Managing the Sun StorageTek T10000 Tape Drive Encryption Solution

Best Practice
June 2007

Revision: 1.3

We welcome your feedback. Please contact Open Systems Customer Emulation Test at:
martha.sammartano@sun.com
greg.drobish@sun.com
brett.hesterberg@sun.com
warner.hersey@sun.com
or
STK Test Engineering
Sun Microsystems, Inc.
One StorageTek Drive
Louisville, CO 80028-9172
USA

Contents

Change History

Document Description	
Document owner	Martha Sammartano–-Senior Systems Integration Engineer
Organization	Tape Engineering Storage Test—Open Systems Customer Emulation Test

Revision	**Date**	**Description**
V1.0	10/31/2006	Initial draft
V1.1	11/10/2006	Final version incorporating reviewers' comments
V1.2	11/17/2006	Revised to include changes from Engineering Added step in Restoring Database to Different KMS section Clarified Item15 in Limitations and Constraints section
V1.3	06/27/2007	Added information pertaining to KMS 1.2 feature enhancements

Audience

Classified as Sun Confidential, this documentation is intended for Sun StorageTek employees.

It can be distributed to field personnel, partners, and customers on a case–by–case basis—such as those interested in a Sun StorageTek encryption solution.

Introduction

Data loss events in the past year have resulted in high-profile exposure for many large corporations and have reinforced the need to secure data. Whether the loss is due to misplaced tape cartridges, stolen computers or outright theft of information, today's enterprise must secure its data.

The Sun StorageTek T10000 tape drive with its encrypting capabilities

> Meets the needs of those customers that want to secure tape-resident data from unauthorized readers.
>
> Meets the needs of the most security conscious governmental organizations in the world. The total solution is one that emphasizes data encryption and physical security and enforces security business practices.

The solution is composed of T10000 encrypting tape drives, key tokens and a Key Management Station (KMS). These components all work together to create a secure system that is also high-performing and reliable.

The product documents listed in the References section present the basics of installation and operation of the components of the T10000 encryption solution.

Note: This paper assumes the reader is familiar with the content of those documents and purposely does not include that material except as needed for emphasis or as context for new material.

Successful deployment of the T10000 encryption solution depends on the development of a key management strategy that is easy to use and flexible enough to support changing business requirements.

The focus of this paper is a discussion of what is involved in designing, implementing and maintaining a successful key management system. It also presents advanced operational information, error scenarios and recovery procedures, and product limitations and constraints that must be considered when developing and implementing a key management system.

Chapter 1: A Key Management Strategy

The security model employed by the StorageTek T10000 version 1 encrypting solution is designed to meet the most stringent security requirements. The sophisticated algorithm employed by this solution protects encrypted data from unauthorized access as long as the physical security of the data center is not breached and encryption keys are kept secure. The goal is to architect an implementation of the solution that guarantees authorized users ready access to encrypted data and is also simple to manage and easy to modify or extend to meet changing business requirements.

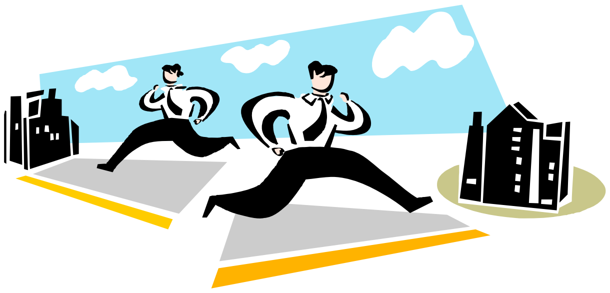
The first step is to determine the key management strategy that your enterprise will employ. Then we will walk you through the process of establishing, implementing and managing your encryption key strategy, pointing out limitations and restrictions that apply.

New Functionality Enhancements

The KMS 1.2 code release is a significant upgrade that introduces several new features. The most significant change involves increased functionality for multi-site support as well as ease of management and remote accessibility enhancements.

Multi-site support and remote accessibility

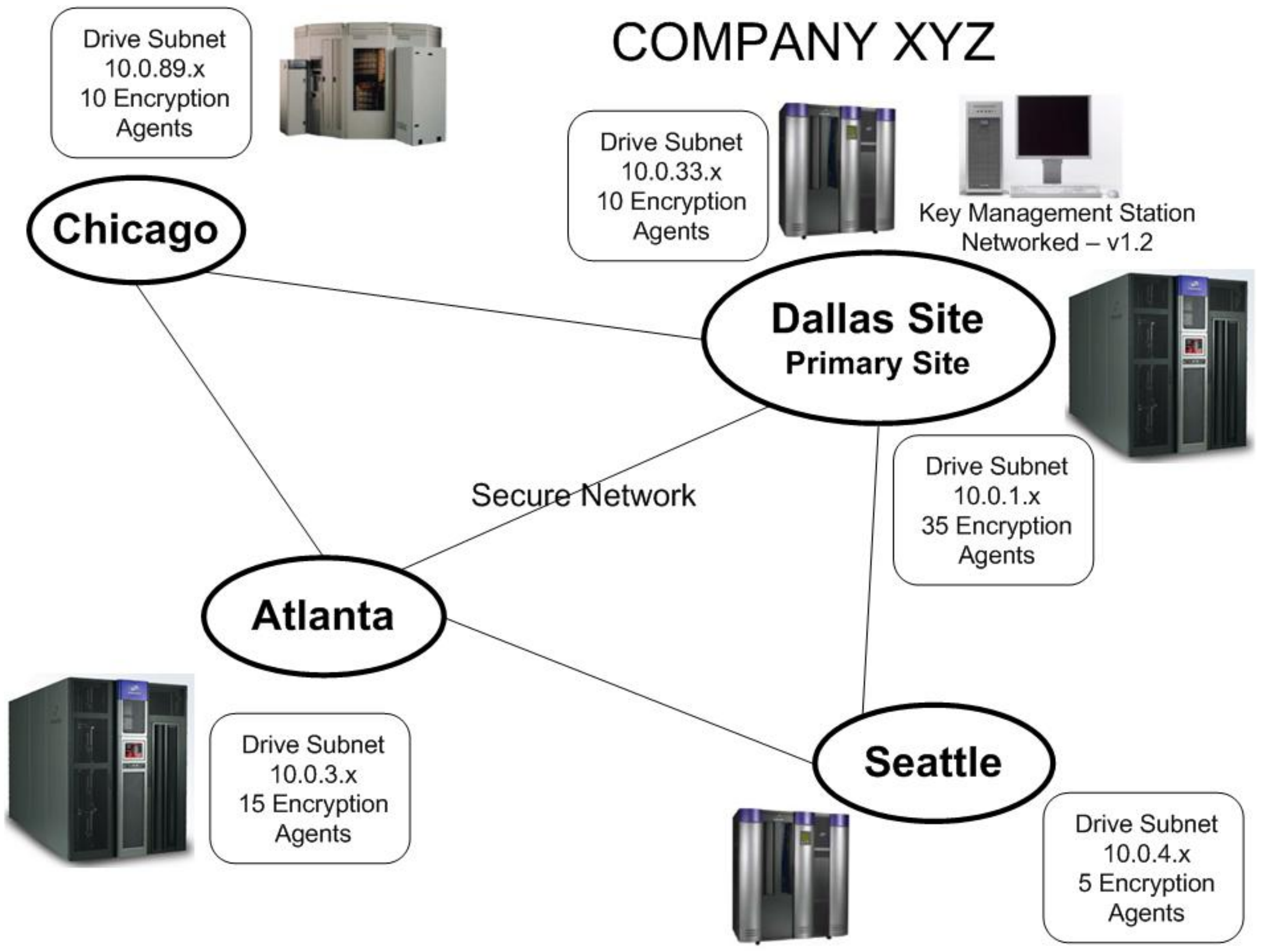
With the introduction of the 1.2 release, the Sun StorageTek Crypto Key Management Station is now capable of running on a secure network and simultaneously managing multiple tokens across several subnets. This enables a centralized KMS to manage encrypted tape drives located on separate subnets in the primary datacenter as well as drives at secondary sites spread around the world without physically moving tokens between locations.



Prior to the 1.2 code release a token exchange was required to change drive assignments, update encryption keys or view error messages. This exchange often involved hand carrying encryption tokens between labs or even between facilities. The multi-site support changes in version 1.2 increases flexibility, ease of management and responsiveness to critical encryption management issues.

The illustration on the following page shows how a fictitious company can manage multiple sites from a single networked key management station.

Company XYZ’s networked multi-site key management implementation:


COMPANY XYZ
Drive Subnet 10.0.89.x 10 Encryption Agents
Chicago
Drive Subnet 10.0.33.x 10 Encryption Agents
Key Management Station Networked – v1.2
Dallas Site Primary Site
Drive Subnet 10.0.1.x 35 Encryption Agents
Secure Network
Atlanta
Drive Subnet 10.0.3.x 15 Encryption Agents
Seattle
Drive Subnet 10.0.4.x 5 Encryption Agents

In addition to this, ease of management is further improved with the introduction of secure remote accessibility options for key management functions. If the KMS is operating on a secure network, it can be accessed remotely via HTTPS and SSH.

To access the GUI, users must connect via HTTPS and login using their previously created login and password. All encryption management functions normally available to that user will be available via remote HTTPS GUI sessions.

code release still supports “air-gap” functionality for customers that would prefer to use hand carried tokens. The air-gap solution functions much in the same way that it did in the previous code release with a change in how token address are acquired. Tokens no longer use DHCP to acquire leased network addresses. They instead rely on static assigned address, which are covered later in this chapter.

Determining Key Distribution

- Where will the data be accessed?
- Which locations will share data?
- Which locations share data because of disaster recovery plans and testing?

The answers to these questions drive the distribution of keys, which is central to the key management process. If you must share data, how will you distribute the required keys to the various sites? If you chose to distribute keys by moving tokens from one location to another, you must do so using a secure, trusted mechanism (for example, courier, secure email, etc.), completely separate from the mechanism used to transport tape cartridges between sites. Placing a copy of the keys in any plaintext form along with tape media and sending it off site would defeat the intent of using the T10000 encryption drive. The opposite exposure also exists. Losing or mismanaging keys results in lack of access to data--in effect, data loss. Enhancements to the KMS software now allow the station to reside on a corporate network, increasing token-management possibilities. Tokens at remote facilities can be managed by a local KMS station assuming network connectivity exists. In this scenario, it is imperative that the keys are transferred from the KMS to remote tokens over a trusted network connection.

The first step in devising a key distribution strategy is to list the locations that will be creating data, then to determine which sites will need to read the data. Business rules that govern the distribution of information determine who will need to read the data (for example, disaster recovery plans).

Since the site creating the data will always have some need to read the data (for example, for validation purposes), the system design assures that the same key assigned as a write key implicitly also acts as a read key.

Because sites are physically separated doesn’t mean that they are logically separated. Your company may have data centers around the world. That doesn’t mean that every data center needs to have a unique key to write data. One large multinational organization has decided to assign keys based on geographies. All data centers in Asia-Pacific will share one key, all centers in North America another key, all centers in South America another and all centers in Europe-Africa-Middle East another key. They have 80 data centers world-wide, but they are managing only four unique write keys at any moment in time.

Key Distribution Example

For the purpose of this document, we use a simple environment for illustration. This example uses data centers in Tokyo, San Francisco, New York, Chicago, London, Singapore, Frankfurt, and Sao Paulo. Each data center will have a number of drives, all assigned to one drive pool (for example, the Tokyo drive pool, the Chicago drive pool, etc.). The business rules are:

- Facilities outside the United States send disaster recovery (DR) data to London.
- Facilities inside the United States send DR data to New York.
- New York sends its DR data to London.
- London sends its DR data to New York.
- Facilities receive keys only on a “need to know” basis.
  That is, if a site has no reason to access another site’s data, the first site will not have a copy of the keys in use at the other site.
- Given the hardened secure nature of their data center sites, only one location is apt to experience a disaster at a time.

If we list the logical association of physical locations to data transfer, the key assignment approach becomes more visible.

Originating Site	Receiving / Recovery Site	Write/Read Key Identifier
Tokyo	London	K1
San Francisco	New York	K2
New York	London	K3
Chicago	New York	K4
London	New York	K5
Singapore	London	K6
Frankfurt	London	K7
Sao Paulo	London	K8

Each site will have its own write/read key. In addition, the DR sites will require read keys for the sites for which they are archiving data. New York will need read keys for San Francisco, Chicago and London; London will require read keys for Tokyo, New York, Singapore, Frankfurt and Sao Paulo. Drive pool mappings in use on Day 1 will be:

Location	Write/Read Key	Read Keys
Tokyo	K1	
San Francisco	K2	
New York	K3	K2, K4, K5
Chicago	K4	
London	K5	K1, K3, K6, K7, K8
Singapore	K6	
Frankfurt	K7	
Sao Paulo	K8	

Determining the Frequency of Key Changes

Changing keys is mandatory if a company has reason to believe that its data security approach has been compromised. Good practice dictates that you change keys periodically to guard against a compromised environment and to limit the exposure if one particular key is compromised. The goal is to change keys frequently enough to protect the system but not frequently enough to present a management or data recovery obstacle. We recommend that you generate and load write keys once a year or perhaps every six months, leaving all previous write keys in the drive pool mapping to provide access to previously written data. More frequent key generation may lead to a more complex management environment, operational issues (for example, having to manually load keys to read data), and the potential for data unavailability.

Key Frequency Example

In the interest of ensuring that data remains secure over time, while also constraining the complexity and overhead of key management, the enterprise has instituted the business rule that they will update keys at midnight on the first day of each year.

At the start of each year, each site will receive a new write/read key, keeping its previous write keys as read keys. Each DR site will need all of its own keys plus those in use at sites for which it is archiving data. At the start of Year 5 of operation, the drive pool mappings in our example environment will look like:

Location	Write/Read Key	Read Keys
Tokyo	K33	K1, K9, K17, K25
San Francisco	K34	K2, K10, K18, K26
New York	K35	K2, K3, K4, K5, K10, K11, K12, K13, K18, K19, K20, K21, K26, K27, K28, K29, K34, K36, K37
Chicago	K36	K4, K12, K20, K28
London	K37	K1, K3, K5, K6, K7, K8, K9, K11, K13, K14, K15, K16, K17, K19, K21, K22, K23, K24, K25, K27, K29, K30, K31, K32, K33, K35, K38, K39, K40
Singapore	K38	K6, K14, K22, K30
Frankfurt	K39	K7, K15, K23, K31
Sao Paulo	K40	K8, K16, K24, K32

A total of 40 keys (8 sites * 1 new write key/year * 5 years) will be needed, with DR sites New York and London requiring 20 and 30 keys, respectively, to have read access for all data archived at those sites. Notice that each of the sites will have fewer than 32 keys, which is the maximum number of keys for a drive pool mapping.

Planning Retention Policies

Retention policies in and of themselves are not related to encryption. However, your key management strategy must guarantee that the required keys are present whenever data needs to be read. Since a T10000 encrypting drive can hold only 32 keys (1 write/read, 31 read) at any one time, you must ensure that your retention policies are compatible with your key distribution and change policies.

For example, if your retention period is 5 years and you want to change your write keys every month, after 2 years and 8 months you will have an issue. The number of keys that you will create in the future plus the number of keys that you have used in the past will exceed the capacity of the drives to store keys. When this happens, ensuring access to all encrypted data written by these drives will require careful planning and possibly manual intervention. To avoid this situation, follow the rule of thumb given below in designing your retention and key management policies.

Recommendation: Define only as many keys as can be held in a drive.

Rule of thumb:

(1 + Number of sites sharing data) * (Data retention period) * (Frequency of key changes) <= 32

We include another example to illustrate how to apply this recommendation.

Retention Policy Example

In our example, London is the site that is sharing data with the most locations. If we assume a five year retention period for data stored in London, we see:

- London needs to write/read data that it creates. This is the one in our formula.
- London needs to read data from 5 remote sites.
- The data retention period is 5 years.
- Write keys change once per year.

Entering these values into our formula, we get

(1 + 5) * 5 years * 1 key change per year = 30 keys,

which is less than the 32 key threshold.

If the company's business rule were to change keys every quarter, then they would have an operational issue. In that case,

(1 + 5) * 5 years * 4 key changes per year = 120 keys,

which exceeds the 32 key threshold.

The T10000 encrypting solution can handle this "overbooking" scenario, but we recommend that you avoid it if possible. To manage this scenario, you must rotate a read key out of each drive pool mapping to allow for the addition of a new write key to each drive pool each quarter. This operation requires the following steps:

1. First, create a new key set and a new key, and assign the new key to the new key set.
2. Next, you must remove one key, probably the oldest key, from the drive pool mapping. To do this, you must first remove the key set containing that old key from the mapping. The steps required to do this depend on how you have created the key sets included in the drive pool mapping.
3. If the old key is the only key in its key set, then remove the key set containing the old key from the mapping, add the key set containing the new key and assign the new key as the write key.
4. If the key set containing the old key contains other keys as well, then remove that key set from the mapping, add the key set containing the new key and assign the new key as the write key. Save the mapping, but do not transfer it to the drive pool. Remove the old key from its key set, and add the modified key set back to the mapping. The mapping now contains the new (write) key and all of the original (read) keys except the old key that you removed.

In addition to the manual steps required to rotate out an older read key to add a new write key, any subsequent request to read data written with that key will require that you remap the deactivated key to the drive pool before the read operation can take place. Loading this key requires the temporary deactivation of another key, using steps similar to those listed above. Then you must reverse the process to restore the newer key to the drive pool mapping.

The discussion above demonstrates clearly the additional overhead of managing more than 32 keys per drive pool. These operations are feasible but require manual intervention, which increases the chance for operator error, and increased management of both drives and media. You will likely find this increase in management overhead unacceptable when you have worked hard to automate and streamline your operations.

Chapter 2: Key Management Operations

A security solution can be successful only if the individuals who implement and manage it are honest and trustworthy. At some level, almost all data security approaches come down to an issue of individual trust; the StorageTek T10000 encrypting solution is no different. Should access to the Key Management Station be compromised, then the security of the whole system is compromised. Should an unauthorized user gain access to the system, log files on the KMS system (`/var/adm/kms.log and /var/adm/messages`) will capture an audit trail of all key transactions and operator actions to enable you to trace what was done and by whom. Defining KMS access and network security practices becomes even more important when the KMS is attached to an accessible (corporate) network.

For activities outside the KMS system (for example, opening a library and adding or removing cartridges), you may require the presence of an auditor. We also recommend that your Electronic Data Processing (EDP) auditors review the T10000 encrypting solution prior to its deployment. They must be aware of how encrypting technology could impact disaster recovery operations as well as day-to-day operations. In addition, you may wish to have an auditor present when an operator is loading encryption enabling keys onto drives or generating media keys.

Setting Up the Key Management Workstation

The Key Management Station Installation and Service Manual, listed in the References section below, describes in detail the setup of the key management workstation and its token bay. We address here only topics that are not included there or that we feel need further elaboration or emphasis.

The system defines three default users: *root*, *kmsadmin* and *kmsuser*. The passwords for these users are common to every KMS manufactured. We recommend changing these passwords immediately according to the guidelines included in the Installation and Service Manual. Because the *root* and *kmsadmin* logins allow access to high-level security functions, we recommend that distribution of these logins be restricted to only a limited number of most-trusted users.

A critical component of key security is the encryption of the KMS database entries (drive PC keys, device and media keys). The KMS employs two levels of encryption.

- First, it encrypts key entries in the database, written to the local hard drive, using the database key.
- Then it encrypts database backup files, written to the external USB drive, using the backup key.

You control these keys and should input them prior to adding any entries into the KMS database. You will use the scripts *`/opt/SUNWkms/app/setup/set_dbkey`* and *`/opt/SUNWkms/app/setup/set_backupkey`* to input these keys.

Each script prompts you to input a 64-character hexadecimal key and validates the number and type of characters input but does not require a second input for verification. These scripts display their input strings on the screen in plaintext. Therefore, only a trusted user should execute these scripts, and you should store these encryption key values securely in hard copy or other format. A restore of the KMS database from one of the backup copies stored on the external USB drive requires you to input the backup key used to encrypt the file.

You may wish to change either or both of these keys periodically to provide extra security. To do so, simply rerun the scripts described above. If you change the database key, the KMS will use the new key value to re-encrypt all stored data encrypted with the previous key. If you change the backup key, the KMS will encrypt subsequent backup files written to the USB drive with the new key. It will not re-encrypt previous backup files, so be sure to store securely both the old and new backup keys and to record the key used to encrypt each file.

Operator Roles

The next step is to log in to Solaris as *kmsuser* and launch the Web Browser to bring up the KMS GUI that controls the KMS application. The system defines three operator roles for the KMS application: Administrator, Security Officer and User. Security concerns may dictate that each of these roles be assigned to different individuals. However, if desired, one individual may fill two or more of these roles, but a different login is required to perform tasks associated with each role. The rights and responsibilities of the three roles have limited overlap, and implementing the key management strategy for any T10000 encryption drive environment will require the use of logins for all three roles.

Initially only the default Administrator operator *KmsAdm* has access to the KMS application. This operator has only the rights and responsibilities of any Administrator. You should use the initial login of this operator to create one or more Security Officer logins and, if desired, other Administrator logins. You will then use the Security Officer login to create one or more User logins and, if desired, other Security Officer logins.

NOTE: Creating a second Administrator login is advisable so that someone with the required permissions is always available to create keys or to change the password on the first Administrator login in case the person with that login breaks trust or will no longer be acting in the Administrator role. (Alternatively, you may mark the login as Inactive. This action also requires an Administrator login.) In addition, the password for the KmsAdm login is common to every KMS manufactured. We strongly recommend that you change this password on the initial login, using the guidelines specified in the Installation and Service Manual.

The system restricts view and modifies access of these logins. An Administrator login has view/modify access to all Administrator and Security Officer logins. A Security Officer login has view/modify access to all Security Officer and User logins. A User login has no view or modifies access to any operator login.

Critical configuration tasks require at least two logins with different privileges. For example, importing raw keys into the KMS database and creating or modifying keys and key sets requires the Administrator login, while only a User operator may modify the mapping of key sets to a drive pool. However, deactivating or reactivating a key requires both the User and Administrator. Only the Security Officer may enable encryption on the T10000 drives selected for archiving the customer's most sensitive data (by writing device keys to a token and transferring those keys to the drives), but the final steps needed to configure the drives to encrypt data require the User login. Once you have defined the drives and configured them for encryption, the User operator can handle the daily key management tasks, but creating and adding a new write key to a drive pool mapping or enabling a replacement drive requires cooperation among operators with different privileges.

The system logs each operator action on the database to the file `/var/adm/kms.log`. However, the system can only tag these log entries with the login of the operator performing the action. Therefore, to ensure a detailed audit trail of all system activities, you should not allow individual operators to share logins but require that each operator log in with a unique identity. Enforcing this restriction will guarantee that each action taken on the database is associated with a specific operator. Only the root login to the KMS workstation has delete permission on the kms.log file, so the audit trail is secure.

Key and Key Set Creation

Once you have selected a key management strategy, implementation of that strategy is the next step. The goal is an implementation that allows efficient day-to-day operations and the flexibility to support changes in business requirements. Attaining this goal will minimize management overhead and maximize satisfaction.

Key Distribution Example Revisited

We return to the key distribution example included earlier. Each site will require a write/read key. By assigning each of these keys to a separate key set, aggregating keys as needed for the DR sites is easy.

Location	**Key Set**	**Write/Read Key Identifier**
Tokyo	KS1	K1
San Francisco	KS2	K2
New York	KS3	K3
Chicago	KS4	K4
London	KS5	K5
Singapore	KS6	K6
Frankfurt	KS7	K7
Sao Paulo	KS8	K8

The drive pool mapping required for each site would be:

Location	**Key Sets in Drive Pool Mapping**	**Write/Read Key**	**Read Keys**
Tokyo	KS1	K1	
San Francisco	KS2	K2	
New York	KS2, KS3, KS4, KS5	K3	K2, K4, K5
Chicago	KS4	K4	
London	KS1, KS3, KS5, KS6, KS7, KS8	K5	K1, K3, K6, K7, K8
Singapore	KS6	K6	
Frankfurt	KS7	K7	
Sao Paulo	KS8	K8	

Creating a key set for each write key facilitates changes to the drive pool mappings needed to reflect changes in business requirements. For example, adding a new data center in Tel Aviv necessitates only the creation of key set KS9 with write/read key K9 and the addition of key set KS9 to the drive pool mapping for London. To support moving the United States DR center from New York to Chicago, simply remove key sets KS2, KS4 and KS5 from the drive pool mapping for New York and add key sets KS2, KS3 and KS5 to the drive pool mapping for Chicago.

Changing Write Keys

As indicated above, changing keys is mandatory if you have reason to believe that your data security approach has been compromised. The ease with which you can complete this operation is a critical test for any key management strategy implementation.

The KMS system provides three methods to assign new write keys. Circumstances will dictate which you should use, but a key management strategy that allows the use of Method 1 is the most efficient.

Method 1

Create a new key set to contain the new key and add to the drive pool mapping.

- Create new key set.
- Create a new key and assign it to the new key set.
- Add the new key set to the existing drive pool mapping, and assign the new key as the write key.
- Write media keys for the new mapping to the token and transfer them to the drive pool (keys will be automatically transferred to the drive pool when a KMS is setup to manage tokens using a network connection).

Method 2

Select a new write key from the read keys in the drive pool mapping.

- Initially the drive pool mapping contains a pool of several keys with one selected as write key.
- Modify the drive pool mapping by selecting a new write key.
- Write media keys for the new mapping to the token and transfer them to the drive pool. (keys will be automatically transferred to the drive pool when a KMS is setup to manage tokens using a network connection)

Method 3

Add a new key to an existing key set mapped to the drive pool.

- Since you cannot modify an active key set, remove the existing key set from the drive pool mapping.
- If this key set is the only key set in the drive pool mapping, map another key set temporarily to the drive pool as a placeholder and designate a key in it as write key. (If no unmapped key set is available for use as a placeholder, creating a placeholder key set and key will be necessary.)
- Create a new key and assign it to the unmapped key set.
- Add the modified key set to the drive pool mapping, remove the placeholder key set (if necessary) and assign the new key as write key.
- Write media keys for the new mapping to the token and transfer them to the drive pool.

NOTE: Method 3 never actually transfers the placeholder key set and write key to the drive pool. They simply allow us to satisfy the application's requirement that a drive pool mapping must always contain at least one key until we can make the desired modification to the previously active key set.

Method 1 is the technique used in our previous example. It requires that you create a new key set for each new key created. This really is not a new constraint, since if all previously generated key sets are mapped to drive pools, as they most likely will be, the application will require that you generate a new key set before it will allow you to create a new key anyway. Using this technique is simple and flexible, satisfying both of our implementation criteria.

Clearly, Method 3 is not the method of choice. However, it illustrates that the KMS application provides a mechanism for modifying a drive pool mapping even when all of the drive's keys are collected into one key set. You may perform both additions and deletions using variants of this technique.

At first glance, generating a pool of keys, placing them all in one key set and using Method 2 to add new keys appears to be the simplest technique, but it may prove too simplistic to be workable in real-world situations. We revisit the previous example to see what issues may arise when using the approach of pre-populating a key set.

Suppose that for the New York DR site, we had a mapping consisting of a single key set containing New York's write/read key K3 and read keys K2, K4 and K5 for data from San Francisco, Chicago and London, respectively. Now suppose that the United States DR site is moved from New York to Chicago. We want to continue to use K3 as New York's write/read key and add keys K2, K3 and K5 to Chicago's drive pool mapping. The customer's business rules also make it necessary to remove keys K2, K4 and K5 from New York's drive pool mapping, since New York no longer needs to have access to data written in San Francisco, Chicago or London. To effect these changes, we must now employ Method 3 (in reverse) to remove keys from the single key set used in New York's drive pool mapping, then again to add keys to Chicago's key set KS4, assuming we wish to continue with just one key set per drive pool mapping. In each case, we must use a placeholder key set since each drive pool mapping contains only one key set.

Clearly, the key creation and assignment strategy illustrated in the key distribution example above, which allows the use of Method 1 to change key assignments, provides much more flexibility to change with changing business requirements. The difference in effort required in these scenarios serves to illustrate the importance of detailed, thoughtful planning before implementing a key management strategy.

Tokens

The Token can operate in two modes in the system. The first mode is that of an Enabling Key Token (EKT) that provides the means of sharing device keys between the KMS and the drive. The second mode is that of an Operational Key Token (OKT) that transmits media keys from the KMS to the drive and provides secure local storage of keys when attached to the drive network.

The security model uses the device keys to encrypt the media keys, securing them when they are stored on the token and during communication between the token and the drive. Under control of the Security Officer operator, the KMS generates device keys without displaying their values to the operator and writes them in encrypted form to the token, creating an EKT. When initializing a drive as encryption-ready for the first time, the KMS encrypts the first set of device keys using the PCkey value stored in the drive during manufacture. If you later update the device keys on a drive, the KMS encrypts the new set of device keys using one of the current device keys assigned to the drive. This process is highly secure since no device key value is available in plaintext while in transit from the KMS to the drive. Furthermore, once the token transmits the device keys to the drive, the drive stores the device keys in its non-readable, non-volatile memory. After successfully transmitting device keys to the drive, the token erases them from its memory. However, it retains the Crypto Serial Numbers (CSNs) for the drives associated with those keys. Token LED 3 (fourth from left) on an EKT flashes green as long as either keys or drive CSNs remain in its memory.

A User operator creates an OKT by writing media keys for a drive pool mapping to the token. As with device keys, the KMS writes media keys in encrypted form to the token without displaying their values to the operator executing the procedure. As noted above, device keys protect these media keys when stored in the token and when in transit to the drive. When transferred to the drive token bay or delivered over the network, the OKT broadcasts a message over the LAN to all drives for which it contains keys. The drives then initiate a communication protocol that transmits their media keys to them.

The drive stores these keys in its volatile memory to ensure that it will not retain them if it loses power for any reason, for example, when someone removes it from the library. Unlike device keys, media keys remain in the OKT's non-volatile memory unless purged by a reset as described below. The OKT should remain in the drive token bay at all times to supply media keys to the drives in the pool as needed. Token LED 3 (fourth from left) on an OKT is solid green.

Resetting Drives and Tokens

Security concerns require that you should clear both device and media keys from the drive's memory before removing it from the library for any reason. Powering off the drive will clear its media keys but not its device keys. Only resetting the drive will clear both device and media keys. You will use the reset button on the front of the token for this task. To reset the drive, you may use any token containing the CSN of the drive, either an EKT previously used to transmit device keys to the drive or an OKT containing media keys for the drive. Place the token in the drive token bay, and press and hold the reset button for at least 3 seconds. The token will transmit a "clear keys" message to all of the drives for which it contains CSNs.

WARNING: This procedure will reset ALL drives on the LAN with the token bay for which the token contains CSNs. Make sure that the token contains CSNs for only those drives you intend to reset.

To reset the token itself, deleting all keys, drive CSNs and IP information from the token's memory, remove the token from the drive bay and reinsert it into the bay while holding the reset button. (Make sure that all drive resets have initiated before resetting the token.)

Using the token to reset drives is the recommended method. In fact, you may use an EKT containing CSNs for all drives on the LAN to clear all keys from all drives on the LAN in case of an emergency. However, if the intent is to reset only one or a few drives, the most efficient method may be to use the Reset Drive function from the Virtual Operator Panel (VOP) management software for the T10000 drive.

OKT Timestamps

Included in the security model for this solution is the assurance that a T10000 encryption drive will accept a new key mapping only if it is newer than the mapping it currently has. Each key mapping assigned to the drive carries with it a timestamp that the drive uses to decide whether to accept or reject a subsequent key message from the token.

When you modify a drive pool mapping, the KMS generates a timestamp for that mapping. When you write the media keys associated with that mapping to the token, the KMS transmits the timestamp associated with the mapping with the media keys. When the drive receives the new media keys and updates its key assignment, it stores the timestamp for the new key mapping in its non-volatile memory, overwriting the timestamp for the outdated key mapping.

NOTE: The timestamp associated with a key mapping is the time when you last updated or refreshed the key mapping for that drive pool, NOT the time when you wrote that mapping to the token.

What happens if a drive reboots, and for some reason the only token available, when it comes online contains an outdated set of media keys? Will the drive acquire and use the outdated keys? No, the drive will still have the timestamp associated with its last set of media keys in its non-volatile memory. If this timestamp is later than that of the key set available from the token, the drive will refuse the offered keys. Its encryption LED will revert to solid amber, indicating its need for media keys, and the status fields on a VOP session screen will report Encryption and Need OKT, respectively.

Be aware that drive sends no error message to the token in this circumstance, and the encryption LED will likely be hidden inside the library. The drive will still log in to the FC switch to which it is attached, but it will refuse any read/write requests with a read error on the media.

A timestamp accompanies each device key mapping as well. However, the KMS creates new device keys each time you write an EKT for a drive, and the token discards device keys from its memory as soon as it transmits them to the drive. Therefore, timestamp issues do arise for device keys.

Redundant Tokens

For security, the drive token bay contains slots for two redundant tokens. Each time you write an OKT, you should write a second OKT for the same drive pool mapping. The time at which you last modified the drive pool mapping determines the OKT timestamp, not the time at which you last wrote the OKT. Therefore, this process produces two OKTs with identical key information. Placing both tokens in the drive token bay greatly decreases the chances of a drive rebooting or losing power without an available source for media key refresh when it comes online.

What happens if you update only one token with a new set of media keys and place it in the drive token bay while the outdated token is still in place? Both tokens will broadcast a “keys available” message to the drives in the drive pool. Each drive will check the timestamp in each message and accept a key update only if the timestamp accompanying the keys is later than the timestamp in its own memory.

Verifying Key Transfer

When you transfer device keys to the drive for the first time, the encryption LED changes from green to amber to signal that the drive now has its device keys and is enabled for encryption. Then when you transfer media keys to the drive for the first time, the encryption LED changes from amber to red to signal that the drive is fully encryption ready and that the drive will encrypt all future data writes. To avoid changing keys in the middle of a write operation, if the token transmits a “keys available” message when the drive contains a tape, the drive accepts the new keys but does not update the key assignment to the encryption engine until the media is unloaded. The Virtual Operator Panel reports **Holding Keys**, and the encryption LED remains amber.

However, when you make subsequent media key updates, you will receive no visible signal to indicate that the key update was successful. So how do you determine if the drive, in fact, has updated its key assignment? In a pool of 100 drives, how do you verify that no glitch has occurred in the transmission of the new key set mapping to any of those drives? Presently, the only method of verifying that a drive has updated its key assignment is through the VOP software. As described above, a timestamp generated when you last modified the drive pool mapping accompanies each new media key update, and the drive stores this timestamp in its non-volatile memory. VOP will report the timestamp of the key mapping currently stored in the drive.

To see this timestamp,

1. Start the VOP application
2. Navigate to File → Connect to Drive,
3. Enter the IP address of the drive. After the software connects to the drive,
4. Navigate to Retrieve → View Drive Data → Encrypt.

The OKT timestamp value displayed should be the same for every drive in the drive pool.

Recording this timestamp before and after each media key update will allow you to verify that every drive in the pool has received and processed the new key update.

Database Backup and Restore

The KMS backs up its database automatically whenever the operator writes keys to a token, logs out of the application or allows a session to expire. If the external USB drive is mounted, the system writes database backups to the `/export/home/kms/mnt_backups` directory on that drive. Otherwise, it writes its database backups to the same directory on the local hard drive. You may restore the database from any of the backup files in the mnt_backups directory using the `run_restore_db` script located in the `/opt/SUNWkms/app/sbin` directory. To run this script, log out of the KMS application, and log in to Solaris as *kmsadmin*.

To display usage information, execute the following command:

```
/opt/SUNWkms/app/sbin/run_restore_db -h
```

To display a list of available backup dump IDs from which a restore can be done, execute

```
/opt/SUNWkms/app/sbin/run_restore_db list
```

To restore from a backup file, you must supply the 64-character hexadecimal backup key used to encrypt the file. You must also supply one of the dump IDs displayed with the list option, or you may use the latest option to restore from the most recent backup taken. The command to restore the database from the most recent backup file is

```
/opt/SUNWkms/app/sbin/run_restore_db -k <backupkey> latest
```

where backupkey is the encryption key mentioned above. If you wish to restore from an earlier backup file, replace the latest argument with the appropriate dump ID from the list of available IDs obtained by using the list argument.

NOTE: If you misplace your copy of the backup key, you can use the CLI command `/opt/SUNWkms/app/cli/view_backup_key` to write the current backup key value to the screen or to a file. (See the Key Sharing section below for more information.) If you have changed the backup key at some point, you will not be able to recover previous backup keys by this method.

Restoring Database to Different KMS

Circumstances may require that you restore a KMS database from a USB drive either to a freshly installed replacement KMS or to an operational KMS in a different location. Doing so will require operators with *root* and *kmsadmin* logins. If you are restoring on a freshly installed replacement unit, you should follow the steps in the **Installation and Service Manual**

IMPORTANT: Do not run the script `kmsdrive_prepare`. This script will format the USB drive and destroy the backup files from which you will restore the database.

If you are restoring to an operational KMS, none of the setup steps should be necessary. Log in to the target system as *root* user, and open a terminal window. Make sure that the directory /export/home/kms/mnt_backups is not open in any terminal window. Unmount the USB drive by executing the following command at the terminal prompt:

```
umount /export/home/kms/mnt_backups
```

Disconnect the existing USB drive, and connect the drive containing the source backup file in the same port.

Execute:

```
/opt/SUNWkms/app/setup/kmsdrive_setup
mount -a
ls -lrt /export/home/kms/mnt_backups
```

to create the mount entry for the new drive, mount it and verify that the source backup file is available.

To restore the database, log in as *kmsadmin*, and execute

```
/opt/SUNWkms/app/sbin/run_restore_db -k <backupkey> latest
```

as described above to restore the database from the most recent backup file, or if desired, replace latest with the dump ID of an earlier backup file.

The database restore will overwrite the KMS ID of the target system with that contained in the source system.

To restore the KMS ID of the target system, execute

```
/opt/SUNWkms/app/setup/set_kmsid
```

The script will prompt you to enter an 8-character string. To restore the KMS ID to its original value, input the last 8 characters from the serial number on the chassis of the target KMS workstation.

Key Sharing

For any of a variety of reasons, you may wish to share data written at one site using a particular encryption key with another site that does not have that key in its KMS system. This could occur as part of a disaster recovery process or if company A procures the assets of company B. To share this information, you must transfer the encryption keys for this data to the destination site.

To facilitate sharing of encryption keys between different KMS systems, our manufacturing process assigns a unique identifier called the KMS ID to each system. The system embeds this identifier in the Key ID of every key generated automatically by the operator. In addition, each Key ID generated on a given system contains an octet containing the ordinal number of the key in the sequence of all keys created on that system. This ordinal number guarantees the uniqueness of all automatically generated keys on a given system, and the KMS ID guarantees the uniqueness of each key across all KMS systems. This uniqueness makes it possible to export a key from one KMS and add it to another KMS with no possibility of duplication.

The KMS provides a function to export keys from its database. To invoke this function from the GUI, log in to the KMS as Administrator and navigate to the Keys → Media Key Export screen. Select some or all of the keys from the displayed list, and enter a file name (name only, no path) to contain the exported key information. The function will create a file with that name in XML format in the keys directory /export/home/kms/mnt_keys and will write the description, key ID and key value for each of the selected keys to it.

WARNING: Exported key information is plaintext and is vulnerable to unauthorized access. We recommend that you impose strict security measures on exported information.

You may want the key information in text format for printing or for sharing media keys with an independent KMS. If you export keys to the file /export/home/kms/mnt_keys/keyfile, then the following command sequence will write the contents of that file in text format to the file keyfile.txt in the same directory:

```
cd /export/home/kms/mnt_keys
/opt/SUNWkms/app/cli/export2text keyfile keyfile.txt
```

You may also use the CLI command /opt/SUNWkms/app/cli/export_key to export keys.

To get usage information, execute the following:

```
cd /opt/SUNWkms/app/cli
man export_key
```

NOTE: CLI commands require either a *kmsadmin* or a *kmsuser* login. The export_key command also requires that you input login information for a KMS Administrator operator. Be sure to check the man page for complete usage information. The information displayed with the –h option does not describe how to export information to the screen.

An administrator on an independent KMS can utilize the media key export file created in the above example to import media keys to the local KMS. This is done through the GUI using the administrator role on the Keys → Media Key Import screen. This can be useful when a customer needs to share encrypted tape cartridges with another organization or a 3rd party disaster recovery site.

KMS Mirroring

The KMS system provides multiple layers of protection for the keys that encrypt your data.

- It generates keys without displaying them to the operator performing the task.
- It uses the database encryption key to encrypt keys before storing them in the database.
- It uses the drive's device keys to encrypt its media keys before writing them to the token.
- It encrypts backups of the key database before writing them to the USB drive.

To provide redundancy for the KMS system itself, you may configure a second KMS system as a standby backup system. In this configuration, the primary KMS writes a set of encrypted database backup files to its local USB drive whenever the operator writes keys to a token, logs out of a session or allows a session to expire. Then at one-minute intervals, if a new set of backup files is available, it copies those files to the USB drive attached to its standby system.

The standby system serves only as a repository for database backup files. It cannot function as a KMS system while serving as a standby for another system. If the primary system fails, requires maintenance or becomes otherwise unusable, you will execute a manual process to shut down the primary (if it is still operational) and to complete the failover to the standby. If the primary system is still functioning, the shutdown steps will take one final database backup, copy it to the standby and stop the KMS processes.

You may choose to shut the system down, leave it up with no KMS functions running or reverse the system's role and make it the standby. Then you will restore the database on the standby system using the latest backup copy and run a script to start the KMS services, making it either a standalone KMS or the primary KMS in a primary-standby pair. The database restore will transmit the KMS ID and the database and backup encryption keys of the original primary KMS to the new primary KMS. Executing the set_kmsid script as described in the Restore Database to Different KMS section above will restore the original KMS ID to the new primary KMS.

The **KMS-to-KMS Setup Cookbook** white paper listed in the References section below contains detailed steps for configuring primary-standby pairs in a variety of scenarios based on the network configurations of the two systems. It also details the steps for failing over a primary system to its standby.

A primary KMS may have only one standby, and there is no provision for a redundant standby. However, the primary KMS system, its local external USB drive and the remote USB drive on the standby would all have to fail to lose key data for your encryption solution.

Chapter 3: Error Scenarios

A successful key management strategy minimizes the risk of occurrence of error scenarios.

TIP: Ensuring that all previous write keys remain in the drive pool mapping as read keys and that all drives in the same library are in the same pool will greatly reduce the potential for errors.

However, should such a scenario occur, the first and most important step is to determine why the error occurred. You carefully designed your key management strategy to avoid such situations. Was it operator error, a failure of the key management strategy or an unauthorized attempt to access data? We recommend that you determine the source of the error before you initiate recovery steps. It may be that your security system is working as designed and has just foiled an attempted security breach. On the other hand, perhaps the failure is a symptom of a flaw in your security strategy that needs to be remedied before you attempt further data accesses. Once you have determined and remedied the source of the error, you may initiate recovery.

Missing Key Errors

The most common error occurs when an encryption-enabled drive attempts to read data for which it does not have the encryption key.

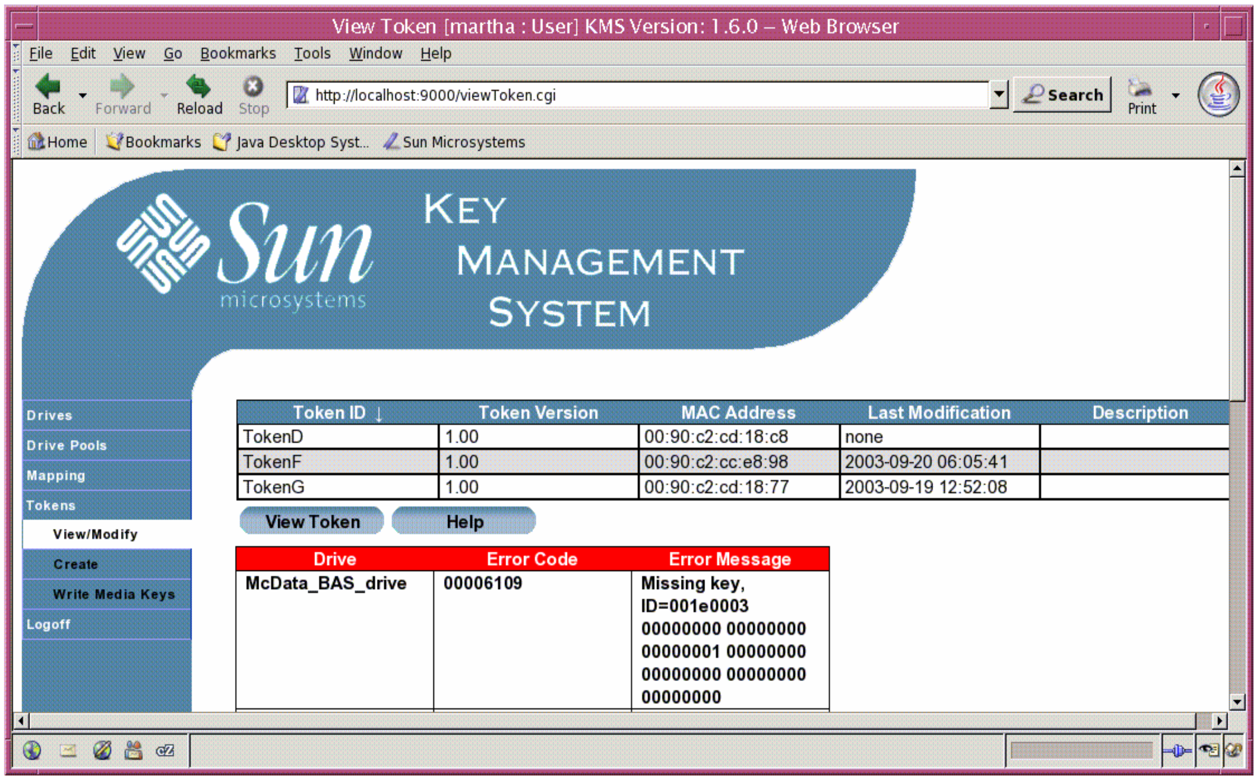
IMPORTANT: Either to read data from or append data to a tape containing encrypted data, the drive's assigned media keys must include ALL keys that were previously used to write data to that tape.

We use a concrete example to illustrate this requirement. Assume that you have written File1 to a tape using write key K1, File2 using write key K2 and File3 using write key K3. Let's look at the key assignments required for the listed tasks:

Task	Read Keys
Retrieve File1	K1
Retrieve File2	K1, K2
Retrieve File3	K1, K2, K3
Append File4 using write key K4	K1, K2, K3, K4

An attempt to execute any of these tasks without all of the required keys in the drive pool mapping will result in the drive reporting a read error to the application, causing the job request to fail. The drive will also report a missing key error to the token and a data decryption error to the operating system on the server that initiated the I/O request. An attempt to erase or re-label a tape without the key used to initialize the tape will generate similar errors, but the operation should succeed.

When the drive returns a Missing Key error to the token in the drive token bay, the token signals the receipt of an error message by flashing its red error LED (the leftmost LED on the front of the token). To retrieve the error message without clearing the error, transfer the token to the KMS token bay and navigate to Tokens → View/Modify. Click on the View Token button to display the error message. You will see an error message something like this:



Token ID ↓	Token Version	MAC Address	Last Modification	Description
TokenD	1.00	00:90:c2:cd:18:c8	none	
TokenF	1.00	00:90:c2:cc:e8:98	2003-09-20 06:05:41	
TokenG	1.00	00:90:c2:cd:18:77	2003-09-19 12:52:08	

Drive	Error Code	Error Message
McData_BAS_drive	00006109	Missing key, ID=001e0003 00000000 00000000 00000001 00000000 00000000 00000000 00000000

In this display, the Drive field is the name of the drive reporting the error. The Error Code 00006109 signifies a Missing Key error, as indicated by the Error Message. The ID shown in the Error Message field is the Key ID of the missing key displayed in 8 hexadecimal octets.

After viewing the error message in the token status field, the error LED will continue to flash since the token still retains the error information in its memory. A write operation to the token will clear the error message. Navigate to Tokens → Write Media Keys. Select the drive pool containing the drive that reported the error and then select the token where the error status was observed and click Apply.

To recover from missing key errors that prevent successful restores of encrypted data,

1. Modify the drive pool mapping to include all of the keys previously used to encrypt data on the media.
2. Write a new OKT for the drive pool, and transfer the new media keys to the drive pool (keys will be automatically transferred to the drive pool when a KMS is setup to manage tokens using a network connection).
3. Check to determine the states of the drive and media. Even though the job request failed, the application should have issued commands to unload the media, remove it from the drive and replaced it in its home slot in the library. However, depending on the circumstance and the application being used, the application may down the drive and/or freeze the media.
4. Restore the drive and media to usable states, and restart the job. The job should complete without error.

Media Format Errors

Appending data to media containing non-encrypted data with an encryption-enabled drive is also disallowed. The write job will fail, and the drive will report an incompatible format error to the OS on the server issuing the I/O request, but it will not report an error to the token in this case. An attempt to use a non-encryption drive to read or to append to media containing encrypted data will generate a similar error.

Synchronization Errors

A more subtle error occurs when the KMS application and the drive get out of sync. In these situations, the drive reports a key message decryption error when it attempts to retrieve keys from the token. We list several such scenarios and recovery steps for each below.

Scenario 1

While enabling a drive for encryption, the operator inadvertently executes the function Write Device Keys, writing keys to the token, then executes it again without transferring the first new set of keys to the drive. He transfers the token to the drive bay (keys will be automatically transferred to the drive pool when a KMS is setup to manage tokens using a network connection), but the drive's encryption LED does not change to amber and the token's error LED starts flashing. An error occurs because the drive and the KMS are now out of sync.

Each execution of the Write Device Keys function generates a new set of keys used to encrypt communications between the token and the drive. The KMS and the drive must have the same set of device keys in order to communicate. If the operator generates a set of device keys but does not transfer them to the drive, then the KMS uses that new set of keys to encrypt the next key transmission. However, the drive has only its old set of device keys to decrypt that key message and reports a key message decryption error to the token.

Recovery

1. The token error LED is flashing. In the GUI, select the token reporting an error to view detailed status.
2. View and clear the error by navigating to Tokens → Write Device Keys. Select the target drive and click Apply. The error information will appear on the screen, but the EKT write will not succeed.
3. Press and hold the reset button on the front of the token for at least 3 seconds to reset the target drive. The drive will reboot. When the drive comes online, its encryption LED will be flashing green, and VOP will display **==RESET==** in its first status field
4. Repeat Steps 1-2 to clear the error from the token. Select the "Use PCKey" checkbox, and click Apply a second time to write the EKT.
5. Verify that the encryption LED on the drive is now amber, indicating that the drive is encryption-enabled.
6. If you wrote an OKT prior to Step 5 in this sequence in preparation for sending media keys to the drive, the device keys used to protect that OKT will now be out of date. Navigate to Tokens → Write Media Keys, and select the drive pool containing the newly enabled drive. Without changing the key assignment, refresh the drive pool mapping by clicking Apply to generate a new timestamp. Write the OKT with the refreshed mapping and transfer to the drive bay. Verify that the encryption LED on the reset drive changes to red. The OKT timestamp displayed in VOP should be the same for each drive in the pool.

Scenario 2

The customer wants to update both device keys and media keys for a drive pool. The Security Officer operator writes an EKT containing new device keys for all of the drives but is interrupted before he transfers the token to the drive bay. The User operator logs in, updates the drive pool mapping with the new key assignment, writes the OKT and transfers the token to the drive bay. The token/drive communication fails because the encryption keys are out of sync between the KMS and the drive.

Recovery

1. Follow Steps 1-5 in the recovery for Scenario 1 except to write new device keys to the entire drive pool, select all drives in the drive pool rather than just a single drive.
2. If the KMS is operating in an “air-gap” configuration and is not connected to the corporate network, transfer the token back to the KMS bay. Write the OKT and transfer it to the drive bay to update the media keys on the drives in the pool. The encryption LED on each drive will change to red to indicate that the drive is fully armed for encryption.

Scenario 3

The operator writes an EKT for all N drives in a drive pool. When he delivers the EKT to the drive bay, the Nth drive is powered off. He returns the token to the KMS bay and overwrites it with media keys. Then he delivers the OKT to the drive bay, transferring media keys to the N-1 drives that have power. Later the Nth drive receives power.

Recovery

3. The token error LED goes on because the Nth drive doesn’t have the device keys used to make the OKT. Follow Steps 1-5 of the recovery for Scenario 1, selecting the Nth drive.
4. If the KMS is operating in an “air-gap” configuration and is not connected to the corporate network, transfer the token back to the KMS bay. Navigate to Mappings → View/Modify, and select the affected drive pool. Without changing the drive pool mapping, click on Apply to refresh the mapping and create a new timestamp. Write the OKT and if necessary, transfer it to the drive bay to update the media keys on the drives in the pool. Drive N now has its media keys, and its encryption LED is red. The other N-1 drives also pick up this retransmission of their media keys since the timestamp is newer than the one they have. All drives are now using the same media keys and share the same OKT timestamp.

Scenario 4

The customer buys N drives. The User operator enters the drives into the KMS, and the Security Officer operator writes an EKT for N drives. However, the N drives selected mistakenly include only N-1 new drives and one old drive. He writes the EKT and if necessary, transfers the token to the drive bay. N-1 new drives get their device keys, one new drive gets no device keys and the error LED on the token is flashing (if the old drive is on the same LAN with the N new drives).

The error LED on the token is flashing because the device key message for the old drive was encrypted using the PCkey for the drive, not its current communications key. Why? Because the operator’s intention was to write an EKT for N new drives, so he selected the Reset Drive box for all of the drives. Now the KMS and the old drive are out of sync, and the only recourse is to reset the old drive.

Recovery

Follow Steps 1-5 of the recovery for Scenario 1, selecting the Nth new drive and the old drive. After completing these steps, all N new drives and the old drive will have new device keys. Then write an

OKT containing media keys for the N new drives and the old drive, if it is to use the same media keys, and if needed, transfer the token to the drive bay. If necessary, write a new OKT for the old drive's pool and transfer it to the drive bay. All of the affected drives will then be fully encryption-enabled and ready to use.

Scenario 5

The operator creates a new EKT for the drives in a drive pool and updates the device keys on the drives. He then writes an OKT for that pool and if necessary, transfers the token to the drive pool to transfer media keys to the drives. When he checks one of the drives through VOP, it is reporting **Holding Keys**, indicating that it has received new device keys but is holding them (i.e., hasn't yet committed them to its encryption engine) until it receives a new media key transmission. Why didn't the media key transfer take place? The timestamp written to the OKT is the time at which the operator last modified the drive pool mapping. Since he made no change to the drive pool mapping before he wrote the OKT, the timestamp written to the token is earlier than the timestamp of the device key packages just received by the drives, so the drives ignored the OKT message. In this state, the encryption LED of the drive alternates between flashing red when doing I/O and solid amber during unloads, rewinds and idle periods to indicate that the operator has completed only half of the key update process.

Recovery

Navigate to Mappings → View/Modify. Select the affected drive pool and click Apply. Even though you made no change in the keys included in the mapping, the KMS will refresh the mapping with a new timestamp. Now write the OKT and if needed, transfer it to the drive bay. The timestamp broadcast by the token is now later than that stored by the drives, and the drives will accept the media keys from the token.

Scenario 6

The operator moves a drive from one drive pool to another. She then writes an OKT for the drive's new pool and transfers it to the drive bay. VOP reports no update in the OKT timestamp on the drive. The mapping for this pool has been in place longer than the mapping for the drive's previous pool. Therefore, the timestamp stored on the drive is later than that broadcast by the OKT, and the drive refuses the new media keys.

Recovery

Use the recovery steps for Scenario 5, selecting the drive's new pool.

Chapter 4: Limitations and Constraints

Why are there limitations and constraints? The need for securing data-at-rest, especially on removable media, is immediate. Given the ecosystem of components that comprise a tape-oriented system (backup applications, drivers, HBAs, switches, drives, libraries, management software), effecting widespread change can take time. Developing a fully integrated encrypting solution involves coordination with partners and standards bodies and a great deal of interoperability and compatibility testing. We are working toward such a fully integrated solution. Meantime the StorageTek T10000 encryption solution delivers an immediate answer to your data security problems that is able to work transparently without requiring changes to all the components in your tape environments. This initial delivery imposes some limitations and restrictions you should consider when deploying the solution. We expect to lift these limitations in the coming months as our industry adopts standards and approaches for encrypting data.

1. You can delete a tape drive database entity if it has been created erroneously, but you may not delete any other database entities such as keys, keys sets or tokens. You should plan very carefully prior to entering any entity into the database to ensure that the resulting configuration precisely implements your business requirements.
2. The name assigned to any key set, drive or token is permanent. If the function of that entity is subject to change, you should include any reference to that function in the description field rather than the name. The description field is editable when the entity is unmapped, i.e., when a key is not assigned to a key set or when a key set or drive is not mapped to a drive pool. You should design the naming scheme used in the creation of the KMS configuration carefully to convey useful information and still allow for functional changes.
3. Each database entity (key, key set, drive, token or operator) has an associated description field that may contain up to 256 characters. Descriptions containing long strings with no spaces (>35 characters in length) may cause display lists to be difficult to view.
4. You must import the PCkey and CSN for each drive one at a time. These keys, which are used to enable a tape drive for encrypted operations, are provided on a CD to automate this process.
5. You must enable encryption on a drive (i.e., transfer device keys to it) before adding it to a drive pool.
6. To perform **any** media access operation, you must provide **all** of the keys present on that media to the tape drive prior to the I/O request. For example, if you are going to append data to a tape with a new write key, the tape drive must have all of the keys previously used for write operations on that tape. Without these keys, the tape drive may not be able to read blocks near the location of the write operation or blocks that need to be read while moving to that location. If not all of the keys for blocks written to the tape in previous write operations are available, then the drive may report a missing key error as it navigates through encrypted data to the current backup. This requirement does not exist for write operations from the beginning of tape.
7. You may create any number of keys, but you may assign at most 32 keys to a key set.
8. You may assign multiple key sets to a drive pool mapping with a 32 key limit per mapping.
9. You cannot modify an active key set. A key set would need to be unmapped from the current drive pool to modify the keys. Alternatively, a new key set could be added and assigned to the active drive pool.
10. You may not permanently deactivate a key. That is, no key state is immutable. If you wish to deactivate a key with the intent of permanently expiring the key, i.e., denying access to all data written with that key, the key should be removed from all available key sets and the description field of the key can be edited with a warning to that effect, for example, ---EXPIRED 8-31-2007 – DO NOT REUSE---. Deactivating a key is not equivalent to deleting the data written with it.

Backups that were written with compromised keys will need to be re-written with active encryption keys.

11. After updating the device keys on a drive or a drive pool, you must refresh the media key mapping for the drive pool and create a new OKT for the drive pool. If you wish to retain the previous media key mapping, simply select the drive pool and click Apply. This will update the timestamp on the mapping without actually changing the media keys included in it.
12. If the KMS is operating in an “air-gap” configuration and is not connected to the corporate network and therefore does not have a communication path to the token, then the KMS will not automatically display an alert when the token in the drive bay receives an error message from a drive. The token error LED will flash, but the operator logged into the KMS may be unaware that an error has occurred since the token is enclosed in a library that is likely remote from the KMS. If it is a missing key error, the backup administrator will see errors in the application job monitor and the system log of the server generating the I/O request. Currently errors reported by the application are read, write or locate errors that are not immediately translatable to a missing encryption key problem, but the data decryption errors posted to the OS log are more helpful. Troubleshooting the problem will require the backup administrator to work with the KMS operator to determine why the error occurred and how to recover.
13. All drives must attach to a LAN containing a token bay. If the drives are distributed among multiple libraries, each library may have its own token bay and token(s), or one token bay may service all of the drives, provided all drives attach to a common LAN. Multiple tokens written by a single KMS may also service drives at different sites. If the KMS is operating in an “air-gap” configuration and is not setup to manage tokens over the network, a process must exist for securely transporting tokens from the KMS site to the token bays in the various data sites. The network constraints described above apply in all of these situations.
14. A primary KMS system may be mirrored to exactly one standby system. The standby is a cold standby, not an active KMS system. Failover from the primary to the standby is a manual process.
15. Closing the browser without first logging out results in no operator access to the KMS application for approximately 10 minutes. The zombie process left running when the browser is closed must time out before any access is allowed. The GUI runs in the browser, and there is no simple way to perform GUI functions with the browser closed. A future release will address this issue. To avoid the 10 minute delay, open a terminal window and type

```
rm /tmp/kms_sessions.pag
```

Summary

The Sun StorageTek T10000 tape drive encryption solution protects your data while at rest and in transit. You will enjoy the superb performance and reliability of the 4-gigabit T10000 tape drive together with the security of a sophisticated, multi-layered encryption scheme.

Careful planning and a full understanding of the features and constraints of this solution will allow you to devise an encryption key management strategy that will satisfy your security needs while at the same time providing the ease of use and flexibility needed to meet your changing business requirements.

References

1. Key Management Station and Data at Rest Encryption, Technical Brief, PN TT0018
2. Key Management Station Installation and Service Manual, PN 96260
3. Key Management Station Configuration and Startup Guide, PN 96261
4. KMS-to-KMS Setup Cookbook, White Paper

Appendix: Encryption Sense Codes

Drive	Tape	Key Status/Operation/Sense

		No Device Keys				Key Status	
		Write Data	WFM	Space/Locate	Read Data	Operation	
Encryption ON	Encrypted	4/2681/343c	4/2681/343c	4/2681/343c	4/2681/343c	Sense	
	Not Encrypted	4/2681/343c	4/2681/343c	4/2681/343c	4/2681/343c	Sense	
Encryption OFF	Encrypted	*No Error Reported 3/3005/32b9 (Reported on previous Space/Locate/ Read)	*No Error Reported 3/3005/32b9 (Reported on previous Space/Locate/ Read)	3/3002/0000	3/3002/0000	Sense	2G drive will report illegal format.
	Not Encrypted	No Error Reported	No Error Reported	No Error Reported	No Error Reported	Sense	

		No Media Key				Key Status	
		Write Data	WFM	Space/Locate	Read Data	Operation	
Encryption ON	Encrypted	4/2681/343c	4/2681/343c	4/2681/343c	4/2681/343c	Sense	
	Not Encrypted	4/2681/343c	4/2681/343c	4/2681/343c	4/2681/343c	Sense	
Encryption OFF	Encrypted	*No Error Reported 3/3005/32b9 (Reported on previous Space/Locate/ Read)	*No Error Reported 3/3005/32b9 (Reported on previous Space/Locate/ Read)	3/3002/0000	3/3002/0000	Sense	These errors would be prevented by "No Device Key" errors
	Not Encrypted	No Error Reported	No Error Reported	No Error Reported	No Error Reported	Sense	

		Matching Media Key Not Found		Key Status
		Space/Locate	Read Data	Operation
Encryption ON	Encrypted	3/2605/6109	3/2605/6109	Sense
	Not Encrypted	No Key Required	No Key Required	Sense
Encryption OFF	Encrypted	Can't Happen	Can't Happen	Sense
	Not Encrypted	Can't Happen	Can't Happen	Sense

		Have Media Keys				Key Status
		Write Data	WFM	Space/Locate	Read Data	Operation
Encryption ON	Encrypted	No Error Reported	No Error Reported	No Error Reported	No Error Reported	Sense
	Not Encrypted	*No Error Reported 7/3005/32ba	*No Error Reported 7/3005/32ba	No Error Reported	No Error Reported	Sense
Encryption OFF	Encrypted	Can't Happen	Can't Happen	Can't Happen	Can't Happen	Sense
	Not Encrypted	Can't Happen	Can't Happen	Can't Happen	Can't Happen	Sense

Sense Format - SK/ASCQ/FSC
SK - Sense Key
ASCQ - Additional Sense Code Qualifier
FSC - Fault Sense Code (Vendor Unique)

NOTE: * From BOT