command, as shown in Figure 4-4.

```
A9000_ITSO>>timezone_list
Timezone
Universal
Egypt
Africa/Nairobi
Africa/Mbabane
Africa/Niamey
Africa/Dakar

Africa/Luanda
...
Europe/Ulyanovsk
NZ-CHAT
EET
GB
GMT
```

Figure 4-4 Excerpt of the timezone_list command

3. Run **timezone_set** to adjust the time zone of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system to match one of the valid time zones that are produced by the command in Figure 4-4, as shown in Figure 4-5.

```
A9000_ITSO>>timezone_set timezone=UTC
Command executed successfully.
```

Figure 4-5 The timezone_set command

4. If the date and time do not match between the key server and the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, you can set it on the A9000 / R by running the **time_set** command, as shown in Figure 4-6.

```
A9000_ITSO>>time_set time=2016-10-18.12:02:00
Command executed successfully.
```

Figure 4-6 The time_set command

5. Also, check whether your IBM FlashSystem A9000 or IBM FlashSystem A9000R system shows the encryption support in the output of the **state_list** command, as shown in Figure 4-7.

```
A9000_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Supported
data_reduction_state     Online
```

Figure 4-7 Show encryption support by running the state_list command

If the encryption category does not state a supported value, contact IBM Support.

Step 3: Creating security admin role users

Preferred practices for managing and configuring encryption with an IBM FlashSystem A9000 or IBM FlashSystem A9000R system requires a minimum of two secadmin role users. To create them, either use the Hyper-Scale Manager User Interface (HSM GUI) or the XCLI.

With the Hyper-Scale Manager GUI, complete the following steps:

1. Open a web browser, and specify the IP address and IP port of the HSM by using the following web address format:

   `https://<HSM_IP>:<HSM_PORT>/`

   For example:

   `https://9.123.123.123:8443/`

2. Enter your IBM FlashSystem A9000 or IBM FlashSystem A9000R system administrator role user ID and password, and then click **Login**.
3. The HSM UI Dashboard indicates that you are successfully logged in to the IBM Hyper-Scale Manager. If you manage multiple IBM FlashSystem A9000 or IBM FlashSystem A9000R systems, choose the correct one from the list in the Dashboard, as shown in Figure 4-8.

Figure 4-8 Choose your IBM FlashSystem A9000 or IBM FlashSystem A9000R system from the dashboard

4. On the Dashboard's left side, click the **Access Views** icon and then click **Users** to open the **Users** menu, as shown in Figure 4-9.

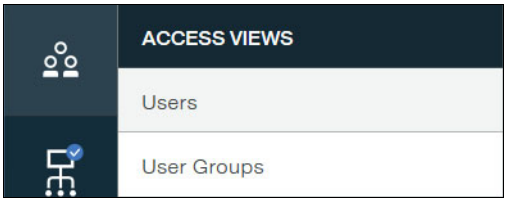

Figure 4-9 Users access view

5. In the Users access view, click the **+** icon to create an object, as shown in Figure 4-10.

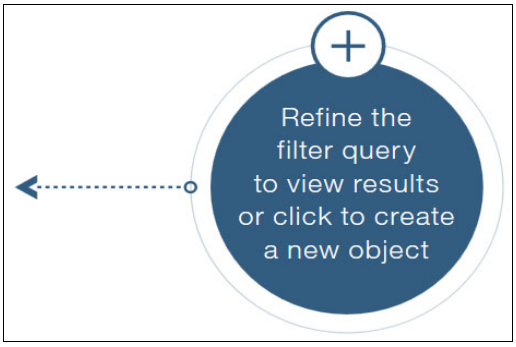

Figure 4-10 Create a user object

Enter the name for your security admin user, select a system and password, change the category to **Security Administrator**, and click **Create**. **Phone** and **Email** are optional fields. You cannot choose a domain because Security Admin is associated to the Global Space only. See Figure 4-11.

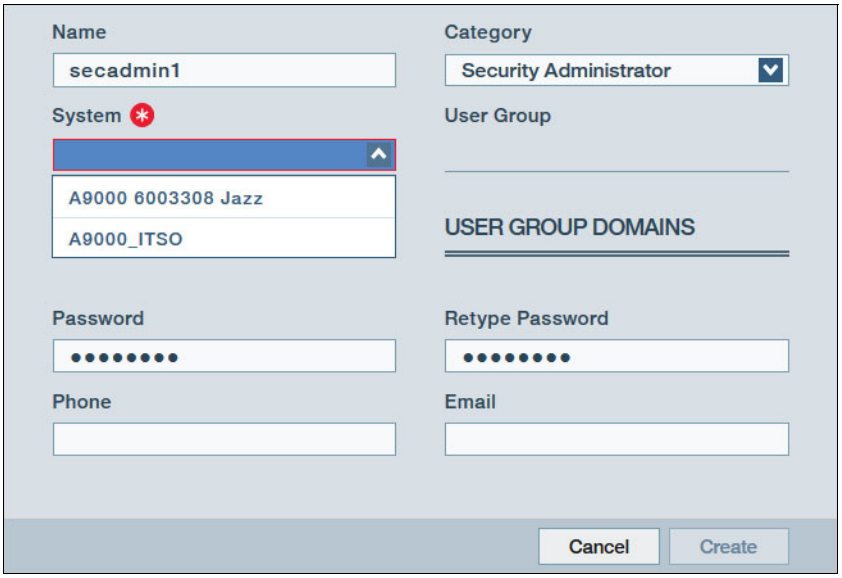

Figure 4-11 Create a secadmin user

6. Repeat these steps for further users. The minimum secadmin users that are required is two.

You can also use the XCLI to add the secadmin role users by running the **user_define** command, as shown in Figure 4-12.

```
user_define user=secadmin1 category=securityadmin password=passw0rd
password_verify=passw0rd

user_define user=secadmin2 category=securityadmin password=passw0rd
password_verify=passw0rd
```

Figure 4-12 Define secadmin users in XCLI

Run the **user_list** command to verify that the creation of the secadmin users is successful, as shown in Figure 4-13.

```
A9000_ITSO>>user_list
Name                Category     Group     Active
xiv_development     xiv_development        yes
xiv_maintenance     xiv_maintenance        yes
admin               storageadmin           yes
technician          technician             yes
xiv_hostprofiler    xiv_hostprofiler       yes
manager_server_user storageadmin           yes
secadmin1           securityadmin          yes
secadmin2           securityadmin          yes
```

Figure 4-13 The user_list command example

You can change certain user's parameters by running the **user_update** command, as shown in Figure 4-14.

```
A9000_ITSO>>user_update
user=              password=          password_verify=   email_address=
number=            area_code=         exclusive=
```

Figure 4-14 The user_update command example

Tip: After a Security Administrator role base user owns a recovery key, the user name cannot be changed anymore, as shown in Figure 4-15.

```
A9000_ITSO>>user_rename user=secadmin1 new_name=secadmin1b
Error:   USER_OWNS_RECOVERY_KEY
Details: User owns recovery key and therefore cannot be deleted or renamed
```

Figure 4-15 User owns the recovery key

Step 4: Managing certificates

To manage and configure the IBM Security Key Lifecycle Manager certificates, log in to the XCLI. You must log on to your IBM FlashSystem A9000 or IBM FlashSystem A9000R system as Security Administrator to complete the following tasks:

1. Copy the IBM FlashSystem A9000 or IBM FlashSystem A9000R device-specific certificate from the IBM FlashSystem A9000 or IBM FlashSystem A9000R system by exporting the certificate, and add it to the IBM Security Key Lifecycle Manager. To get the certificate name that is required in step 2 on page 67, run the **pki_list** command, which shows the IBM FlashSystem A9000 or IBM FlashSystem A9000R default device-specific certificate that was installed during manufacturing, as shown in Figure 4-16.

```
A9000_ITSO>>pki_list
Name Fingerprint                      Has signed certificate Services
XIV  2c4cd4f951e48d664ebd978357e5a16b yes
                  XCLI,CIM,IPSEC,KMIP
```

Figure 4-16 The pki_list command

2. Using the name that you get from the output (this example says <*default certificate*>), run **pki_show_certificate name=<*default_certificate*>**, as shown in Figure 4-17.

```
A9000_ITSO>>pki_show_certificate name=XIV
Certificate:
-----BEGIN CERTIFICATE-----
MIIDVjCCAj6gAwIBAgIEAJiaaDANBgkqhkiG9w0BAQsFADAiMQswCQYDVQQGEwJV
UzETMBEGA1UEChMKaWJtWElWRGlzazAeFw0xNjEwMTcyMzIzMDVaFw0OMTEwMTEy
..
..
aBcZn89DohhoURXMQvxVvkLQ+Q2LWJ7fV/zrNxgFQHIEbUdpu2PJwRFWqzbbc3Bm
Jou1rZ2FlmYSwrFbremhA54VtjogP35UC/Vb9wAPTYgNxF822vF1mWE4jVYr2xys
UzWUZM8/HdMz1sWycpcedhR6xELTZ2HMrql3XJRHSPk9W/R9XY4UHbEb
-----END CERTIFICATE-----
```

Figure 4-17 The pki_show_certificate excerpt

3. Copy the portion that includes `----BEGIN CERTIFICATE----` through `----END CERTIFICATE----`, paste it into a blank text editor, and save the file as <*filename*>`.pem`.
4. Place this file on the SKLM server so that it can be located during the SKLM Client Certificate process in the following steps.

   As you can see, this manufacturing default certificate is readable. After the data-at-rest encryption is activated, the public key is wrapped, and it is encrypted.
5. Inside SKLM, from the Configuration tab, select **Keep pending client device communication certificates**, which can be set in the Key Serving Parameters tab of the Configuration tab, as shown in Figure 4-18 on page 68.

   When the setting is disabled, client device communication certificates must be manually imported in the wsadmin CLI and do not show up in the Welcome window automatically.

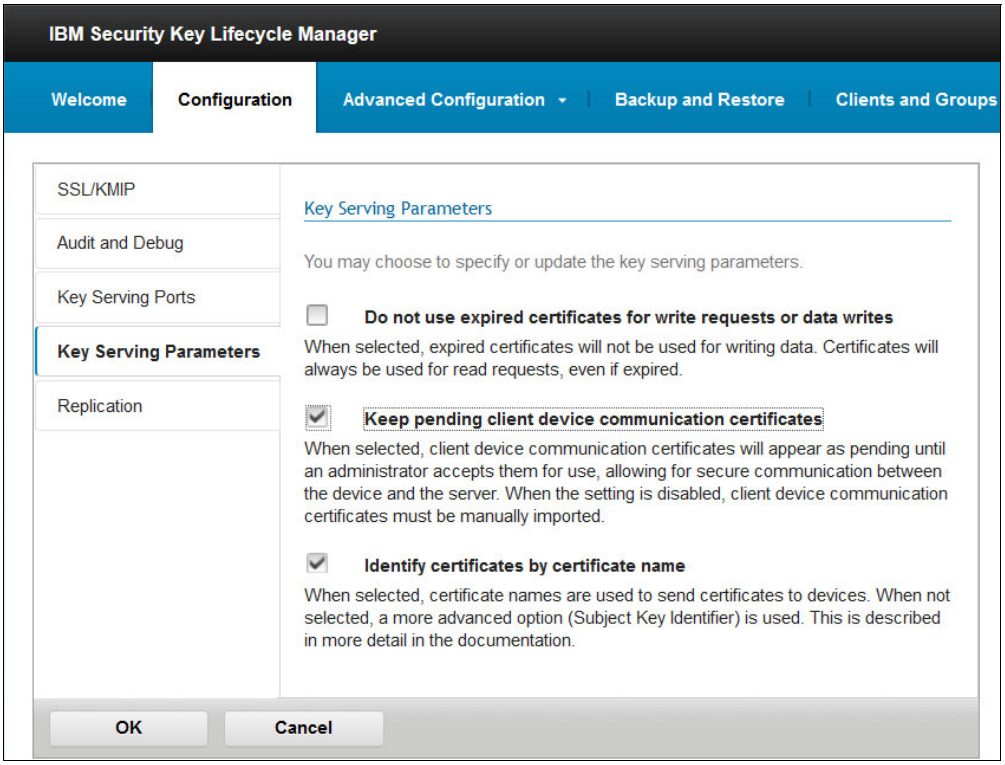

Figure 4-18 Keep pending client device communication certificates

6. In the Welcome window of the IBM Security Key Lifecycle Manager GUI, select the IBM FlashSystem A9000 or IBM FlashSystem A9000R system Device Group and then select **Go to** → **Manage keys and devices**, as shown in Figure 4-19.

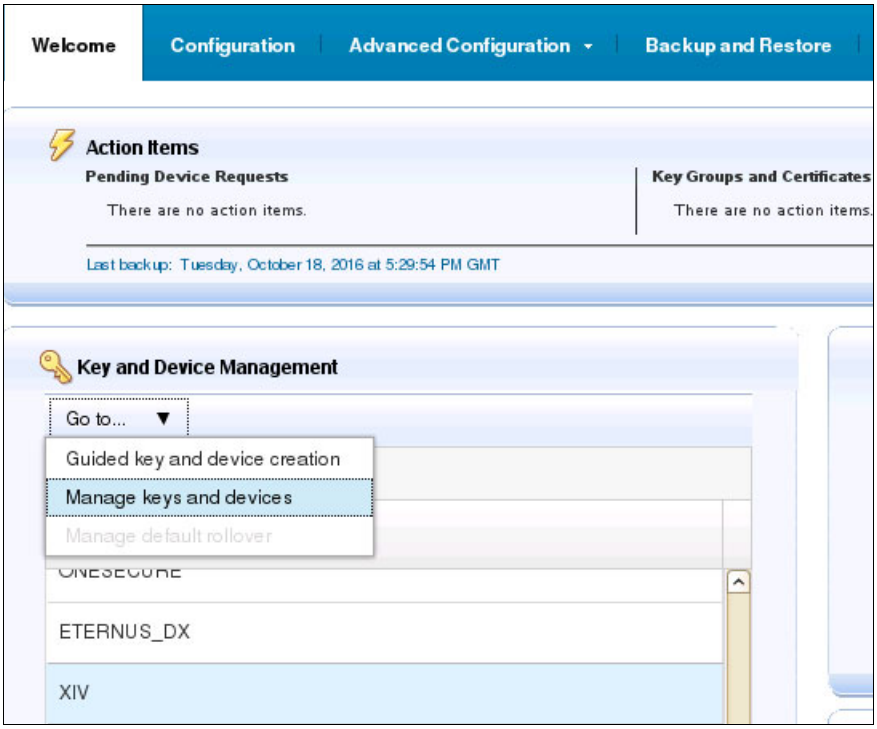

Figure 4-19 Manage keys and devices window

7. Select **Machine affinity** and **Hold new device requests pending my approval** as shown in Figure 4-20. This action ensures that when you generate the recovery key or add the key server in the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, that it shows as **Pending** in the Welcome window.

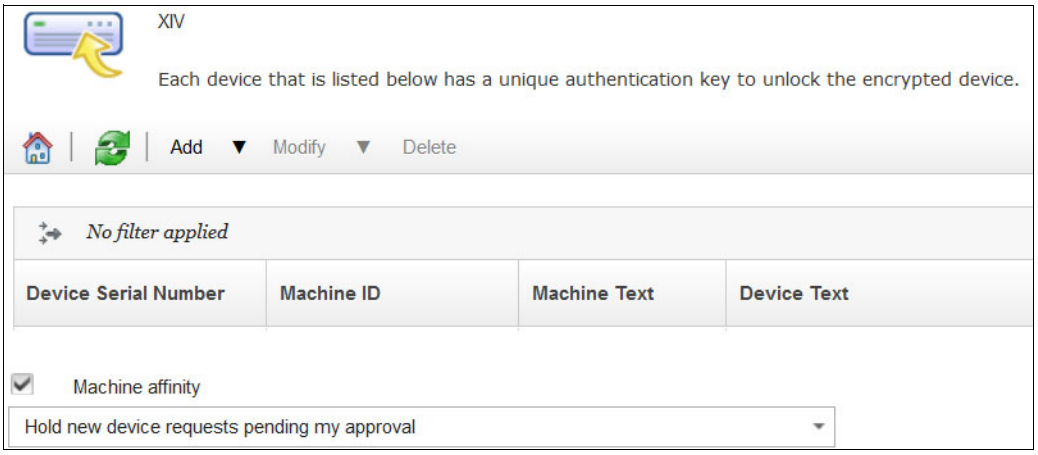

Figure 4-20 Machine affinity and Hold new device requests pending approval

8. With the these options set, import the newly created SSL certificate as “trusted” into the IBM Security Key Lifecycle Manager web GUI. Click **Advanced Configuration** → **Client Certificates** as shown in Figure 4-21.

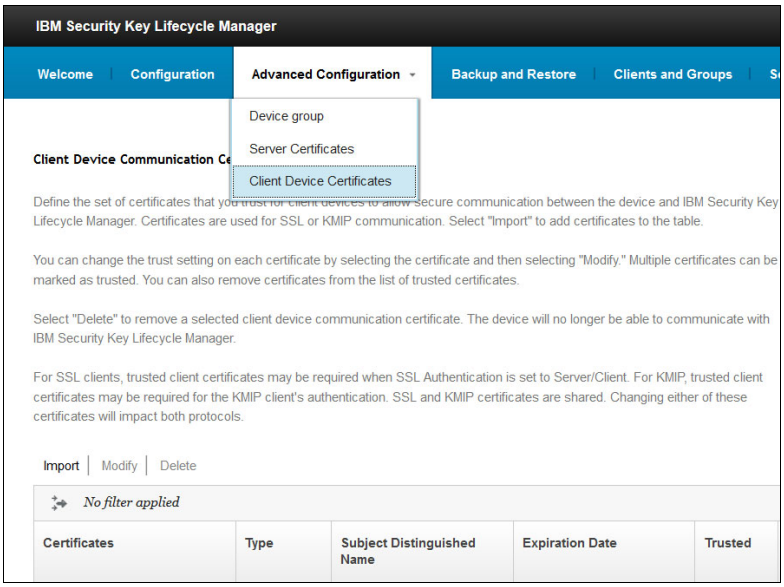

Figure 4-21 Client Device Communication Certificates display

9. Click **Import** under the SSL/KMIP Certificate for Clients section, as shown in Figure 4-22.

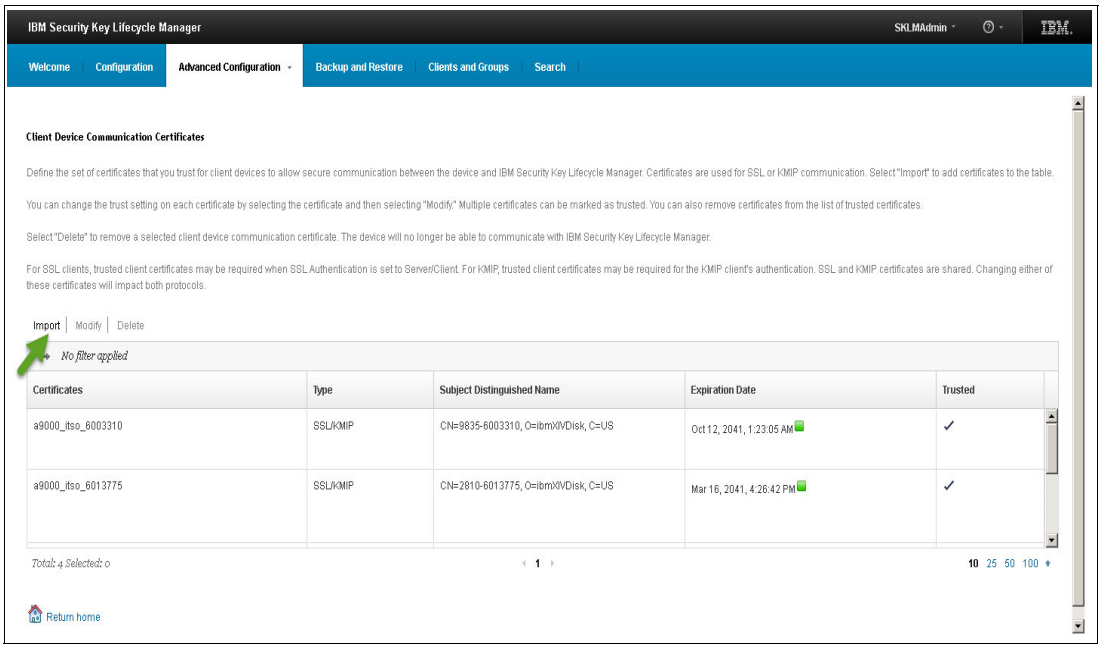

Figure 4-22 External Key Server, SKLM Import Certificate

10. Browse to find the certificate as shown in Figure 4-23.

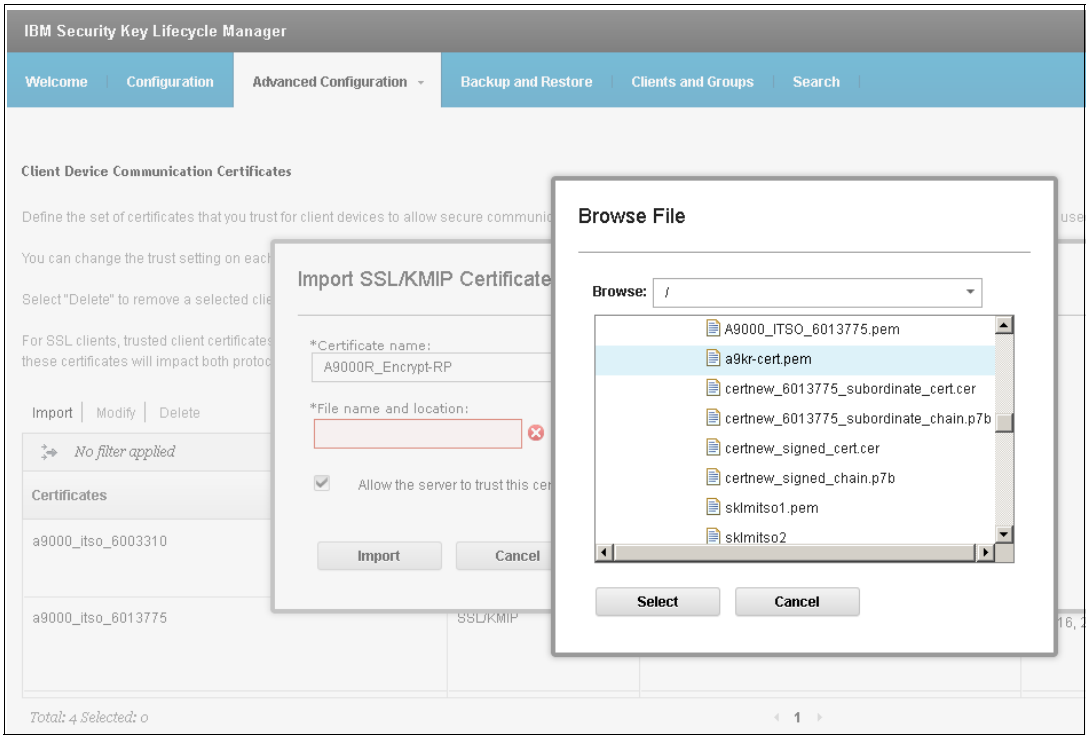

Figure 4-23 SKLM External Certificate Import

11. Click **Import** as shown in Figure 4-24.

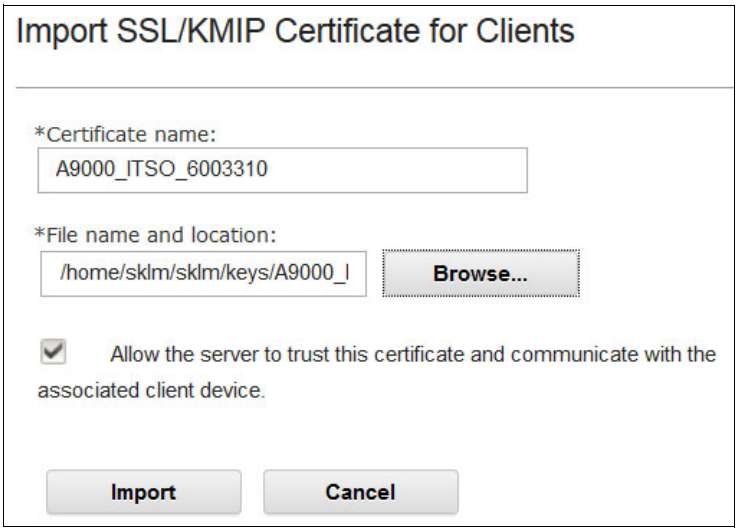

Figure 4-24 Import SSL/KMIP Certificate for Clients

The certificate shows as trusted with a type of **SSL/KMIP** and its name and expiration date are displayed, as shown in Figure 4-25.

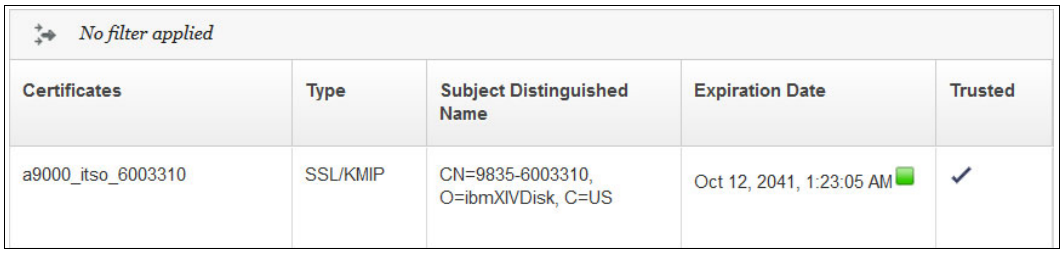

Figure 4-25 Show the trusted device-specific certificate

12. To define the key server in the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, you must create and export the key server certificate on the IBM Security Key Lifecycle Manager. Then, add it to the IBM FlashSystem A9000 or IBM FlashSystem A9000R system afterward.

    Create an IBM Security Key Lifecycle Manager self-signed certificate in the IBM Security Key Lifecycle Manager GUI (add the SSL/KMIP certificate for key serving).

If this is a new key server installation, you must have an Action Item in the IBM Security Key Lifecycle Manager Welcome window to create the server certificate, as shown in Figure 4-26.

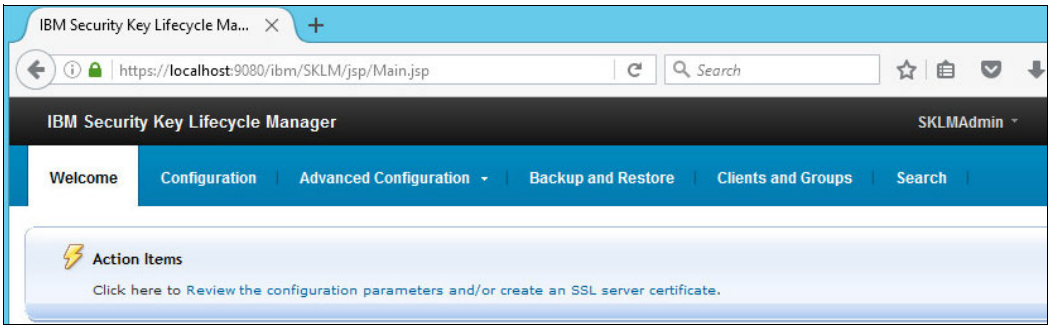

Figure 4-26 IBM Security Key Lifecycle Manager Welcome with initial configuration Action Item

If the Action Item in the Welcome window is no longer available, the server certificate can be created by completing the following steps:

a. In the window that is shown in Figure 4-26, under Advanced Configuration, click **Create self-signed certification**.

b. When the window that is shown in Figure 4-27 opens, create the certificate that is used to encrypt data for secure communication over SSL.

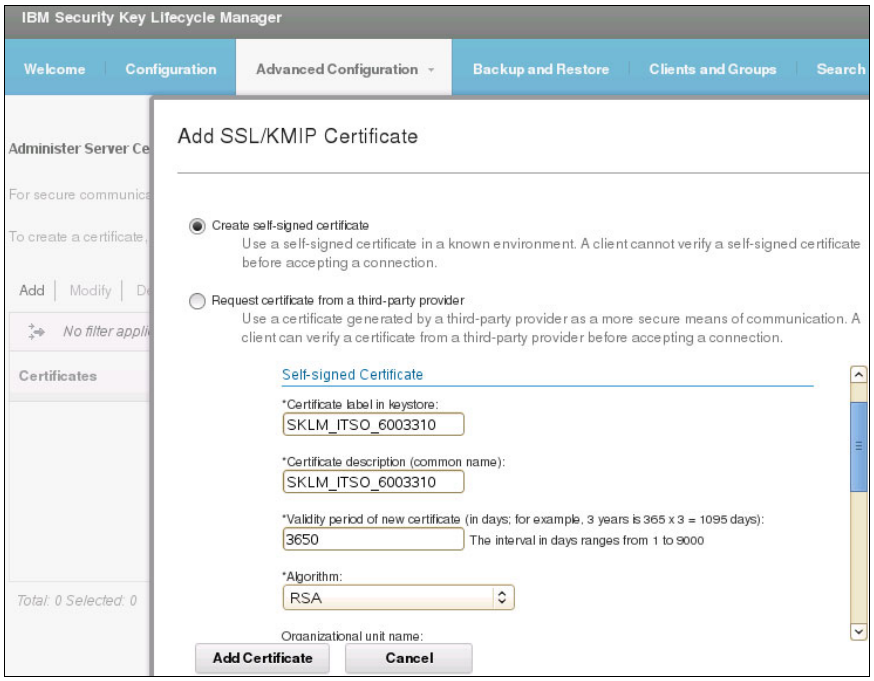

Figure 4-27 Add SSL/KMIP Server Certificate

> **Tip:** Do not confuse this certificate with the *device* certificate that is associated with the IBM FlashSystem A9000 or IBM FlashSystem A9000R system.

c. Select **Create self-signed certificate**. Third-party signed certificates are also supported.

Caution: Although using an existing certificate from the keystore is possible, using the same certificate to encrypt disk data and protect the communication protocol is *not* recommended.

d. Enter a descriptive label and a certificate expiration validity in days in accordance with your security guidelines. You can also enter certificate parameters.

e. Click **Add Certificate**.

As indicated in the Warning notice that is shown in Figure 4-28, the SSL/KMIP certificate is updated. For this change to take effect, you must restart the server. Stopping and starting the key server is described in 5.2, “Starting and stopping an external IBM Security Key Lifecycle Manager server” on page 121. Also, create a backup to ensure that you can restore this data.

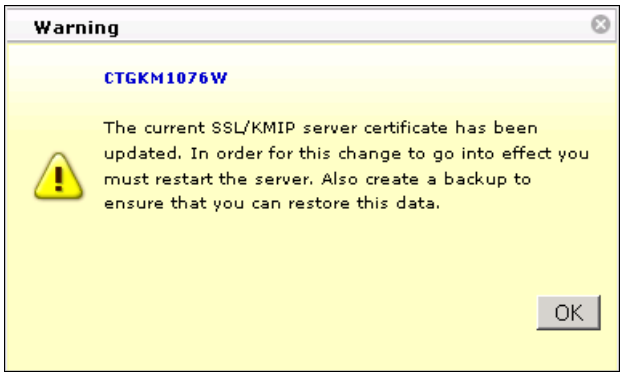

Figure 4-28 Reminder to restart the server

13. In the left pane of the IBM Security Key Lifecycle Manager GUI, click **Welcome** to return to the Welcome window, as shown in Figure 4-29.

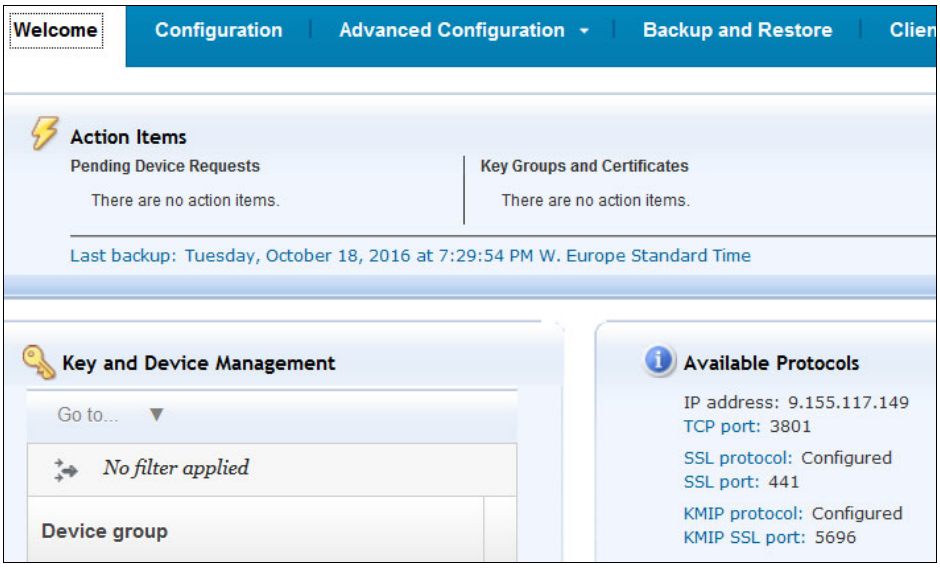

Figure 4-29 Welcome window

You have now created the IBM Security Key Lifecycle Manager master keystore and the SSL certificate. As a result, the Available protocols section now has both *SSL protocols* and *KMIP protocols* configured.

After it is created, you must export the certificate. In versions earlier than IBM Security Key Lifecycle Manager V2.6, you can do this export only by running **`wsadmin`**.

14. Exporting IBM Security Key Lifecycle Manager server certificates can be done in the IBM Security Key Lifecycle Manager GUI starting with Version 2.6 by completing the following steps:

    a. Log in to the IBM Security Key Lifecycle Manager server after if restarts and click **Advanced Configuration** → **Server Certificates**, as shown in Figure 4-30.

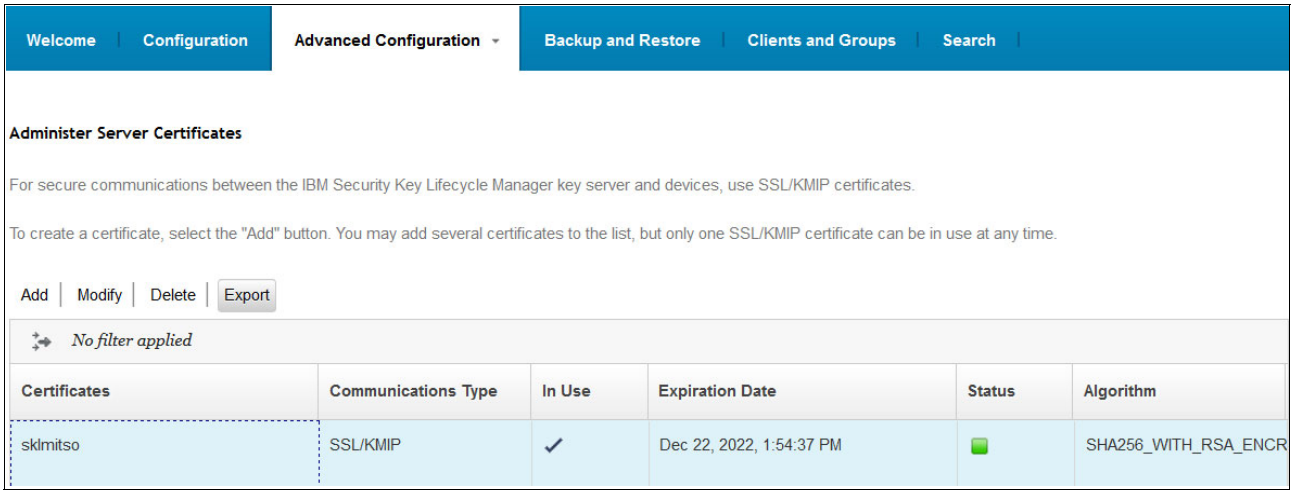

Figure 4-30 Export the server certificate

    b. Select the server certificate file name that you want, including the `.pem` file ending, and browse to the file location where you want it to be saved. Choose certificate type **base64** because the DER format is not supported in an A9000 system, as shown in Figure 4-31.

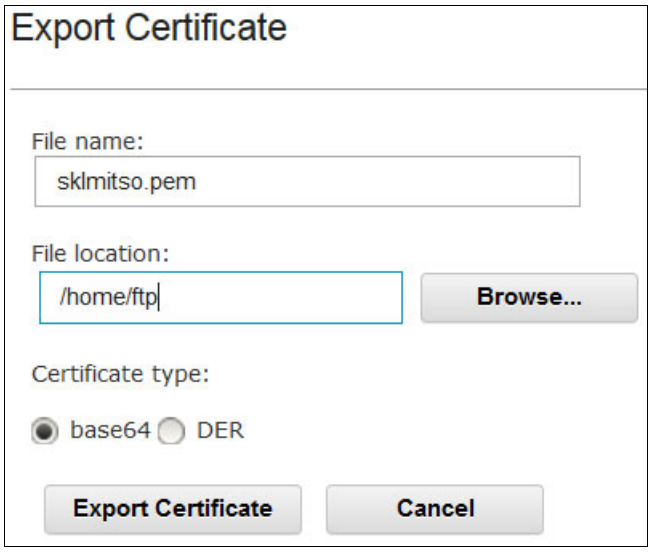

Figure 4-31 Export certificate type base64

15. For IBM Security Key Lifecycle Manager Version 2.5 and earlier, run **wsadmin** to export the server certificate:
    - For a Microsoft Windows operating system, open a DOS prompt with Administrator privileges. Run the following **wsadmin** command:

      ```
      cd <WAS_HOME> wsadmin -username sklmadmin -password <sklmadmin password>
      -lang jython
      ```

      <WAS_HOME> is, for example, C:\Program Files (x86)\IBM\WebSphere\AppServer.
    - For a Linux operating system, open a UNIX terminal session, and run the following command:

      ```
      cd <WAS_HOME>/bin>rm -f ./tmp/cert.der
      ./wsadmin.sh -username SKLMAdmin -password <sklmadmin password> -lang jython
      ```

      <*WAS_HOME*> is, for example, /opt/IBM/WebSphere/AppServer/bin/.
16. To view all existing certificates, run the **print AdminTask.tklmCertList()** command that is shown in Figure 4-32.

```
wsadmin>print AdminTask.tklmCertList('[-alias a9000_itso_6003310]')
CTGKM0001I Command succeeded.

uuid = CERTIFICATE-23882e94-73b3-4282-b253-185569a76a4d
alias = a9000_itso_6003310
Keystore name = defaultKeyStore
key state = ACTIVE
issuer name = O=ibmXIVDisk,C=US
subject name = CN=9835-6003310,O=ibmXIVDisk,C=US
creation date = 10/18/16 6:22:45 PM GMT+00:00
expiration date = 10/12/41 1:23:05 AM GMT+00:00
serial number = 10001000
```

Figure 4-32 List all certificates command in wsadmin

17. Print the certificate that is created in step 12 on page 71 by running the **print** command:

    ```
    print AdminTask.tklmCertList('[-alias <label provided in Step 2>]')
    ```

    Figure 4-33 shows an example of the result.

```
wsadmin>print AdminTask.tklmCertList('[-alias a9000_itso_6003310]')
CTGKM0001I Command succeeded.

uuid = CERTIFICATE-23882e94-73b3-4282-b253-185569a76a4d
alias = a9000_itso_6003310
Keystore name = defaultKeyStore
key state = ACTIVE
issuer name = O=ibmXIVDisk,C=US
subject name = CN=9835-6003310,O=ibmXIVDisk,C=US
creation date = 10/18/16 6:22:45 PM GMT+00:00
expiration date = 10/12/41 1:23:05 AM GMT+00:00
serial number = 10001000
```

Figure 4-33 List a specific certificate in wsadmin

18. Take the Universally Unique Identifier (UUID) information from the output of step 16 on page 75 and use it to export the certification file. You might want to change the **-fileName** option to something other than **/tmp/cert.der** if you want to save it in a different folder.

    The specified folder and file name are relative. Therefore, if you specify **/tmp/cert.der**, it is saved in a subdirectory of your IBM Security Key Lifecycle Manager installation directory. On a Windows server, you can find it in this path:

    ```
    C:\ibm\tivoli\tiptklmV2\products\tklm\tmp\cert.der
    ```

19. Export the certification file by running the following command:

    ```
    print AdminTask.tklmCertExport('[-uuid
    CERTIFICATE-a44aba79-6bcc-47dd-94c0-23ddb5db102c -format base64 -fileName
    /tmp/cert.pem ]')
    ```

    This is a successful output response:

    ```
    CTGKM0001I Command succeeded /tmp/cert.pem
    ```

    This `.pem` file is the certificate that is passed by a parameter in the XCLI **encrypt_keyserver_define** command (as described in “Step 5: Defining the external key server on FlashSystem A9000/A9000R”).

Step 5: Defining the external key server on FlashSystem A9000/A9000R

You can define a key server on an IBM FlashSystem A9000 or IBM FlashSystem A9000R system by adding the IBM Security Key Lifecycle Manager certificate that you just exported to the key server. To do so, run the XCLI **encrypt_keyserver_define** command.

Each operating system has its own way of creating line breaks in a file. Those breaks are not always visible, and if the server certificate is specified to be added as a file name, it can lead to a `bad cert` status after the key server is added.

Important: A common installation error occurs due to the line breaks contained inside the certificate, which must be removed.

Next, we provide some examples that help identify and prevent such problems.

Notice the **^M** in the VI output that is shown in Figure 4-34.

```
-----BEGIN CERTIFICATE-----^M
MIIEhTCCA22gAwIBAgIBADANBgkqhkiG9w0BAQsFADCBjTELMAkGA1UEBhMCVVMx^M
CzAJBgNVBAgTAk1BMRIwEAYDVQQHEwlMaXR0bGV0b24xDDAKBgNVBAoTA0lCTTEU^M
MBIGA1UECxMLSUJNLVhJVi1sYWIxGDAWBgNVBAMTD3hpdi1DZXJ0aWZpY2F0ZTEf^M
MB0GCSqGSIb3DQEJARYQbGthdHpAdXMuaWJtLmNvbTAeFw0xNTExMDkwNTA3MzVa^M
Fw0yNTExMDcwNTA3MzVaMIGNMQswCQYDVQQGEwJVUzELMAkGA1UECBMCTUExEjAQ^M
...
```

Figure 4-34 Certificate line break in UNIX

These line breaks must be removed. This process can be done manually, or by using the **dos2unix** utility.

Windows uses a combination of two characters (0xD 0xA) for a newline, which must be removed by using a text editor that can show invisible characters or a hexdump utility as shown by the red characters in Figure 4-35.

```
root@nextra-1310050-module-1:/local/scratch# hexdump -C xiv-cert.crt
00000000 2d 2d 2d 2d 2d 42 45 47 49 4e 20 43 45 52 54 49 |-----BEGIN CERTI|
00000010 46 49 43 41 54 45 2d 2d 2d 2d 2d 0d 0a 4d 49 49 |FICATE-----..MII|
00000020 45 68 54 43 43 41 32 32 67 41 77 49 42 41 67 49 |EhTCCA22gAwIBAgI|
00000030 42 41 44 41 4e 42 67 6b 71 68 6b 69 47 39 77 30 |BADANBgkqhkiG9w0|
00000040 42 41 51 73 46 41 44 43 42 6a 54 45 4c 4d 41 6b |BAQsFADCBjTELMAk|
00000050 47 41 31 55 45 42 68 4d 43 56 56 4d 78 0d 0a 43 |GA1UEBhMCVVMx..C|
...
```

Figure 4-35 Certificate line break in Windows

To effectively eliminate these errors, edit the server certificate in a full featured text editor, which can show line breaks, hex, or “invisible characters” and create the whole **encrypt_keyserver_define** command as one single line, including the certificate, and add the asterisk (*) character after each line break that you remove, as shown in Figure 4-36.

```
A9000_ITSO>>encrypt_keyserver_define name=sklm1 ipv4=9.123.123.123 master=yes
certificate="-----BEGIN
CERTIFICATE-----*MIICyTCCAbGgAwIBAgIGKSiyd1FPMAOGCSqGSIb3DQEBCwUAMBQxEjAQBgNVBA
MTCXNrbG1pdHNv*MzAeFwOxNjEwMjEwOTIxMzlaFwOyMjEyMjQwOTIxMzlaMBQxEjAQBgNVBAMTCXNr
bG1pdHNvMzCC*ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAIGeknLONw39+wYw4ZTvUWYjtN
uOOeUJ1weO*UgG6pdfZR1N5xj5ijEYENiWMHFxkKO6Su87lQLXTEvL3QsQlxrAmOKMuitOQxNugguEs
LaCim1Nhawv1Da*Rtd1eDlAKeDlAfERReiRaZHaiVnUw8uvURzbLlofgg+/iKfnPeFLBeBqrnkIl7Fq
rdVNOTObbFxPxoV6*DfM49pjG8lM5kAR93RO5UysSlz6NOw9srK6eGCfNJGWqzpsHYK8gBmfxDHco/j
/hD4+1TjLbnxna*w8gdQYTFRDkWR4oteFKDBlLM7szIHLZ3VS2CjkYW/VZhXGcqzzALy3ZP5uvWFFKW
T5THO3I8OtJe*xpkCAwEAAaMhMB8wHQYDVROOBBYEFDUo+63Y6PQCGt9qM8P6Oj+cZfC7MAOGCSqGSI
b3DQEBCwUA*A4IBAQBwx7WrZGN9E77LcFQmjkWty9Oy44cPHuf5B7dO64x4q3pZOqFAuY1OBpAExwOg
hDgW6IeY*nkaWWGfIfTie/NA+U/sP2toVVtR/xuNQSp2WKnxMnd8ddNosHo2q9Xprx4d2CJS1H7oUj+
Maf7OL*napObaHKyaa3veLHOfxyN/HWFQTfOKfOqYwqlCypx3nLw3XF+a3PKAAZA1hSnMfdNG59RJY+
xoa4*fzrmAleVx6iUqZV3jy3Eg2mf3Pam/qPlPbelmzl4SdJBc+XMfJ2dquidQKedVYymSVy977NF4T
ws*erD6HgQHSkfR3FEM+b7EfOUPFIBrys8rKtLRbWvovebq*-----END CERTIFICATE-----"
```

Figure 4-36 The encrypt_keyserver_define command that uses the server certificate

You can verify that the key server was added successfully by running the **encrypt_keyserver_list** command that is shown in Figure 4-37.

```
A9000_ITSO>>encrypt_keyserver_list
Module Name App/Key Status Last_time_checked Master Port Address Keyserver Type
1      sklm1 NOKEY         2016/10/26 13:48:45 yes    5696 9.123.123.123    TKLM
2      sklm1 NOKEY         2016/10/26 13:48:45 yes    5696 9.123.123.123    TKLM
3      sklm1 NOKEY         2016/10/26 13:48:45 yes    5696 9.123.123.123    TKLM
```

Figure 4-37 The encrypt_keyserver_list command

The NOKEY status shows that the IBM FlashSystem A9000 or IBM FlashSystem A9000R system has not been accepted on the key server. You will see a Pending Devices link in the Welcome window of the external SKLM key server GUI, as shown in Figure 4-38.

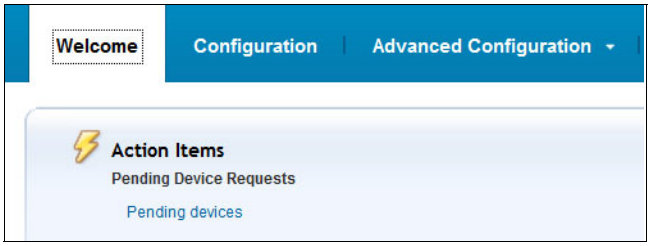

Figure 4-38 Pending Devices in the Welcome window

Complete the following steps:

1. Click the **Pending Devices** action item and the IBM FlashSystem A9000 or IBM FlashSystem A9000R system that you added is displayed, as shown in Figure 4-39.

Figure 4-39 Accept the pending IBM FlashSystem A9000 or IBM FlashSystem A9000R system device

2. Highlight the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and click **Accept** to add it to the key server. If this device served keys in the past from that key server, a warning message, as shown in Figure 4-40, appears and advises you to take a backup before accepting the pending device. This situation can happen during migrations or if encryption was enabled and disabled before. It is a preferred practice to create a backup now to be able to revert to the current state of configuration of the key server.

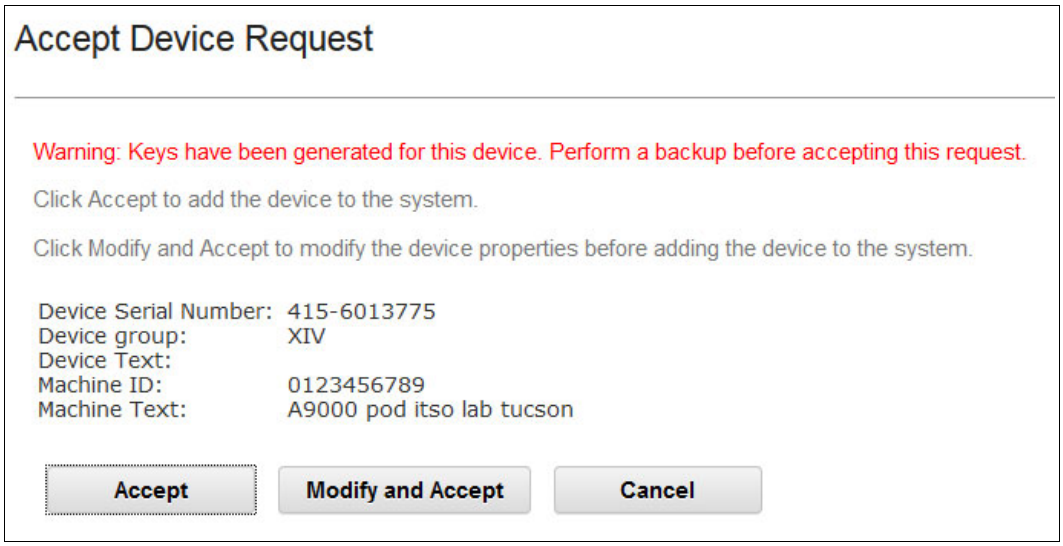

Figure 4-40 Accept Device Request

After the Device Request is accepted, it shows as a Client Device Communication Certificate in the Advanced Configuration section, as shown in Figure 4-41.

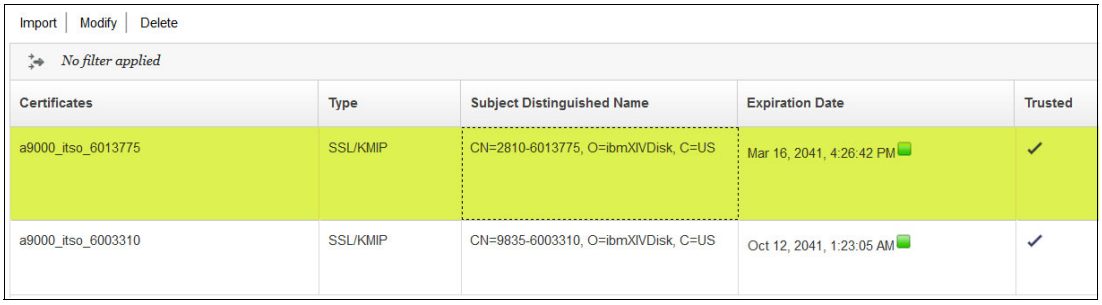

Figure 4-41 Client Device Communication Certificates

If the `NOAPP` status is shown in the XCLI, verify that the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is not being blocked from communications to the external key server.

Important: As a next step, and *before* encryption is activated, we strongly advise that you define recovery keys. For more information, see 4.3, “Recovery key use and maintenance” on page 97; then, see 4.4, “Activating and deactivating encryption” on page 104.

4.2.5 Setting up SafeNet KeySecure encryption

In this section, we describe the slightly modified steps to configure and implement the IBM FlashSystem A9000 or IBM FlashSystem A9000R system with Gemalto's SafeNet KeySecure KMIP key server. This section also applies when converting from an internal key server to an external one.

Step 1: Verifying the date and time of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and the key server

When you implement external key server encryption, you must have a matching date and time to avoid validity issues, especially when the key server and the system to be encrypted are in different time zones. A mismatch might lead to certificates becoming valid at a future time in the other time zone.

Complete the following steps:

1. Log in to the IBM FlashSystem A9000 or IBM FlashSystem A9000R XCLI by using an administrator role-based user and run **time_list**, as shown in Figure 4-42.

```
A9000_ITSO>>time_list
Time       Date         Time Zone   Daylight Saving Time
16:07:38   2016-10-17   UTC         no
```

Figure 4-42 The time_list command

2. You can match times between the external key server and the IBM FlashSystem A9000 or IBM FlashSystem A9000R system by adjusting the time zone or setting the time manually. If you choose to adjust the time zone, make sure that you use a valid time zone. Otherwise, your IBM FlashSystem A9000 or IBM FlashSystem A9000R system might disappear from the HSM UI. If this issue occurs, deleting the IBM FlashSystem A9000 or IBM FlashSystem A9000R system entry in the HSM UI and adding it again brings it back. Valid time zones can be shown by running the **timezone_list** command, as shown in Figure 4-43.

```
A9000_ITSO>>timezone_list
Timezone
Universal
Egypt
Africa/Nairobi
Africa/Mbabane
Africa/Niamey
Africa/Dakar

Africa/Luanda
...
Europe/Ulyanovsk
NZ-CHAT
EET
GB
GMT
```

Figure 4-43 Excerpt of the timezone_list command

3. Run the **timezone_set** command (see Figure 4-44) to adjust the time zone of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system to match one of the valid time zones that are produced by the command that is run as shown in Figure 4-44.

```
A9000_ITSO>>timezone_set timezone=UTC
Command executed successfully.
```

Figure 4-44 The timezone_set command

4. If the date and time do not match between the key server and the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, you can set it on the A9000R by running the **time_set** command, as shown in Figure 4-45.

```
A9000_ITSO>>time_set time=2016-10-18.12:02:00
Command executed successfully.
```

Figure 4-45 The time_set command

5. Check whether your IBM FlashSystem A9000 or IBM FlashSystem A9000R system shows the encryption support in the output of the **state_list** command, as shown in Figure 4-46.

```
A9000_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Supported
data_reduction_state     Online
```

Figure 4-46 Show encryption support by running the state_list command

Contact IBM Support if the encryption category does not state a supported value.

Step 2: Creating the A9000 Security Admin role users

Preferred practices for managing and configuring encryption with an IBM FlashSystem A9000 or IBM FlashSystem A9000R system requires a minimum of two secadmin role users. To create them, use the Hyper-Scale Manager User Interface (HSM GUI) or the XCLI.

Complete the following steps by using the Hyper-Scale Manager GUI:

1. Open a web browser and specify the IP address and IP port of the HSM by using the following web address format:

   `https://<HSM_IP>:<HSM_PORT>/`

   For example:

   `https://9.123.123.123:8443/`

2. Enter your IBM FlashSystem A9000 or IBM FlashSystem A9000R system administrator role user ID and password, and then, click **Login**.

3. The HSM UI Dashboard indicates that you are successfully logged in to the IBM Hyper-Scale Manager. If you manage multiple IBM FlashSystem A9000 or IBM FlashSystem A9000R systems, choose the correct one from the list in the Dashboard, as shown in Figure 4-47.

Figure 4-47 Choosing IBM FlashSystem A9000 or IBM FlashSystem A9000R system

4. On the Dashboard's left side, click the **Access Views** icon and then click **Users** to open the **Users** menu, as shown in Figure 4-48.

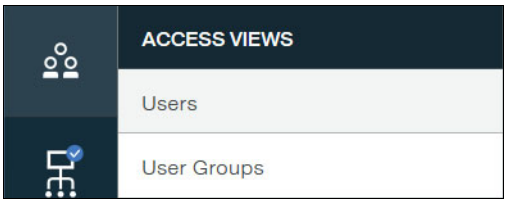

Figure 4-48 Users access view

5. In the Users access view, click the **+** icon to create an object, as shown in Figure 4-49.

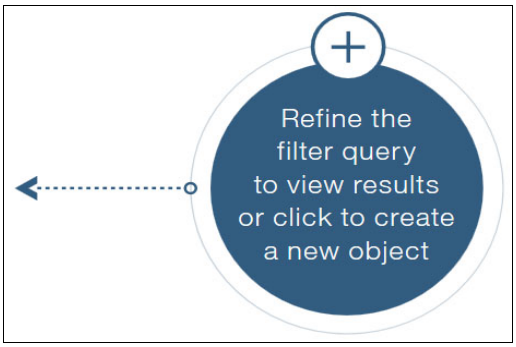

Figure 4-49 Create a user object

Enter the name for your security admin user, select a system and password, change the category to **Security Administrator**, and click **Create**. **Phone** and **Email** are optional fields. You cannot choose a domain because Security Admin is associated to the Global Space only (see Figure 4-50 on page 83).

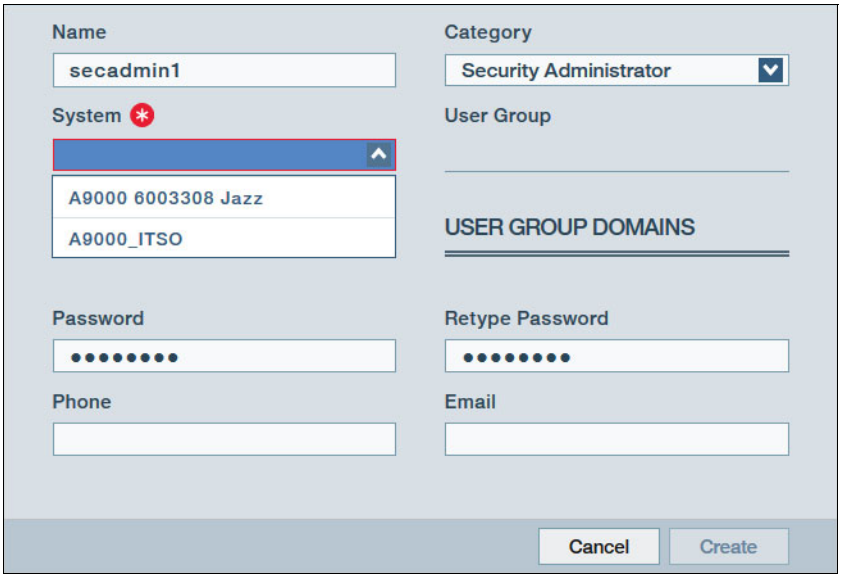

Figure 4-50 Create a secadmin user

6. Repeat these steps for other users. The minimum secadmin users that are required is two.

You can also use the XCLI to add the secadmin role users by running the **user_define** command, as shown in Figure 4-51.

```
user_define user=secadmin1 category=securityadmin password=passw0rd
password_verify=passw0rd

user_define user=secadmin2 category=securityadmin password=passw0rd
password_verify=passw0rd
```

Figure 4-51 Define secadmin users in XCLI

Run the **user_list** command to verify that the creation of the secadmin users is successful, as shown in Figure 4-52.

```
A9000_ITSO>>user_list
Name                Category     Group     Active
xiv_development     xiv_development        yes
xiv_maintenance     xiv_maintenance        yes
admin               storageadmin           yes
technician          technician             yes
xiv_hostprofiler    xiv_hostprofiler       yes
manager_server_user storageadmin           yes
secadmin1           securityadmin          yes
secadmin2           securityadmin          yes
```

Figure 4-52 The user_list command example

You can change certain user's parameters by running the **user_update** command, as shown in Figure 4-53.

```
A9000_ITSO>>user_update
user=              password=          password_verify=   email_address=
number=            area_code=         exclusive=
```

Figure 4-53 The user_update command example

Tip: After a Security Administrator role base user owns a recovery key, the user name cannot be changed anymore, as shown in Figure 4-54.

```
A9000_ITSO>>user_rename user=secadmin1 new_name=secadmin1b
Error:   USER_OWNS_RECOVERY_KEY
Details: User owns recovery key and therefore cannot be deleted or renamed
```

Figure 4-54 User owns the recovery key

4.2.6 Installing and configuring SafeNet KeySecure key server

This section describes the installation and configuration of the SafeNet KeySecure key server for operation with FlashSystem A9000 or A9000R.

Installing and configuring licenses on the SafeNet Application

Per Gemalto's best practices, install and ensure logical licensing are applied. For more information, see this web page:

https://safenet.gemalto.com/data-encryption/enterprise-key-management/key-secure

Note: Skip this section if you have an active KeySecure server.

Install the license file provided by Gemalto. From the Gemalto SafeNet KeySecure Management Console, select **Device** → **System Information & Upgrade** and upload the file, as shown in Figure 4-55 on page 85.

Make sure that the license status is Active and the expiration date conforms to the license.

If your production environment includes two or more key managers, make sure that all servers are active and the cluster is configured, as described in the Gemalto user guide.

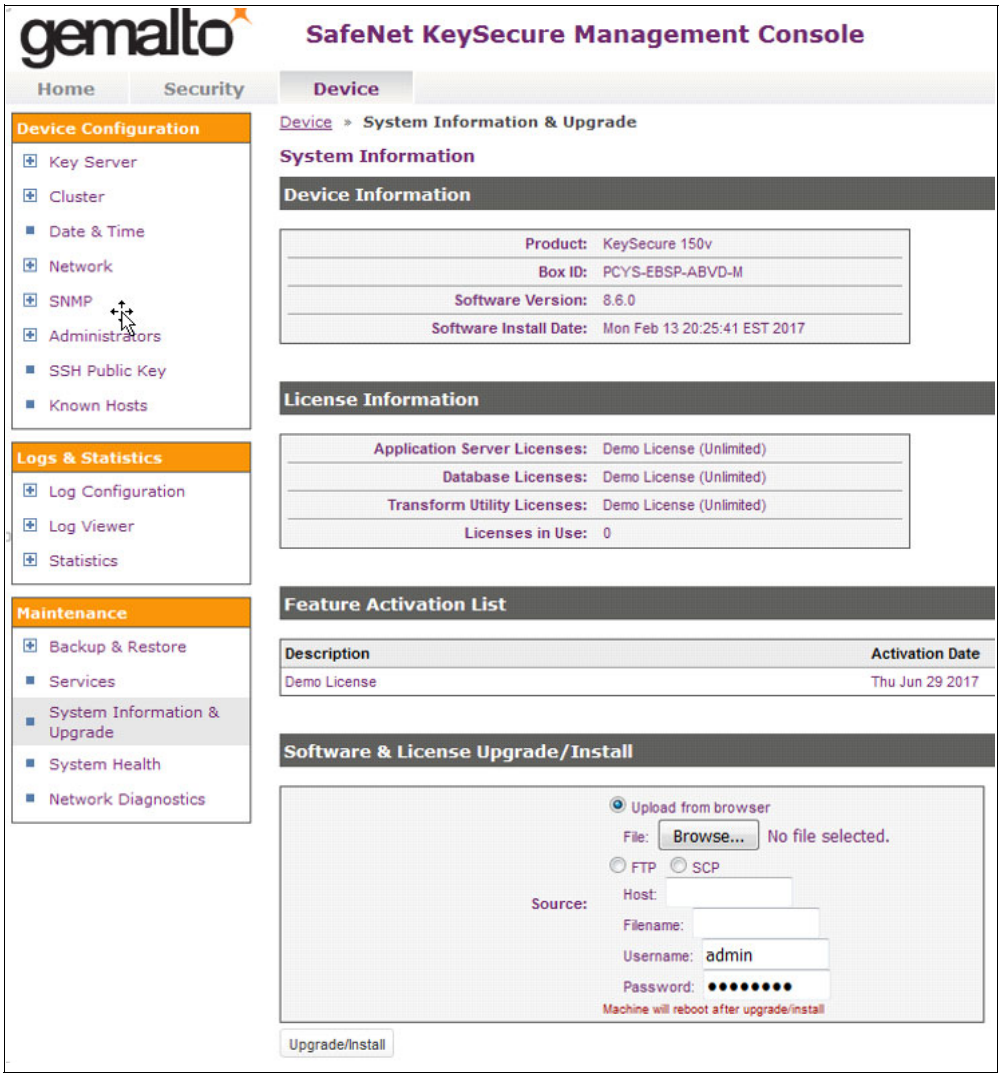

Figure 4-55 Gemalto Safenet KeySecure Licensing

Configuring the Safenet key server

Configuring the Safenet key server entails the following overall steps:

- Configuring High Security
- Creating a local Certificate Authority (CA)
- Creating an SSL certificate for the KMIP Key Server and signing with the local CA
- Adding a KMIP key server
- Creating an AES encryption key
- Making a copy of the CA certificate to be installed on FlashSystem A9000/R

These steps are described next.

Configuring High Security

In the Gemalto SafeNet KeySecure Management Console, select **Security** → **High Security** and verify that **Disable Creation and Use of Global Keys** is not selected, as shown in Figure 4-56.

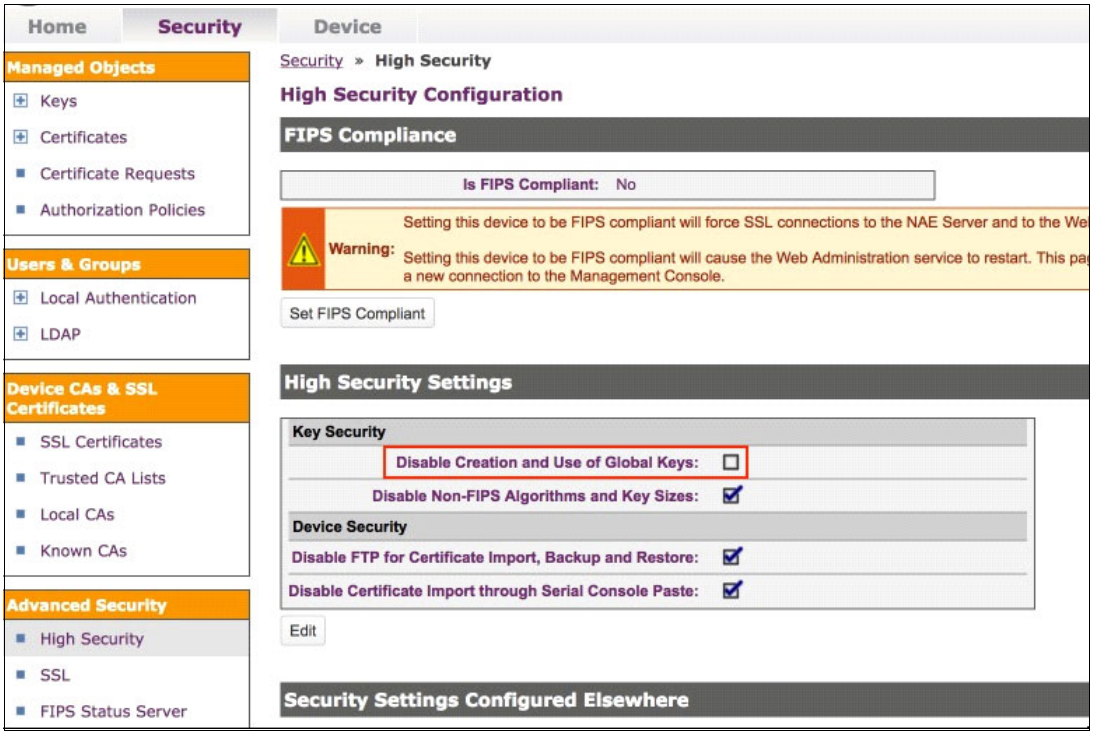

Figure 4-56 Key Secure items to UNcheck for proper operation

Creating a local Certificate Authority

Note: Skip this step if your environment includes a local or external Certificate Authority (CA).

Complete the following steps (see Figure 4-57 on page 87) to create a local CA:

1. In the Gemalto SafeNet KeySecure Management Console, select **Security** → **Local CAs**.
2. In the Create Local Certificate Authority pane, complete all of the blank fields.
3. Ensure that the Key Size value is 2048.
4. Ensure that **Self-signed Root CA** is selected.
5. Click **Create**.

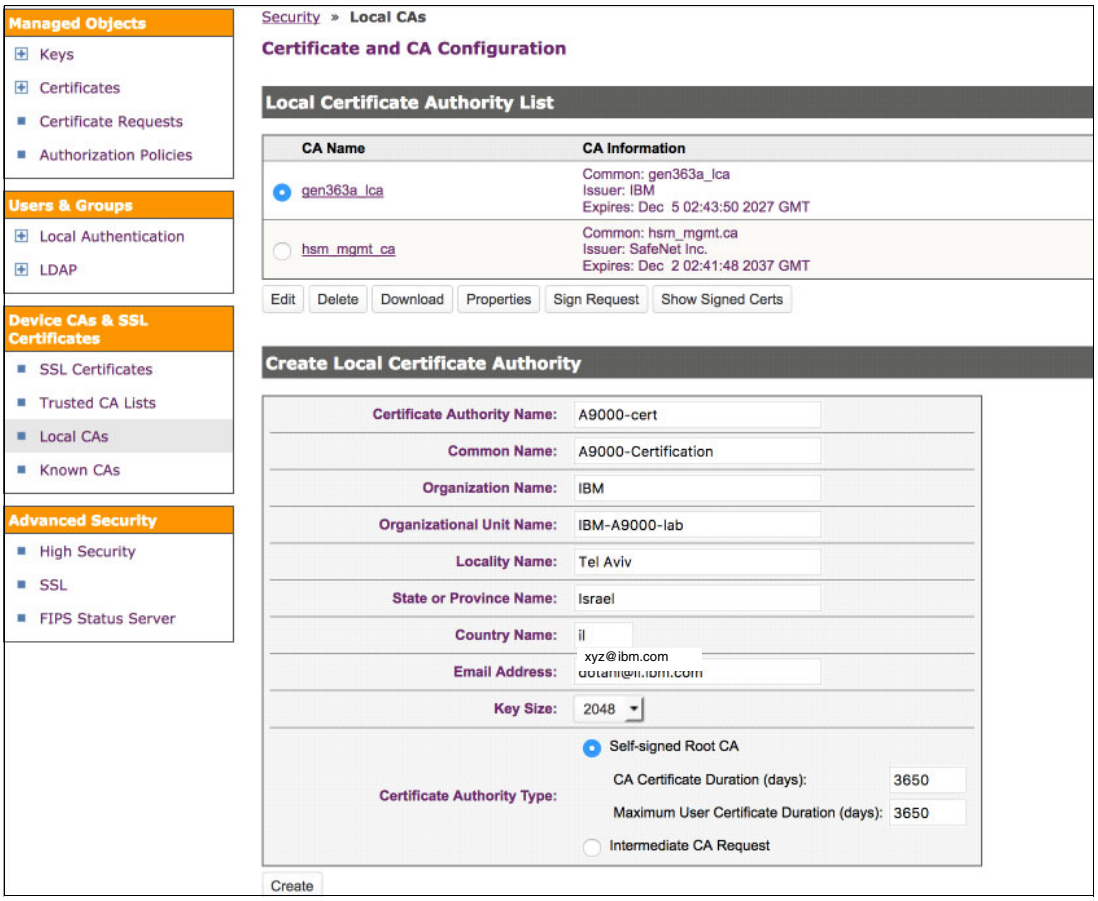

Figure 4-57 KeySecure Certificate Authority {CA} Creation

Creating an SSL certificate and signing with the local CA

An SSL certificate is required for secure communication between the key server and FlashSystem A9000/R.

Complete the following steps:

1. To create an SSL Certificate for the key server and sign with the local CA, in the Gemalto SafeNet KeySecure Management Console, select **Security** → **SSL Certificates**.

2. In the Create Certificate Request pane, complete all of the blank fields and click **Create Certificate Request**, as shown in Figure 4-58.

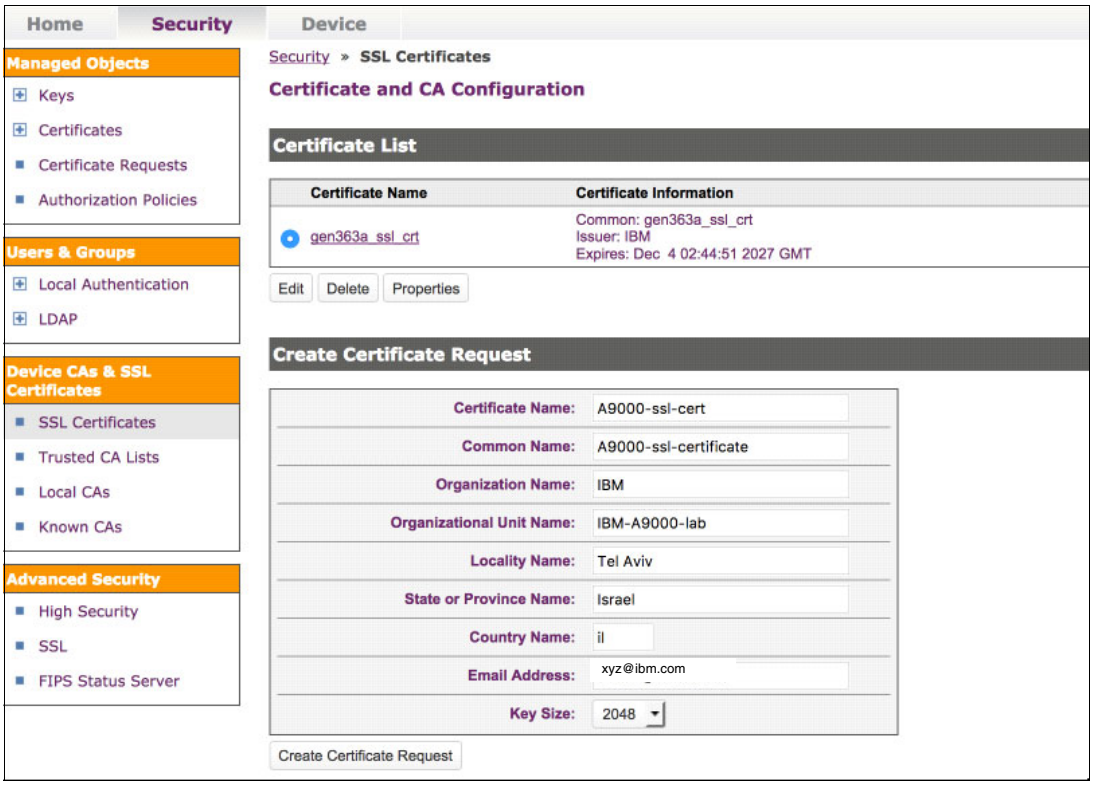

Figure 4-58 KeySecure Certificate Request

3. Review and ensure that the certificate is added, as shown in Figure 4-59.

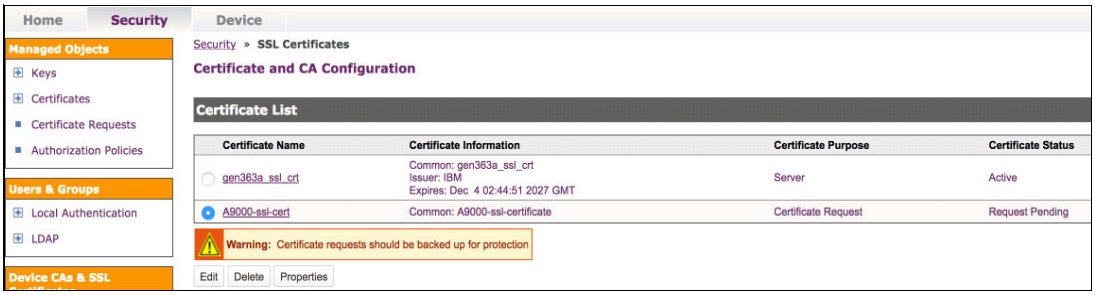

Figure 4-59 Confirmation of Certificate being added to KeySecure

4. Now that we have the certificate, we need to have it signed. Click **Properties** and copy the entire text of the A9000 SSL certificate, including the “-----BEGIN CERTIFICATE REQUEST----- ..... “ to the end of “-----END CERTIFICATE REQUEST-----”, as shown in the red box in Figure 4-60 on page 89.

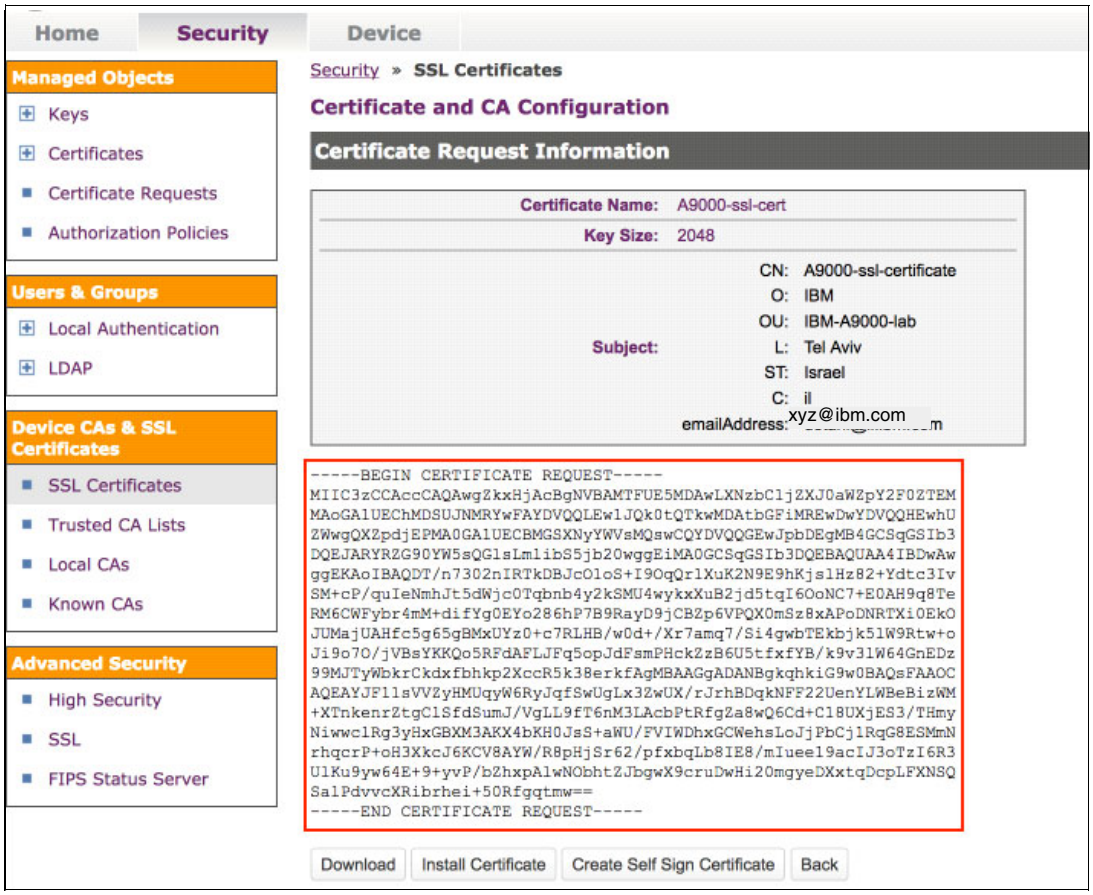

Figure 4-60 KeySecure Certificate and sign dialog box

5. Browse to **Security** → **Local CAs** and click **Sign Request**, as shown in Figure 4-61.

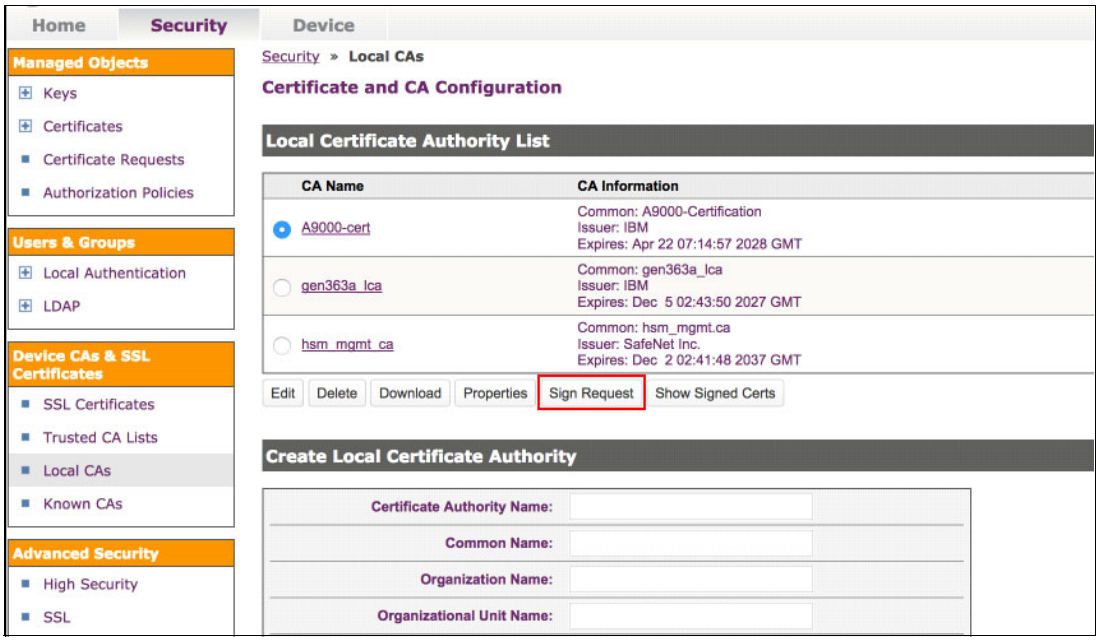

Figure 4-61 KeySecure Sign Request for CA

6. Paste the certificate text that was copied earlier into the Sign Certificate Request box, as shown in Figure 4-62.

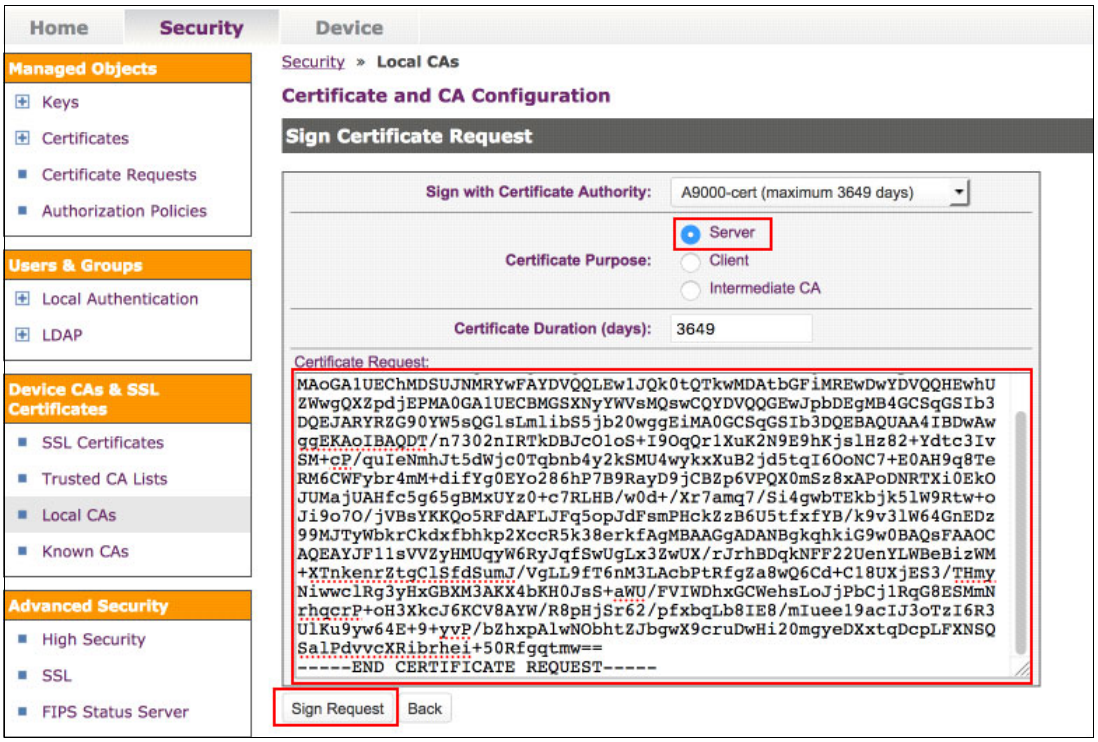

Figure 4-62 KeySecure Paste of Certificate

7. Inside the Certificate Purpose area, select **Server** and click **Sign Request**.
8. You must sign the SSL certificate with the CA certificate. From the CA Certificate Information panel that is shown in Figure 4-63 on page 91, copy the entire certificate text, including the '-----BEGIN CERTIFICATE-----' and '-----END CERTIFICATE-----'lines, as shown by the red box.

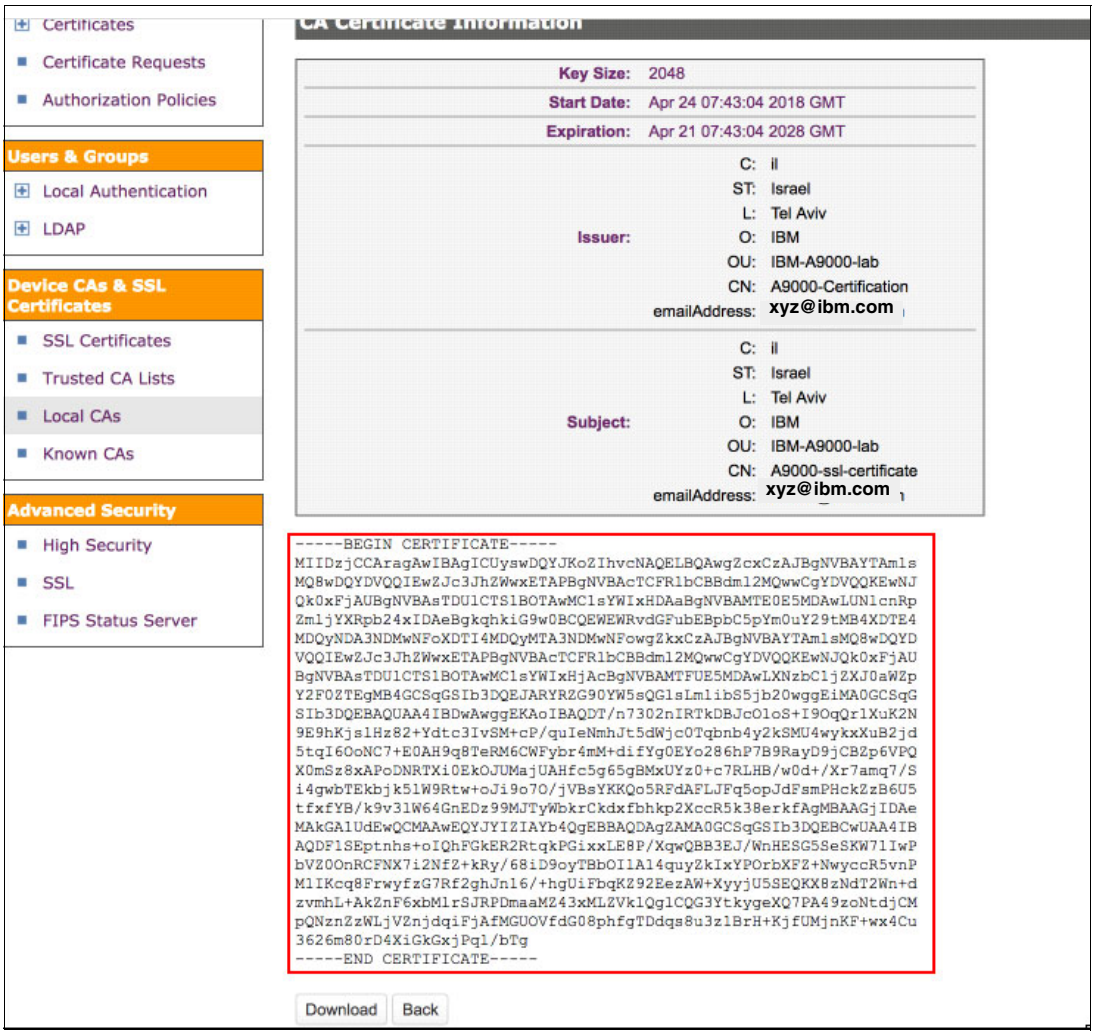

Figure 4-63 Copy of Certificate again for KeySecure

Important: As shown in Figure 4-63, the certificate starts with “--BEGIN...” and ends with “END CERTIFICATE----”. This text is critical to ensure no other characters, such as Carriage Return <CR> or Line Feeds <LF>, can be added at any point. Adding characters causes the validation process to fail.

Ensure that any copy of the certificate or similar maintains this strict format: a single line without any breaks. If the CLI is used, you can use asterisks { *} instead of any line breaks or carriage returns.

9. Go to **Security** → **SSL Certificates** and click **Properties**. Then, at the bottom of the page, click **Install Certificate**, as shown in Figure 4-64 on page 92.

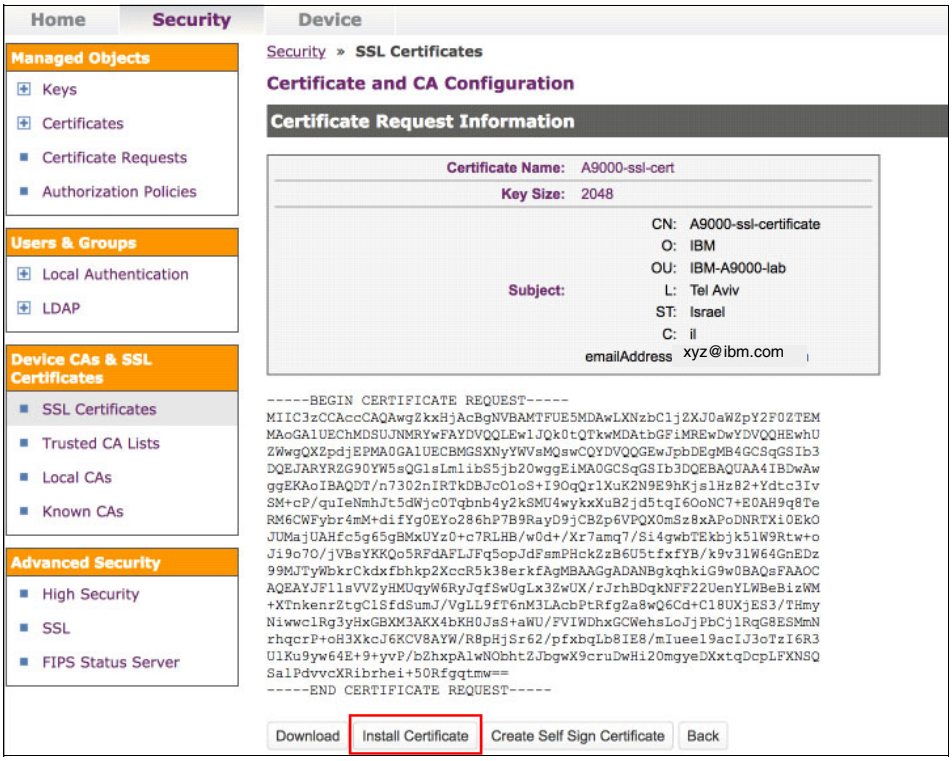

Figure 4-64 Install Certificate again for KeySecure

10. Paste the copied text and click **Save,** as shown in Figure 4-65.

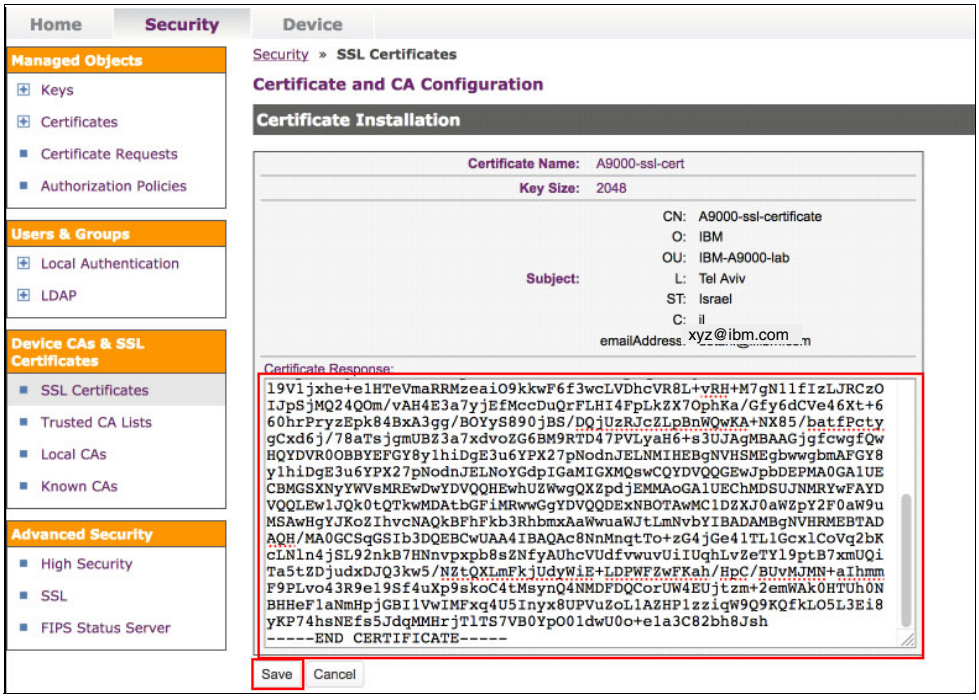

Figure 4-65 Installing the Certificate for KeySecure

11. Ensure that the Certificate Status is Active, as shown in Figure 4-66.

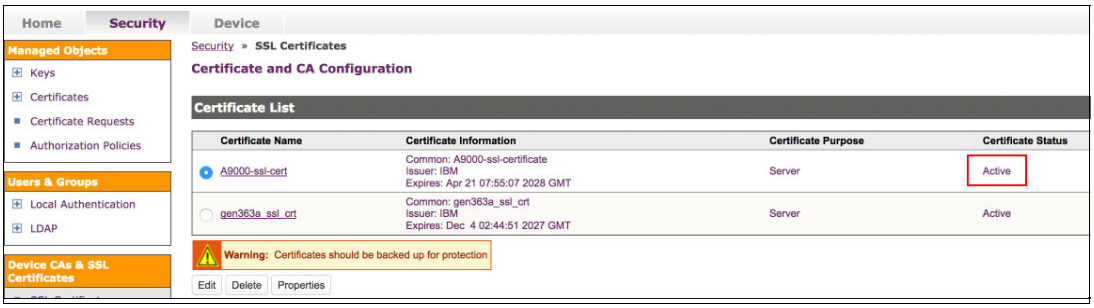

Figure 4-66 Confirmation of the active certificate for KeySecure

Adding a KMIP Key Server

Complete the following steps to add a KMIP Key Server:

1. In the Gemalto SafeNet KeySecure Management Console, select **Device** → **Key Server** and click **Add** (see Figure 4-67).

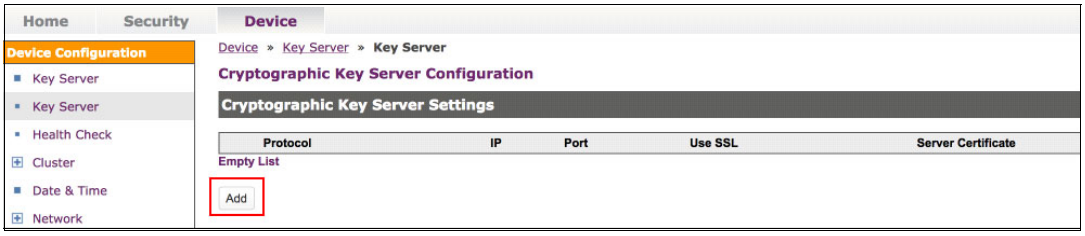

Figure 4-67 KeySecure Adding KMIP

2. Select the following settings from the drop-down menu:
   - Protocol = KMIP
   - Port = 5696
   - Use SSL
   - The Server Certificate that was defined in “Creation of an SSL certificate for the KMIP Key Server and signing with the local CA” from the drop-down box

   Then, click **Save** (see Figure 4-68).

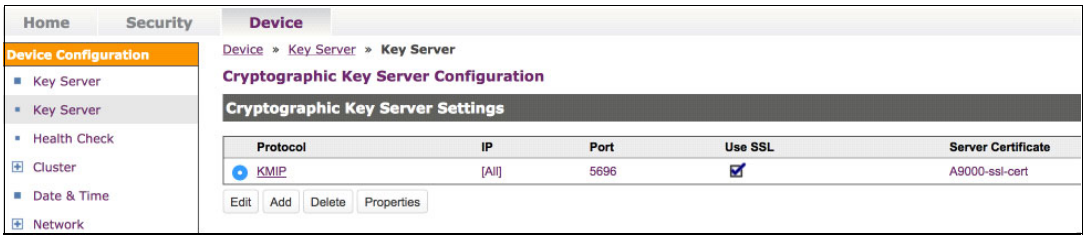

Figure 4-68 KeySecure Confirmation of KMIP configuration

Note: For security, only the KMIP server should remain.

Creating an AES encryption key

Click **Security** → **Keys**, and define the settings that are shown in Figure 4-69.

Key Name	FlashSystem A9000/R system model serial. For example: 415-6013833
Exportable	Check
Deletable	Check

Figure 4-69 KeySecure AES Key Settings

The values of the rest of the fields can be kept as the default values, as shown in Figure 4-70. Click **Create**.

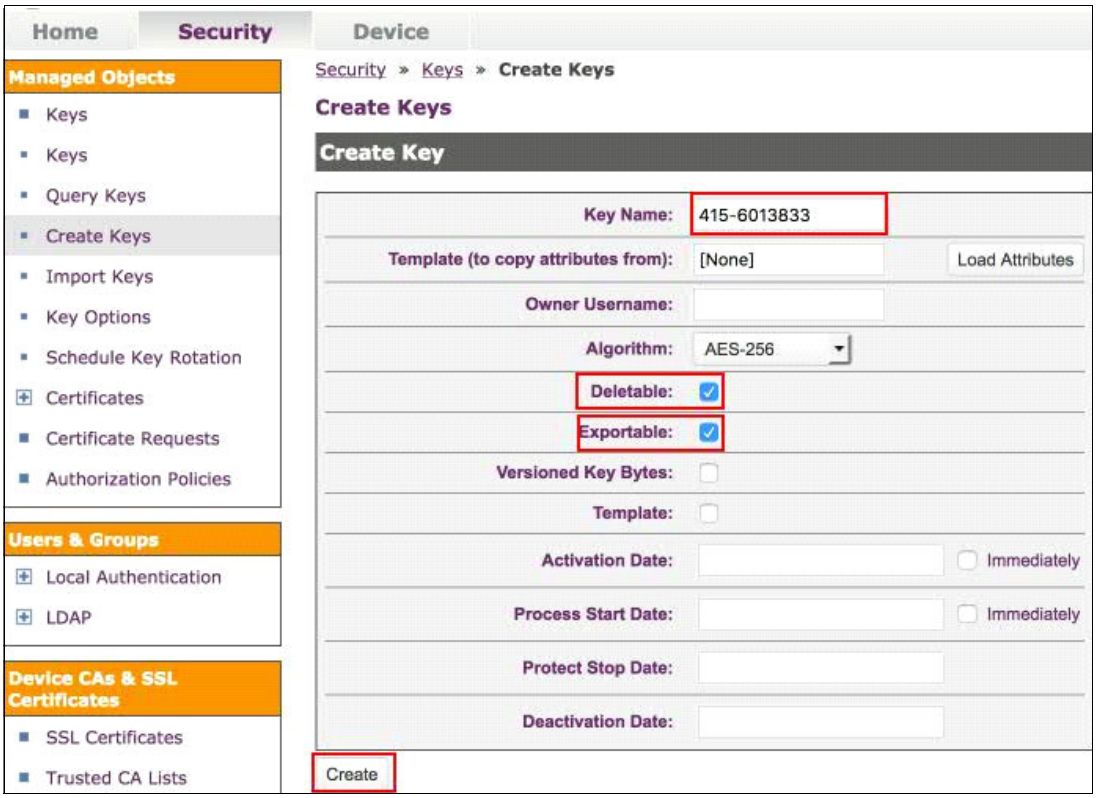

Figure 4-70 KeySecure Creating AES Key

Making a copy of the CA certificate for installation on FlashSystem A9000/R

Complete the following steps to make a copy of the CA Certificate that will be installed on FlashSystem A9000/R:

1. In the Gemalto SafeNet KeySecure Management Console, click **Security** → **Local CAs** and then, click **Properties**.
2. Copy the entire certificate to the clipboard, as shown in Figure 4-71 on page 95.

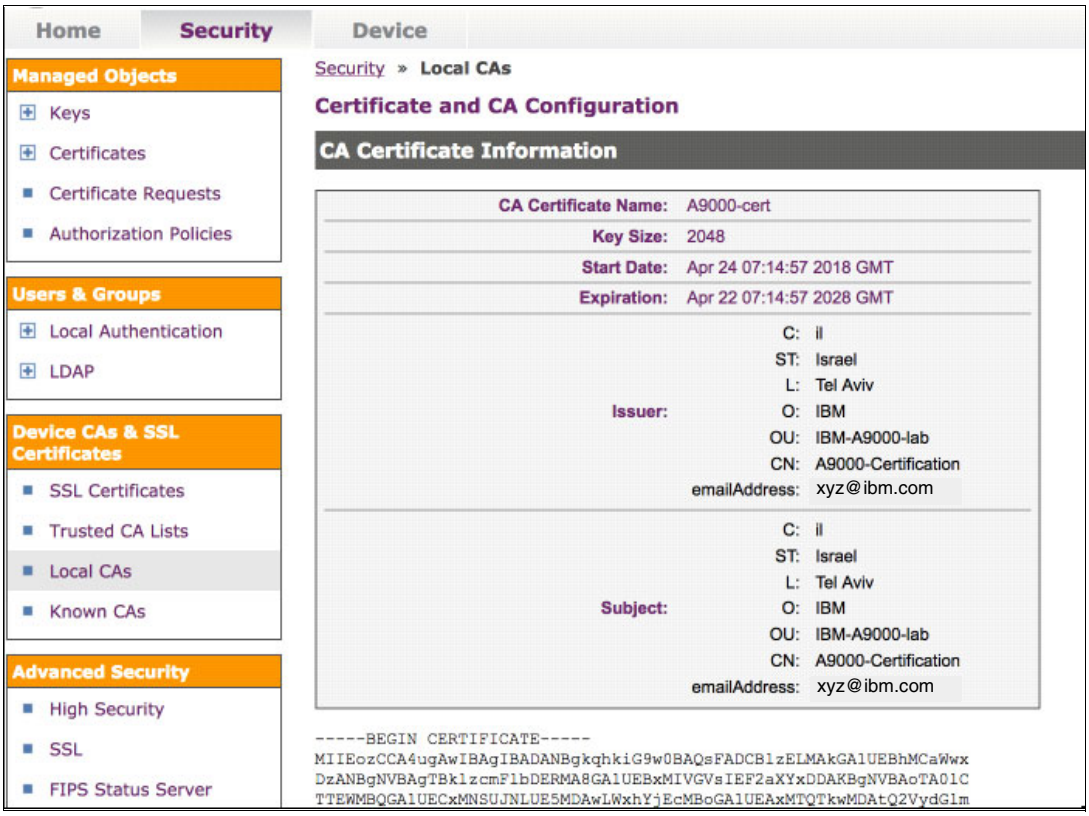

Figure 4-71 Review of the Certificate inside KeySecure

3. Copy the certificate text to the clipboard.

Important: Make sure to include the first and the last lines (-----BEGIN CERTIFICATE-----and -----END CERTIFICATE-----), and copy only the text in the certificate. Do not copy any extra white space.

4.2.7 Installing the KeySecure certificate on the A9000/R

We suggest using a Linux host for these steps to ensure that no invisible characters become embedded inside the certificate, which causes configuration errors. On the host that is used to run the CLI commands, create a text file (for example, `/tmp/test.crt`) and paste the contents of the certificate without trailing white spaces.

Use the **`encrypt_keyserver_define`** FlashSystem A9000/R CLI command to install the key server certificate on FlashSystem A9000/R, as shown in Figure 4-72.

```
A9000-Site_B>>encrypt_keyserver_define keyserver_type=KeySecure master=yes
certificate="$(tr "\n" "*" </tmp/test.crt)"
```

Figure 4-72 A9000 CLI command for KeySecure definition with certificate

The **`encrypt_keyserver_define`** CLI command was updated as of A9000 /R software version 12.3 or later; with earlier versions, you must contact IBM Support to ensure the correct parameters are updated.

You can also paste the entire certificate in the command; however, ensure that the certificate does not include line breaks or similar invisible formatting commands. You can use asterisks (*) to replace any CR or LF.

Run the **pki_list** CLI command to ensure that the default certificate is installed, as shown in Figure 4-73.

```
A9000-Site_B>>pki_list
Name    Fingerprint                      Has signed certificate   Services
A9000   bb1de87ce18795b88c09d9bd42d91eed   yes                    XCLI,CIM,IPSEC,KMIP,QUORUM
```

Figure 4-73 Verify that the certificate is installed

If updates re needed, run the **encrypt_keyserver_update** command, as shown in Figure 4-74. You can modify the default port (5696), by specifying the “port” parameter.

```
 A9000-Site_B>> encrypt_keyserver_update name=SafeNet-A9000415
certificate="$(tr "\n" "*" </tmp/test.crt)"
```

Figure 4-74 A9000 CLI Key Server Update command

Enable the status by running the command that is shown in Figure 4-75.

```
A9000-Site_B>> encrypt_enable recovery_keys=no -y
```

Figure 4-75 A9000 enable SafeNet KeySecure encryption

Check the key server, as shown in Figure 4-76.

```
A9000-Site_B>> encrypt_keyserver_list
Module  Name             App/Key Status  Last time checked    Master  Port  Address       Keyserver Type
1       SafeNet-A9000415 NOKEY           2018/04/30 12:16:01  yes     5696  9.15x.1xx.xx  KeySecure
```

Figure 4-76 A9000 CLI Key Server List command

After the encryption is successfully enabled, the flash enclosure status starts changing from `Ready` state to `Enrolled` state.

As a last verification of the flash enclosure encryption status, log in as Administrator and run the **flash_enclosure_list** command, as shown in Figure 4-77.

```
 A9000-Site_B>> flash_enclosure_list
Component ID        Status  Currently Functioning  Control Path Status  Cluster IP     Redundancy State  FW level       Has
Spare  Array Rebuild Percentage  Encryption State  Machine Model
---- --------------- ------------------ ---------------- ----------- -------------------------- ------------------ ---------------

1: Flash_Enclosure:4  OK    yes              OK              14.10.204.239  online         1.5.0.0-436.13  yes      None
Enrolled         AE1
```

Figure 4-77 A9000 CLI command to confirm encryption enrollment status

4.3 Recovery key use and maintenance

This section focuses on the definition and use of the recovery keys. These processes are applicable to the local and external key server, such as SKLM methodologies. There are only a few slight differences, which are noted.

To protect against the possibility (following a disaster, for example) that a key server becomes unusable and unrecoverable, the IBM FlashSystem A9000 or IBM FlashSystem A9000R systems enables you to create a *recovery key*, as shown in Figure 4-78. With a recovery key, Security Administrators can unlock an IBM FlashSystem A9000 or IBM FlashSystem A9000R system without the involvement of an external key server, such as IBM Security Key Lifecycle Manager server or in case something happens to the master key on the local key server.

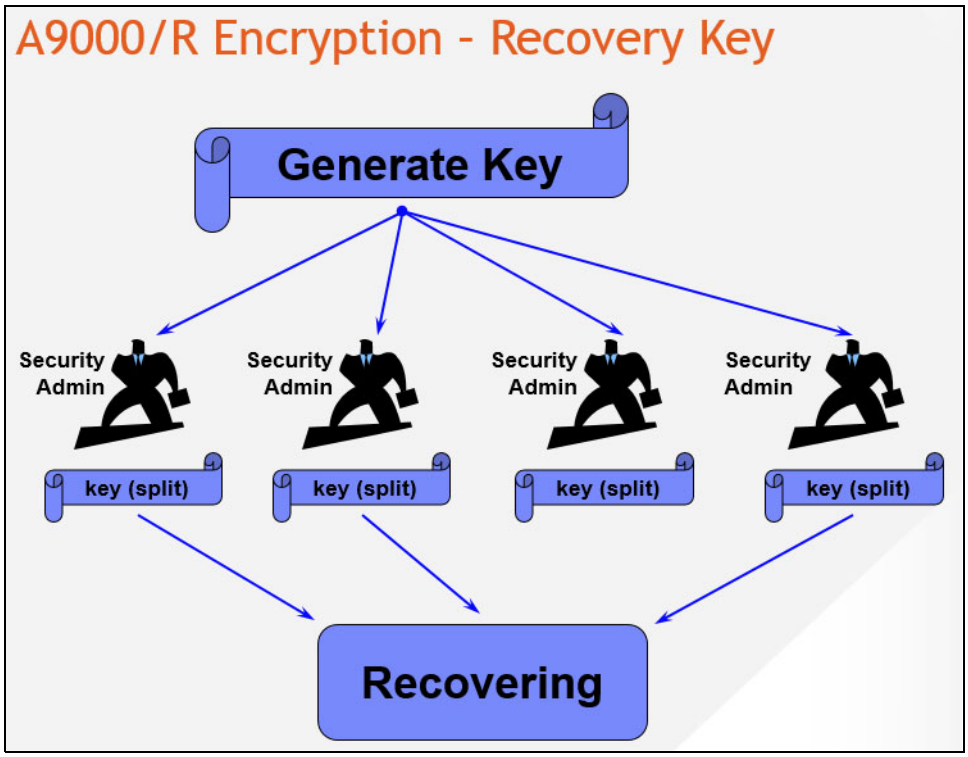

Figure 4-78 Recovery key

Encryption must be activated in the XCLI. The option to activate data-at-rest encryption without recovery keys is possible, but only through the XCLI by running the **`encrypt_enable`** command with the **`recovery_keys=no`** flag. This is *not* the preferred practice.

Normally, the recovery keys are split according to the number of defined Security Administrators (secadmins) and created separately for each Security Administrator.

The recovery key is used to unwrap the XMK, which unlocks the micro-latency modules.

Important: A recovery key can be created only if data-at-rest encryption is not yet enabled. You cannot create a recovery key when IBM FlashSystem A9000 or IBM FlashSystem A9000R encryption is already activated. However, you can issue a rekey command to change an existing recovery key.

Managing the recovery key requires at least two Security Administrators. They maintain the recovery key and keep it safe.

Client responsibility: Although an IBM FlashSystem A9000 or IBM FlashSystem A9000R system supports two group admin roles: Storage Administrator and Security Administrator. Only the Security Administrator is allowed to do encryption tasks, such as the recovery key. The client is responsible for assigning at least two *separate* individuals as Security Administrators to prevent data access by a single person.

4.3.1 Process for recovery keys

The recovery keys can be generated only if data-at-rest encryption is deactivated on the IBM FlashSystem A9000 or IBM FlashSystem A9000R system. Make sure that at least two Security Administrators are defined on the system. Recovery key creation requires communication with the key server when using the external key encryption solution.

Attention: Most of these recovery key steps will have been done while setting up and enabling the local key server encryption option. You can review them in “Step 3: Creating and verifying encryption recovery keys” on page 113.

The following steps are required to configure recovery keys:

1. Generate recovery keys for each Security Administrator.
2. Get keys for each Security Administrator (make a note of them for later).
3. Verify keys for each Security Administrator to make the keys usable.

The IBM FlashSystem A9000 or IBM FlashSystem A9000R system generates a random recovery key and a related wrapping key. The recovery key can also be rekeyed, which generates a new recovery key. That new key must be acquired and verified again by each defined Security Administrator.

4.3.2 Recovery key generation and verification with XCLI

If you prefer, you can generate the recovery key through the XCLI by completing the following steps:

1. Start with the **`encrypt_recovery_key_generate`** command that is shown in Table 4-1.

Table 4-1 The encrypt_recovery_key_generate command

Category	Command	Description
System	**`encrypt_recovery_key_generate`**	Specifies which Security Administrators receive recovery key shares and the minimum number of recovery key shares that must be entered

Note that you must specify the SecAdmin users on the command line. An example of this command is shown in Example 4-1.

Example 4-1 Generate Recovery Key by using XCLI

```
A9000_ITSO>>encrypt_recovery_key_generate min_req=2
users=itsosecadmin1,itsosecadmin2

Command executed successfully.
```

2. Each defined Security Administrator must then, individually, using their specific credentials log in to the XCLI, collect, and then verify, the keys that have been generated in the prior step, as shown in Table 4-2 and in Example 4-4.

Table 4-2 The encrypt_recovery_key_get command

Category	Command	Description
System	**encrypt_recovery_key_get**	Retrieves the recovery key share that is generated for the current user

Here is an example of the first part of the command, without the verify, as shown in Example 4-2.

Example 4-2 Recovery Key get command by using XCLI

```
A9000_ITSO>>encrypt_recovery_key_get
key=628O7CB19O2AMO74EDLA4EV8FOC574E40A1564F5557OCDEEBED37BC3876789

Command executed successfully.
```

An example of this command is shown in Example 4-3.

Example 4-3 Recovery key verification by using XCLI

```
A9000_ITSO>>encrypt_recovery_key_verify
key=628O7CB19O2AMO74EDLA4EV8FOC574E40A1564F5557OCDEEBED37BC3876789
Command executed successfully.
recovery_status=Key accepted, 1 of 2 fragments have been verified
remaining_fragments=1
```

3. All defined Security Administrators must collect, and then verify, their keys. If this process is not done, based on the previously defined Security Administrators, then the IBM FlashSystem A9000 / R displays an error as shown in Example 4-4.

Example 4-4 Enforcement of recovery key retrieval and verification

```
[1st user]
A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=29AEC7E516EB65943874817B910853892C65BC22BED6CC84E1EFFC49F7992E55
A9000>>

A9000>>encrypt_recovery_key_verify
key=29AEC7E516EB65943874817B910853892C65BC22BED6CC84E1EFFC49F7992E55
Command executed successfully.
    recovery_status=Key accepted, 1 of 3 fragments have been verified
    remaining_fragments=2
```

```
A9000>>

A9000>>encrypt_recovery_key_list
Key Created           Number of Shares   Min Required
2017-04-28 09:56:59   3                  2
A9000>>
....

[2nd user]

A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=6EBF275AC35FABE05A826F385F3B97B3915679067F55736E35A80B2C9C41AAA2

A9000>>encrypt_recovery_key_verify
key=6EBF275AC35FABE05A826F385F3B97B3915679067F55736E35A80B2C9C41AAA2
Command executed successfully.
    recovery_status=Key accepted, 2 of 3 fragments have been verified
    remaining_fragments=1
....

[3rd user]
A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=B3D086D06FD2F22C7C905CF52D6EDBDDF64735EA3FD41A5889621A1040E926EE
A9000>>

A9000>>encrypt_recovery_key_verify
key=B3D086D06FD2F22C7C905CF52D6EDBDDF64735EA3FD41A5889621A1040E926EE
Command executed successfully.
    recovery_status=all recovery key fragments have been verified
    remaining_fragments=0
A9000>>
```

4. All Security Administrators users that are defined by the **encrypt_recovery_key_generate** command must verify their recovery keys, as shown in Table 4-3.

Table 4-3 The encrypt_recovery_key_verify command

Category	Command	Description
System	**encrypt_recovery_key_verify**	Confirm s that the current user correctly copied the recovery key share that is presented by the **encrypt_recovery_key_get** command

Here is an example of this command:

```
A9000_ITSO>>encrypt_recovery_key_verify
key=62807CB1902AM074EDLA4EV8F0C574E40A1564F55570CDEEBED37BC3876789
Command executed successfully.
recovery_status=Key accepted, 1 of 2 fragments have been verified
remaining_fragments=1
```

5. The state of verification can be checked by running the **encrypt_recovery_key_status** command that is shown in Table 4-4.

Table 4-4 The encrypt_recovery_key_status command

Category	Command	Description
System	**encrypt_recovery_key_status**	Shows the status of the recovery keys

Here is an example of this command:

```
A9000_ITSO>>encrypt_recovery_key_status
Date Created          User          Status
2016-11-24 11:55:15   itsosecadmin1 Verified
2016-11-24 11:55:15   itsosecadmin2 Unverified
```

You can run **encrypt_recovery_key_list** to show the number of shares that have recovery keys and how many of them are required for recovery.

6. Each Security Administrator must log in to the XCLI with the correct credentials and repeat the Activate Recovery Key procedure.
7. Save the key in a text file and keep it in a secure place that is physically separate from both the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and the IBM Security Key Lifecycle Manager servers.

Important: The rekey process is not finished until all defined Security Administrators have confirmed their keys. Until all newly generated keys are confirmed, the “old” keys remain active. The key switch from old to new is an atomic operation carried after all new keys are confirmed.

After all defined Security Administrators have collected and verified their keys, the IBM FlashSystem A9000 or IBM FlashSystem A9000R data-at-rest encryption can be activated. For more information, see 4.4.1, “Activating data-at-rest encryption” on page 105.

4.3.3 Recovery key rekey

Rekeying is the process of changing cryptographic values in the chain between the key server, recovery key, and DAKs so that the previous value no longer enables access to the system.

The rekey and Verify Recovery Key functions can be performed any time while the recovery key is configured and the appropriate key server is available.

For the local key server option, this procedure is always available because the key server is software inside the IBM FlashSystem A9000 / R.

For the external key server, such as SKLM, the server must be online and reachable to enable the IBM FlashSystem A9000 or IBM FlashSystem A9000R system to verify and update keys. Only then will the external key server generate a new recovery key.

Using the external key server methodology, on an IBM FlashSystem A9000 or IBM FlashSystem A9000R system that was stolen and put in a separate environment, rekeying the recovery key is not possible.

Unfortunately, if the local key server methodology is used on an IBM FlashSystem A9000 or IBM FlashSystem A9000R system that was stolen and put in a separate environment, there is the possibility of data theft.

During a rekeying operation, for an external key server the following actions are performed:

1. The IBM FlashSystem A9000 or IBM FlashSystem A9000R system sends the ESK to the IBM Security Key Lifecycle Manager and requests a rekey validation.
2. The IBM Security Key Lifecycle Manager verifies the identity of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system by using its certificates.
3. The IBM Security Key Lifecycle Manager signals the IBM FlashSystem A9000 or IBM FlashSystem A9000R system that it can proceed to generate a recovery key.
4. The IBM FlashSystem A9000 or IBM FlashSystem A9000R system generates a recovery key.

A more streamlined process exists for the local key rekey operation:

1. A Security Admin logs in to the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and requests a rekey operation. This process sends the Random Wrapper Key (RWK) to the local key server and requests an immediate rekey validation.
2. The local key management process signals the IBM FlashSystem A9000 or IBM FlashSystem A9000R system that it can proceed to generate a recovery key. The RWK will then rewrap the Master key or XMK.
3. The IBM FlashSystem A9000 or IBM FlashSystem A9000R system generates a recovery key.

Changing the recovery key does not erase the data.

An Unconfigure function of the recovery key is not available after data-at-rest encryption is activated, but you can use the Regenerate Recovery Key function to change your keys.

The recovery key can also be rekeyed to replace the current recovery key with a new one. All defined Security Administrators must collect and verify the new recovery key.

The recovery key can be rekeyed in the XCLI by running the **encrypt_recovery_key_rekey** command that shown in Table 4-5.

Table 4-5 encrypt_recovery_key_rekey

Category	Command	Description
System	**encrypt_recovery_key_rekey**	Restarts the recovery key generation process that is described in **encrypt_recovery_key_generate**

Figure 4-79 shows the command.

```
A9000 ITSO>>encrypt_recovery_key_rekey
Command executed successfully.
```

Figure 4-79 The encrypt_recovery_key_rekey command

After the recovery key is replaced with a new one, the same recovery key verification procedure that is shown in Step 4 on page 100 must be performed again by designated Security Admins.

4.3.4 Using a recovery key to unlock IBM FlashSystem A9000/A9000R

On an IBM FlashSystem A9000 or IBM FlashSystem A9000R system with a recovery key that is configured, an option exists to let a Security Administrator enter the recovery key.

After a power-off followed by a power-on action, or after a simple reboot, if the IBM FlashSystem A9000 or IBM FlashSystem A9000R system cannot get the required data key from the SMK server, it attempts to contact all other configured IBM Security Key Lifecycle Manager servers to obtain the required key.

In some key server versions, a secondary key server must be promoted to master first. If that is not successful, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system enters `maintenance_mode` status, which can be verified by running **`state_list`**, as shown in Figure 4-80.

```
A9000_ITSO>>state_list
Category                 Value
------------------------ -------------
data_protection_status   N/A
data_reduction_state     Online
encryption               Enabled
safe_mode                no
shutdown_reason          No Shutdown
system_state             maintenance
target_state             on
```

Figure 4-80 state_list command

In this case, the Security Administrators must provide their recovery keys. The IBM FlashSystem A9000 or IBM FlashSystem A9000R system uses the recovery key to unwrap the XIV SMK that unlocks the IBM FlashSystem A9000 or IBM FlashSystem A9000R system. After access to data is restored, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is available to serve host I/O again.

Each Security Administrator enters their individual parts of the recovery key until the number of defined minimum required Security Administrators is reached and it is again possible to unlock the IBM FlashSystem A9000 or IBM FlashSystem A9000R system. The XCLI command to do so is **`encrypt_recovery_key_enter`**, as shown in Table 4-6.

Table 4-6 encrypt_recovery_key_enter

Category	Command	Description
System	**`encrypt_recovery_key_enter`**	Unlocks encrypted disks when the system reboots and cannot access any of the defined key servers if the recovery keys were defined

This example shows the command:

```
encrypt_recovery_key_enter
key=62807CB1902AM074EDLA4EV8F0C574E40A1564F55570CDEEBED37BC3876789
```

As soon as the last one of the minimum number of defined Security Administrators has logged in with credentials and entered a recovery key, the IBM FlashSystem A9000 or IBM FlashSystem A9000R system unlocks and activates the data-at-rest encryption again, as shown in Figure 4-81.

```
secadmin1 user:
A9000_ITSO>>encrypt_recovery_key_enter
key=7177O8CB7DE9AOBCB40A619B17B82ADD8390C032965EED89B76401C6C20A3F71
Command executed successfully.
    recovery_status=1 of 2 recovery keys accepted
secadmin2 user:
A9000_ITSO>>encrypt_recovery_key_enter
key=E042B767755D599F726071DB095016BDBA20F638B0659D320E16D3E2BA3BD5AE
Command executed successfully.
    recovery_status=2 of 2 recovery keys accepted - enabling storage access
```

Figure 4-81 The encrypt_recovery_key_enter command

After the minimum required number of keys are entered, a Storage Administrator or a Security Administrator user must change the state of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system from `maintenance` to `on` by running the **`encrypt_recovery_key_finish`** command, as shown in Figure 4-82.

```
A9000_ITSO>>encrypt_recovery_finish
Command executed successfully.
```

Figure 4-82 The enctrypt_recovery_finish command

Now, you can verify by running the **`state_list`** command again that the system_state changed to `on`, as shown in Figure 4-83.

```
A9000_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
```

Figure 4-83 The state_list command

4.4 Activating and deactivating encryption

Now that the implementation and configuration of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system and its corresponding IBM Security Key Lifecycle Manager server are finished, you can enable (activate) the data-at-rest encryption in IBM FlashSystem A9000 or IBM FlashSystem A9000R system.

More data-at-rest encryption-related XCLI commands are listed in IBM Knowledge Center:

https://www.ibm.com/support/knowledgecenter/en/STJKN5_12.1.0/c_category__Encryption_enablement_and_support_commands.html

4.4.1 Activating data-at-rest encryption

For data-at-rest encryption to complete successfully, all of these prerequisites must be fulfilled:

- The current encryption state must be DISABLED (displayed as `Supported` in the **`state_list`** output).
- One master key server must be configured successfully, and recovery keys must be generated and verified by at least two separate Security Administrators, unless a **`recovery_keys=no`** parameter was passed.

This command is entered by a Security Administrator to enable the data protection feature.

The data-at-rest encryption can be activated from the XCLI by using the command that is shown in Table 4-7.

Table 4-7 The encrypt_enable command

Category	Command	Description
System	**`encrypt_enable`**	Enables the data protection feature

Figure 4-84 shows the output of the command.

```
ITSO A9000>>encrypt_enable
Warning:   ARE_YOU_SURE_YOU_WANT_TO_ENABLE_ENCRYPTION y/n: y
Command executed successfully.
```

Figure 4-84 Output of the encrypt_enable command

It can take a couple of minutes to enroll the MicroLatency modules and the vaulting SSDs. After the procedure finishes, a successful encryption of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system can be seen by running the **`state_list`** command, as shown in Figure 4-85.

```
A9000_ITSO>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
```

Figure 4-85 The state_list command to verify that encryption is enabled

4.4.2 Deactivating data-at-rest encryption

This **encrypt_disable** command disables the data protection feature. A prerequisite for this action is that no volumes and no pools are defined on the system. In addition to disabling the data protection, a cryptographic erase is performed on all protected storage to ensure that all existing user data is no longer accessible. After the command completes successfully, all MicroLatency modules and vault SSDs are left in an unlocked state. Disabling encryption when the encryption state is other than ACTIVE causes an error (the **state_list** command must show it as `Enabled`).

In XCLI, run **encrypt_disable** and the system prompts you to verify that you want to deactivate encryption on the IBM FlashSystem A9000 or IBM FlashSystem A9000R system, as shown in Figure 4-86.

```
A9000_ITSO>>encrypt_disable
Warning:   Are you sure you want to disable encryption on this system? y/n: y
Command executed successfully.
```

Figure 4-86 The encrypt_disable command

4.5 Verifying the encryption state

These actions enable users to verify the encryption state of the system:

- With IBM FlashSystem A9000 or IBM FlashSystem A9000R software Version 12.0.1 or later, the Encryption column in the output of the **state_list** command shows whether the system is encrypted, as shown in Figure 4-87. The **state_list** command can show any of these states: `Supported` (encryption is disabled), `Enabling/Activating`, `Partial`, `Enabled/Activated`, `Enabling on Boot`, or `Disabling`.

```
A9000_ITSO>>state_list
Category                Value
system_state            on
target_state            on
safe_mode               no
shutdown_reason         No Shutdown
data_protection_status  Fully Protected
encryption              Enabled
data_reduction_state    Online
```

Figure 4-87 The state_list command

- The key server state can be checked by running the **encrypt_keyserver_list** command, as shown in Figure 4-88. The status is checked once every hour or if something has changed or was updated.

```
A9000_ITSO>>encrypt_keyserver_list
Module Name App/Key Status Last_time_checked Master Port Address Keyserver Type
1      sklm1 ACTIVE        2016/10/26 13:48:45 yes   5696  9.123.123.123   TKLM
2      sklm1 ACTIVE        2016/10/26 13:48:45 yes   5696  9.123.123.123   TKLM
3      sklm1 ACTIVE        2016/10/26 13:48:45 yes   5696 9.123.123.123   TKLM
```

Figure 4-88 The encrypt_keyserver_list command

- To force an update of the **encrypt_keyserver_list** state, you can run the **encrypt_keyserver_check_status** command, as shown in Figure 4-89.

```
A9000_ITSO>>encrypt_keyserver_check_status
Command executed successfully.
```

Figure 4-89 The encrypt_keyserver_check_status command

- The **flash_enclosure_list** command gives you various information about its state, as shown in Figure 4-90. One is the encrypted value that shows yes if the whole the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is encrypted and no if it is not. The key_needed value shows you whether the flash enclosure and its MicroLatency modules are locked. If they are locked, the value is yes.

```
A9000_ITSO>>flash_enclosure_list -x
<XCLIRETURN STATUS="SUCCESS" COMMAND_LINE="flash_enclosure_list -x">
   <OUTPUT>
      <flash_enclosure id="4d90e000000">
         <component_id value="1:Flash_Enclosure:1"/>
         <status value="OK"/>
         <currently_functioning value="yes"/>
         <control_path_status value="OK"/>
         <fru_part_number value="00DH521"/>
         <fru_identity value="11S00DH412YS16BG66N03Z"/>
         <temperature_state value="cool"/>
         <fw_level value="1.4.4.1-199.102"/>
         <required_service value=""/>
         <service_reason value=""/>
         <enabled value="yes"/>
         <cluster_ip value="14.10.204.35"/>
         <array_status value="OK"/>
         <redundancy_state value="online"/>
         <has_spare value="yes"/>
         <fw_upgrade_progress value="0"/>
         <fw_upgrade_status value="invalid"/>
         <fw_upgrade_committed value="no"/>
         <target_fw_version value=""/>
         <fw_file_name value=""/>
         <utility_file_name value=""/>
         <cluster_id value="000002006d2ab064"/>
         <encrypted value="yes"/>
         <key_needed value="no"/>
         <base_guid value="5005076061D24B80"/>
         <charging value="no"/>
         <flash_status value="ok"/>
      </flash_enclosure>
   </OUTPUT>
</XCLIRETURN>
```

Figure 4-90 The flash_enclosure_list -x command

- The **vault_device_list** command shows you the state of the SSDs that are used as vaulting devices in the IBM FlashSystem A9000 or IBM FlashSystem A9000R system. The syntax of the command is **vault_device_list [module=ModuleNumber | vault_device=ComponentId]**. If you do not specify the vault device, it shows all the existing SSDs. After the encryption of the IBM FlashSystem A9000 or IBM FlashSystem A9000R system is enabled, the **encryption_state** shows the value `Enrolled`, and **sw_encryption_active** shows the value `yes`, as shown in Figure 4-91.

```
A9000_ITSO>>vault_device_list vault_device=1:Vault_Device:1:1 -x
<XCLIRETURN STATUS="SUCCESS" COMMAND_LINE="vault_device_list
vault_device=1:Vault_Device:1:1 -x">
   <OUTPUT>
      <vault_device id="51313b00004">
         <component_id value="1:Vault_Device:1:1"/>
         <status value="OK"/>
         <currently_functioning value="yes"/>
         <capacity_in_bytes value="250059350016"/>
         <capacity value="250GB"/>
         <target_status value=""/>
         <model value="HUSMR1625ASS20E"/>
         <vendor value="LENOVO-X"/>
         <serial value="0PVKLBMA"/>
         <part_number value="00NA685"/>
         <requires_service value=""/>
         <service_reason value=""/>
         <temperature value="24"/>
         <firmware value="P4C9"/>
         <original_firmware value=""/>
         <revision value=""/>
         <drive_pn value=""/>
         <encryption_state value="Enrolled"/>
         <security_state value="Unchecked"/>
         ...etc...
         <accessible_modules id="14" value="no"/>
         <desc>
            <disk_id value="1"/>
            <read_fail value="no"/>
            <smart_code value="NO ADDITIONAL SENSE INFORMATION"/>
            <smart_fail value="no"/>
            <power_on_hours value="0"/>
            <power_on_minutes value="0"/>
            <last_sample_time value="0"/>
            <last_sample_serial value="0PVKLBMA"/>
            <last_time_pom_was_mod value="0"/>
            <temperature_status>
            .......
                        <power_is_on value="no"/>
            <bgd_scan value="0"/>
            <sw_encryption_active value="yes"/>
         </desc>
      </vault_device>
   </OUTPUT>
</XCLIRETURN>
```

Figure 4-91 The vault_device_list -x command

4.6 Local encryption mechanism and process

The local key process is similar, still using several levels of symmetric key encryption for the data-at-rest solution. A key is encrypted (or wrapped) between components as shown in Figure 4-92.

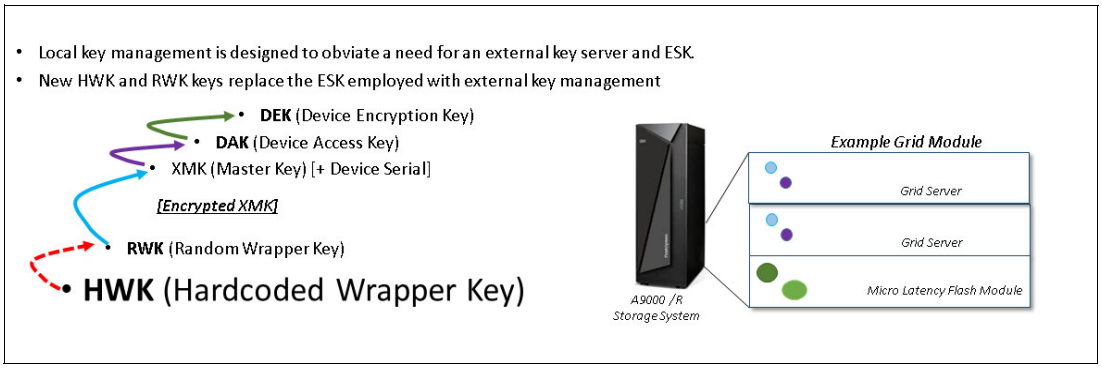

Figure 4-92 Local Key Encryption Process Flow

With the local key option specified, during the encryption setup, IBM FlashSystem A9000 or IBM FlashSystem A9000R generates a RWK to encrypt a Master Key (XMK). The RWK replaces the ESK discussed in 4.2.1, “External key server encryption process overview” on page 59. The RWK is itself encrypted with a Hardcoded Wrapper Key (HWK) that is contained but obfuscated in the storage system code.

All requirements and key management remain the same as described in the external key section. During normal boot operations, the encrypted RWK is retrieved from the key repository inside the system, and decrypted by using the HWK. The encrypted XMK is retrieved from the key repository and decrypted by using the RWK. After the XMK is decrypted, the generation of DAKs is unchanged.

Important: To prevent theft of data, if the system is to be transported to a new data center, a secure erase of the storage system should be performed.

Furthermore, as a practical concern, if any grid servers or their boot SSDs are replaced, a Security Admin should issue the transparent rekey command after the new component has been phased in. This process ensures a new RWK is randomized and used.

4.6.1 Configuring local encryption

When the user enables encryption, the system generates an RWK that encrypts the Master key (XMK). This key is protected (*wrapped*) by the HWK.

During a normal startup operation, the encrypted RWK is retrieved from the system key repository and decrypted by using the HWK. Then, the encrypted XMK is retrieved from the key repository and decrypted by using the RWK. After the XMK is decrypted, the generation of DAKs continues, and is the same as with the external key server.

REKs are strongly encouraged to ensure data access. They are the only option to enable the possibility to recover the master key if it is lost.

Important: A recovery key can be created only if data-at-rest encryption is not yet enabled. You cannot create a recovery key after encryption is activated. However, you do have the option of rekeying.

When you enable encryption on an IBM FlashSystem A9000 or IBM FlashSystem A9000R system, the following keys are generated:

- For each MicroLatency module, in all flash enclosures, a Device Access Key (DAK) is generated that is based on the XMK and the serial number of the flash enclosure.
- For the vaulting devices (SSDs), the device access key is generated that is based on the XMK and the grid controllers' MegaRaid adapter serial number.

The encryption for the IBM FlashSystem A9000, or IBM FlashSystem A9000R can be enabled during the installation of the system or later. If encryption is not enabled, the system might not meet the customer's compliance standards or the legal compliance standards and the data might not be protected against security threats.

Important: The encryption for IBM FlashSystem A9000 or IBM FlashSystem A9000R can only be disabled when there are no pools or volumes defined. Therefore, you must back up any data that must be kept, or migrate it to another system, before you deactivate encryption on any of those systems.

4.6.2 Local encryption configuration and process

This section describes the steps to configure and implement IBM FlashSystem A9000 or IBM FlashSystem A9000R system using the local encryption.

Except where specifically noted, at the time of writing this material, most of configuration steps could be done with XCLI only.

Step 1: Verifying encryption status on the A9000 / R

The first step is to verify that IBM FlashSystem A9000 / A9000R has its software license accepted and it is ready to encrypt (see Example 4-5).

Example 4-5 Verify Internal Encryption Status

```
A9000>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Supported
data_reduction_state     Online
A9000>>
```

Step 2: Creating and verifying secadmin users

The next step is to create and verify that there are a minimum of three (3) users with the role of secadmin defined inside the A9000 / R. This step can be accomplished in the GUI as described next, or the XCLI, as shown in Example 4-6 on page 112, along with an error if you try to proceed without all of them properly defined.

To use the HSM GUI, complete the following steps:

1. Open a web browser, and specify the IP address and IP port of the HSM by using the following web address format:

   `https://<HSM_IP>:<HSM_PORT>/`

   For example:

   `https://9.123.123.123:8443/`

2. Enter your IBM FlashSystem A9000 or IBM FlashSystem A9000R system administrator role user ID and password, and then click **Login**.

3. The HSM UI Dashboard indicates that you are successfully logged in to the IBM Hyper-Scale Manager. If you manage multiple IBM FlashSystem A9000 or IBM FlashSystem A9000R systems, choose the correct one from the list in the Dashboard, as shown in Figure 4-93.

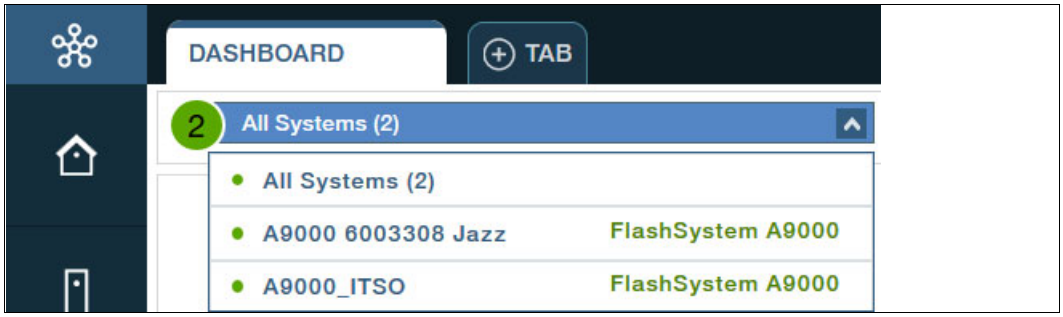

Figure 4-93 Choose your IBM FlashSystem A9000 or IBM FlashSystem A9000R system from the dashboard

4. On the Dashboard's left side, click the **Access Views** icon and then click **Users** to open the **Users** menu, as shown in Figure 4-94.

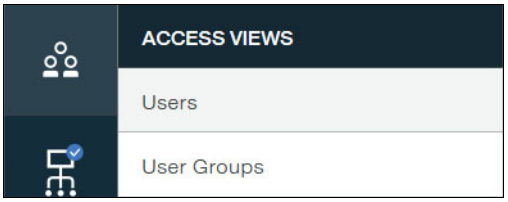

Figure 4-94 Users access view

5. In the Users access view, click the **+** icon to create an object, as shown in Figure 4-95.

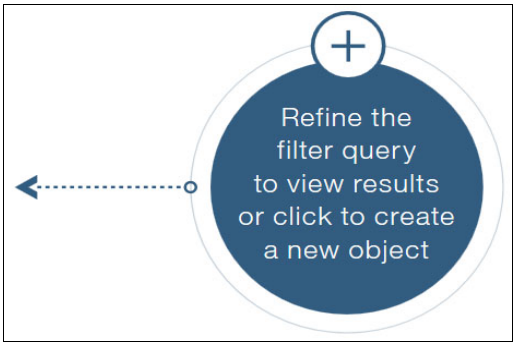

Figure 4-95 Create a user object

6. Enter the name for your security admin user, select a system and password, change the category to **Security Administrator**, and click **Create**. **Phone** and **Email** are optional fields. You cannot choose a domain because Security Admin is associated to the Global Space only. See Figure 4-96.

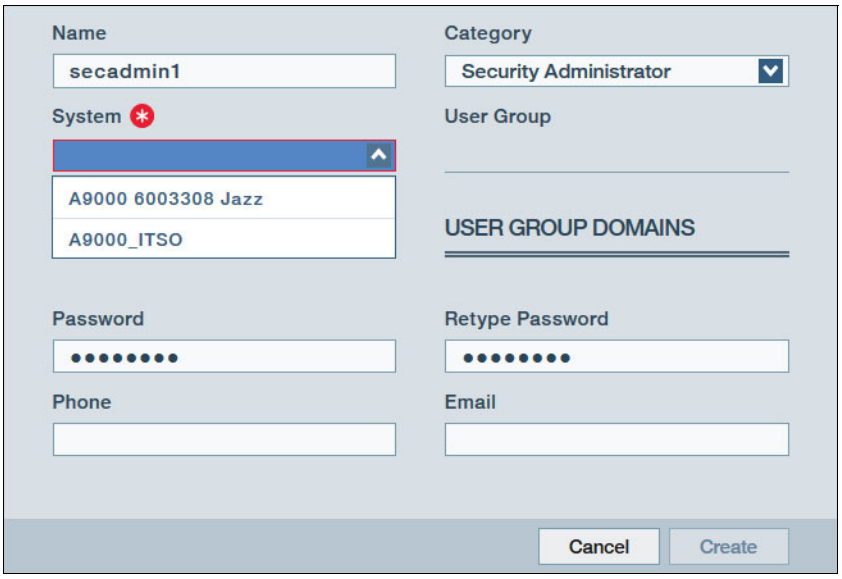

Figure 4-96 Create a secadmin user

Repeat these steps for further users. The minimum secadmin users that are required is two. The XCLI can also be used to create the required users. Note the error that is also shown in Figure 4-6 if the required users are not created.

Example 4-6 Create Minimum SecAdmin users inside A9000 / R

```
A9000>>user_define user=SecAdmin password=SecAdmin1 password_verify=SecAdmin1
category=securityadmin
Command executed successfully.

A9000>>encrypt_recovery_key_generate key_scheme=local users=SecAdmin
Error:   INSUFFICIENT_RK_ADMINS
Details: Number of users must be greater than or equal to the minimal required
number.

A9000>>user_define user=SecAdmin2 password=123456 password_verify=123456
category=securityadmin
Command executed successfully.

A9000>>user_define user=SecAdmin3 password=123456 password_verify=123456
category=securityadmin
Command executed successfully.

A9000>>user_list
Name                    Category           Group   Active   Email Address
admin             storageadmin          yes
SecAdmin           securityadmin         yes
SecAdmin2          securityadmin         yes
SecAdmin3          securityadmin         yes
A9000>>
```

Step 3: Creating and verifying encryption recovery keys

The next step is to create and verify the recovery key. This step cannot be done from a user with storageadmin rights. It can only be done as one of the secadmin role users. In the prior step, you created the minimum number. The process is shown in Example 4-7.

Tip: For more information about the recovery keys, see 4.3, “Recovery key use and maintenance” on page 97.

Example 4-7 Create encryption recovery key

```
A9000>>encrypt_recovery_key_generate
Error:   OPERATION_FORBIDDEN_FOR_USER_CATEGORY
Details: This operation cannot be performed by users of this category

---Log back in as a user with SecAdmin role:

Machine IP/Hostname: 9.155.120.218
User name: SecAdmin
Password: *********
connecting.
A9000>>
A9000>>encrypt_recovery_key_generate key_scheme=local
users=SecAdmin,SecAdmin2,SecAdmin3

Command executed successfully.
```

As shown in Example 4-7, one XCLI command creates the three split recovery keys and specifies the users and their accounts. The next step is to get and then verify each of the keys assigned to each of the specified secadmin users.

To save time verify with the existing user now, and then log off, and back in using the other two specified users. This process is shown in Example 4-8.

Example 4-8 Retrieve and Verify Recovery keys for each specified secadmin user

```
A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=29AEC7E516EB65943874817B910853892C65BC22BED6CC84E1EFFC49F7992E55
A9000>>
A9000>>encrypt_recovery_key_verify
key=29AEC7E516EB65943874817B910853892C65BC22BED6CC84E1EFFC49F7992E55
Command executed successfully.
    recovery_status=Key accepted, 1 of 3 fragments have been verified
    remaining_fragments=2
A9000>>

We repeat the above steps for remaining secadmin users - note the keys are
different for each!

-2nd user
A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=6EBF275AC35FABE05A826F385F3B97B3915679067F55736E35A80B2C9C41AAA2
```

```
A9000>>encrypt_recovery_key_verify
key=6EBF275AC35FABE05A826F385F3B97B3915679067F55736E35A80B2C9C41AAA2
Command executed successfully.
    recovery_status=Key accepted, 2 of 3 fragments have been verified
    remaining_fragments=1

- 3rd user
A9000>>encrypt_recovery_key_get
Command executed successfully.
    key=B3D086D06FD2F22C7C905CF52D6EDBDDF64735EA3FD41A5889621A1040E926EE
A9000>>

A9000>>encrypt_recovery_key_verify
key=B3D086D06FD2F22C7C905CF52D6EDBDDF64735EA3FD41A5889621A1040E926EE
Command executed successfully.
    recovery_status=all recovery key fragments have been verified
    remaining_fragments=0
A9000>>
```

Notice in the last step above, that after the third user has retrieved and then verified their key, IBM FlashSystem A9000 / A9000R has confirmed that it is ready to encrypt.

Step 4: Enabling encryption using the internal keyserver

The next step is a simple XCLI command using one of the secadmin users. The system ensures that you confirm this step, which is shown in Example 4-9.

Example 4-9 Enable Internal Keyserver Encryption

```
A9000>>encrypt_enable key_scheme=local recovery_keys=yes
Warning:  Are you sure you want to enable encryption with local key management on
this system? y/n: y
Command executed successfully.
A9000>>
- In progress:

A9000>>state_list
Category                Value
system_state            on
target_state            on
safe_mode               no
shutdown_reason         No Shutdown
data_protection_status  Fully Protected
encryption              Enabling
data_reduction_state    Online
A9000>>

-When completed:
A9000>>state_list
Category                Value
system_state            on
target_state            on
safe_mode               no
shutdown_reason         No Shutdown
data_protection_status  Fully Protected
encryption              Enabled
```

```
data_reduction_state     Online
A9000>>
```

The entire process is logged in the IBM FlashSystem A9000 / R events, visible in both the GUI and the XCLI. For clarity, this output is shown in reverse order in Example 4-10.

Example 4-10 Event listing of Internal Keyserver Encryption process

Severity	Event Code	User	Description
Informational	ENCRYPT_ENABLE_COMPLETED		Encryption is in effect.
Informational	FLASH_ENCRYPTION_STATUS_CHANGED		Encryption enabled changed to yes. Enclosure id 1:Flash_Enclosure:1.
Informational	ENCRYPT_ENABLE_STARTED	SecAdmin3	Starting encryption activation. This process can take several minutes to complete.
Informational	ENCRYPT_RECOVERY_KEY_ALL_SHARES_VERIFIED	SecAdmin3	All recovery key shares have been verified.
Informational	ENCRYPT_RECOVERY_KEY_VERIFIED	SecAdmin3	Recovery key verifed successfully for user 'SecAdmin3'.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'SecAdmin3' from IP '9.244.87.93' successfully logged into the system.
Informational	ENCRYPT_RECOVERY_KEY_VERIFIED	SecAdmin2	Recovery key verifed successfully for user 'SecAdmin2'.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'SecAdmin2' from IP '9.244.87.93' successfully logged into the system.
Warning	USER_LOGIN_HAS_FAILED		User 'SecAdmin2' from IP '9.244.87.93' failed logging into the system.
Informational	ENCRYPT_RECOVERY_KEY_VERIFIED	SecAdmin	Recovery key verifed successfully for user 'SecAdmin'.
Informational	ENCRYPT_RECOVERY_KEYS_GENERATED	SecAdmin	Recovery keys created.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'SecAdmin' from IP '9.244.87.93' successfully logged into the system.
Informational	USER_DEFINED	admin	A user with name 'SecAdmin3' and category securityadmin was defined.
Informational	USER_DEFINED	admin	A user with name 'SecAdmin2' and category securityadmin was defined.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'admin' from IP '9.244.80.65' successfully logged into the system.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'SecAdmin' from IP '9.244.87.93' successfully logged into the system.
Informational	USER_LOGIN_HAS_SUCCEEDED		User 'admin' from IP '9.244.87.93' successfully logged into the system.

4.7 Converting from external to local key encryption

These steps enable a Security Admin to change from using an external encryption server, such as IBM Secure Key Lifecycle Server, to the local encryption within IBM FlashSystem A9000 or IBM FlashSystem A9000R. This flow is transparent to production workflow and data and is considered nondisruptive.

Attention: The reverse action of migrating from a local key encryption to using an external key server cannot be accomplished without first erasing all data (volumes and pools).

All of the steps must be done inside the XCLI, and require the same prerequisites as an initial configuration, which is detailed in Figure 4-11.

Example 4-11 Conversion from an external keyserver to an internal one

```
---Verify correct state and ready to change encryption:

A9000R>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
A9000R>>

---Change the encryption key server:
```

```
A9000R>>encrypt_change_key_scheme key_scheme=local

Warning:   Are you sure you want to change the key management scheme? This
operation is irreversible. y/n: y
Command executed successfully.
A9000R>>

A9000R>>encrypt_key_scheme_get
Command executed successfully.
    encrypt_key_scheme=LOCAL
A9000R>>
```

Confirm that you are not using external anymore and also clean up those entries detailed in Example 4-12.

Example 4-12 Confirm change of keyserver and cleanup

```
A9000R>>encrypt_keyserver_list
Module   Name    App/Key Status   Last time checked     Master   Port   Address         Keyserver Type
1        sklm1   UNKNOWN          2017/05/03 23:49:13   yes      5696   9.155.117.149   TKLM
2        sklm1   UNKNOWN          2017/05/03 23:49:13   yes      5696   9.155.117.149   TKLM
3        sklm1   UNKNOWN          2017/05/03 23:49:13   yes      5696   9.155.117.149   TKLM

A9000R>>

Now we need to remove these entries:

A9000R>>encrypt_keyserver_delete name=sklm1
Command executed successfully.
A9000R>>

We will verify that no keyservers exist:

A9000R>>encrypt_keyserver_list
None Defined
A9000R>>
```

The next step is ensuring that the new encryption method has fully recoverable keys and verification of the secadmins, and their individual encryption keys for that process shown in Example 4-13.

Example 4-13 Rekey for local and recovery keys after keyserver change

```
A9000R>>encrypt_local_rekey
Warning:   Are you sure you want to change the local key? y/n: y
Command executed successfully.
A9000R>>

Now we will do the same for the recovery keys and verify:

A9000R>>encrypt_recovery_key_rekey
Command executed successfully.

A9000R>>encrypt_recovery_key_status
Date Created          User        Status
2017-05-03 19:39:12   SecAdmin    Verified
2017-05-03 19:39:12   SecAdmin2   Verified
2017-05-04 00:47:42   SecAdmin    Unverified
```

```
2017-05-04 00:47:42   SecAdmin2   Unverified
A9000R>>
```

Now we will need to safeguard the new keys and verify them as before, using each secadmin user:

```
A9000R>>encrypt_recovery_key_get
Command executed successfully.
    key=83CF9E8C24BCF221E60AFA078F17C502BFB50CCBE8E1E88E9D5603BD006414FD
A9000R>>
```

---- Using the same process as during the initial config, we now need to verify the keys:

```
A9000R>>encrypt_recovery_key_verify
key=83CF9E8C24BCF221E60AFA078F17C502BFB50CCBE8E1E88E9D5603BD006414FD
Command executed successfully.
    recovery_status=Key accepted, 1 of 2 fragments have been verified
    remaining_fragments=1
A9000R>>
```

---Same step as above, but as the other secadmin user defined:

```
Machine IP/Hostname: 9.155.116.200
User name: SecAdmin2
Password: ******
connecting..
A9000R>>
A9000R>>encrypt_recovery_key_get
Command executed successfully.
    key=77C3116C4A34C973F6BDFD523564D78D18EA43826ED6F5E1B8F492E68115B279
A9000R>>
A9000R>>encrypt_recovery_key_verify
key=77C3116C4A34C973F6BDFD523564D78D18EA43826ED6F5E1B8F492E68115B279
Command executed successfully.
    recovery_status=all recovery key fragments have been verified
    remaining_fragments=0
A9000R>>
A9000R>>encrypt_recovery_key_status
Date Created          User        Status
2017-05-04 00:47:42   SecAdmin    Verified
2017-05-04 00:47:42   SecAdmin2   Verified
A9000R>>

A9000>>state_list
Category                 Value
system_state             on
target_state             on
safe_mode                no
shutdown_reason          No Shutdown
data_protection_status   Fully Protected
encryption               Enabled
data_reduction_state     Online
A9000>>
```

As you can see from the last step, the encryption keys have been verified and the keyserver has been changed from external to local. Example 4-14 is an illustration of the events that are shown in the logs.

Example 4-14 Events log for encryption keyserver change

```
2017-05-03 19:50:41 FLASH_ENCRYPTION_STATUS_CHANGED Encryption enabled changed to yes. Enclosure id
1:Flash_Enclosure:1.
2017-05-03 19:50:49 FLASH_ENCRYPTION_STATUS_CHANGED Encryption enabled changed to yes. Enclosure id
1:Flash_Enclosure:2.
2017-05-03 19:50:49 ENCRYPT_ENABLE_COMPLETED Encryption is in effect.
2017-05-04 00:39:08 ENCRYPT_CHANGE_KEY_SCHEME_COMPLETED Change key scheme from external to local key completed.

-Below entries show the removal and clean up of old key server:


2017-05-04 00:42:33 ENCRYPT_KEYSERVER_DELETED Key server 'sklm1' was deleted.
2017-05-04 00:45:53 ENCRYPT_LOCAL_REKEY_COMPLETED Local key rekey completed.
2017-05-04 00:49:46 ENCRYPT_RECOVERY_KEY_VERIFIED Recovery key verifed successfully for user 'SecAdmin2'.
2017-05-04 00:51:29 USER_LOGIN_HAS_SUCCEEDED User 'SecAdmin2' from IP '9.65.129.164' successfully logged into the
system.
2017-05-04 00:53:04 USER_LOGIN_HAS_SUCCEEDED User 'SecAdmin' from IP '9.65.129.164' successfully logged into the
system.
2017-05-04 00:54:30 ENCRYPT_RECOVERY_KEY_REKEY_SUCCESS Recovery key rekey was successful.
2017-05-04 00:54:30 ENCRYPT_RECOVERY_KEY_VERIFIED Recovery key verifed successfully for user 'SecAdmin'.
A9000R>>
```

5

Maintaining

This chapter explains maintenance tasks that are related to data-at-rest encryption for the IBM XIV and the IBM FlashSystem A9000 and IBM FlashSystem A9000R systems.

It covers these topics:

- Automated replication
- Starting and stopping an external IBM Security Key Lifecycle Manager server
- Encryption server rekey
- Encryption deadlock
- Component replacements

5.1 Automated replication

This function is not applicable to the local key server option for the IBM FlashSystem A9000 / R.

Automated replication provides a way of cloning an external key server automatically. This function avoids the need for manual or scripted backup and restore operations.

Supported: IBM Security Key Lifecycle Manager Version 2.6 or later supports automated replication functions through the GUI.

The replication capabilities depend on your version of IBM Security Key Lifecycle Manager:

- IBM Security Key Lifecycle Manager Version 2.6

  IBM Security Key Lifecycle Manager Version 2.6 provides an operating system-independent GUI for automated replication configuration. You can configure the replication program to replicate IBM Security Key Lifecycle Manager critical data across clone servers when news keys are added to the master server.

  The automated replication process enables cloning of IBM Security Key Lifecycle Manager environments to multiple servers in a manner that is independent of operating systems and the directory structure of the server. For example, you can replicate data from a master server on a Windows system to a clone server on a Linux system. You can clone a master IBM Security Key Lifecycle Manager server with up to 20 copies. For more information, see IBM Knowledge Center:

  `http://www.ibm.com/support/knowledgecenter/en/SSWPVP_2.6.0/com.ibm.sklm.doc/admin/cpt/cpt_ic_admin_replication_clone_master.html`

- IBM Security Key Lifecycle Manager Version 2.5

  IBM Security Key Lifecycle Manager version 2.5 automated clone replication uses a program to clone a master IBM Security Key Lifecycle Manager with up to five copies. You can configure the program to replicate the keys and also other configuration information, such as when new keys are rolled over.

  This program automates the replication of everything that is needed. IBM Security Key Lifecycle Manager provides a set of operations to replicate current active files and data across systems. This replication enables cloning of IBM Security Key Lifecycle Manager environments to multiple servers.

  You can replicate the following data:

  - Tables in the IBM Security Key Lifecycle Manager database.
  - All keys materials in the IBM Security Key Lifecycle Manager database.
  - IBM Security Key Lifecycle Manager configuration files apart from the replication configuration file.

    Note: This data is taken as part of an IBM Security Key Lifecycle Manager backup. During a replication, the replication configuration file is not backed up and passed to the clone.

  You can configure IBM Security Key Lifecycle Manager replication with the `ReplicationSKLMgrConfig.properties` configuration file. You must specify the replication configuration file on all systems that are involved in the replication process.

Each instance of IBM Security Key Lifecycle Manager is defined as either the master (the system that is to be cloned) or a clone (the system that the data is being replicated on). The master and clone systems must be identical. The operating system, directory structures, and IBM DB2® admin user must be same on all systems, or there might be unpredictable results.

5.2 Starting and stopping an external IBM Security Key Lifecycle Manager server

You might have to use the **startServer** or **stopServer** command to start or stop the IBM Security Key Lifecycle Manager server. Restarting that server, for example, is required after the completion of a restore task.

Versions 2.5 and 2.6 or later of IBM Security Key Lifecycle Manager server automatically restart after a backup file is restored when the `autoRestartAfterRestore` property value is `true` (default value) in the `SKLMConfig.properties` file.

5.2.1 Starting and stopping the server by using scripts

Scripts to start and stop the IBM Security Key Lifecycle Manager server are in the `WAS_HOME/bin` directory for Linux and IBM AIX platforms. In the commands, the **server1** parameter is the default name of the configured IBM Security Key Lifecycle Manager server instance.

Starting the IBM Security Key Lifecycle Manager server

To start the server, run the command for your system:

- Microsoft Windows systems:

  ```
  StartServer.bat server1
  ```

- Linux and IBM AIX systems:

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

Stopping the IBM Security Key Lifecycle Manager server

To stop the server, use the **stopServer** script for your system:

- Windows systems:

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

- Linux and AIX systems:

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

  When global security is enabled (do not disable global security when you use IBM Security Key Lifecycle Manager), enter the user ID and password of the IBM Security Key Lifecycle Manager console administrator as parameters for the **stopServer** script. The script prompts for these parameters if they are omitted, but you can specify them on the command line.

5.2.2 Determining status

If you want to determine whether the IBM Security Key Lifecycle Manager server is running, try to log in to the IBM Security Key Lifecycle Manager web console. If the login is successful, the IBM Security Key Lifecycle Manager service is running. Otherwise, you can issue the **serverStatus** command (in the `WAS_HOME/bin` folder) with the server instance, user name, and password parameters, as illustrated in Example 5-1.

Example 5-1 Check server status

```
SKLM26SLES11:~ # /opt/IBM/WebSphere/AppServer/bin/serverStatus.sh server1
-username wasadmin -password Password
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/KLMProfile/logs/server1/serverStatus.log
ADMU0128I: Starting tool with the KLMProfile profile
ADMU0500I: Retrieving server status for server1
ADMU0508I: The Application Server "server1" is STARTED
```

For Windows systems, you can also check in the Services window to verify that the service is running, as shown in Figure 5-1.

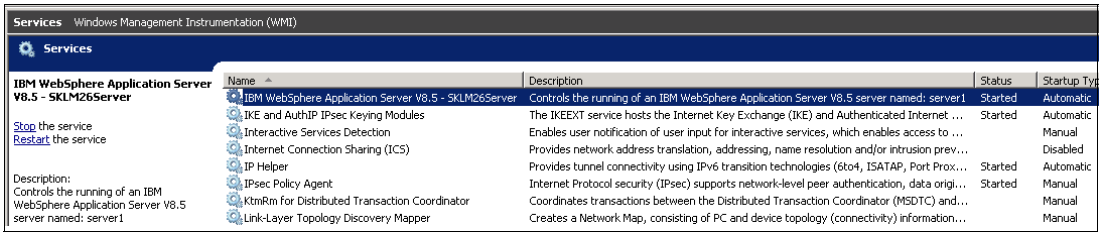

Figure 5-1 Windows server service state check

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

Example 5-2 Linux server state check

```
SKLM26SLES11:~ # ps -ef |grep server1|grep -v grep
root     18905     1  0 Oct27 ?        00:12:17
/opt/IBM/WebSphere/AppServer/java_1.7.1_32/bin/java -Declipse.security
-Dwas.status.socket=34282 -Dosgi.install.area=/opt/IBM/WebSphere/AppServer
...etc...
/opt/IBM/WebSphere/AppServer/profiles/KLMProfile/config SKLMCell SKLMNode server1
```

Run the **tklmKeyExport** command with the **-alias**, **-fileName**, **-keyStoreName**, and **-type** parameters to export secret or private keys. The **-alias** parameter is the name from the IBM Security Key Lifecycle Manager server in the XIV or in the IBM FlashSystem A9000 and IBM FlashSystem A9000R system. The **-keyStoreName** is the master keystore name for the IBM Security Key Lifecycle Manager server. See Example 5-3.

Example 5-3 Export the keystore

```
wsadmin>print AdminTask.tklmKeyExport('[-alias 1310092 -fileName TKLM_XIV
-keyStoreName "defaultKeyStore_xiv" -type privatekey -password xxxxxxxx]')
CTGKM0001I Command succeeded.
```

The `TKLM_XIV` file was created in the `/opt/IBM/tivoli/tip/products/tklm` folder.

Copy and archive the exported key to the new IBM Security 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.

5.3 Encryption server rekey

The encryption server rekey option is available on the XIV and on IBM FlashSystem A9000 and IBM FlashSystem A9000R systems. This option enables a user in the Security Administrator role to rekey against the master key server. As a preferred security practice, use this function to periodically change the keys. With an XIV system, you can use either the XIV GUI or the XIV XCLI. With IBM FlashSystem A9000 and IBM FlashSystem A9000R, you must use the XCLI.

For more information, see 5.3.2, “XIV and IBM FlashSystem A9000 or A9000R server rekey by using XCLI” on page 125.

5.3.1 XIV system external encryption rekey by using the XIV GUI

The following procedure describes how to rekey the server:

1. In the XIV GUI, click **All systems** → **List**, select your XIV system, right-click it, and choose the **Server Re-Key** menu entry, as shown in Figure 5-2.

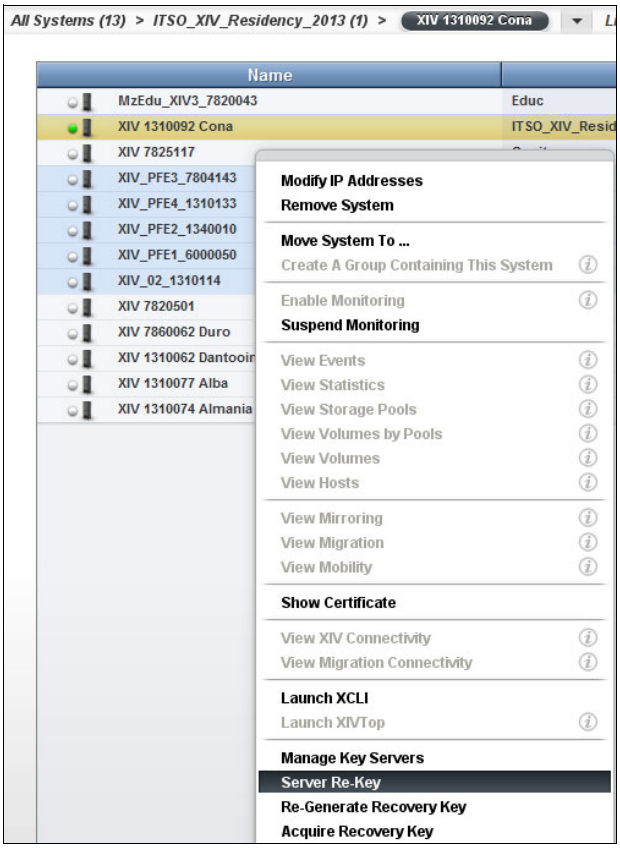

Figure 5-2 Server Re-Key

2. When the Server Re-key window that is shown in Figure 5-3 opens, select the XIV system from the drop-down menu, and click **Start**.

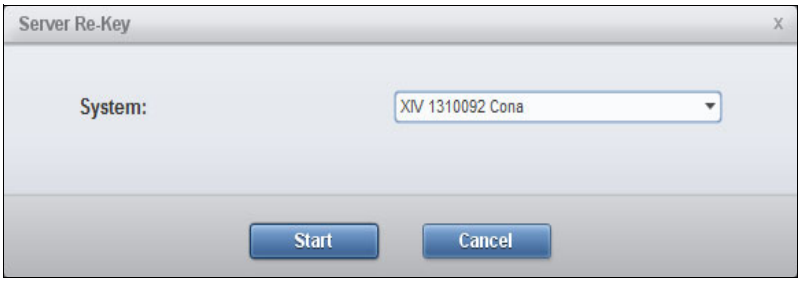

Figure 5-3 Server Re-Key start

The key server must be available to process the rekey request. If it is not available, the error message that is shown in Figure 5-4 is displayed.

Figure 5-4 Server rekey error message

If the key server is available, it creates a key. The `Completed Successfully` confirmation message looks like the example in Figure 5-5.

Figure 5-5 Server Rekey results

3. You can click **Show Results** to see the `Completed Successfully` message, as shown in Figure 5-6.

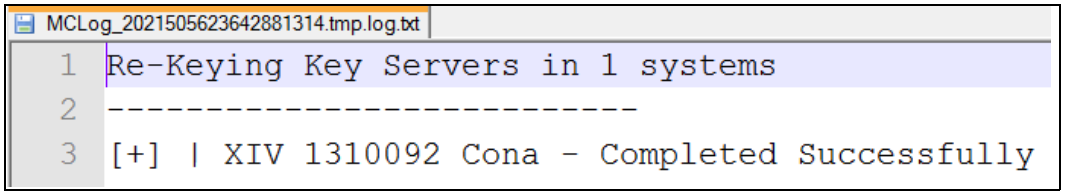

Figure 5-6 Rekey completed successfully

5.3.2 XIV and IBM FlashSystem A9000 or A9000R server rekey by using XCLI

For XIV and IBM FlashSystem A9000 and IBM FlashSystem A9000R systems, the server key can be rekeyed by using the **encrypt_keyserver_rekey** XCLI command, as shown in Table 5-1.

Table 5-1 The encrypt_keyserver_rekey command

Category	Name	Description
System	**encrypt_keyserver_rekey**	Initiates a rekey against the master key server

5.4 Encryption deadlock

The key server platform provides the operating environment on which the key server application runs, accesses its keystore on persistent storage. It also interfaces with client storage devices, such as the XIV or IBM FlashSystem A9000 and IBM FlashSystem A9000R systems, that require key server services.

The keystore data is accessed by the key server application through a password that is specified by the customer. As such, the keystore data is encrypted at rest, independently from 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 cannot 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 plus any other data that is required by that operating system and its associated software stack to run the key server application. This action is necessary to allow the key server 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 a storage device that has a dependency on a key server to access data.

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

Important (encryption deadlock): Any data that is required to make the IBM Security 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 to 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 encryption deadlock can be temporary or permanent:

Temporary encryption deadlock The temporary encryption deadlock indicates a situation where the XIV or IBM FlashSystem A9000 or A9000R system cannot access its disk devices because IBM Security Key Lifecycle Manager servers are not online, the network is down, or there are other temporary hardware-related errors. This temporary failure can be fixed at the client site.

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

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

- As the availability of encryption-capable devices becomes more pervasive, 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 that is required by the key server from non-encrypted to encrypted storage.
- Generally, several layers of virtualization in the I/O stack hierarchy can cause difficulties for the client in maintaining awareness of where all of the files that are necessary to make the key server and its associated keystore available are stored. The key server can access its data through a database that runs on a file system that runs on a logical volume manager. The volume manager communicates with a storage subsystem that provisions logical volumes with capacity that is obtained from other subordinate storage arrays. The data that is 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 by.
- 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 IBM Security Key Lifecycle Manager server is mandatory, and the client must be directly involved in managing the encryption environment. For more information about this topic, see Chapter 2, “Planning” on page 7, Chapter 3, “Implementing encryption on XIV” on page 21, and Chapter 4, “Implementing encryption on IBM FlashSystem A9000 and A9000R” on page 57.

5.5 Component replacements

The following scenarios are considered:

- XIV disk and module replacement

  If a disk or multiple disks must be replaced, the new disks must be unlocked upon power-on and contacting the key server to get the Disk Access Key (DAK). If a disk drive that does not support encryption is added to an encryption-enabled XIV system, it fails the component test and cannot be included in the running XIV configuration. This action ensures that no unencrypted data is on any disk drives inside the XIV system.

- IBM FlashSystem A9000 and A9000R MicroLatency module and SSD replacement

  If MicroLatency modules or vaulting SSDs must be replaced, the new parts must be encryption-capable, just like the original parts that are being replaced:

  – Preferred practice when using the local encryption key server is to do a rekey operation whenever any grid server, vaulting SSD, or boot device has been replaced.

  – Because the MicroLatency Cards have on-board hardware encryption, a rekey is not as important if any of them need to be replaced.

Related publications

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

IBM Redbooks

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

- *IBM FlashSystem A9000 and IBM FlashSystem A9000R Architecture and Implementation*, SG24-8345
- *IBM XIV Storage System Architecture and Implementation*, SG24-7659

You can search for, view, download, or order these documents and other Redbooks, Redpapers, web docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications and online resources

These publications are also relevant as further information sources:

- IBM Fix Central:

  http://www.ibm.com/support/fixcentral/

- IBM FlashSystem A9000 on IBM Knowledge Center:

  http://www.ibm.com/support/knowledgecenter/STJKMM

  The following publications are at this website:

  - *IBM FlashSystem A9000 Command-Line Interface (CLI) Reference Guide*, SC27-8559
  - *IBM FlashSystem A9000 Models 9836-415 and 9838-415 Deployment Guide*, GC27-8564
  - *IBM FlashSystem A9000 Product Overview,* GC27-8583-00
  - *Hyper-Scale Manager 5.0 REST API Specification*, SC27-6440-01
  - *Hyper-Scale Manager 5.0 User Guide*, SC27-8560

- IBM FlashSystem A9000R on IBM Knowledge Center:

  http://www.ibm.com/support/knowledgecenter/STJKN5

  The following publications are at this website:

  - *IBM FlashSystem A9000R Command-Line Interface (CLI) Reference Guide*, SC27-8711
  - *IBM FlashSystem A9000R Model 415 Deployment Guide*, GC27-8565
  - *IBM FlashSystem A9000R Product Overview,* GC27-8558-00

- IBM FlashSystem A9000 product page:

  http://www.ibm.com/systems/storage/flash/a9000
- IBM FlashSystem A9000R product page:

  https://www.ibm.com/systems/storage/flash/a9000r/
- IBM Offering Information page (announcement letters and sales manuals):

  http://www.ibm.com/common/ssi/index.wss?request_locale=en

  On this page, enter A9000, select the information type, and click **Search**. On the next page, narrow your search results by geography and language.
- IBM System Storage Interoperation Center (SSIC) website:

  http://www.ibm.com/systems/support/storage/ssic/interoperability.wss
- *IBM XIV Storage System Planning Guide*, GC27-3913
- *IBM XIV Storage System: Product Overview*, GC27-3912
- *IBM XIV Storage System User Manual*, GC27-3914
- *IBM XIV Storage System XCLI Utility User Manual*, GC27-3915

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

REDP-5402-02

ISBN 0738457574

Printed in U.S.A.