IBM Encrypted Storage Overview and Customer Requirements

External Version 1.0b
July 6, 2009

Rick Ripberger
Enterprise Disk Architecture
Dept CSVA/ IBM Tucson

Summary of Changes

The following versions of the document have been released with the indicated changes:

1. External Version 1.0a, April 7, 2009

   The following changes were made:

   - Recovery site was clarified in section “Best Practices for Encrypting Storage Environments”.

   - Introduces IKS as acronym for isolated key server in section “Best Practices for Encrypting Storage Environments”.

2. External Version 1.0b, July 6, 2009

   All changes for this version are in blue.

   The following changes were made:

   - A sentence alluding to a possible future zSeries IKS was removed from section “IBM Guidelines and Requirements for Encrypting Storage Installations”.

Overview

Use of encryption technology has a number of considerations that are critical for the user to understand in order to maintain the security and accessibility of encrypted data. This document is intended to contain critical information that you will need to know in order to manage IBM encrypted storage and to comply with IBM requirements for using IBM encrypted storage.

Failure to follow these requirements can result in a *permanent encryption deadlock* which can result in the permanent loss of all key-server-managed encrypted data at all installations at the customer account.

Encryption Concepts

Encryption is a technique used to encode data with an encryption key such that the information content of the data can generally only be decoded with knowledge of a *decryption key*. Data that is encrypted is referred to as *ciphertext*. Data that is not encrypted is referred to as *plaintext* or *cleartext*. With an appropriately derived encryption key and an appropriate encryption algorithm, it is prohibitively difficult to guess the decryption key or otherwise determine the plaintext from the ciphertext through any known mechanism (i.e., It would take a prohibitively long time to attempt enough guesses to find the correct decryption key). As such, data that is encrypted into ciphertext is considered securely secret from anyone who does not have possession of the decryption key.

In the context of electronic data processing systems encryption and decryption keys are (preferably) random numbers of a bit length specified by the chosen encryption algorithm. There are multiple classes of encryption algorithms. For the purpose of this paper, we will describe two algorithm classes: symmetric and asymmetric.

Symmetric encryption algorithms use the same key for both the encryption key and the decryption key. Symmetric algorithms are preferred for encryption of data because they require significantly less compute resource to encode and decode data streams. One issue with symmetric algorithms when being used to transfer data among separate parties is a secure mechanism must be found to send the key to the receiving party.

Asymmetric encryption algorithms use different keys for the encryption key and the decryption key. The encryption key and the decryption key are mathematically related but different for asymmetric algorithms. The two related keys are sometimes referred to as a public/private key pair. Either key can be used to encrypt data but then the other key must be used to decrypt the resulting message to a readable form. Asymmetric encryption algorithms provide a mechanism to securely transfer keys between separate parties.

Security vs. Accessibility

As a result of the nature of encryption, the security of, and accessibility to, data that is encrypted is dependent on the security of, and accessibility to, the decryption key needed to decrypt the data. The disclosure of a decryption key to an unauthorized agent

(individual person or system component) creates a security exposure if that agent also has access to the ciphertext generated with the associated encryption key. If all copies of the decryption key are lost (whether intentionally or accidentally), then there is no feasible way to decrypt the associated ciphertext, and the data contained in the ciphertext is said to have been *cryptographically erased*. If the only copies of some data that exist are cryptographically erased ciphertext, then access to that data has been permanently lost for all practical purposes. The security and accessibility characteristics of encrypted data create considerations for the customer that do not exist with storage devices that do not encrypt data. Encryption key material must be kept secure from disclosure or use by any agent that does not have authority to it; at the same time it must be accessible to any agent that has both the authority and need to use it at the time of need.

- To preserve the security of encryption keys, many techniques can be used in an implementation to ensure that no one agent has access to all the information necessary to determine an encryption key.

  For example, an implementation using only symmetric encryption might manage encryption keys such that the "data" key (used to encrypt/decrypt data) is encrypted (wrapped) with a "wrapping" key (used to encrypt/decrypt "data" keys). In order to decrypt the data ciphertext in this case, the wrapping key is first used to decrypt (unwrap) the data key ciphertext to obtain the data key in plaintext which is then used to decrypt the data ciphertext to obtain the data in plaintext. If one agent stores the wrapping key and a second agent stores the encrypted data key, then neither agent alone has sufficient information to determine the plaintext data key. For the IBM Encrypting Storage implementations that will be discussed subsequently, the agent that stores the wrapping keys will be referred to as a *key server* and the agent that stores or has access to the encrypted data keys will be referred to as a *storage device*.

  This separation of secrets between multiple entities is important because it becomes increasingly difficult to compromise the integrity (security) of a set of entities as the number of entities required increases.

- To preserve the access to encryption keys, many techniques can be used in an implementation to ensure that more than one agent has access to any single piece of information necessary to determine an encryption key.

  For example, redundancy is provided by having multiple independent key servers that have multiple independent communication paths to encrypting storage devices. Additionally, backups of each key server's data are maintained. Failure of any one key server or any one network does not prevent storage devices from obtaining access to data keys required to provide access to data. Redundancy is also provided in the storage device (or on the storage media) by keeping multiple copies of the encrypted data key.

The sensitivity of possessing and maintaining encryption keys as well the complexity of managing the number of encryption keys in a typical environment results in a customer requirement for a *key server*. A key server is integrated with encrypting storage products to resolve most of the security and usability issues associated with key management for encrypted storage. However, the customer must still be sufficiently aware of how these

products interact in order to provide appropriate management of the compute environment. Even with a key server, there is generally at least one encryption key (e.g., the overall key that manages access to all other encryption keys, a key that encrypts the data used by the key server, etc.) that must be maintained manually.

On critical consideration with a key server implementation is that all code and data objects required to make the key server operational must not be stored on storage that is dependent on any key server to be accessed. A situation where all key servers cannot become operational because there is data or code that cannot be accessed without an operational key server is referred to as an *encryption deadlock*. It is analogous to having a bank vault that is unlocked with a combination and the only copy of the combination is locked inside the vault. This situation, as well as policies and mechanisms required to avoid it, will be discussed more fully in subsequent sections.

Tivoli Key Lifecycle Manager (TKLM)

The IBM Tivoli Key Lifecycle Manager (TKLM) is an IBM program product that provides a key server application that integrates with certain IBM storage products. The Tivoli Key Lifecycle Manager (TKLM) program product can be installed on a set of supported servers to implement a set of redundant key servers. Supported server platforms include:

- z/OS V1.9 and V1.10
- AIX 5.3 and 6.1
- Red Hat AS 4.0 x86
- Suse Linux 9.0 and 10 x86
- Solaris 10 Sparc
- Windows Server 2003

Encrypting storage devices that require key services are configured to communicate with one or more key servers (some devices may require at least two key servers). The TKLM application is configured with the devices to which it is authorized to provide key services.

The TKLM uses a wrapped key method to deliver keys to encrypting storage devices. The IBM DS8000 Disk Subsystem and the IBM TS1120 and TS1130 Tape Devices implement this wrapped key method to secure keys by separating the storage of a wrapped data key stored on the media or within the device from the storage of the wrap/unwrap keys within the key server. The wrap/unwrap keys on the TKLM are a public/private asymmetric key pair referred to as the pubic Key Encrypting Key (KEK) and the private Key Encrypting Key (KEK'), respectively. The wrap/unwrap key pair is used to wrap and unwrap data keys for the client storage devices as will be discussed subsequently. The key generation and propagation processes on the TKLM associate a *key label* with each wrap/unwrap key pair. This key label is a user specified text string that is retained with each wrap/unwrap key pair.

When a storage device initiates communication with a TKLM, the storage device sends a request to the TKLM that provides the device's certificate and the device's identification. The TKLM verifies that the device has a valid certificate signed by an appropriate certificate authority and verifies that the TKLM is authorized to talk to the device. The TKLM uses the public key in the device's certificate to authenticate messages sent by the device.

The storage device generates a device session public/private key pair (DSK/DSK') for securely communicating with the TKLM. The private device session key (DSK') is kept secret by the storage device. As described subsequently, the storage device sends the public device session key (DSK) to the TKLM, and the TKLM uses the DSK to wrap data keys (DKs) such that only the storage device can unwrap the wrapped key using the private device session key (DSK'). When transferring the DSK to the TKLM, the storage device signs the DSK with its own device-specific private key to create a signed public session key to the TKLM, the signature authenticating the origin of the public key.

The TKLM maintains a public/private key encrypting key pair (KEK/KEK') for each key label. The public and private key of this key pair is kept secret by the TKLM. As described subsequently, the public key encrypting key (KEK) is used by the TKLM to wrap data keys (DKs) such that only the TKLM can unwrap the wrapped using the private key encrypting key (KEK').

In the wrapped key method, there are two high level functions that an encrypting storage device can initiate with a TKLM key server:

1. Request a new data key

   The storage device requests a new data key. Either the storage device specifies a key label to select the Key Encrypting Key (KEK), or the TKLM uses the default key label specified for this device in the TKLM configuration information to select the Key Encrypting Key (KEK). (The DS8000 always specifies a key label). After authenticating the device and obtaining the public device session key (DSK), the TKLM sends a properly generated 256 bit random data key (DK) to the storage device in two forms:

   a. Externally Encrypted Data Key (EEDK)

      The TKLM wraps the data key (DK) with the selected public wrapping key (KEK) and sends it to the storage device in a structure referred to as the externally encrypted data key (EEDK). This structure also contains sufficient information to determine the KEK used to wrap the data key in the EEDK.

   b. Session Encrypted Data Key (SEDK)

      The TKLM wraps the data key (DK) with the public device session key (DSK) and sends it to the storage device in a structure referred to as the session encrypted data key (SEDK).

   The EEDK is persistently stored by the storage device for future use as described subsequently. The SEDK is decrypted by the storage device using the storage device's private key to obtain the data key (DK). The data key (DK) is then used

to symmetrically encrypt and/or decrypt either data (in the case of tape devices) or other subordinate data keys that are required to encrypt, decrypt, or gain access to data (in the case of DS8000).

2. Unwrap an existing data key

   The storage device requests the TKLM to unwrap an existing wrapped data key by sending the request to the TKLM with the EEDK and the public device session key. After authenticating the device and obtaining the public device session key, the TKLM unwraps the data key (DK) in the EEDK with the private key wrapping key (KEK') specified by the EEDK, wraps the DK with the public device session key, imbeds the wrapped DK in an SEDK structure, and sends both the EEDK and SEDK to the storage device.

   The SEDK is decrypted by the storage device using the private device session key (DSK') to obtain the data key (DK). The data key (DK) is then used to symmetrically encrypt and/or decrypt either data (in the case of tape devices) or to symmetrically wrap and unwrap other subordinate data keys that are required to encrypt, decrypt, or gain access to data (in the case of DS8000).

Security is improved by implementing dual control across two agents. The storage device does not maintain a persistent copy of the data key in the clear. It is unable to encrypt or decrypt data without access to the TKLM.

Different use cases and storage mechanisms are appropriate for different types of storage devices. For instance, the EEDK for a removable media device might be stored on the media when it is initially written and the DK is overwritten to erase it (i.e. it is zeroized) by the storage device when the media is dismounted such that each time the medium is mounted, the storage device must communicate with the TKLM to obtain the DK in order to encrypt write data or decrypt read data on the medium. The EEDK for a non-removable storage device might be stored as persistent metadata within the storage device and the DK zeroized by the storage device when it is powered off such that each time the storage device is powered on, it must communicate with a TKLM to obtain the DK. In all cases where the described TKLM wrapped key model is used, access to data encrypted with a DK requires access to both the EEDK from the medium or storage device and to a TKLM that has the private key wrapping key (KEK) needed to decrypt the EEDK to obtain the DK.

DS8000 Disk Encryption

The IBM DS8000 disk subsystem supports data encryption with the IBM Full Disk Encryption (FDE) drive. These enterprise-class disks are available in 146 GB, 300 GB, or 450 GB capacities and with 15K RPM speed. These drives contain encryption hardware and can perform symmetric encryption and decryption of data at full disk speed with no measurable impact on performance. The encryption algorithm is AES (advanced encryption standard).

To use data encryption, an IBM DS8000 must be ordered from the factory with all FDE drives. At this time, DS8000 does not support intermix of FDE and non-FDE drives in the same storage facility so additional drives added to a DS8000 must be consistent with

the already installed drives. An IBM DS8000 with FDE disks is referred to as being *encryption-capable*. Each storage facility image on an encryption-capable DS8000 can be configured to either enable or disable encryption for all data stored on customer disks.

In order to enable encryption, each DS8000 storage facility image must be configured as follows:

1. Configure at least two and up to four TKLM key servers. The physical connection between the DS8000 HMC and the key server is through a TCP/IP network. The configuration process specifies the IP addresses of key servers.
2. Configure an encryption group on the storage facility image. The configuration operation specifies a key label for the encryption group. The DS80000 uses the key label to request a new data key (DK) from an attached TKLM for the encryption group when the encryption group is created and stores the EEDK for subsequent use. It also generates a random 256 bit group key (GK) for the encryption group, AES wraps the group key with the DK producing the encrypted group key (EGK), and stores the EGK for future use, and then zeroizes the DK. Both the EEDK and the EGK are stored in multiple places for reliability.
3. After the encryption group is configured, ranks may be created that are associated with a configured encryption group such that data stored on these ranks is encrypted and the rank is said to be encryption enabled. Ranks that are not associated with an encryption group are not encrypted and are said to be encryption disabled. All ranks on a storage facility image are encryption enabled, or all ranks on a storage facility image are encryption disabled. The first rank configured determines how the remaining ranks must be configured. Deleting the last rank on the storage facility image allows the choice of encryption enabled or encryption disabled to be made when the first rank is configured.
4. The customer assigns one or more ranks to one or more extent pools. All ranks an extent pool must be encryption enabled or encryption disabled. The extent pool is set to be encryption enabled or encryption disabled based whether the ranks in the extent pool are encryption enabled or encryption disabled.
5. After extent pools are configured, the customer configures logical volumes in each extent pool. Data associated with logical volumes configured in an encryption enabled extent pool is encrypted.

Each IBM FDE drive has an encryption key for the region of the disk that contains customer data. When the customer data region is *locked*, the encryption key for the region is wrapped with an access credential and stored on the disk media. Read/write access to the data on a locked region is blocked following a power loss until the initiator accessing the drive authenticates by supplying the currently active access credential. The access credential assigned to the locked customer data region by the storage facility image is unique to each disk and is derived from the group key (GK) and the disk serial number using a secure hash algorithm.

When the customer data region is *unlocked*, the encryption key for the region is wrapped with a unique access credential that is assigned to this particular disk and stored on the disk media. This access credential is accessible to the device, to any attached initiator,

and is visible on the device external labeling to the customer. Read/write access to the data on an unlocked region does not require an access credential or any interface protocols that are not used on a non-FDE disk. The FDE disk still encrypts/decrypts data with an encryption key, but it does so transparently to the initiator.

On DS8000, an IBM FDE disk that is a member of an encryption-enabled rank is locked. An IBM FDE disk that is unassigned, is a spare, or is a member of an encryption-disabled rank is unlocked. Locking occurs when a FDE disk is added to an encryption-enabled rank (either at rank creation or during sparing). Unlocking occurs when an encryption-enabled rank is deleted or when an encryption-enabled rank member is re-purposed as a spare. Unlocking always entails a cryptographic erasure of an IBM FDE disk. IBM FDE disks are also cryptographically erased when an encryption-disabled rank is deleted. The customer may cryptographically erase the data for a set of logical volumes in an encryption-capable extent pool by deleting all the ranks associated with that extent pool.

IBM FDE disks are NOT cryptographically erased when the disk fails. In this case, there is no guarantee that the device-adapter can communicate with the disk, and more specifically, the device adapter intentionally fences the failing drive from the device interface as soon as possible to prevent it from causing any other problems on the interface.

Once an encryption group is configured, additional interaction with the TKLM is not required to allow access to data except when the storage facility image powers on. The DS8000 must be able to communicate with a TKLM after a power on in order to allow access to the disks that have encryption enabled. In this case, DS8000 asks the TKLM to unwrap the EEDK, the TKLM returns an SEDK, the DS8000 unwraps the SEDK to obtain the DK, the DS8000 unwraps the EGK with the DK to obtain the GK, the DS8000 uses the GK to generate the access credentials to authenticate with the FDE disks in the encryption group, the FDE disks in the encryption group use their access credential to unwrap the encryption/decryption key used on customer data in the customer data band. Without access to the TKLM, the data at rest on any locked FDE disks is secure.

In the current implementation of an encryption-capable DS8000, any customer data is persistently stored in one of three places:

1. On customer disks

   Customer data on customer disks (i.e. DDM installed via DDM Install Group features) that are members of an encryption-enabled rank is managed through a data key obtained from the TKLM key server. As such the data is encrypted with an encryption key that is managed though an externally encrypted encryption key. As such, this data is **encrypted** and is considered secure.

   Customer data on customer disks that are members of a rank that is not encryption-enabled is encrypted with an encryption key that is encrypted with a derived key and stored on the disk. As such, this data is **obfuscated** on the medium, but since the drive allows any initiator to access the medium without requiring an access credential, it is effectively plaintext and is not considered secure.

2. NVS dump data on system disks

   If a force power off sequence is initiated, write data in flight in the NVS memory is encrypted with an encryption key and stored on the system disks in the DS8000 server. The data is limited to at most 8 GBs. The encryption key is encrypted with a derived key and stored on the system disk. As such, this data is only **obfuscated** and is not considered secure. The data on the system disk is cryptographically erased after power is restored once the data has been restored to the NVS memory during the initial microcode load (IML).

3. APU dump data in device adapter flash memories

   If a force power off sequence is initiated, atomic parity write data in flight in the device adapter memory for RAID 6 arrays is encrypted with an encryption key and stored in a flash memory on the device adapter card in the DS8000 server. The data is limited to at most 32 MB per device adapter or 512 MB per storage facility in a maximum configuration. The encryption key is a derived key. As such, this data is only **obfuscated** and is not considered secure.

Note: The power off requests issued through the DS GUI/CLI interfaces or through the zSeries power control interfaces do not initiate a force power off sequence. Activation of the Force Power Off service switch or loss of AC power does initiate a force power off sequence.

IBM Tape Encryption

Most IBM Ultrium 4 LTO Tape Drives (FC, SAS, and Ultra-320 parallel SCSI) are encryption-capable. IBM Ultra-160 parallel SCSI LTO-4 drives are not encryption-capable. Although LTO-4 drives can read LTO-2 cartridges, and can read/write LTO-3 cartridges, the LTO-4 drives will only do encrypting writes to, and decrypting reads from LTO-4 cartridges consistent with the LTO Consortium specification.

All IBM TS1130 Tape Drives are encryption-capable. All IBM TS1120 Tape Drives with Feature Code 5592 or 9592 are encryption-capable. This means that they are functionally capable of performing hardware encryption, but this capability must be encryption-enabled in order to perform hardware encryption. In an IBM System Storage TS3500 Tape Library, TS1130 and encryption-capable TS1120 Tape Drives can be encryption-enabled through the IBM System Storage Tape Specialist. For all other TS1130 and encryption-capable TS1120 Tape Drives this process typically consists of a storage administrator making configuration changes such that the drive will encrypt and decrypt. Only encryption-enabled TS1120 Tape Drives can be used to do decrypting reads from or encrypting writes to 3592 tape cartridges.

Encryption can involve the use of several kinds of keys for different purposes. How these keys are generated, maintained, controlled, and transmitted depends upon the operating

environment where the TS1120, TS1130, or Ultrium 4 Tape Drive is installed. Some applications are capable of performing key management. For environments that do not consist entirely of such applications, or where application key management is undesirable, IBM provides either the IBM Encryption Key Manager (EKM) component or the Tivoli Key Lifecycle Manager (TKLM) program product to perform all necessary key management tasks.

When encryption-enabled, these tape devices can interface with a key server to obtain new encryption keys for a tape cartridge that is written from scratch. In the case of TS1120 and TS1130, two copies of the encryption key are stored on the tape medium in the form of two EEDKs, each of which is wrapped with a different wrapping key on the key server. The second EEDK provides a mechanism for the customer to export an encrypted tape medium to external environments without revealing the local secret private key needed to unwrap the first EEDK. When an existing encrypted tape is mounted, the tape device reads the wrapped key from the tape medium and requests the TKLM or EKM to unwrap the key to allow the tape device to read/write data on the tape cartridge.

Encryption Deadlock

As previously stated, a situation where all key servers cannot become operational because there is data or code that cannot be accessed without an operational key server is referred to as an *encryption deadlock*. The critical consideration with a key server implementation for storage devices is that all data and code objects required to make key servers operational *must not* be stored on storage that is dependent on any key server to be accessed.

If this constraint is not met, the key servers may not be able to complete their initial program load (IPL) and become operational or may not be able to provide key services to their storage clients. The required code and data objects includes not only the boot image for the operating system that runs on the key server, but also any other data required by that operating system and their associated software stacks to run the key server application, to allow the key sever to access its key store, and to allow the key server to communicate with its storage device clients. Similarly, any backups of the key store must not be stored on storage that has a dependency on a key server to access data.

While an encryption deadlock exists, the customer is unable to access any encrypted data on the encrypted storage devices. If all possible key server instances are in encryption deadlock and, additionally, all backups of the key store also happen to be stored on encrypting storage that is dependent on a key server, the encryption deadlock can become a permanent encryption deadlock such that all encrypted data managed by the key servers is permanently lost. Permanent encryption deadlock could result in failure of the business.

When IBM delivered encrypting tape technology, the product documentation advised the customer against storing key server and key store backups on encrypted tapes. Customers were specifically advised to make their backups of the key server environment without invoking the encryption capability of the tape device in order to avoid encountering an

encryption deadlock condition during disaster recovery. With the introduction of encrypting disks to the environment, the risk of an encryption deadlock increases significantly due to the following factors:

- The various layers of virtualization in the I/O stack hierarchy make it difficult to maintain awareness of where all the files necessary to make the key server, and its associated key store, available are stored. The key server may access its data through a database that runs on a file system that runs on a logical volume manager which communicates with a storage subsystem that provisions logical volumes with capacity obtained from other subordinate storage arrays. The data required by the key server may end up provisioned over various storage devices, each of which may be independently encryption-capable or encryption-enabled.
- Various layers within this I/O stack hierarchy may provide for transparent data relocation either autonomically or as a result of customer-initiated operations.
- As the availability of encryption-capable devices becomes pervasive and storage of customer data on encrypted storage becomes a general requirement, more and more data will be migrated from non-encrypted storage to encrypted storage. In the course of such an extensive migration activity, it is possible that a storage administrator might accidentally migrate some data required by the key server from non-encrypted to encrypted storage.
- Consolidation of servers and storage tends to drive data migration and tends to move more and more data under a generalized shared storage environment which will tend to be encryption-capable as time goes on.
- The ability to detect that a single key server's data access has been compromised cannot be absolutely detected except by power cycling the entire environment with only that key server activated. With multiple key servers activated, it is not possible to detect that all key servers except one have been dependent on that one key server's access to data. As such, it is possible for a set of independent key servers to gradually become interdependent over a long period of time as the result of successive inappropriate data migrations. Such only, a single additional change is required to compromise the access of the last independent key server and to enable an encryption deadlock. Furthermore, the act of power cycling the entire environment results in an encryption deadlock if all key servers' data access has been compromised.
- All IBM server platforms support fabric-attached boot devices and storage. Some IBM servers do not support internal boot devices. It is common for boot devices to be present within the generalized storage environment and accessible to generalized storage management tools that support data management and relocation.

To reduce the risk of encountering an encryption deadlock, the customer must exercise a very high standard of care in managing the encryption environment such that he assures that an encryption deadlock will not occur.

Best Practices for Encrypting Storage Environments

The following information is not intended to be comprehensive, but may include some key techniques for mitigating the risk of an encryption deadlock:

ITEMS RELATED TO SECURITY ONLY:

General:

- The customer should manage the physical security of access to hardware in general. In particular, the customer may want to provide additional physical security around the hardware, network, and media components that comprise the key servers.

Key Store Related:

- The initiation of a TKLM key server involves the specification of a password that is used to access the key store. The customer must decide whether the TKLM password must be provided by a human manually or whether there is some mechanism to automatically provide the password to the TKLM. If a startup script is used on the TKLM server that contains the password, the script file should have access controls to prevent unauthorized access to the file and password (e.g. file permissions that prevent read, write, or execute access by unauthorized personnel).
- Maximum security is achieved when the Key Material is stored securely using the services of a Hardware Secure Module (HSM) and the master key for that module is formally managed, and only distributed in split key form, via key ceremonies. At present, IBM provides the capability for customers to realize this level of security on System z servers using ICSF and the CryptoExpress 2 HSM at this time.

ITEMS RELATED TO AVAILABILITY:

Key Server Related:

- Redundant key servers must be configured to each encrypting storage device. Each independent customer recovery site must have independent and redundant key servers. TKLM supports secure network connectivity for delivery of keys across a client WANs; however each customer site should be evaluated to assure sufficient local availability. Many situations may require more than two local, redundant key servers to achieve RTO/RPO objectives for the site.
- For sites with requirements to initiate key server operation after power on without human intervention, key servers must be configured to automatically power on when power is available and to automatically initiate the key server application. In this case, the key server application should be configured to automatically boot.
- The configuration of redundant network fabrics between key servers and encrypting storage should also be evaluated with respect to the availability requirements of the implementation. Most storage products support two or more network connections. Providing independent network paths through independent to independent key servers improves robustness.

DS8000 Related:

- The DS8000 should be configured with the dual HMC option to provide redundant access to the customer network. Dual HMCs can be provided by cross-coupling the HMCs on two DS8000s or by providing an additional standalone HMC for a single DS8000. The inability of a DS8000 to communicate with a key server when it powers on will prevent access to encrypted storage on the DS8000.

ITEMS RELATED TO ENCRYPTION DEADLOCK PREVENTION:

General:

- The information in this document should be reviewed, understood, and appropriately incorporated into requirements documents and into design, system, and process implementations.
- All personnel who have any of the following assignments or capabilities should be required to review a customer document which describes these risks and the processes adopted to mitigate them at least annually.
    - Responsibility for the implementation of TKLM key servers or encrypted storage products.
    - Responsibility to manage the placement or relocation of data related to, or required by, any TKLM key server.
    - Access authority to configure TKLM key servers or encrypted storage products.
- Customers should thoroughly review and update systems management processes, especially configuration and change management processes, to ensure adherence to guidelines required to ensure proper configuration and isolation of key servers, configuration and utilization of encrypted storage, and placement of code and data objects related to key servers.
- The customer should implement automated monitoring of the availability of any equipment associated with management of key services and take appropriate action to keep them operational. This equipment would include but is not limited to key servers, SNMP masters, domain name servers, and DS8000 HMCs.
- The customer should pay particular attention to disaster recovery plans and scenarios and consider the availability of key servers, key server backups, and key server synchronization. It is desirable to establish the independence of each recovery site from the other recovery site.
- Isolation of network paths to remote key servers in the context of a site power cycle is one way to test that the key servers at that site are not encryption deadlocked within that site. If such a test is performed, it may be helpful to attempt the power cycle with the isolated key servers (see subsequent information) offline to verify that the un-isolated key servers are not encrypted deadlocked.

Key Server Related:

- The configuration of at least two redundant key servers is required. Redundancy implies that the key servers have no common single points of failure and would typically mean that they are implemented on independent servers and independent storage devices. For key servers operating in LPARs, do not use data sharing techniques that result in one copy of the data being shared by independent key servers.
- The configuration of at least one *isolated key server* (IKS) per recovery site is required. An isolated key server is a dedicated set of server resources running, only the TKLM application and its associated software stack, that are directly attached (i.e. no switches) to dedicated non-encrypting storage resources containing only TKLM key server code and data objects. A recovery site is any site that must operate independently of all other sites. The objective of this requirement is to avoid encryption deadlock by:
    - Implementing a key server environment that is independent of all non-key server applications so that management of the key server can be restricted to those personnel specifically authorized to manage key servers.
    - Implementing a key server that is physically and logically isolated from other applications that may require access to encrypting storage so that the key server environment does not need to be configured with access to any encrypting storage.
    - Implementing a key server that is physically and logically isolated from encrypting storage so that the risk storing (initially or through data migration) code and data objects required by the key server on encrypting storage is eliminated.
    - Ensuring that a recovery site can operate independently of any other sites by configuring a key server that is not subject to encryption deadlock due to the characteristics of an isolated key server.

  The simplest approach to eliminating the possibility that a virtualization layer, data migration tool, or storage administrator accidentally moves required objects to an encrypting storage component (creating a potential deadlock situation) is to create an Isolated Key Server on a server with only internally attached, non-encrypting disks. The currently supported set of isolated key servers of this type supported by TKLM is defined in section “IBM Guidelines and Requirements for Encrypting Storage Installations”.

  Note: IBM DS8000 requires at least one isolated key server be configured, but recommends two for redundancy.
- Configuration of additional key servers on generalized server hardware and generalized storage is allowed, but appropriate procedures and controls should be established to prevent these key servers from having their data access compromised by storing the data on key server managed encrypting storage. In the subsequent discussion, these key servers are referred to as *general key servers*.

- Configuration of key servers at independent sites is recommended and provides additional immunity to encryption deadlocks since it reduces the probability that all key servers will experience a simultaneous power loss.
- The utilization of uninterruptible power supplies (UPSs) on certain key servers may provide additional immunity to an encryption deadlock. (Note: The isolated servers should be already safe from deadlock, so the un-isolated servers are the candidates to consider here).
- The initiation of a TKLM key server involves the specification of a password that is used to access the key store. The customer should ensure appropriate retention of the password as well as limit access to the password to appropriate personnel. Loss of a password is a cryptographic erasure of the key store for the associated key server(s). Loss of one or more redundant key server increases the probability of an encryption deadlock. The permanent loss of all encryption key servers is equivalent to a permanent encryption deadlock.
- The customer must ensure that all key servers that a given storage device is configured to request key services from have current and consistent key store content relative to any wrapping keys that will be required by the storage device. Any wrapping key that will be used by a device must be propagated across the set of key servers that service the storage device before the storage device is configured to use the wrapping keys. Failure to synchronize the key stores effectively eliminates one or more key servers from the set of redundant key servers for a device that uses the keys that are not synchronized.
- The customer should backup key server data after it is updated. The backups should not be stored on encrypted storage media that is dependent on a key server.
- The customer should periodically audit that all online and backup data required to make each key server operational is stored on storage or media that is not dependent on a key server to access the data.
- The customer should not delete keys on the key server under normal circumstances. The appropriate action to remove a key from a key server is almost always to archive the key. If the wrong key is inadvertently archived causing the loss of access to encrypted data at some point in the future, the archive action allows the key to be restored from the archive (provide the archive remains accessible). Deletion of all copies of a key is a cryptographic erase of all encrypted data that is encrypted under this key.

DS8000 Related:

- Since they are generally only configured once on the TKLM and there are generally not that many DS8000s in an installation, it is suggested that the customer manually configure DS8000 devices on the TKLM key server. The option to automatically configure them may be used, but increases the risk that an unauthorized DS8000 might gain access to a key server.

- Each DS8000 storage facility image should be assigned an independent key encrypting key (KEK) with a unique key label on the TKLM. This facilitates independent management of each storage facility image.
- The DS8000 supports up to four key server ports. IBM requires that at least one of the key server ports be assigned to an isolated key server. IBM suggests that two of the key server ports be assigned to isolated key servers. The remaining ports can be connected to general key servers. Key servers at the local site should generally be preferred over key servers at a remote site to improve availability (e.g. each site needs to be sufficiently locally robust to meet customer requirements in the event of a remote site or network connectivity failure, local robustness implies at least two local key servers, one of which should be an isolated key server).
- The DS8000 verifies that at least two key servers are configured on and accessible to the DS8000 when the DS8000 is configured to enable encryption. The configuration request is rejected if this condition is not met.
- If encryption has not been activated on the DS8000, the DS8000 will reject the configuration of ranks and extent pools with a non-zero encryption group specified. See section “IBM Guidelines and Requirements for Encrypted Storage Installations” for more information on encryption activation.
- The DS8000 will monitor all configured key servers. Customer notification is provided for loss of access to key servers and other key server related errors through DS8000 customer notification mechanism (SNMP traps and/or email, when configured). The customer should set up monitoring for these indications and take corrective action when a condition is detected which reflects a degraded key server environment. The following conditions are monitored and reported:
    - If, at power on, the DS8000 cannot obtain a required unwrapped data key for a configured encryption group from a key server, it reports an error condition to the customer and to IBM. In this case, encrypted logical volumes associated with the encryption group are inaccessible to attached hosts. If subsequent to reporting this error, the DS8000 is able to obtain the required key services from a key server, it reports the condition to the customer and to IBM and makes the associated logical volume accessible.
    - DS8000 access to each configured key servers is verified at five minute intervals. Loss of access is reported to the customer.
    - The ability of each key server to unwrap data keys configured on the DS8000 is verified at 8 hour intervals. Loss of the ability unwrap a configured data key is reported to the customer and to IBM.
    - The DS8000 detects if there are fewer than two key servers configured, or fewer than two key servers that are available, or there are fewer than two key servers that can unwrap data keys configured on the DS8000 at 8 hour intervals. If detected, this condition is reported to the customer and to IBM.

Tape Related:

- The validation of key store backups to assure they are not being encrypted is advised. Validation can be performed by reading the backup through a storage device that has been confirmed as being not encryption capable or as having no access to a key manager.

IBM Guidelines and Requirements for Encrypting Storage Installations

IBM has the following requirements relative to the use of DS8000 encrypting storage:

- IBM defines an isolated key server as the following equipment that can be ordered through eConfig:
    1) IBM System L5420
        - Quad-core Intel Xeon Processor X5420 (2.5 GHz / 12 MB L2 / 1.0 GHz FSB / 50 W)
        - 6 GB Memory:
        - 146 GB SAS RAID 1 Storage:
        - SUSE Linux 9.0 - 32 bit. (v10 or v9):
        - Dual Gigabit Ethernet ports (Standard)
        - Power Supply
    2) Tivoli Key Lifecycle Manager v1
        - Includes DB2 9.1 FB4

    No other hardware or software is allowed on this server. An isolated server must only use internal disk for all files necessary to boot and have the TKLM key server become operational.

    Note: This is the same hardware platform used for SSPC with a different software load installed by manufacturing.

    Note: Other isolated key servers may be added to the list of supported isolated key servers.
- IBM requires the use of at least one isolated key server per site with a DS8000 that is encryption enabled. This key server can be configured to serve keys to any TKLM supported device, including IBM tape.
- IBM requires at least one isolated key server to be configured to each DS8000 that is encryption enabled.
- IBM DS8000 requires at least two key servers to be configured to each DS8000 that is encryption enabled.

- To use encryption on a DS8000, the customer must complete a process (described subsequently) for using encryption on each DS8000 storage facility image (SFI). Once the customer has completed the process, IBM will enable the encryption function on a SFI for the customer.
- The ordering, installation, and encryption activation of an encryption-capable DS8000 involves the following steps:
    1) The customer signs the ICA Attachment for IBM Encrypting Storage and selects one of the following:
        a. The customer agrees to contract with IBM Lab Services to provide customer education and to setup the configuration of key servers configured with the DS8000 and to maintain the configuration in compliance with IBM guidelines for encrypting storage. At the completion of the setup, IBM Lab Services enables encryption of the storage facility images on the machine.
        b. The customer agrees to setup the configuration of key servers configured with the DS8000 and to maintain the configuration in compliance with IBM guidelines for encrypting storage. At the completion of the setup, IBM enables encryption of the storage facility images on the machine.
    2) The IBM sales representative or IBM business partner places a customer order for a DS8000 from IBM with encryption-capable DDMs and returns the signed ICA Attachment to IBM.
    3) IBM files the ICA Attachment.
    4) IBM delivers the DS8000 and the IBM CE installs the DS8000.
    5) IBM Lab Services or the customer configures the key servers to be used with the DS8000.
    6) IBM Lab Services or the customer configures the TKLM program product to add the DS8000 SFI(s) to the device table and configures a key-label for the DS8000 SFI(s).
    7) IBM Lab Services or the customer configures the DS8000 with the IP addresses of the associated key server ports.
    8) IBM Lab Services or the customer configures an encryption group on the DS8000 SFI(s) with a key-label defined on the TKLM.
    9) IBM Lab Services or the customer requests encryption enablement for the DS8000.
    10) IBM provides an Encryption Authorization LIC feature key(s) to the customer for the storage facility image(s) on the machine. Each LIC feature key is unique to the SFI it is generated for.
    11) IBM Lab Services or the customer installs the LIC feature key on the SFI.

12) The account may now configure ranks or extent pools for the configured encryption group.

    Note: All ranks and extent pools on a given encryption-capable DS8000 SFI must be configured with the same encryption group attribute. The first rank or encryption group configured determines what the remaining objects must be configured with. A value of 0 indicates encryption-disabled. A value of 1 indicates encryption enabled.

    Note: In order to change between encryption-enabled and encryption-disabled, all ranks and extent pools must be deconfigured. Deconfiguring an encryption-enabled rank causes any data that was stored on the rank to be cryptographically erased and, subsequently, overwritten to re-initialize the rank.

13) When a machine is discontinued, the customer must request IBM to perform a box discontinue action. This action will cause CE to disable encryption on all storage facility images on the storage facility by installing a disablement feature key for encryption. Subsequent use of the box will require the procedure to enable encryption to be performed again. In the event that this process is not performed, the customer assumes responsibility for any liability for any misuse of the box by any subsequent owners.

IBM Guidelines and Requirements for Key Server Management

The customer is responsible for maintaining physical and logical security of key servers.

The customer should backup a key server any time new keys are created which are to be maintained by that key server. The backup should be performed before these new keys are used by any client storage devices (e.g. before there is device is configured to communicate with the key server to request data keys for the associated key label).

The customer is responsible for maintaining synchronization of key stores between key servers and for back up of key store information.

- If the customer provisions more than one type of key server (refer to supported TKLM platforms for available types), at this time, the user must use the TKLM supported secure PKCS12 key export method to transfer keys between heterogeneous key server types. Backup/restore methods can be used between homogeneous key servers.
- Exporting/Importing of keys across heterogeneous TKLM server platforms for DS8000 keys can be accomplished with the following procedure. The customer may want to implement automation to reduce the amount of manual effort involved.

1. Get a list of all the known DS8000 devices from the TKLM. This can be done using the **tklmDeviceList** with the '-type' set to 'DS8K' and with the '-v' option set to 'y'. Here is an example:

   wsadmin>print AdminTask.tklmDeviceList ('[-type DS8K] [–v y]')
   CTGKM0001I Command succeeded.

   Description = salesDivisionDrive
   Serial Number = CCCB31403AFF
   Device uuid = DEVICE-5023fd36-cf2a-4406-80cc-fc2ed4065460
   Device type = DS8K
   World wide name = 61041
   Key alias 1 = certb
   Key alias 2 = certb

2. Using **tklmServedDataList** get the list of all the keys that have been served to all devices.
3. Search the output from step 2 using the Drive serial number gotten from step 1 and find the last occurrence of the Drive serial number in this output. Alias 1 is the key alias that needs to be exported, so write this down. Note: Check the output from step 1 and verify that this alias is the same one associated with the device. If it isn't, then write this alias as well.
4. Repeat 3 until all Drive Serial Numbers are done from step 1.
5. Now use the key alias list from above and for each one run **tklmKeyExport** using a '-type' for 'privatekey'. This will create a file for each key alias. Here is an example:

   wsadmin>print AdminTask.tklmKeyExport ('[-alias certa -fileName mysecretkeys1 -keyStoreName "Tivoli Key Lifecycle Manager Keystore" -type privatekey -keyAlias certa]')

6. FTP these files to the other server where the second TKLM instance is running.
7. At the second TKLM instance, make sure that **ds8k.acceptUnknownDrives** is set to **true** in the TKLM configuration file to allow requests from unknown DS8000 storage images.
8. At the second TKLM instance, use **tklmKeyImport** for each of these files. Note: the password you need to specify is the password that was used for the key store of the TKLM on the machine these files came from.
9. Optionally, add the DS8000 devices listed in step 1 to second TKLM instance using TKLMDeviceAdd.

For more information on the commands see http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.tklm.doc/welcome.htm

IBM DS8000 and IBM TKLM Considerations

The following information may be helpful in using data encryption on DS8000:

- An IBM DS8000 ships from the factory with encryption disabled on each SFI. This is true even if the DS8000 is encryption-capable. The customer must follow the procedure described in section "IBM Guidelines and Requirements for Encrypted Storage Installations" to have IBM activate encryption on each IBM DS8000 SFI.
- An encryption-capable DS8000 may be configured to either enable or disable encryption. Ensure that the desired configuration is achieved before storing data on any configured storage.
- The DS/GUI and DS/CLI must be upgraded to the appropriate level in order to enable encryption on an encryption capable DS8000. Use of a down level GUI or CLI will generally result in a DS8000 configured with encryption disabled.
- CIM support for DS8000 encrypting storage at this time does not support the configuration of TKLM IP ports, encryption groups, encrypting ranks, or encrypting extent pools. A box which has been already configured with encrypting extent pools can use CIM to configure encrypting logical volumes and host attachments for encrypting logical volumes.
- When defining a DS8000 in the TKLM device table, the device identifier is in the format TTTT-PPSSSSS where TTTT is the emulated machine type of the storage facility image, PP is the plant of manufacture of the storage facility image, and SSSSS is the sequence number of the storage facility image. The emulated machine type is 2107 (even though the ordered machine type may be a 2421, 2422, 2423, or 2424). This matches the emulated machine type identification presented to all software systems on DS8000 storage systems. The storage facility image plant of manufacture and sequence number is the same as the storage facility plant of manufacture and sequence number except that the storage facility sequence number ends in 0 and the storage facility image sequence number ends in 1 or 2 for storage facility image 1 or 2, respectively. For example, on a 2421-922-75FA120 storage facility with two storage facility images, the device type identifiers the two storage facility images would be 2107-75FA121 and 2107-75FA122.
- TKLM v1 has a policy input for setting the length of time that key label remains valid (i.e. validity period for new certificate). This input controls the period of time that a key label will support requests for a new data key. It does not prevent any existing data keys created for that key label from being unwrapped. This input is set for each key label as it is created. Since disks typically obtain a new key only once when an encryption group is configured, the expiration of the TKLM certificate is of no significance to on going operation of currently installed and configured encryption groups. It does effect whether a new encryption group can be configured with that key label. The default validity period is 20 years.

- Prior to the shipment of the first fix pack for TKLM v1, TKLM for open systems generates 1,024 bit wrapping keys on all platforms.
- On z/OS 1.9, when using the RACF key store, the RACF key stored does not support 2,048 bit data keys. As such, the TKLM generates 1,024 bit wrapping keys when running on this platform. TKLM key servers running on other platforms can import 1,024 bit wrapping keys, but generates 2,048 bit wrapping keys. In order to support export of keys between Systems z running z/OS 1.9 and other key server platforms (e.g. the isolated key server), key labels must be created on the z/OS platform and exported to the other platforms.
- On z/OS 1.10, when using the RACF key store, the RACF key store supports 2,048 bit data keys. As such, there is no limitation on which TKLM platform is used to create key labels.
- On z/OS when using the ICSF key store, the ICSF key store supports 2,048 bit data keys. The ICSF "Clear Key" mode must be selected so that key labels can be exported from System z to the isolated key server. In this case, key labels can be created on any platform and exported to the other platforms. ICSF "Secure Key" mode cannot be used in conjunction with the heterogeneous platforms required to support an isolated key server.
- The script described in section IBM Guidelines and Requirements for Key Server Management only exports IBM DS8000 key labels and does not export tape key labels. The customer must ensure that there are sufficient redundant key servers to support encrypting tape devices.

The following hardware features are related to DS8000 Encryption:

- IBM DS8000 FDE Disk Hardware Features
  - 146, 300, 450 GB / 15 RKM / 16 DDM Install Groups
    (No factory intermix of FDE and Non-FDE disks)
  - Field MES for additional FDE disks on DS8000 shipped from factory with all FDE disks
- IBM DS8000 Redundant (External) HMC
- IBM DS8000 Billable Service Action to Set Up Dual HMC Configuration
- IBM DS8000 Billable Service Action to discontinue box and disable encryption
- IBM DS8000 Billable Service Action to perform box secure erase
- Hardware Platform For Isolated Key Server
  - IBM System L5420 ordered to configure the subsequent software.
  - SUSE Operating system
  - TKLM Program Product